eqexam: An Exam Construction Package

D. P. Story
Email: dpstory@uakron.edu

Processed January 7, 2020

Contents

1 Introduction

This package is my attempt at writing a set of macros for creating exams. The package can be used with or without web or exerquiz. When used without, what is produced is a document suitable for printing. When exerquiz is used, the same document is produced, but with hypertext links to solutions. This may be useful for publishing solutions on the web, or publishing pretests with/without solutions.

The package must be as flexible as possible vis-à-vis PDF. (1) web only does not add much, it does input hyperrref and test info such as \title, \subject, etc are placed in the Document Info fields of the PDF. (2) If exerquiz is also input (prior to eqexam), then hyperlinks will be created to the solutions to the test, if solutions are included at the end of the document. (3) If web and exerquiz are input, and the online option is taken, then the checkboxes will be come interactive, the space left to work the problems will be multi-line text boxes, fill-ins limited to True/False and simple text fill-in the blank will also become text boxes.

When in online mode, the student can take the test in a computer lab, the completed exam can be printed and handed in, or perhaps submitted server-side script.

(2011/05/13) The version of eqexam is a departure from previous versions. Previously, the list of problems were not in list, they were left-justified, with the problem number extending out into a little area determined by \oddsidemargin. This makes it hard to reformat a list of problems to fit into a custom book format. This new version defines a new environment, eqequestions, that makes each problem into a list. The list environment allows for an easy redesign of the formatting of the problems. The purpose of this new scheme, is to open up eqexam as a format package that can be used by author for writing a textbook.

The fortextbook option supplies support for authors writing a textbook. The exam environment is re-cast into the probset environment, it can be used to write problem sets within the text. there is a version for the instructor and the student. The instructors version writes answers to the problem sets to the margins (or inline). Two solutions are offered, short and long. The short solutions appear in the back of the book (odd-numbered ones for the student edition. The long solutions are used to build the stand-alone solutions manuals for both the student and the instructor. More details are found in 'Concerning the fortextbook option' on page 118, see also 'fortextbook Style File' on page 147.

2 Package options and Process Options

Let us catalog the options of this package.

2.1 Early inclusion of packages

We include these packages early in the process of loading.

```
⟨*package⟩
```

```
\RequirePackage{ifpdf}[2006/02/20]
\RequirePackage{ifxetex}[2006/08/21]
```

Very useful package for defining key-values.

```
\usepackage{xkeyval}
```

2.2 Options New to eqexam

`usecustomdesign`

Here are some options unique to this package. Use this option to avoid eqexam from setting up the "standard" page layout.

```
\DeclareOptionX{usecustomdesign}{\eqcustomdesigntrue}
\newif\ifeqcustomdesign \eqcustomdesignfalse
\DeclareOptionX{nocustomdesign}{\let\eqe@nocustomdesign=1}
\let\eqe@nocustomdesign=0
```

`fortextbook`

An option to extend the application of eqexam to provide support (exercises, providing solutions, short solutions, answers, and hints) for authors writing textbooks. See 'Concerning the `fortextbook` option' on page 118.

```
\DeclareOptionX{fortextbook}{\eqfortextbooktrue}
\newif\ifeqfortextbook \eqfortextbookfalse
```

`forinstr`
`forstudent`

These two options simply set a switch to signal the intention of the document author.

```
\DeclareOptionX{forinstr}{\eqforinstrtrue}
\DeclareOptionX{forstudent}{\eqforinstrfalse}
\newif\ifeqforinstr \eqforinstrfalse
```

`nomarginwrite`

The switch `\ifeqwritetomargins` is used by the `fortextbook` option. It is normally `true`, but if set to `false`, the `\AddToShipoutPicture` is not generated at the beginning of the document. Here is the code taken from below:

```
\ifeqfortextbook\ifeqwritetomargins
\AtBeginDocument{\chkmarginboxwidth
    \AddToShipoutPicture{\eqe@tb@shipout}}
\fi\fi
```

Using this option, the check for the margin width is not done, and writing to the margins is turned off. (`\marginpar` still works)

```
\DeclareOptionX{nomarginwrite}{\eqwritetomarginsfalse}
\newif\ifeqwritetomargins\eqwritetomarginstrue
```

Configuration Files. This section contains options for the configuration files.

cfg

The `cfg` option is used to specify a named configuration file, extension must be `.cfg`; usage `cfg=hwdoc`.

```
\define@key{eqexam.sty}{cfg}[]{%
    \def\arg@i{#1}\ifx\arg@i\@empty
    \PackageWarning{eqexam}{No value for ‘cfg’ specified}\else
    \def\ifeqexamCFG{true}\def\eqexamCFG{#1.cfg}%
    \AtEndOfPackage{\InputIfFileExists{#1.cfg}
    {\typeout{Inputting #1.cfg}}{\PackageWarning{eqexam}{%
        Cannot find configuration file #1.cfg}}}\fi
    }
\def\ifeqexamCFG{false}
\let\eqexamCFG\@empty
```

myconfig
myconfigi...myconfigvi

We offer seven sets of configuration files, that should be enough, especially light of the new `cfg` option, defined above.

```
\@for\eqe@tmp@i:={},i,ii,iii,iv,v,vi\do{%
    \edef\eqe@tmp@exp{%
    \noexpand\DeclareOptionX{myconfig\eqe@tmp@i}%
        {\noexpand\AtEndOfPackage{\expandafter\noexpand
            \csname eqemyconfig\eqe@tmp@i\endcsname}}%
    }\eqe@tmp@exp
}
```

Point options. Options relating to points, points on left, right, both, no points, totals on left and right. We offer options for points and totals.

pointsonleft
pointsonright
pointsonboth
nopoints
totalsonleft
totalsonright
nozerototals

```
\DeclareOptionX{pointsonleft}{\AtEndOfPackage{\PointsOnLeft}}
\DeclareOptionX{pointsonright}{\AtEndOfPackage{\PointsOnRight}}
\DeclareOptionX{pointsonboth}{\AtEndOfPackage{\PointsOnBothSides}}
\DeclareOptionX{nopoints}{\AtEndOfPackage{\NoPoints}}
\DeclareOptionX{totalsonleft}{\AtEndOfPackage{\TotalsOnLeft}}
\DeclareOptionX{totalsonright}{\AtEndOfPackage{\TotalsOnRight}}
\DeclareOptionX{nozerototals}{\AtEndOfPackage{\noZeroTotals}}
```

nototals
noparttotals
parttotalsonright
parttotalsonleft
noseparationrule
nosummarytotals

Totals options. Options relating to totals

```
\let\eqe@YES=y \let\eqe@NO=n
\let\eqe@One=1 \let\eqe@Zero=0
\let\eqe@Two=2 \let\eqe@Three=3 \let\eqe@Four=4
\def\eqe@h{h}
\DeclareOptionX{nototals}{\AtEndOfPackage{\NoTotals}}
\DeclareOptionX{noparttotals}{%
    \AtEndOfPackage{\let\eq@parttotals\eqe@NO}}
\DeclareOptionX{parttotalsonright}{%
    \def\eqeomarginbox{\eqeomarginboxright}}
\DeclareOptionX{parttotalsonleft}{%
    \def\eqeomarginbox{\eqeomarginboxleft}}
\def\eqeomarginbox{\eqeomarginboxright}
\DeclareOptionX{noseparationrule}{%
```

```
    \AtEndOfPackage{\separationruleOff}}
%    \AtEndOfPackage{\let\separationrule\relax}}
\DeclareOptionX{nosummarytotals}{\let\eq@nosummarytotals\eqe@YES}
```

coverpage

cover page options. There are two such options, coverpage and coverpagesumry. If this option is taken, a cover page is generate.

```
\DeclareOptionX{coverpage}{\def\eqex@coverpage{\eqexcoverpage}%
    \setcounter{page}{0}}
```

coverpagesumry

If this option is taken, an **Exam Record** is generated on the cover page, provided the coverpage option is taken. Possible values aer byparts, bypages, or none.

```
\define@choicekey+{eqexam.sty}{coverpagesumry}[\val\nr]%
    {byparts,bypages,none}{%
    \ifcase\nr\relax
        \def\sumryAnnots{\cpSumrybyparts}\or
        \def\sumryAnnots{\cpSumrybypages}\or
        \let\sumryAnnots\relax
    \fi
}{\PackageWarning{aeb}{Bad choice for coverpagesumry, permissible values
   are byparts, bypages, and none. Try again}}
\let\sumryAnnots\relax
```

nospacetowork

Options related to how the document is built. The vertical space defined by the solution environment is removed.

```
\DeclareOptionX{nospacetowork}{%
    \AtEndOfPackage{\let\eq@insertverticalspace\eqe@NO}}
```

answerkey

Equivalent to solutionsafter and proofing.

```
\@ifundefined{ifanswerkey}{\newif\ifanswerkey\answerkeyfalse}{}
\@ifundefined{ifsolutionsAtEnd}
    {\newif\ifsolutionsAtEnd\solutionsAtEndtrue}{}
```

A more intelligent processing of certain options is deployed: Now, we allow only one of the options answerkey, nosolutions, vspacewithsolns, solutionsafter, and solutionsonly at a time.

```
\let\thisOpt@OK \eqe@YES
\def\w@rningBadOpts{\PackageWarningNoLine{eqexam}{You've chosen two
    conflicting options,\MessageBreak \l@stPO\space and \CurrentOption.
    We will recognize\MessageBreak the \l@stPO\space option.
    Please correct\MessageBreak this if my guess is wrong}}
\def\eqe@optiont@kenMsg{\PackageInfo{eqexam}
  {Option \CurrentOption\space taken}}
\DeclareOptionX{answerkey}{\ifx\thisOpt@OK\eqe@YES
  \def\l@stPO{answerkey}\let\thisOpt@OK\eqe@NO
  \eqe@optiont@kenMsg\expandafter\AnswerKey\else
  \w@rningBadOpts\fi}
\def\AnswerKey{\solutionsAtEndfalse\answerkeytrue
  \eq@proofingtrue\eq@solutionsaftertrue\displayworkareafalse}
```

vspacewithsolns

ftbsolns

\ifkeyalt

When vspacewithsolns is used, vertical space is created by the solutions environment, and the solutions are written to the end of the file. Added ftbsolns as an alias for vspacewithsolns . This is implemented through a Boolean switch \ifvspacewithsolns, which I'm sorry now I've made so long. To make up for that bad decision, I also define \fkeyalt to be \ifvspacewithsolns with making conditional decisions, of course, one cannot say \keyalttrue.

```
\newif\ifvspacewithsolns\vspacewithsolnsfalse
\def\ifkeyalt{\csname ifvspacewithsolns\endcsname}
  \let\keyalttrue\vspacewithsolnstrue
  \let\keyaltfalse\vspacewithsolnsfalse
%\def\ifkeyalt{\csname ifvspacewithsolns\endcsname}
\def\ifkeyOrkeyalt{\ifanswerkey
    \def\eqe@next{\csname iftrue\endcsname}\else
    \ifvspacewithsolns\def\eqe@next{\csname iftrue\endcsname}\else
    \def\eqe@next{\csname iffalse\endcsname}\fi\fi\eqe@next}
\DeclareOptionX{vspacewithsolns}{\ifx\thisOpt@OK\eqe@YES
    \def\l@stPO{vspacewithsolns}\let\thisOpt@OK\eqe@NO
    \eqe@optiont@kenMsg
    \vspacewithsolnstrue\expandafter\displayworkareatrue
    \else\w@rningBadOpts\fi}
\DeclareOptionX{ftbsolns}{\ExecuteOptionsX{vspacewithsolns}}
```

flextended

The flextended is an (experimental) option to allow filler lines (fl) to be superimposed under the solutions, when the answerkey option is taken.

```
\DeclareOptionX{flextended}{\AtEndOfPackage{\flextendedInput}}
\def\flextendedInput{\eqe@flextendedtrue
    \InputIfFileExists{flextended.def}
        {\PackageInfo{eqexam}{Inputting flextended.def for
            flextended\MessageBreak option}}
        {\eqe@flextendedfalse\PackageWarning{eqexam}
    {Cannot find flextended.def for flextended\MessageBreak
    option. You should rebuild the package. Removing\MessageBreak
    flextended for now}}}
\@ifundefined{ifeqe@flextended}
  {\newif\ifeqe@flextended \eqe@flextendedfalse}{}
```

useforms

Use forms (if online option is taken); otherwise draw rectangles for multiple choice/multiple selection questions.

```
\DeclareOptionX{useforms}{\AtEndOfPackage{\def\sqstar{*}}}
```

allowcirc4mc

This option uses lcircle10 to draw circles around multiple choice questions.

```
\newif\ifallowcircmc \allowcircmcfalse
\DeclareOptionX{allowcirc4mc}{\allowcircmctrue}
```

online

pdf

links

email

PDF Options The various options to go beyond paper!

Options related to the interactive capability of eqexam. For each of the PDF options, we auto-check for pdftex and xetex.

```
\def\eqe@auto@chk@drivers{\ifpdf\ExecuteOptionsX{pdftex}\else
```

```
    \ifxetex\ExecuteOptionsX{xetex}\fi\fi}
\newif\ifeqeonline \eqeonlinefalse
\DeclareOptionX{online}{\eqe@auto@chk@drivers
    \let\eq@online\eqe@YES\eqeonlinetrue\ExecuteOptionsX{links}}
\DeclareOptionX{pdf}{\eqe@auto@chk@drivers\let\load@web\eqe@YES}
\DeclareOptionX{links}{\eqe@auto@chk@drivers
    \let\load@web\eqe@YES\let\load@exerquiz\eqe@YES}
\DeclareOptionX{email}{\eqe@auto@chk@drivers
    \let\use@email\eqe@YES\ExecuteOptionsX{online}}
```

obeylocalversions

This option is used for multiple versions of a document.

```
\newif\ifeqobeylocalversion \eqobeylocalversionfalse
\DeclareOptionX{obeylocalversions}{\eqobeylocalversiontrue}
```

usexkv

Causes the xkeyval package to be input, this option extends the option list of `\fillIn`.

```
\DeclareOptionX{usexkv}{\let\eq@usexkeys\eqe@YES}
```

2014/12/19 Now, the `usexkeys` is on by default.

```
\let\eq@usexkeys\eqe@YES
```

max
rendition

Renditions Options relating to renditions. The `max` and `rendition` option can be used instead of the `\numVersions{2}` and `\forVersion{a}`, respectively. These options allow you to set the version information though a package option. This allows us, for example, to use a cfg file such as `rendition.cfg` to dynamically set the version. This feature is used primarily by AeB Exam Builder.

```
\let\eq@renditionOptions\@empty
\let\eq@max@selected\@empty \let\eq@ren@selected\@empty
\DeclareOptionX{max}{\def\eq@max@selected{#1}%
    \g@addto@macro\eq@renditionOptions{\numVersions{#1}}}
\DeclareOptionX{rendition}{\def\eq@ren@selected{#1}%
    \g@addto@macro\eq@renditionOptions{\forVersion{#1}}}
```

allowrandomize

Randomization Options relating to randomization. Use this option to randomize the choices of a multiple choice question.

```
\DeclareOptionX{allowrandomize}{\AtEndOfPackage{\inputRandomizeChoices}}
\def\inputRandomizeChoices{\InputIfFileExists{aebrandom.def}
    {\typeout{inputting aebrandom.def}}{cannot find aebrandom.def}}
```

Set Misc. Defaults and Helper Macros. We set some defaults, and define macros for use by the document author.

```
\let\eq@online\eqe@NO
\def\ifisOnline{\ifx\eq@online\eqe@YES}
\let\use@email\eqe@NO
\let\load@web\eqe@NO
\let\load@exerquiz\eqe@NO
\def\sqLinks{\def\sqstar{}}\sqLinks
\def\sqForms{\def\sqstar{*}}
```

```
\def\NoSpaceToWork{\let\eq@insertverticalspace\eqe@NO}
\def\SpaceToWork{\let\eq@insertverticalspace\eqe@YES}
\let\eq@nototals\eqe@NO
\let\eq@nosummarytotals\eqe@NO
\let\eq@parttotals\eqe@YES
\let\eqx@separationrule\eqe@YES
\let\eq@insertverticalspace\eqe@YES
\let\eqex@coverpage\relax
\def\@reportpoints{0}
\let\marginpoints\@empty
\let\totalsbox=\hfil
```

2.3 Options from and for web

forpaper
forcolorpaper
forcolorpaper*

Here is the list of options of exerquiz we plan to recognize.

```
\@ifundefined{ifequsecolor}{\newif\ifequsecolor\equsecolorfalse}{}
\DeclareOptionX{forpaper}{\equsecolorfalse\eqforpapertrue
    \PassOptionsToPackage{monochrome}{\eq@ColorPackage}}
\DeclareOptionX{forcolorpaper}{\equsecolortrue\eqforpapertrue}
\newif\ifForceNoColor\ForceNoColorfalse
\DeclareOptionX{forcolorpaper*}{\equsecolortrue\eqforpapertrue
    \ForceNoColortrue} % for print
```

preview

Preview shows outlines for form fields.

```
\DeclareOptionX{preview}{\previewtrue}
```

webOpts

Pass options to web, when pdf or higher is used.

```
\define@key{eqexam.sty}{webOpts}[]{\def\eqe@webOpts{#1,\eqDriverName}}
\def\eqe@webOpts{\eqDriverName}
```

2.4 Options from and for exerquiz

exerquizOpts

Options from exerquiz that are useful for this package. Pass options to exerquiz, when links or higher is used.

```
\define@key{eqexam.sty}{exerquizOpts}[]{\def\eqe@eqOpts{#1,nodljs}}
\def\eqe@eqOpts{nodljs}
```

nosolutions
nohiddensolutions
noHiddensolutions
solutionsafter
solutionsonly

Solutions related options

```
\DeclareOptionX{nosolutions}{\ifx\thisOpt@OK\eqe@YES
    \def\l@stPO{nosolutions}\let\thisOpt@OK\eqe@NO
    \eqe@optiont@kenMsg\eq@nolinktrue\eq@nosolutionstrue
    \expandafter\displayworkareatrue
    \else\w@rningBadOpts\fi}
\DeclareOptionX{nohiddensolutions}{\eq@globalshowsolutionstrue}
\DeclareOptionX{noHiddensolutions}%
    {\eq@globalshowsolutionstrue\AtBeginDocument{\def\Hidesymbol{h}}}
\DeclareOptionX{solutionsafter}{\ifx\thisOpt@OK\eqe@YES
    \def\l@stPO{solutionsafter}\let\thisOpt@OK\eqe@NO
    \eqe@optiont@kenMsg
    \eq@solutionsaftertrue\expandafter
    \displayworkareafalse\else\w@rningBadOpts\fi}
\DeclareOptionX{solutionsonly}{\ifx\thisOpt@OK\eqe@YES
    \def\l@stPO{solutionsafter}\let\thisOpt@OK\eqe@NO
    \eqe@optiont@kenMsg\AtEndOfPackage{\therearesolutionstrue
        \let\exerSolnsHeadnToc\relax}
        \solutionsonlytrue\answerkeytrue\expandafter
        \displayworkareafalse\else\w@rningBadOpts\fi}
```

proofing

The proofing option sets a switch that controls whether the checkbox for multiple choice questions appears, and whether the answer for the \fillin command appears. \ifeq@proofing is set to true when the answerkey option is taken.

```
\DeclareOptionX{proofing}{\eq@proofingtrue}
```

\showproofing
\hideproofing

We provide two helper commands for turning on or off proofing. These are \showproofing for turning on proofing and \hideproofing for turning off proofing. There was some reason for defining these two, but can't remember now.

```
\newcommand{\showproofing}{\eq@proofingtrue}
\newcommand{\hideproofing}{\eq@proofingfalse}
```

showgrayletters

When this option is in effect, capital letters in gray appear under the multiple choice question boxes.

```
\newif\ifaebshowgrayletters\aebshowgraylettersfalse
\DeclareOptionX{showgrayletters}%
    {\AtEndOfPackage{\aebshowgrayletterstrue}}
```

\ifdisplayworkareaOn
\ifdisplayworkareaOff

Define \ifdisplayworkareaand user commands to turn off and on.

```
\@ifundefined{ifdisplayworkarea}{\newif\ifdisplayworkarea
  \displayworkareafalse}{}
\providecommand\displayworkareaOn{\displayworkareatrue}
\providecommand\displayworkareaOff{\displayworkareafalse}
```

Color packages We set the color package, xcolor preferred.

```
\IfFileExists{xcolor.sty}%
{\def\eq@ColorPackage{xcolor}\def\eqe@color@opt{table}%
    \PassOptionsToPackage{xcolor}{table}}
{\def\eq@ColorPackage{color}\let\eqe@color@opt\@empty}
```

noxcolor

noxcolor forces the use of the color package.

```
\DeclareOptionX{noxcolor}{\def\eq@ColorPackage{color}}
```

2.4.1 Title page options

useclassmaketitle

Tosupport the use of eqexam outside a straight exam document, we allow the user to bypass the redefinition of \maketitle. This is automatic with the fortextbook option.

```
\newif\if@eqeuseclassmaketitle \@eqeuseclassmaketitlefalse
\DeclareOptionX{useclassmaketitle}{\@eqeuseclassmaketitletrue}
\newif\if@bypasseqexamheading \@bypasseqexamheadingfalse
\DeclareOptionX{bypasspkgpagestyle}{\@bypasseqexamheadingtrue
    \ExecuteOptionsX{nototals}}
```

2.5 Drivers Recognized

These drivers are only relevant when a PDF option is taken (pdf, links, online, email). For ordinary paper documents, it is not necessary to specify the driver. If you put the assignment/homework/test (solns) on the web, suggested option is pdf, this inputs hyperref, and the document info dialog is filled in.

dvipsone
dvips
pdftex
dvipdfm
dvipdfmx
xetex
textures

```
\DeclareOptionX{dvipsone}{%
    \def\eqe@drivernum{0}\def\eqDriverName{dvipsone}%
    \PassOptionsToPackage{\eq@ColorPackage}{dvipsone}}
\DeclareOptionX{dvips}{\def\eqe@drivernum{0}\def\eqDriverName{dvips}%
    \PassOptionsToPackage{\eq@ColorPackage}{dvips}}
\DeclareOptionX{pdftex}{\def\eqe@drivernum{1}\def\eqDriverName{pdftex}%
    \PassOptionsToPackage{\eq@ColorPackage}{pdftex}}
\DeclareOptionX{dvipdfm}{%
    \def\eqe@drivernum{2}\def\eqDriverName{dvipdfm}%
    \PassOptionsToPackage{\eq@ColorPackage}{dvipdfm}}
\DeclareOptionX{dvipdfmx}{\def\eqe@drivernum{2}%
    \def\eqDriverName{dvipdfmx}%
    \PassOptionsToPackage{\eq@ColorPackage}{dvipdfmx}}
\DeclareOptionX{xetex}{\def\eqe@drivernum{2}\def\eqDriverName{xetex}%
    \PassOptionsToPackage{\eq@ColorPackage}{xetex}}
\DeclareOptionX{textures}{%
    \def\eqe@drivernum{3}\def\eqDriverName{textures}%
    \PassOptionsToPackage{\eq@ColorPackage}{textures}}
\DeclareOptionX*{%
    \PassOptionsToPackage{\CurrentOption}{\eq@ColorPackage}}
\def\eqe@drivernum{5}
\let\eqDriverName\@empty
```

If exerquiz is not loaded, when we need to define some of the switches that were defined in exerquiz.

The following switches are used in the options above, and are also defined in web, exerquiz, or eforms.

```
\newif\ifeq@solutionsafter \eq@solutionsafterfalse
\def\ifsolutionsafter{\csname ifeq@solutionsafter\endcsname}
 \let\solutionsaftertrue\eq@solutionsaftertrue
 \let\solutionsafterfalse\eq@solutionsafterfalse
\newif\ifsolutionsonly\solutionsonlyfalse
\newif\ifeq@hidesolution \eq@hidesolutionfalse
\newif\ifeq@globalshowsolutions \eq@globalshowsolutionsfalse
\newif\ifeq@nosolutions \eq@nosolutionsfalse
\newif\ifeq@proofing \eq@proofingfalse
\newif\ifeq@nolink \eq@nolinkfalse
\@ifundefined{ifpreview}{\newif\ifpreview \previewfalse}{}
\newif\ifeqforpaper \eqforpaperfalse
```

We define the commands for inputting the CFG files.

```
\def\eqe@csarg#1#2{\expandafter#1\csname#2\endcsname}
\@for\eqe@tmp@i:={},i,ii,iii,iv,v,vi\do{\eqe@csarg
    \edef{eqemyconfig\eqe@tmp@i}{\noexpand
    \InputIfFileExists{eqexam\eqe@tmp@i.cfg}{}{}}%
    \eqe@tmp@exp
}
```

2.6 Bring in Config Files

First read `web.cfg`, to possibly get the driver, then input `eqecus.opt`, which is used to create convenient custom options.

Here is an example of usage for defining your own custom options, must be based on current options, this code would be in the file `eqecus.opt`.

```
\DeclareOptionX{atbdbopts}
{%
    \ExecuteOptionsX{online}
    \ExecuteOptionsX{forcolorpaper}
    \ExecuteOptionsX{nosolutions}
    \ExecuteOptionsX{nopoints}
    \ExecuteOptionsX{nototals}
    \ExecuteOptionsX{nospacetowork}
    \ExecuteOptionsX{obeylocalversions}
    \ExecuteOptionsX{myconfig}
}
```

The following config files are input prior to `\ProcessOptionsX`, and can, therefore, contain declaration of options. `web.cfg` usually only specifies the default driver. `eqecus.opt` is used by @EASE, but can be used locally.

```
\let\bWebCustomize\endinput
\let\eWebCustomize\relax
\InputIfFileExists{web.cfg}{}{}
\InputIfFileExists{eqecus.opt}{}{}
```

These two are used by the rendition package and the exam builder utility.

```
\InputIfFileExists{rendition.cfg}{}{}
\InputIfFileExists{exambuilder.cfg}{}{}
```

2.7 Process Options

Now process the options.

```
\ProcessOptionsX
```

When the online or email option is taken, as well as answserkey, we cancel the `flextended` option.

```
\ifeqeonline
    \ifanswerkey
    \def\flextendedInput{\let\turnfl@nskeyMsg\@empty
    \PackageWarningNoLine{eqexam}
    {You’ve chosen the online or email option\MessageBreak
    with the flextended option. This is not\MessageBreak
    supported, removing the flextended option}}\fi
\fi
```

The \selectedMC command contains the choice for the styling for the region multiple choice questions: rectangles or circles. The default is rectangles.

```
\AtEndOfPackage{\selectedMC}
```

(2015/07/12) Move the inclusion of the color package prior to loading many of the other required packages.

```
\edef\eqe@tmpexp{\noexpand
\RequirePackage{\eq@ColorPackage}}\eqe@tmpexp
```

We require a minimal version for xcolor.

```
\@ifpackageloaded{xcolor}{\AtEndOfPackage{\let\CT@cell@color\relax
    \let\CT@arc@\relax}%
    \@ifpackagelater{xcolor}{2004/07/04}{}{\PackageError{eqexam}{%
    ************************************************\MessageBreak
    * Your Version of ‘xcolor.sty’ is too old!\MessageBreak
    * You need the version from 2004/07/04 or newer\MessageBreak
    * or use: \string\usepackage[noxcolor]{eqexam}\MessageBreak
    * or \string\documentclass[noxcolor]{article}\MessageBreak
    ************************************************}{}%
    }%
}{}
```

If nocustomdesign option is taken, we set the switch \eqcustomdesignfalse.

```
\if\eqe@nocustomdesign1\eqcustomdesignfalse\fi
```

Define a \immediate\write helper macro.

```
\long\def\eqe@IWO#1{\immediate\write#1}
```

Early definitions for the fortextbook option.

\showAllAnsAtEnd If the user has chosen the vspacewithsolns option, we must turn of all other solution options, namely answerkey. This command is used internally.

(2016/10/02) Include \eqTopOfSolnPage from exerquiz to support copying question to solution page.

```
\newcommand{\showAllAnsAtEnd}{%
    \makeAnsEnvForSolnsAtEnd\eqTopOfSolnPage
    \answerkeytrue\eq@proofingtrue
    \eq@solutionsaftertrue\vspacewithsolnstrue
    \displayworkareafalse
}
```

\makeAnsEnvForSolnsAtEnd One user wanted to be able to use the answers environment in the solutions section at the end of the document (when the vspacewithsolns is used). Here it is. This definition is added to the definition of \showAllAnsAtEnd.

```
\newcommand{\makeAnsEnvForSolnsAtEnd}{%
```

```
%  \proofingsymbol{\ding{52}}%
  \let\answers\answers@sq
  \let\endanswers\endanswers@sq
  \let\manswers\manswers@sq
  \let\endmanswers\endmanswers@sq
}
```

```
\newcommand{\solAtEndFormatting}[1]{\def\eqeAEFormatting{#1}}
\let\eqeAEFormatting\@empty
```

\writeAllAnsAtEnd writes the \showAllAnsAtEnd command to the solutions file.

```
\def\writeAllAnsAtEnd{\ifsolutionsonly\else
    \let\quiz@solns\ex@solns
    \eqe@IWO\quiz@solns{\string\showAllAnsAtEnd}%
    \ifx\eqeAEFormatting\@empty\else
        \eqe@IWO\quiz@solns{\string\eqeAEFormatting}\fi
    \fi
}
\def\writeWithSolDocTrue{\writeT@SolnFile{\protect\withinsoldoctrue}}
```

\setSolnMargins Sets the value of \eqemargin in the context of the solution file, this command is redefined later.

```
\newcommand{\setSolnMargins}[1]{\setlength\eqemargin{#1}\ignorespaces}
```

(2011/05/08) In the new version of eqexam, the one that makes the problems within an exam environment, into a list, the solutions file that appears at the end of the document also needs to be put into a list. Here, we define the command that writes the beginning of the eqequestions environment to the beginning of the \jobname.sol file. We hard-wire write \setSolnMargins{\the\eqemargin} to the solution file, but leave a back door open to write an alternate string.

\altSetSolnMargins (2014/03/20) \altSetSolnMargins allows you, through its argument to pass an alternate string.

```
\def\altSetSolnMargins#1{\def\alt@SetSolnMargins{#1}}
\let\alt@SetSolnMargins\@empty
\def\writeBeginEqeQuestions{\ifsolutionsonly\else
  \ifOKToWriteExamData
    \let\quiz@solns\ex@solns
    \ifx\alt@SetSolnMargins\@empty
      \writeT@SolnFile{\string\setSolnMargins{\the\eqemargin}}\else
      \writeT@SolnFile{\alt@SetSolnMargins}\fi
    \writeT@SolnFile{\protect\eqgriii\string\noindent
      \string\begin{eqequestions}}%
    \fi
  \fi
}
```

(2011/05/08) We define the command that writes the end of the eqequestions environment to the beginning of the \jobname.sol file.

```
\newcommand{\setBtwnExamSkip}[1]{\gdef\btwnExamSkipAmt{#1}%
    \def\btwnExamSkip{\ifdim#1=0pt\else
    \vskip#1\relax\fi}}
\setBtwnExamSkip{6pt}
\def\writeEndEqeQuestions{%
    \ifsolutionsonly\else
    \ifOKToWriteExamData
    \let\quiz@solns\ex@solns
    \eqe@IWO\quiz@solns{\string\eqgrii
      \string\end{eqequestions}^^J}%
    \writeT@SolnFile{\string\btwnExamSkip^^J}%
    \fi\fi
}
```

If `\ifvspacewithsolns` we set the switches need to simulate `nosolutions`.

```
\def\csarg#1#2{\expandafter#1\csname#2\endcsname}
\def\saveIFEQE#1{\def\ARG{#1ifSave}%
  \expandafter\csarg\expandafter
  \let\expandafter\ARG\csname#1\if#1true\else false\fi\endcsname}
\saveIFEQE{vspacewithsolns}\saveIFEQE{answerkey}
\saveIFEQE{eq@proofing}\saveIFEQE{eq@solutionsafter}
\saveIFEQE{eq@nolink}\saveIFEQE{eq@nosolutions}
\saveIFEQE{displayworkarea}
\def\vpwsSimulateNoSolns{% dps28
  \ifvspacewithsolns
    \answerkeyfalse\eq@proofingfalse\eq@solutionsafterfalse
    \eq@nolinkfalse\eq@nosolutionsfalse\displayworkareatrue
  \fi}
\def\reverseVSWS{\vspacewithsolnsfalse
  \answerkeyifSave\eq@proofingifSave
  \eq@solutionsafterifSave\eq@nolinkifSave
  \eq@nosolutionsifSave\displayworkareaifSave
}
\vpwsSimulateNoSolns
```

2.8 Save Switch Values

Now, save the current state of the switches defined above. When, and if, the packages web, exerquiz and eforms are loaded, they will overwrite the choices set by the author, so we save them.

```
\let\savedeq@online\eq@online
\let\savedifeq@solutionsafter\ifeq@solutionsafter
\let\savedifeq@hidesolution\ifeq@hidesolution
\let\savedifeq@globalshowsolutions\ifeq@globalshowsolutions
\let\savedifeq@nosolutions\ifeq@nosolutions
\let\savedifeq@proofing\ifeq@proofing
\let\savedifeq@nolink\ifeq@nolink
\let\savedifpreview\ifpreview
\let\savedifeqforpaper\ifeqforpaper
\let\ifnosolutions\ifeq@nosolutions
```

3 Required Packages

The following are the required packages for eqexam.

```
\RequirePackage{amstext,amssymb}
\@ifundefined{if@fleqn}{\let\fleqnOn\relax\let\fleqnOff\relax}
    {\def\fleqnOn{\@fleqntrue}\def\fleqnOff{\@fleqnfalse}}
```

Bring the comment package in early, before verbatim, these two clash a bit.

```
\RequirePackage{aeb-comment}
```

If \BeforeIncludedComment is defined, the old version of aeb-comment is used; otherwise, the new version of aeb-comment is being used. The new version supports utf-8.

```
\@ifundefined{BeforeIncludedComment}{\let\aebc@end\endgroup}
  {\let\aebc@end\relax}
\def\eqe@commentChkMsg{\@ifpackageloaded{comment}
  {\PackageWarningNoLine{eqexam}
    {The comment package is incompatible with the\MessageBreak
     aeb-comment package, do not use the comment package}}{}}
\AtBeginDocument{\eqe@commentChkMsg}
```

The macro \includeexersolutions is defined in eqexam.def. We execute the command \include@solutions before the web package is loaded. The web package has a \AtEndDocument as well, and inserts a new page that we don't want.

```
\AtEndDocument{\includeexersolutions}
```

If user has specified one of the pdf options (pdf, links, online, email), we bring in the web package.

```
\@ifpackageloaded{web}{\let\load@web\eqe@YES}{%
  \ifx\load@web\eqe@YES\ifnum\eqe@drivernum=5
    \PackageInfo{eqexam}{You have not selected a driver %
      for eqexam. Perhaps the \MessageBreak
      driver is introduced through web.cfg}\fi
    \expandafter\RequirePackage\expandafter[\eqe@webOpts]{web}%
    \edef\@pdfcreator{\@pdfcreator, The eqexam Package}\fi
}
```

If user has specified links, online or email, we bring in the exerquiz package.

```
\@ifpackageloaded{exerquiz}{\let\load@exerquiz\eqe@YES}{%
  \let\symbolchoice\@gobble
  \ifx\load@exerquiz\eqe@YES\expandafter
    \RequirePackage\expandafter[\eqe@eqOpts]{exerquiz}[2011/08/30]
```

We input exerquiz with the nodljs, we don't need all the JavaScript to process interactive shortquizzes or quizzes, but we do want the option of adding in document JavaScript, so after we input exerquiz, we set the switches to allow these features.

```
    \let\importdljs\eqe@YES\let\execjs\eqe@YES
  \fi
}
\@ifpackageloaded{eforms}{}{%
```

If eforms is not loaded, we define \symbolchoice, which is defined in eforms to gobble up its argument, we don't need it.

```
    \let\symbolchoice\@gobble
}
```

Here is a fix to a problem I've been having previewing in dviwindo. I've traced the problem down to \@pdfviewparams. Redefining \@pdfviewparams as follows.

```
\def\eqDvipsone{dvipsone}
\@ifpackageloaded{hyperref}
  {\ifx\eqDriverName\eqDvipsone
      \renewcommand\@pdfviewparams{ null null null}\fi
  }{\let\textorpdfstring\@firstoftwo}
```

Now that we have possibly input web or exerquiz, we need to restore the authors options.

```
\let\eq@online\savedeq@online
\let\ifeq@solutionsafter\savedifeq@solutionsafter
\def\ifsolutionsafter{\ifeq@solutionsafter} % user interface
\let\ifeq@hidesolution\savedifeq@hidesolution
\let\ifeq@globalshowsolutions\savedifeq@globalshowsolutions
\let\ifeq@nosolutions\savedifeq@nosolutions
\let\ifeq@proofing\savedifeq@proofing
\let\ifeq@nolink\savedifeq@nolink
\let\ifpreview\savedifpreview
\let\ifeqforpaper\savedifeqforpaper
```

Other packages of interest.

```
\RequirePackage{calc}
\RequirePackage{pifont}
\RequirePackage{array}
```

Here, I input the verbatim package after the comment package.

```
\RequirePackage{verbatim}
\@ifundefined{dlcomment}{\typeout{!! defining dlcomment}%
  \let\dlcomment\comment
  \let\enddlcomment\endcomment
  \let\eqSavedComment\dlcomment
  \let\endeqSavedComment\enddlcomment
}{}
```

When constructing paper tests, I often use a multi-column format for some of the questions, so let's require this package

```
\RequirePackage{multicol}
\setlength\columnseprule{.4pt}
\raggedcolumns\multicolsep=3pt
\newcommand{\setmulticolprob}{%
    \setlength{\linewidth}{\linewidth+\eqemargin}}
```

For the fortextbook option, we require eso-pic.

```
\edef\eqe@reqPack{\ifeqfortextbook\noexpand\RequirePackage{eso-pic}\else
\relax\fi}
\eqe@reqPack
```

We've processed the options, I need to detect whether the document author has not chosen any solutions options.

```
\ifvspacewithsolns\solutionsAtEndtrue\fi
\ifanswerkey\solutionsAtEndfalse\fi
\ifeq@nosolutions\solutionsAtEndfalse\fi
\ifeq@solutionsafter\solutionsAtEndfalse\fi
```

4 Page Layout

(2011/05/08) The revised version of eqexam allows the document author to more easily design the size of the page; the new version makes all content inside the exam environment into a list, this gives us better control over the margins and spacing.

\eqexammargin

(2011/05/08) Use this command to set the margin for the exam environment.

```
\@ifundefined{eqemargin}{\newlength{\eqemargin}}{}
\providecommand{\prbDecPt}[1]{\def\eqe@decPointPrb{#1}}
\def\eqe@decPointPrb{.}\def\eqe@dpsepPrb{\ }
\providecommand{\prbPrtsep}[1]{\def\eqe@prtsepPrb{#1}}
\def\eqe@prtsepPrb{\ }
\providecommand{\prbNumPrtsep}[1]{\def\eqe@hspannerPrb{#1}}
\def\eqe@hspannerPrb{\ }
\newcommand{\eqexammargin}[2][\normalsize\normalfont\bfseries]{%
    \settowidth{\eqemargin}{#1#2\eqe@decPointPrb\eqe@hspannerPrb}}
```

(2011/05/08) The default margin for the eqexam environment, two digits and a space.

```
\eqexammargin{00}
```

```
\def\eqe@hspannerSoln{\ }   % space after prob number
\providecommand{\solNumPrtsep}[1]{\def\eqe@hspannerSoln{#1}}
```

\eqeSetExamPageParams

(2011/05/08) The default spacing maximizes the amount of space on the page.

```
\newcommand{\eqeSetExamPageParams}{%
    \setlength{\headheight}{12pt}
    \setlength{\topmargin}{-.5in}
    \setlength{\headsep}{20pt}
    \setlength{\oddsidemargin}{0pt}
    \setlength{\evensidemargin}{0pt}
    \setlength{\marginparsep}{11pt}
    \setlength{\marginparwidth}{35pt}
    \setlength{\footskip}{11pt}
}
```

\eqExamPageLayout

Set the basic parameters of this exam page package

```
\newcommand{\eqExamPageLayout}{%
    \setlength\textwidth\paperwidth
    \addtolength{\textwidth}{-2in}
    \addtolength{\textwidth}{-\oddsidemargin}
    \setlength\textheight{\paperheight}
    \addtolength\textheight{-2in}
    \addtolength\textheight{-\headheight}
    \addtolength\textheight{-\headsep}
    \addtolength\textheight{-\topmargin}
    \addtolength\textheight{-\footskip}
}
```

(2011/05/08) If `usecustomdesign` is used it is expected that `\eqeSetExamPageParams` and `\eqeSetExamPageParams` are redefined in he preamble, otherwise, we set up the standard parameters; otherwise

```
\ifeqcustomdesign\else
\eqeSetExamPageParams
\eqExamPageLayout
\fi
```

A simple page layout scheme for this exam.

```
\newcommand{\ps@eqExamheadings}
{%
    \renewcommand{\@oddhead}{%
    {\normalfont\normalsize\ifnum\value{page}<2
        \hfil\else\eqExamRunHead\fi}}%
    \renewcommand{\@evenhead}{\@oddhead}
    \renewcommand{\@oddfoot}{\settotalsbox\runExamFooter}
    \renewcommand{\@evenfoot}{\@oddfoot}
}
\raggedbottom
```

5 Counters, Lengths and Tokens

`eqpointsofar`
`eqpointsthispage`
`eq@numparts`

Some counters to keep track of things. The first two counters keep track, respectively, of the total points so far up the current page, and the number of points on the current page. The counter `eq@numparts` holds the number of parts of the multi-part question.

```
\newcount\eqe@tempcnta
\newcounter{eqpointsofar}
\newcounter{eqpointsthispage}
\newcounter{eq@numparts}
\newcounter{eq@count}
\newtoks\partNames \partNames={}
\newlength{\eq@tmplengthA}
\newlength{\eq@tmplengthB}
\newbox{\eq@pointbox}
\newlength{\eq@pointboxtotalheight}
```

Some scratch registers to do calc calculations.

```
\newlength{\eqetmplengtha}
\newlength{\eqetmplengthb}
```

6 Some Macros to Support the Options

We make a few definitions to support various options.

```
\def\PointsOnLeft{\def\@reportpoints{1}\let\marginpoints\eqleftmargin}
\def\PointsOnRight{\def\@reportpoints{2}\relax
    \let\marginpoints\eqrightmarginbox}
\def\PointsOnBothSides{\def\@reportpoints{3}\relax
    \let\marginpoints\eqbothmargins}
\newif\ifeqe@nopoints \eqe@nopointsfalse
\def\NoPoints{\if\isInExamEnv\eqe@NO
    \eqe@nopointstrue\def\@reporttotals{0}\let\totalsbox=\hfil
    \let\marginpoints\@empty\let\eq@nosummarytotals\eqe@YES\else
    \PackageWarning{eqexam}{The \string\NoPoints\space ignored;
    it needs to be executed\MessageBreak outside of an
    exam environment}\fi}
\def\TotalsOnLeft{\def\@reporttotals{1}\def\totalsbox{\totalsboxleft}}
\def\TotalsOnRight{\def\@reporttotals{2}\def\totalsbox{\totalsboxright}}
\def\noZeroTotals{\let\eqe@zeroTotalsAllowed\eqe@NO}
\def\allowZeroTotals{\let\eqe@zeroTotalsAllowed\eqe@YES}
\allowZeroTotals
\def\NoTotals{\def\@reporttotals{0}\let\totalsbox=\hfil}
\def\SummaryTotalsOn{\let\eq@nosummarytotals\eqe@NO}
\def\SummaryTotalsOff{\let\eq@nosummarytotals\eqe@YES}
\def\eoeTotalOff{\let\eq@parttotals\eqe@NO}
\def\eoeTotalOn{\let\eq@parttotals\eqe@YES}
\def\separationruleOn{\let\eqx@separationrule\eqe@YES}
\def\separationruleOff{\let\eqx@separationrule\eqe@NO}
\def\AllowFitItIn{\global\let\eq@fititin\eqfititin}
\def\DoNotFitItIn{\global\let\eq@fititin\@gobble}
```

```
\def\NoSolutions{\eq@nolinktrue\eq@nosolutionstrue
    \displayworkareatrue}
\@onlypreamble\NoSolutions
```

Added \chngToNoSolns, useful for book authors that want to switch between \AnswerKey, \chngToNoSolns, and \SolutionsAtEnd.

```
\def\chngToNoSolns{\solutionsAtEndfalse\answerkeyfalse
    \eq@proofingfalse\eq@solutionsafterfalse
    \eq@nosolutionstrue\displayworkareatrue}
```

\vspacewithkeyOn
\vspacewithkeyOff

User interface to keeping the declare vspace, even when the answerkey (or solutionsafter) option is taken. The switch \ifkeepdeclaredvspacing is defined in eqexam.def/exerquiz.

```
\def\vspacewithkeyOn{\keepdeclaredvspacingtrue}
\def\vspacewithkeyOff{\keepdeclaredvspacingfalse}
```

\displayPointsOn
\displayPointsOff

\displayPointsOn displays the points, if not otherwise overridden and the command \displayPointsOff turns off the display of points. In either case, points are calculated.

```
\newif\ifdispl@yPoints\displ@yPointstrue
\newcommand{\displayPointsOn}{\displ@yPointstrue}
\newcommand{\displayPointsOff}{\displ@yPointsfalse}
\newif\ifl@stDispl@yPoints\l@stDispl@yPointsfalse
```

\encloseProblemsWith

\encloseProblemsWith to support the solutionsonly option

```
\def\encloseProblemsWith#1{%
    \ifsolutionsonly\excludecomment{#1}\else
    \includecomment{#1}\fi
}
```

7 Colors

\proofingsymbolColor
\instructionsColor
\eqCommentsColor
\universityColor
\titleColor
\authorColor
\subjectColor
\linkcolor
\nolinkcolor
\fillinColor
\forceNoColor
\eqEndExamTotalColor

Here we list commands for controlling colors. There are some other colors defined in the stand alone code.

```
\providecommand{\proofingsymbolColor}[1]{\def\@proofingsymbolColor{#1}}
\proofingsymbolColor{red}
\providecommand{\instructionsColor}[1]{\def\@instructionsColor{#1}}
\instructionsColor{blue}
\providecommand{\eqCommentsColor}[1]{\def\@eqCommentsColor{#1}}
\eqCommentsColor{blue}
\providecommand{\eqCommentsColorBody}[1]{\def\@eqCommentsColorBody{#1}}
\eqCommentsColorBody{black}
\providecommand{\universityColor}[1]{\def\webuniversity@color{#1}}
\universityColor{blue}
\providecommand{\titleColor}[1]{\def\webtitle@color{#1}}
\titleColor{black}
\providecommand{\authorColor}[1]{\def\webauthor@color{#1}}
\authorColor{black}
\providecommand{\subjectColor}[1]{\def\websubject@color{#1}}
\subjectColor{blue}
\providecommand{\linkcolor}[1]{\def\@linkcolor{#1}}
\linkcolor{blue}
\providecommand{\nolinkcolor}[1]{\def\@nolinkcolor{#1}}
\nolinkcolor{black}
\providecommand{\eqEndExamTotalColor}[1]{\def\endexamtotal@color{#1}}
\eqEndExamTotalColor{black}
\newcommand\fillinColor[1]{\def\eq@fillinColor{#1}}\fillinColor{red}
\providecommand{\sectionColor}[1]{\def\web@sectionsColorOld{#1}
    \def\aeb@sectioncolor{#1}}
\sectionColor{blue}
\def\forceNoColorSet{\proofingsymbolColor{black}%
    \instructionsColor{black}%
    \eqCommentsColor{black}\universityColor{black}%
    \titleColor{black}\authorColor{black}%
    \subjectColor{black}\linkcolor{black}%
    \nolinkcolor{black}\fillinColor{black}%
    \instructionsColor{black}\eqCommentsColor{black}%
    \eqCommentsColorBody{black}%
    \eqEndExamTotalColor{black}\ckboxColor{}%
    \ckcirColor{}\if\load@web\eqe@YES
        \sectionColor{black}\fi
}
\ifForceNoColor
    \AtBeginDocument{\forceNoColorSet}
\fi
\newcommand{\forceNoColor}{\ifForceNoColor\else\ForceNoColortrue
    \AtBeginDocument{\forceNoColorSet}\fi}
```

\ckboxColor
\ckcirColor

The \ckboxColor provides color for the MC and MS boxes, then a PDF-related option is not in effect, while \ckcirColor provides color for circular checkboxes. Usually, these are black or both the same color.

```
\providecommand{\ckboxColor}[1]{\def\@rgi{#1}\ifx\@rgi\@empty
    \let\ckbox@Color\relax\else
    \def\ckbox@Color{\color{#1}}\fi}\ckboxColor{}
\providecommand{\ckcirColor}[1]{\def\eq@rgi{#1}\ifx\eq@rgi\@empty
    \let\ckcir@Color\relax\else
    \def\ckcir@Color{\color{#1}}\fi}\ckcirColor{}
```

8 Version Control

Here are some simple macros use to create two versions, version A and version B, of the same test.

\examNum

Convenience macro for holding the exam number. It sets the value of \nExam.

```
\def\examNum#1{\def\nExam{#1}}
\examNum{1}
```

\Exam
\sExam

Convenience macros for titling the exam. Usage:

```
\VersionAtext{Test~\nExam--Version A}
\VersionBtext{Test~\nExam--Version B}
\shortVersionAtext{T\nExam A}
\shortVersionBtext{T\nExam B}

\examNum{1}
\forVersion{c}
\subject[C3]{Calculus III}
\title[\sExam]{\Exam}
\author{Dr.\ D. P. Story}
```

These next two definitions are overwritten by the two commands \longTitleText and \shortTitleText.

```
\def\Exam{\ifAB{\eq@VersionAtext}{\eq@VersionBtext}}
\def\sExam{\ifAB{\eq@shortVersionAtext}{\eq@shortVersionBtext}}
```

\VersionAtext
\VersionBtext
\shortVersionAtext
\shortVersionBtext

Convenience macros for entering the text for the title, long and short for versions A and B.

```
\def\VersionAtext#1{\def\eq@VersionAtext{#1}}
\def\VersionBtext#1{\def\eq@VersionBtext{#1}}
\def\shortVersionAtext#1{\def\eq@shortVersionAtext{#1}}
\def\shortVersionBtext#1{\def\eq@shortVersionBtext{#1}}
\VersionAtext{Exam~\nExam--Version A}
\VersionBtext{Exam~\nExam--Version B}
\shortVersionAtext{Exam~\nExam A}
\shortVersionBtext{Exam~\nExam B}
```

In this section we introduce a new set of commands that supersedes the commands defined above. Those commands were limited to only two versions. The ones below can handle up to 26 versions.

```
\newtoks\eqtemptokena
\newtoks\eqtemptokenb
```

`\numVersions`

In the preamble, declare the number of versions for this document using `\numVersions`, e.g., `\numVersions{3}`. This sets the value of `\eq@nVersions`

```
\def\numVersions#1{\ifnum#1>26\def\eq@nVersions{26}%
    \PackageWarning{eqexam}{The value of \string\numVersions\space is
    too large. \MessageBreak Choose a natural number less than 27}
    \else\def\eq@nVersions{#1}\fi}
```

`\longTitleText`
`\endlongTitleText`
`\shortTitleText`
`\endshortTitleText`

Next we state the long and short titles for our document, one for each of our declare number of versions given earlier. For example, we can use the value `\nExam` in out titles. Usage:

```
\longTitleText
    {Test~\nExam--Version A}
    {Test~\nExam--Version B}
    {Test~\nExam--Make Up}
\endlongTitleText
\shortTitleText
    {T\nExam A}
    {T\nExam B}
    {T\nExam MU}
\endshortTitleText
```

I've added markers that delimit the end of the arguments. In this way, the end of the list of titles can be detected, even though the number of titles is not the same as what is declared by the `\numVersions`.

If there are more titles than what is declared, the rest are absorbed (gobbled). If there are fewer titles than declared, a LaTeX package error is generated, and substitute titles are generated. Modified `\longTitleText` and `\shortTitleText` to have an optional argument (A–Z;a–z). You can select a particular title from a list of titles. If no optional argument is passed, then the title determined by `\forVersion` is used.

```
\newcommand{\longTitleText}[1][]{%
    \ifeqglobalversion\let\eq@selectedVersion@save\eq@selectedVersion
    \else\let\eq@selectedVersion@save\relax\fi
    \uppercase{\def\eqe@localTextTitle{#1}}%
    \ifx\eqe@localTextTitle\@empty\else
    \expandafter\forVersion\expandafter{\eqe@localTextTitle}\fi
    \eqe@contTitleText{\Exam}{\endlongTitleText}%
}
\def\endlongTitleText{l}
\newcommand{\shortTitleText}[1][]{%
    \ifeqglobalversion\let\eq@selectedVersion@save\eq@selectedVersion
    \else\let\eq@selectedVersion@save\relax\fi
    \uppercase{\def\eqe@localTextTitle{#1}}%
    \ifx\eqe@localTextTitle\@empty\else
    \expandafter\forVersion\expandafter{\eqe@localTextTitle}\fi
    \eqe@contTitleText{\sExam}{\endshortTitleText}%
}
\def\endshortTitleText{s}
```

Both title commands, above, call this macro which sets the environment for `\@gatherTitleText`, which gathers the list of titles.

```
\def\eqe@contTitleText#1#2{%
    \setcounter{eq@count}{0}%
    \eqtemptokena={}\let\endtitleMarker#2
    \@gatherTitleText{#1}%
}
```

This command gathers each title and places it as the argument of a `\v<LETTTER>` command. These are accumulated in token registers then saved in `\Exam` and `\sExam`.

```
\def\@gatherTitleText#1#2{%
    \def\eqe@argii{#2}
    \if\endtitleMarker\eqe@argii
```

Encountered the end marker. See if we have collected the correct number of titles declared. If we have collected too few, we note an warning in the log, and create titles.

```
        \ifnum\value{eq@count}>\eq@nVersions\let\eqe@next\relax
        \else\def\eqe@next{\eq@shortTitlesFix{#1}}\fi
    \else
    \stepcounter{eq@count}
        \eqtemptokenb=\expandafter{#2}
        \xdef#1{\the\eqtemptokena\expandafter\noexpand
        \csname v\Alph{eq@count}\endcsname{\the\eqtemptokenb}}
        \xdef\sExam{\the\eqtemptokena\expandafter\noexpand
        \csname v\Alph{eq@count}\endcsname{\the\eqtemptokenb}}
        \eqtemptokena=\expandafter{#1}
        \ifnum\value{eq@count}<\eq@nVersions
            \def\eqe@next{\@gatherTitleText{#1}}%
        \else
            \def\eqe@next{%
                \if\endtitleMarker\endlongTitleText
                    \expandafter\eqe@absorbTokensLong
                \else
                    \expandafter\eqe@absorbTokensShort
                \fi
            }%
        \fi
    \fi
    \eqe@next
}
\long\def\eqe@absorbTokensLong#1\endlongTitleText{%
    \protected@xdef\Exam{\Exam}\ifx\eq@selectedVersion@save\relax
    \eqe@offVersion\else\expandafter\forVersion\expandafter
    {\eq@selectedVersion@save}\fi}
\long\def\eqe@absorbTokensShort#1\endshortTitleText{%
    \protected@xdef\sExam{\sExam}\ifx\eq@selectedVersion@save\relax
    \eqe@offVersion\else\expandafter\forVersion\expandafter
    {\eq@selectedVersion@save}\fi}
```

We have reached \endtitleMarker, but the count is still less than \eq@nVersions, so we'll warn the user, and create titles for user.

```
\def\eq@shortTitlesFix#1{%
    \PackageWarning{eqexam}{You have defined an insufficient number
    of titles\MessageBreak for the number of versions declared in
    \string\numVersions.\MessageBreak Please fix the problem}%
    \stepcounter{eq@count}%
    \if\endtitleMarker\endlongTitleText
        \edef\eqe@tmp{\noexpand\@gatherTitleText{\noexpand#1}
            {??---Title \# \the\value{eq@count}---??}%
            \noexpand\endlongTitleText}
    \else
        \edef\eqe@tmp{\noexpand\@gatherTitleText{\noexpand#1}
            {T\#\the\value{eq@count}??}\noexpand\endshortTitleText}
    \fi
    \addtocounter{eq@count}{-1}%
    \eqe@tmp
}
```

Here, we define \ifAB so that document under the old system still work properly, I hope. Usage of \ifAB at this point is discouraged.

```
\def\ifAB#1#2{\if\eq@selectedVersion A#1%
    \else\if\eq@selectedVersion B#2\fi\fi}
\def\eq@replaceToken#1{#1}
```

\forVersion Here is the command that does all the work. It creates alternate text macros for each of the versions declared using \numVersions.

For example, assuming \numVersions{3} appeared earlier, the command \forVersion{a} (or \forVersion{A}) defines 3 text commands \vA, \vB and \vC, each taking one argument, the text you want to display:

```
Name the \vA{place}\vB{date}\vC{year} of the signing of the Declaration
of independence.
```

Since we said `forVersion{a}` only the `\vA` text is displayed, the others are gobbled up, etc. But wait, the `\forVersion` does more than that! It also creates a series of comment environments `\begin{verA}`/`\end{verA}`, `\begin{verB}`/`\end{verB}`, `\begin{verC}`/`\end{verC}`, etc., where only the version for which this compile applies will be typeset, the others are commented out.

```
\numVersions{3}
\forVersion{b}
...
\begin{document}
...
 Solve the equation for $\vA{x}\vB{y}\vC{z}$:
\[
\begin{verA}
       2x + 4 = 7
\end{verA}
\begin{verB}
       5y + 2 = 4
\end{verB}
\begin{verC}
       3z - 2 = 2
\end{verC}
\]
```

```
\newif\ifeqglobalversion \eqglobalversionfalse
\newif\ifeqlocalversion \eqlocalversionfalse
\newif\if@templocalversion \@templocalversionfalse
\def\eqe@initializeMultiVersions{%
  \let\save@message\message\let\message\@gobble
  \@tfor\eqe@tmp:=ABCDEFGHIJKLMNOPQRSTUVWXYZ\do{%
    \csarg\let{v\eqe@tmp}\@gobble
    \edef\exp@temp{\noexpand\excludecomment{ver\eqe@tmp}}\exp@temp
    \csarg\let{Afterver\eqe@tmp Comment}\aebc@end
  }\let\message\save@message
}
\AtBeginDocument{\let\eqe@initializeMultiVersions\relax}
```

(09/10/04) Trying to fix a bug in the case when the version selected is greater then the number of versions available for a given problem; that is, when modular arithmetic occurs (in `\selectVersion`).

```
\let\eqe@@onVersion\@empty
\def\eqe@onVersion{\g@addto@macro\eqe@@onVersion}
\let\eqe@@offVersion\@empty
\def\eqe@offVersion{\g@addto@macro\eqe@@offVersion}
\let\eqe@@holdTemp\@empty
\def\eqe@holdTemp{\g@addto@macro\eqe@@holdTemp}
```

Two commands to turn on and off versions (the `\v<LETTER>` and the `ver<LETTER>` environment).

Throughout the definitions below, we use \csarg, a command that is defined in the comment package.

```
\def\eqe@showArg#1{#1}
\def\eqe@turnOnComment#1{%
%    \csarg\let{v#1}\@empty
    \csarg\let{v#1}\eqe@showArg
    \edef\exp@temp{\noexpand\includecomment{ver#1}}\exp@temp
}
\def\eqe@turnOffComment#1{%
    \csarg\let{v#1}\@gobble
    \edef\exp@temp{\noexpand\excludecomment{ver#1}}\exp@temp
    \csarg\let{Afterver#1Comment}\aebc@end
}
```

\selVersion

Finally, the \forVersion command. \selVersion holds the version selected in upper-case.

```
\def\forVersion#1%
{%
    \eqe@initializeMultiVersions
    \let\eqe@@onVersion\@empty
    \let\eqe@@offVersion\@empty
    \global\eqglobalversiontrue
    \setcounter{eq@count}{0}%
    \uppercase{\edef\eq@selectedVersion{#1}}%
    \edef\selVersion{\eq@selectedVersion}%
    \@ifundefined{eq@nVersions}{\PackageInfo{eqexam}{%
        \string\numVersions\space has not been declared, \MessageBreak
        taking the number of versions to be 2.}\def\eq@nVersions{2}}{}%
    \loop
        \stepcounter{eq@count}%
        \expandafter\if\Alph{eq@count}\eq@selectedVersion
            \xdef\eq@nSelectedVersion{\the\value{eq@count}}%
            \setcounter{eq@count}{27}\fi
        \ifnum\value{eq@count}<26\repeat
        \ifnum\eq@nSelectedVersion >\eq@nVersions
            \PackageError{eqexam}
            {The value of \string\forVersion
             \space(\eq@selectedVersion)\MessageBreak
             exceeds the value of \string\numVersions\space
             (\eq@nVersions)}%
            {Decrease the value of \string\forVersion.}%
        \fi
    \setcounter{eq@count}{0}%
    \let\save@message\message\let\message\@gobble
    \loop
        \stepcounter{eq@count}%
        \csarg\let{After\Alph{eq@count}Comment}\relax
        \lowercase
        {%
            \if#1\alph{eq@count}%
```

```
                \eqe@turnOnComment{\Alph{eq@count}}%
```

(09/10/04) Save the commands for turning on the version with `\eqe@onVersion`, and for turning it off with `\eqe@offVeresion`.

```
                \edef\temp@exp{\noexpand
                    \eqe@turnOnComment{\Alph{eq@count}}}%
                \expandafter\eqe@onVersion\expandafter{\temp@exp}%
                \edef\temp@exp{\noexpand
                    \eqe@turnOffComment{\Alph{eq@count}}}%
                \expandafter\eqe@offVersion\expandafter{\temp@exp}%
            \else
                \eqe@turnOffComment{\Alph{eq@count}}%
            \fi
        }%
        \ifnum\value{eq@count}<\eq@nVersions\repeat
        \let\message\save@message
}
```

Let us assume version A initially, user with reset this in document.

```
\AtEndOfPackage{\numVersions{26}\forVersion{A}%
    \eq@renditionOptions}
```

`\selectVersion` When an exam has questions in which the number of variations are not all the same, then you can locally change the version between problems. If the first argument is empty, the first variation is chosen. The syntax is

```
\selectVersion{2}{3}
```

This command says that the next problem has 3 variations, and here we select the second one.

```
\def\selectVersion#1#2{% #1 \le #2
    \xdef\nLocalSelection{#1}\xdef\nLocalVersions{#2}%
```

If obey local version is in effect and the local selection is not empty, no modular arithmetic is needed in the first loop below of `\nLocalSelection`.

```
    \let\needsModArith\eqe@YES
    \ifeqobeylocalversion\ifx\nLocalSelection\@empty\else
    \let\needsModArith\eqe@NO\fi\fi
```

When the solutions appear at the end of the document, the version may not match the version for the question. We need to use a private hook defined in exerquiz (and `eqexam.def`) to reproduce the same settings going into each solution at the end. So, we write the `\selectVersion` to the solution file.

```
    \edef\exer@solnheadhook{%
        \string\selectVersion{#1}{#2}}%
```

Turn off messaging.

```
    \let\save@message\message\let\message\@gobble
```

Reset the selected version, the one selected in the preamble. A previous problem may have changed the version due to modular arithmetic.

```
    \eqe@@onVersion
    \ifx\eqe@@holdTemp\@empty\else
```

If `\eqe@holdTemp` is non-empty, this means that modular arithmetic was performed on the previous problem. We need to turn on the original choice, and turn off the temporary choice, then clear the command `\eqe@@holdTemp`.

```
        \eqe@@onVersion\eqe@@holdTemp
        \let\eqe@@holdTemp\@empty
    \fi
```

If `\eqglobalversion` is `true`, then a `\forVersion` has been executed. If the number of versions declared by `\numVersions` is greater than the number of local versions for this problem, then we perform modular arithmetic to get an appropriate alternative. It may be necessarily to temporarily put `\eqobeylocalversion` to true to accomplish, but we use change it back at the end.

```
\ifx\needsModArith\eqe@YES
    \ifeqglobalversion\ifnum\eq@nSelectedVersion>\nLocalVersions
```

If we perform modular arithmetic, turn off original choice.

```
        \eqe@@offVersion
```

Now perform mod arithmetic

```
        {\count0=\eq@nSelectedVersion \count2=\count0
         \advance\count0by-1 \divide\count0by\nLocalVersions
         \multiply\count0by\nLocalVersions %\count2=\eq@nSelectedVersion
         \advance\count2by-\count0
         \xdef\nLocalSelection{\the\count2 }%
         \ifeqobeylocalversion\else
            \global\@templocalversiontrue
            \global\eqobeylocalversiontrue\fi}%
         \global\let\eqe@@holdTemp\@empty
    \fi\fi
\fi
    \ifeqobeylocalversion
        \global\eqlocalversiontrue
        \setcounter{eq@count}{0}%
```

If local selection is empty, use the version requested by `\forVersion`, modular arithmetic may have to be performed. arithmetic to resolve the matter

```
        \ifx\nLocalSelection\@empty
            \def\nLocalSelection{\eq@nSelectedVersion}%
        \fi
```

If local selection exceeds number of local versions, use modular arithmetic to resolve the matter

```
        \ifnum\nLocalSelection>\nLocalVersions
            {\count0=\nLocalSelection
            \count2=\count0
             \advance\count0by-1 \divide\count0by\nLocalVersions
             \multiply\count0by\nLocalVersions
             \advance\count2by-\count0
             \xdef\nLocalSelection{\the\count2 }}%
        \fi
        \let\save@message\message\let\message\@gobble
        \loop
            \stepcounter{eq@count}%
            \csarg\let{Afterver\Alph{eq@count}Comment}\aebc@end
            \lowercase
            {%
                \ifnum\value{eq@count}=\nLocalSelection
                   \eqe@turnOnComment{\Alph{eq@count}}%
                    \edef\temp@exp{\noexpand
                        \eqe@turnOffComment{\Alph{eq@count}}}%
                    \expandafter\eqe@holdTemp\expandafter{\temp@exp}%
                \else
                    \eqe@turnOffComment{\Alph{eq@count}}%
                \fi
            }%
            \ifnum\value{eq@count}<\nLocalVersions\repeat
            \let\message\save@message
    \fi
    \if@templocalversion\global\eqobeylocalversionfalse\fi
```

added 09/10/03 reset back to default

```
    \@templocalversionfalse
    \let\message\save@message
}
```

9 Title Definitions from Web

\title \subject \author \email \keywords \university

Make Title Definitions taken from the Web package. This is to maintain compatibility with Web.

```
\@ifpackageloaded{web}{}{%
    \@ifpackageloaded{hyperref}{}{\let\texorpdfstring\@firstoftwo}%
    \let\web@save@title\title
    \def\title{\@ifnextchar[{\@web@title}{\@web@title[]}}
    \def\@web@title[#1]#2{\gdef\webtitle{#2}%
        \@ifundefined{hypersetup}{}{\hypersetup{pdftitle={#2}}}%
        \def\webArg{#1}\ifx\webArg\@empty\gdef\shortwebtitle{#2}\else
        \gdef\shortwebtitle{#1}\fi\web@save@title{#2}}
    \let\web@saved@author\author
    \def\author#1{\gdef\webauthor{#1}%
        \@ifundefined{hypersetup}{}{\hypersetup{pdfauthor={#1}}}%
        \web@saved@author{#1}}
    \def\subject{\@ifnextchar[{\@subject}{\@subject[]}}
    \def\@subject[#1]#2{\def\webArg{#1}%
        \ifx\webArg\@empty\gdef\shortwebsubject{#2}\else
        \gdef\shortwebsubject{#1}\fi\gdef\websubject{#2}%
        \@ifundefined{hypersetup}{}{\hypersetup{pdfsubject={#2}}}}
    \def\email#1{\gdef\webemail{#1}}
    \def\keywords#1{\gdef\webkeywords{#1}%
        \@ifundefined{hypersetup}{}{\hypersetup{pdfkeywords={#1}}}}
    \def\university#1{\gdef\webuniversity{#1}}
    \def\copyrightyears#1{\gdef\webcopyrightyears{#1}}
    \def\version#1{\gdef\web@version@value{#1}%
        \edef\webversion{\ifx\web@version@value\@empty\else
            \noexpand\web@versionlabel\noexpand\
            \noexpand\web@version@value\fi}%
    }
    \let\web@version@value\@empty
    \def\versionLabel#1{\def\web@versionlabel{#1}}
    \versionLabel{Version}
    \def\web@toc{Table of Contents}
    \def\web@continued{cont.}
% set  some defaults
    \title{}\author{}\email{}\subject{}\keywords{}\university{}
    \providecommand{\optionalPageMatter}[2][]{%
        \def\optionalpagematter{#2}}
    \def\optionalpagematter{}
}
```

`\date` LaTeX (TeX) defines a `\date` command that is also used by eqexam.

```
\def\duedate#1{\def\theduedate{#1}}
\duedate{}
```

`\duedate` In addition to these, we also define a `\duedate` macro, may be useful for writing assignments with a due date.

```
\def\duedate#1{\def\theduedate{#1}}
\duedate{}
```

`\thisterm` The command `\thisterm` can be used in the `\date` field to indicate the term of this test, for example, `\date{\thisterm, \the\year}` This command may be redefined to conform to your own academic terms.

```
\newcommand\thisterm{%
% if prior to June (Jan--May)
    \ifnum\month<6Spring\else
% if prior to August (June-July)
        \ifnum\month<8Summer\else
% August or later; if Sept--Dec
            \ifnum\month>8Fall\else
% Month of August; if after 25th
                \ifnum\day>25Fall\else
% if it's 25th or earlier in month of August
                    Summer\fi\fi\fi\fi
}
```

10 Identification Information

We define a series of commands in support of building an exam: Lines to identify the student and his/her student id (SID), the instructors email address, the name of the test and the course.

\eqExamName \examNameLabel provides a line for the student to enter his/her name into the exam. The command \examNameLabel can be used to define the name label, the default is Name:

Will insert a text box as well if the option is taken in addition to nosolutions and with solutionsafter not taken. This macro defines \eq@ExamName, which actually contains the code. The first (optional) parameter is passed to \insTxtFieldIdInfo, and can be used to change the appearance of the text field created; the second required parameter is the width of the field.

```
\newcommand{\examAnsKeyLabel}[1]{%
    \def\@examAnsKeyLabel{\ifanswerkey\space #1\fi}}
\examAnsKeyLabel{Answer Key}
\newcommand\examNameLabel[1]{\gdef\@examNameLabel{#1\@examAnsKeyLabel}}
\examNameLabel{Name:}
\newcommand{\idinfoHighlight}[1]{\def\eqe@idinfohl{#1}}
\idinfoHighlight{\underbar}
\newcommand\eqExamName[2][]{%
    \def\eqExamName@argi{#1}\def\eqExamName@argii{#2}}
\def\eq@ExamName{\bgroup
    \settowidth\eq@tmplengthA{\@examNameLabel\ }%
    \@tempdima=\eqExamName@argii\relax
    \advance\@tempdima by-\eq@tmplengthA
    \eqe@idinfohl{\makebox[\eqExamName@argii][l]{\@examNameLabel}}%
    \expandafter\insTxtFieldIdInfo\expandafter[\eqExamName@argi]%
        {\@tempdima}{IdInfo.Name}\egroup}
```

Here we set the field to be a required field with width of 2.25 inches

```
\eqExamName[\Ff\FfRequired]{2.25in}
```

\eqSID provides a line for the student to enter his/her ID number (SID).

Will insert a text box as well if the option is taken in addition to nosolutions and with solutionsafter not taken. The first (optional) parameter is passed to \insTxtFieldIdInfo, and can be used to change the appearance of the text field created; the second required parameter is the width of the field.

```
\newcommand\examSIDLabel[1]{\gdef\@examSIDLabel{#1}}
\examSIDLabel{SID:}
\newcommand\eqSID[2][]{\def\eqSID@argi{#1}\def\eqSID@argii{#2}}
\def\eq@SID{\bgroup\settowidth\eq@tmplengthA{\@examSIDLabel\ }%
    \@tempdima=\eqSID@argii\relax\advance\@tempdima by-\eq@tmplengthA
    \eqe@idinfohl{\makebox[\eqSID@argii][l]{\@examSIDLabel}}%
    \expandafter\insTxtFieldIdInfo\expandafter[\eqSID@argi]%
        {\@tempdima}{IdInfo.SID}\egroup}
```

Here we set the field to be a required field with width of 2.25 inches

```
\eqSID[\Ff\FfRequired]{2.25in}
```

\eqEmail provides a line for the student to enter his/her email address. Useful for documents submitted by email, the instructor can reply.

Will insert a text box as well if the option is taken in addition to nosolutions and with solutionsafter not taken. The first (optional) parameter is passed to \insTxtFieldIdInfo, and can be used to change the appearance of the text field created; the second required parameter is the width of the field.

```
\newcommand\examEmailLabel[1]{\gdef\@examEmailLabel{#1}}
\examEmailLabel{Email:}
\newcommand\eqEmail[2][]{%
    \def\eqExam@argi{#1}\def\eqEmail@argii{#2}}
\gdef\eq@Email{\bgroup\settowidth\eq@tmplengthA{\@examEmailLabel\ }%
    \@tempdima=\eqEmail@argii\relax\advance\@tempdima by-\eq@tmplengthA
    \underbar{\makebox[\eqEmail@argii][l]{\@examEmailLabel}}%
    \expandafter\insTxtFieldIdInfo\expandafter[\eqEmail@argi]%
        {\@tempdima}{IdInfo.email}\egroup}
```

Here we set the field to be a field with width of 2.25 inches. (Not set to be a required field.)

```
\eqEmail{2.25in}
```

`\insTxtFieldIdInfo` The above macros (`\eqExamName`, `\eqSID` and `\eqEmail`) all call this macro, which inserts a Acroforms text field if the `option` is taken in addition to `nosolutions` and with `solutionsafter` not taken.

The first (optional) parameter is used to change the appearance of the text field. The second parameter is the width of the field, and the third is the field name.

```
\def\insTxtFieldIdInfo[#1]#2#3{%
    \@ifundefined{@quiz}{}{\if\eq@online\eqe@YES
        \ifeq@nosolutions\ifeq@solutionsafter\else
            \raisebox{-1bp}{\makebox[0pt][r]{%
                \textField[\BC{}#1]{#3}{#2}{11bp}}}%
        \fi\fi\fi
    }%
}
```

`\SubmitInfo` is required when the `email` option is taken, and should appear in the preamble. The first argument is the URL to the eqAttach.asp code on the server, and the second is the email of the instructor is to receive the results. (Multiple recipients can be specified by separating each with a comma.)

```
\def\SubmitInfo#1#2{%
    \def\EqExam@SubmitURL{#1}\def\@EmailInstr{#2}%
}
```

`\EmailCourseName` is used to specify the course name of the course. The default value for this is `\websubject`, obtained from the `\subject` macro used in the preamble; however, if you want a different name in the email, perhaps with more information included, you can redefine the value using this macro.

```
\def\EmailCourseName#1{\def\@EmailCourseName{#1}}
```

Here's the default value.

```
\EmailCourseName{\websubject}
```

`\EmailExamName` is used to specify the exam name of the course. The default value for this is `\webtitle`, obtained from the `\title` macro used in the preamble; however, if you want a different name in the email, perhaps with more information included, you can redefine the value using this macro. (Multiple recipients can be specified by separating each with a comma.)

```
\def\EmailExamName#1{\def\@EmailExamName{#1}}
```

Here's the default value.

```
\EmailExamName{\webtitle}
```

\EmailSubject The document author mail want a custom subject in the email, instead of the standard one. By using this macro, he can design his own email subject.

```
\def\EmailSubject#1{\def\@EmailSubject{#1}}
```

Here's the default value, which generates no custom subject line.

```
\EmailSubject{}
```

In this case eqAttach.asp inserts the standard one.

```
Exam Results: \webtitle of \websubject
```

The email would read like "`Exam Results: Test 1 of Calculus I`", for example.

\ServerRetnMsg Unless submitted in `silent` mode, the eqAttach.asp returns a message acknowledging the receipt of the data. `\ServerRetnMsg` is used to customize this message.

```
\def\ServerRetnMsg#1{\def\@ServerRetnMsg{#1}}
```

Here's the default value, which generates no custom return message

```
\ServerRetnMsg{}
```

\SubmitButtonLabel is the label that appears on the submit button.

```
\def\SubmitButtonLabel#1{\def\@SubmitButtonLabel{#1}}
```

Here's the default value.

```
\SubmitButtonLabel{Submit}
```

\SubmitButton is the macro that provides the submit button when the `email` option is taken. It appears automatically at the top of the first page of the exam, and appears only if `nosolutions` has has been taken, and `solutionsafter` has not been taken.

```
\let\priorSubmitJS\@gobble
\let\postSubmitJS\@empty
\def\SubmitButton
{%
    \ifx\use@email\eqe@YES\ifeq@nosolutions\ifeq@solutionsafter\else
        \makebox[0pt][l]{\pushButton
            [\CA{\@SubmitButtonLabel}\A{\JS{%
                var _eqEok2Submit = true;\r
                var aSubmitFields = new Array("eqexam", "IdInfo");\r
                \priorSubmitJS\r
                if(_eqEok2Submit) this.submitForm("\EqExam@SubmitURL",
                    true, false, aSubmitFields);\r
                \postSubmitJS
                }}]{Submit}{1.5in}{16bp}}%
        \makebox[0pt][l]{\textField[\F\FHidden\DV{\@EmailInstr}
            \V{\@EmailInstr}]{IdInfo.mailTo}{11bp}{11bp}}%
        \makebox[0pt][l]{\textField[\F\FHidden\DV{\@EmailCourseName}
            \V{\@EmailCourseName}]{IdInfo.courseName}{11bp}{11bp}}%
        \makebox[0pt][l]{\textField[\F\FHidden\DV{\@EmailExamName}
            \V{\@EmailExamName}]{IdInfo.examName}{11bp}{11bp}}%
        \makebox[0pt][l]{\textField[\F\FHidden\DV{\@EmailSubject}
            \V{\@EmailSubject}]{IdInfo.subject}{11bp}{11bp}}%
        \makebox[0pt][l]{\textField[\F\FHidden\DV{\@ServerRetnMsg}
            \V{\@ServerRetnMsg}]{IdInfo.retnmsg}{11bp}{11bp}}%
    \fi\fi\fi
}
```

```
\def\thequizno{\if\probstar*\Alph{quizno}\else\alph{quizno}\fi}
\def\linkContentFormat{%
    \if\probstar*\Alph{quizno}\else\alph{quizno}\fi}
\def\linkContentWrapper{(\hfil\linkContentFormat\hfil)}%
\def\Ans@r@l@Defaults
{%
    \BC{}\S{S}\W{1}\Ff{\FfNoToggleToOff}
    \textSize{12}\textColor{0 g}
}
```

(2014/03/12) Added \eq@hspanner\ignorespaces, in the next two commands. Fixes problems with spacing following the choice box.

```
\newcommand{\optsRadioBtnf}[1]{\def\eqe@optsRadiof{#1}}
\newcommand{\optsRadioBtnl}[1]{\def\eqe@optsRadiol{#1}}
\let\eqe@optsRadiof\@empty\let\eqe@optsRadiol\@empty
\def\eqExam@Ans@sq@l{\leavevmode
    \if\eq@listType1\stepcounter{quizno}\else\ifwithinsoldoc
    \stepcounter{quizno}\else\refstepcounter{quizno}\fi\fi
    \PBS\raggedright\settowidth{\eq@tmplength}{\eq@lw@l}%
    \sbox{\eq@tmpbox}{\eq@l@l}\eq@tmpdima=\wd\eq@tmpbox
    \def\link@@Content{\linkContentWrapper}%
    \hangindent=\eq@tmplength\hangafter=1\relax
    \edef\fieldName{%
        \if\probstar*eqexam.\curr@quiz.\theeqquestionnoi.part\thepartno%
        \else
            eqexam.\curr@quiz.\theeqquestionnoi%
        \fi
    }%
    \if\eq@online\eqe@YES\relax
    \makebox[0pt][l]{%
        \radio@@Button{\presets{\eqe@optsRadiol}}{\fieldName}%
        {\eq@tmpdima}{\RadioFieldSize}{\Ans@choice\alph{quizno}}%
        {\eq@protect\A}{\eq@setWidgetProps\eq@l@check@driver}%
        {\Ans@r@l@Defaults\every@RadioButton\every@qRadioButton}}%
    \else
        \edef\@linkcolor{\@nolinkcolor}%
    \fi
    \ifeq@nosolutions\edef\@linkcolor{\@nolinkcolor}\fi
    \textcolor{\@linkcolor}{\makebox[\eq@tmpdima]{\link@@Content}}%
    \Ans@proofing{\eq@tmpdima}%
    \eq@hspanner\ignorespaces
}
```

(2019/10/28) Legacy assignment, in case eqexam.def/exerquiz are still using \eq@RadioCheck@driver.

```
\@ifundefined{eq@RadioCheck@driver}{}
 {\let\eq@Radio@driver\eq@RadioCheck@driver}
\def\eqExam@Ans@sq@f{\if\eq@listType1\stepcounter{quizno}\else
    \ifwithinsoldoc\stepcounter{quizno}\else
      \refstepcounter{quizno}\fi\fi
    \PBS\raggedright\settowidth{\eq@tmplength}{\eq@lw@f}%
    \eq@tmpdima=\wd\eq@tmpbox%
    \hangindent=\eq@tmplength\hangafter=1\relax
    \if\eq@online\eqe@NO\previewtrue
        \insertGrayLetters
        \Ans@sq@f@driver
    \else
        \ifanswerkey\previewtrue
            \Ans@sq@f@driver
        \else
            \edef\fieldName{%
                \if\probstar*eqexam.\curr@quiz.\theeqquestionnoi.%
                    part\thepartno%
                \else
                    eqexam.\curr@quiz.\theeqquestionnoi%
                \fi
            }\insertGrayLetters
            \radio@@Button{\presets{\eqe@optsRadiof}}%
            {\fieldName}{\RadioFieldSize}%
            {\RadioFieldSize}{\Ans@choice\alph{quizno}}{\eq@protect\A}%
            {\eq@setWidgetProps\eq@Radio@driver}%
            {\@@Ans@sq@f@Defaults\Ans@sq@f@Actions\every@RadioButton
            \every@sqRadioButton\insert@circlesymbol}%
        \fi
    \fi
    \Ans@proofing{\RadioFieldSize}%
    \eq@hspanner\ignorespaces
}
\newcommand{\optsCkBxf}[1]{\def\eqe@optscbf{#1}}
\newcommand{\optsCkBxl}[1]{\def\eqe@optscbl{#1}}
\let\eqe@optscbl\@empty\let\eqe@optscbf\@empty
\def\eqExam@Ans@ck@sq@l{\leavevmode
    \if\eq@listType1\stepcounter{quizno}\else
    \ifwithinsoldoc\stepcounter{quizno}\else
      \refstepcounter{quizno}\fi\fi
    \PBS\raggedright\settowidth{\eq@tmplength}{\eq@lw@l}%
    \sbox{\eq@tmpbox}{\eq@l@l}\eq@tmpdima=\wd\eq@tmpbox
    \def\link@@Content{\linkContentWrapper}%
    \hangindent=\eq@tmplength\hangafter=1\relax
    \edef\fieldName{%
        \if\probstar*eqexam.\curr@quiz.\theeqquestionnoi.%
            part\thepartno.\alph{quizno}%
    \else
      eqexam.\curr@quiz.\theeqquestionnoi.\alph{quizno}%
    \fi
  }%
  \if\eq@online\eqe@YES\relax
  \makebox[0pt][l]{\check@@Box{\presets{\eqe@optscbl}}{\fieldName}%
    {\eq@tmpdima}{\RadioFieldSize}{\Ans@choice\alph{quizno}}%
    {\eq@protect\A}{\eq@setWidgetProps\eq@l@check@driver}%
    {\Ans@r@l@Defaults\every@RadioButton\every@qRadioButton}}%
  \else
    \edef\@linkcolor{\@nolinkcolor}%
  \fi
  \ifeq@nosolutions\def\@linkcolor{\@nolinkcolor}\fi
  \textcolor{\@linkcolor}{\makebox[\eq@tmpdima]{\link@@Content}}%
  \Ans@proofing{\eq@tmpdima}%
  \eq@hspanner\ignorespaces
}
\def\eqExam@Ans@ck@sq@f{\if\eq@listType1\stepcounter{quizno}\else
  \ifwithinsoldoc\stepcounter{quizno}\else
   \refstepcounter{quizno}\fi\fi
  \PBS\raggedright\settowidth{\eq@tmplength}{\eq@lw@f}%
  \eq@tmpdima=\wd\eq@tmpbox%
  \hangindent=\eq@tmplength\hangafter=1\relax
  \if\eq@online\eqe@NO\previewtrue
    \insertGrayLetters
    \Ans@sq@f@driver
  \else
    \ifanswerkey\previewtrue
      \Ans@sq@f@driver
    \else
      \edef\fieldName{%
        \if\probstar*eqexam.\curr@quiz.\theeqquestionnoi.%
          part\thepartno.\alph{quizno}%
        \else
          eqexam.\curr@quiz.\theeqquestionnoi.\alph{quizno}%
        \fi
      }\insertGrayLetters
      \mbox{\check@@Box{\presets{\eqe@optscbf}}{\fieldName}%
        {\RadioFieldSize}{\RadioFieldSize}%
        {\Ans@choice\alph{quizno}}{\eq@protect\A}%
        {\eq@setWidgetProps\eq@Radio@driver}%
        {\@@Ans@sq@f@Defaults\Ans@sq@f@Actions\every@RadioButton
        \every@sqRadioButton}}%
    \fi
  \fi
  \Ans@proofing{\RadioFieldSize}%
  \eq@hspanner\ignorespaces
}
```

`\optsMlTextField` is used to pass options to the multi-line text field.

```
\def\optsMlTextField#1{\def\eqe@optsmltf{#1}}
\let\eqe@optsmltf\@empty
\def\eqExamPriorVspace#1{%
    \edef\fieldName{%
        \if\probstar*eqexam.\curr@quiz.%
            \theeqquestionnoi.part\thepartno.solution%
        \else
            eqexam.\curr@quiz.\theeqquestionnoi.solution%
        \fi
    }%
    \nobreak\noindent\textField[\BC{}\presets{\eqe@optsmltf}
        \Ff\FfMultiline]{\fieldName}{\linewidth}{#1}\@gobble
}
```

Test to see if `exerquiz` is loaded. If not, we input the 'stand alone', `eqalone.def`, followed by `eqexam.def`. The latter definition file is maintained in `exerquiz.dtx` under the `eqexam` option.

```
\@ifpackageloaded{exerquiz}{%
    \let\Ans@sq@l\eqExam@Ans@sq@l
    \let\Ans@sq@f\eqExam@Ans@sq@f
    \let\Ans@ck@sq@l\eqExam@Ans@ck@sq@l
    \let\Ans@ck@sq@f\eqExam@Ans@ck@sq@f
    \def\eqexheader@wrapper{\makebox[0pt][r]{%
        \hypertarget{qex.\the@exno}{\eqexheader}}}%
    \if\eq@online\eqe@YES\relax
        \newcounter{@cntfillin}%
        \let\eqPriorVspace\eqExamPriorVspace
    \fi
}%
{%
    \input{eqalone.def}
    \input{eqexam.def}

    \@ifl@ter{def}{eqexam}{\eqexamdefReq}{}
    {\PackageWarningNoLine{eqexam}
    {This version of eqexam requires eqexam.def\MessageBreak
    dated \eqexamdefReq\space or later}}
    \def\eqexheader@wrapper{\makebox[0pt][r]{\eqexheader}}
}
```

We wrote `\begin{eqequestions}` to the top of the solutions file (`\jobname.sol`.

```
%\writeBeginEqeQuestions
```

If the `vspacewithsolns` is in effect, we write solutions to the end of the document.

```
\AtBeginDocument{%
    \ifvspacewithsolns\writeAllAnsAtEnd\else
    \ifeqfortextbook\writeAllAnsAtEnd\else
    \writeWithSolDocTrue\fi\fi
}
```

We execute \vspacewithkeyOff, which sets \ifkeepdeclaredvspacing to false, the default behavior of eqexam before the new feature.

```
\vspacewithkeyOff
```

```
⟨/package⟩
```

11 Stand alone Code

```
⟨*standalone⟩
```

Now we begin the listing of the stand alone code. This code is necessary if exerquiz has not been loaded, which is the case if there is no PDF options or if the pdf option is taken.

Many of the following definitions are given in eforms, which was recently separated from exerquiz and is now maintained as a separate package.

```
\ProvidesFile{eqalone.def}
 [2012/25/01 v3.0t Minimal code used by eqexam (dps)]
\@ifundefined{eq@tmpbox}{\newsavebox{\eq@tmpbox}}{}% defined in eforms
\@ifundefined{eq@tmpdima}{\newdimen\eq@tmpdima}{}  % defined in eforms
\def\RadioFieldSize{11bp}
```

```
\newdimen\eqcenterWidget
```

This macro is used to vertically center the response box on the line. Seems to work well.

```
\def\centerWidget
#1{%
    \eqcenterWidget=#1
    \eqcenterWidget=.5\eqcenterWidget
    \advance\eqcenterWidget by-4bp
}
```

\eqe@Bbox When the preview option has been used, draw a frame box around the bounding rectangle.

```
\def\eqe@BboxRect#1#2{\hbox{\ckbox@Color\vbox{\hrule width #1
  \hbox to#1{\vrule height#2\hfill\vrule height#2}\vfill\hrule}}}
\def\useRectForMC{\let\Bbox\eqe@BboxRect}
\def\selectedMC{\useRectForMC}
```

\Rect is used internally to color a link.

```
\def\Rect#1{\textcolor{\@linkcolor}{#1}}
```

\ReturnTo The auxiliary file eqexam.def, created by exerquiz, writes \ReturnTo to the SOL file in the form \ReturnTo{page.1}{\mbox{}}. We want to remove the \mbox because it causes, at times, more vertical space that is wanted in an exam document.

```
\def\eqe@striphbox\mbox#1{#1}
\newcommand{\ReturnTo}[2]{\eq@fititin{\eqe@striphbox#2}}
```

12 Switching proofing symbols

\proofingsymbol The definition of the proofing symbol, this symbol marks the correct answer of a multiple choice question when the `proofing` option is used.

```
\newcommand{\proofingsymbol}[1]{%
    \def\@proofingsymbol{\textcolor{\@proofingsymbolColor}{#1}}}
```

This is the answers macro for the link-style and is called from the `eqexam.def` file.

```
⟨/standalone⟩
```

```
⟨*package⟩
```

In response to the `allowcircmc`, we load `lcircle10` and use the ‘h’ and ‘x’ glyph.

```
\def\selectedMC{\useRectForMC}
\ifallowcircmc
    \font\eqe@lcir=lcircle10 at 12pt
    \bgroup
```

Get the width of the ‘h’, the circle has zero height and depth. Set the diameter and radius of the circle.

```
        \setbox0=\hbox{\eqe@lcir h}
        \xdef\eqe@cirDiam{\the\wd0}
        \@tempdima=.5\wd0
        \xdef\eqe@cirRadius{\the\@tempdima}
    \egroup
```

A command to use the circle (h) and the filled circle (x).

```
    \def\circ@Glyph#1#2{\hbox{\smash{\raisebox{\eqe@cirRadius}%
        {\makebox[\eqe@cirDiam]{\llap{\ckcir@Color
        \rlap{\eqe@lcir#1}\hskip#2\relax}}}}}}
```

Added preview color to the circle version of the bounding box. The color accessed through `\previewColor`, its default is `black`.

```
    \def\eqe@BboxCirc#1#2{\hbox{\ckcir@Color{\circ@Glyph{h}{0pt}}}}
```

For proofing purposes, prepare the filled circle in the form of the command `\circProofingForCirc`.

```
    \def\circProofingForCirc{\centerWidget\RadioFieldSize
        \lower\eqcenterWidget\circ@Glyph{x}{1.4pt}}
    \if\eq@online\eqe@YES
```

If online (`online` or `email` option), we cancel these commands.

```
        \def\useCircForMC{\let\Bbox\eqe@BboxCirc\useMCCircles}
        \def\selectedMC{\useCircForMC}
        \@ifundefined{eqe@BboxRect}
            {\def\useRectForMC{\let\Bbox\ef@Bbox\useMCRects}}
            {\def\useRectForMC{\let\Bbox\eqe@BboxRect\useMCRects}}
%        \let\useRectForMC\relax
    \else % if not online
        \@ifundefined{eqe@BboxRect}
            {\def\useRectForMC{\let\Bbox\ef@Bbox}}
            {\def\useRectForMC{\let\Bbox\eqe@BboxRect}}
        \def\useCircForMC{\let\Bbox\eqe@BboxCirc}
        \def\selectedMC{\useCircForMC}
    \fi
    \def\useCircForProof{\symbolchoice{circle}%
        \proofingsymbol{\circProofingForCirc}}
\else
```

If allowcircmc is not taken

```
    \if\eq@online\eqe@YES
        \def\useCircForMC{\let\Bbox\eqe@BboxCirc\useMCCircles}
        \@ifundefined{eqe@BboxRect}
            {\def\useRectForMC{\let\Bbox\ef@Bbox\useMCRects}}
            {\def\useRectForMC{\let\Bbox\eqe@BboxRect\useMCRects}}
    \else
        \@ifundefined{eqe@BboxRect}
            {\def\useRectForMC{\let\Bbox\ef@Bbox}}
            {\def\useRectForMC{\let\Bbox\eqe@BboxRect}}
        \def\useCircForMC{\useRectForMC}
        \def\selectedMC{\useRectForMC}
    \fi
    \let\useCircForProof\relax
\fi
```

\useRectForMS
\useCircForMS

\useRectForMS When declared, rectangles are used for multiple selection, similarly, \useCircForMS uses circles if allowcirc4mc is in effect.

```
\newif\ifuserectforms
\def\useRectForMS{\userectformstrue}
\def\useCircForMS{\userectformsfalse}
```

```
\newcommand{\useCheckForProof}{\symbolchoice{check}%
    \proofingsymbol{\ding{52}}}
\useCheckForProof
```

Similarly, for a cross, we get a nice ‘handwritten’ cross \ding{56}, !when we latex the document; for the online or email option, we get a simple cross in the active radio button field.

```
\newcommand{\useCrossForProof}{\symbolchoice{cross}%
    \proofingsymbol{\raisebox{-1pt}{\rlap{\kern-1pt\Large\ding{56}}}}}
```

13 The Main Code

We now continue with the main package. Mostly, we define macros specific to the eqexam package: define the problem and problem* environments, macros for calculating totals per page, etc.

```
\def\MCcolor{black}
\def\Ans@sq@l@driver{\edef\@linkcolor{\MCcolor}%
    \Rect{\makebox[\eq@tmpdima]{\linkContentWrapper}}%
    \Ans@proofing{\eq@tmpdima}%
}
```

This is the answers macro for the form-style and is called from the eqexam.def file.

```
\def\Ans@sq@f@driver{%
    \centerWidget\RadioFieldSize
    \leavevmode\lower\eqcenterWidget\Bbox %\eqe@Bbox
        {\RadioFieldSize}{\RadioFieldSize}%
    \Ans@proofing{\RadioFieldSize}%
}
```

Write quiz solutions to the exercise solutions file

```
\def\eq@sqsllabel{\string\textbf{Solution to Quiz:}}
\def\sqsllabel{\eq@sqsllabel}
```

\writeToSolnFile General purpose command for writing to the solution file.

\preExamSolnHead Executed just before a user friendly name

\examSolnHeadFmt Format for the user friendly name

\postExamSolnHead Executed just after a user friendly name

```
\let\quiz@solns\ex@solns
\newcommand{\preExamSolnHead}{\goodbreak\noindent}
\newcommand{\examSolnHeadFmt}[1]{\textbf{#1}}
\newcommand{\postExamSolnHead}{\par\medskip}
```

Write to solution file if *not* solutions-only

```
\@ifundefined{ifOKToWriteExamData}{\newif\ifOKToWriteExamData
  \OKToWriteExamDatatrue}{}
\newcommand{\writeToSolnFile}[1]{%
  \ifsolutionsonly\else\ifOKToWriteExamData
    \let\quiz@solns\ex@solns
    \set@display@protect
    \eqe@IWO\quiz@solns{#1}%
    \set@typeset@protect
  \fi\fi}
\let\writeT@SolnFile\writeToSolnFile
```

We will write all solutions to the `.sol` auxiliary file.

```
\def\eqe@writetoSolns#1{% dpsD17
  \ifsolutionsonly\else
  \ifOKToWriteExamData
  \set@display@protect
  \eqe@IWO\quiz@solns{\string\preExamSolnHead
      \string\examSolnHeadFmt{#1}\string\postExamSolnHead}%
  \set@typeset@protect\fi\fi}
\def\eqe@writetoAux#1{%
    \set@display@protect
    \eqe@IWO\@auxout{#1}%
    \set@typeset@protect}
```

Turn off interactivity of short quiz.

This macro is defined in exerquiz, but has a little different definition for eqexam.

```
\def\Ans@proofing
#1{%
    \ifeq@proofing\if\Ans@choice1\relax
        \llap{\rlap{\,\@proofingsymbol}\hskip#1\relax}%
    \fi\fi
}
```

\eq@ExamLastPage This macro gets the page number of the last page of the exam. It is read in through a macro definition made and written to the .aux file.

```
\def\eq@ExamLastPage{\csname eqExamLastPage\endcsname}
```

\nPagesOnExam is the number of pages in the exam.

```
\newcommand{\nPagesOnExam}{\csname eqExamLastPage\endcsname}
```

\nPagesOnQues is the number of pages in the questions.

```
\newcommand{\nPagesOfQues}{\ifvspacewithsolns
    \csname eqExamQuesLastPage\endcsname\else
    \csname eqExamLastPage\endcsname\fi}
```

\nPagesOnSols is the number of pages of solutions.

```
\newcommand{\nPagesOfSols}{\def\eqExamNumPagesSolns{0}%
    \csarg\ifx{eqExamLastPage}\relax\else
    \csarg\ifx{eqExamLastPage}\relax\else
    {\count\z@=\nPagesOnExam\relax\advance
     \count\z@-\nPagesOfQues\relax
     \xdef\eqExamNumPagesSolns{\the\count\z@}}\fi\fi
     \eqExamNumPagesSolns}
```

\lastPageOfExam Returns the page number of the end of the exam with a name of #1.

\firstPageOfExam Returns the page number of the beginning of the exam with a name of #1.

```
\newcommand{\lastPageOfExam}[1]{\pageref{#1PageEnd}}
\newcommand{\firstPageOfExam}[1]{\pageref{#1PageBegin}}
```

\eqe@defNumRefii The last two commands are meant to produce typeset numbers; however, there is a need to convert these to numbers that tex's registers can manipulate. Here goes. \eqe@defNumRefii takes its argument and strips away the other arguments of \pageref; it picks off the second of two or five, depending if hyperref is loaded. It defines a macro \csname nRefii@#1\endcsname whose value is a page number of the referenced object.

```
\newcommand{\eqe@defNumRefii}[1]{%
    \@ifundefined{hyperref}{\let\@getsecondOf\@secondoftwo}
    {\let\@getsecondOf\@secondoffive}%
    \@ifundefined{r@#1}{%
```

If the reference r@#1 is undefined, define the value to be 0

```
    \csarg\gdef{nRefii@#1}{0}}{%
```

If the reference r@#1 is defined, define the value to be the second argument of \r@#1 expanded

```
    \expandafter\xdef\csname nRefii@#1\endcsname
        {\expandafter\expandafter\expandafter
    \@getsecondOf\csname r@#1\endcsname}%
    }%
}
```

\eqe@numRefii takes one argument, the control name. Its value is zero or \nRefii@#1. This expands to a number in all cases. It can be used in tex comparisons.

```
\def\eqe@numRefii#1{%
    \csarg\ifx{nRefii@#1}\relax 0\else
    \@nameuse{nRefii@#1}\fi}
```

\numLastPageOfExam
\numFirstPageOfExam

This is the user-interface to acquiring the first and last page numbers of the exam with name #1. These can be used in comparisons, e.g.

```
\rfooteqe{\ifnum\value{page}<\numLastPageOfExam{<myTest>}%
    \textbf{Test Continues}\fi}
```

```
\newcommand{\numLastPageOfExam}[1]{\eqe@numRefii{#1PageEnd}}
\newcommand{\numFirstPageOfExam}[1]{\eqe@numRefii{#1PageBegin}}
```

```
\newcommand{\makeRefsNums}{%
    \@ifundefined{thePartNames}{}{\begingroup
        \def\\##1{\PackageInfo{eqexam}{processing exam: ##1}%
            \eqe@defNumRefii
            {##1PageEnd}\eqe@defNumRefii{##1PageBegin}}%
        \thePartNames
    \endgroup}%
}
```

```
\AtBeginDocument{\makeRefsNums}
```

13.1 Running Heads and Feet

We develop a series of macros for creating running headers and footers for the exam.

\lheadeqe
\cheadeqe
\rheadeqe

Set the left, center, and right running headers.

```
\newcommand{\lheadeqe}[1]{\def\eq@lhead{#1}}
\lheadeqe{\shortwebsubject/\shortwebtitle}
\newcommand{\cheadeqe}[1]{\def\eq@chead{#1}}
\cheadeqe{-- Page \arabic{page} of {\nPagesOnExam} --}
\newcommand{\rheadeqe}[1]{\def\eq@rhead{#1}}
```

The default is \eq@ExamName, which is defined by \eqExamName, the default displays the word "Name" and an underlined horizontal space for the student to enter his/her name.

```
\rheadeqe{\eq@ExamName}
```

\lhead \chead \rhead These are the original names for the headers, we'll keep them if fancyhdr is not already loaded to maintain compatibility with previous versions of eqexam. The use of these commands is *discouraged.*

```
\@ifpackageloaded{fancyhdr}{}{%
    \let\lhead\lheadeqe
    \let\chead\cheadeqe
    \let\rhead\rheadeqe
}
```

\runExamHeader \eqExamRunHead The running header of the exam, may be redefined.

```
\newcommand{\runExamHeader}{\eq@lhead\hfill\eq@chead\hfill\eq@rhead}
\newcommand\eqExamRunHead{%
    \addtolength\textwidth{\oddsidemargin}%
    \noindent\hspace*{-\oddsidemargin}\makebox[\textwidth]
    {\runExamHeader}%
}
```

Running footers. One or two users wanted to use running footers, so here they are.

\lfooteqe \cfooteqe \rfooteqe There has been some demand for running footers. You have to be a little careful, eqexam uses the footer for the command \settotalsbox, which puts in the totals for the pages either on the left (totalsonleft) or right (totalsonright) side, depending on the option.

```
\newcommand{\lfooteqe}[1]{\def\eq@lfoot{#1}}
\lfooteqe{}
\newcommand{\cfooteqe}[1]{\def\eq@cfoot{#1}}
\cfooteqe{}
\newcommand{\rfooteqe}[1]{\def\eq@rfoot{#1}}
\rfooteqe{}
```

\runExamFooter Considering the defaults for the left, center, and right footer elements, the default footer contributes nothing, except inserting \settotalsbox (see the definition of \@oddfoot)

```
\newcommand{\runExamFooter}{\eq@lfoot\hfill\eq@cfoot\hfill\eq@rfoot}
```

Running headers for solutions. We provide a special set of headers for the solution pages. The document author needs to manage running footers for the solution pages.

\lheadSol \cheadSol \rheadSol \runExamHeaderSol \eqsolutionshook The running header of the exam, when solutions are included at the end of the document, perhaps for posting the solutions to the exam, or publication of a "pretest". Note that \eqsolutionshook is defined in exerquiz/eqexam.def. May be redefined.

```
\newcommand{\lheadSol}[1]{\def\eq@lheadSol{#1}}
\lheadSol{\shortwebsubject/\shortwebtitle}
\newcommand{\cheadSol}[1]{\def\eq@cheadSol{#1}}
\cheadSol{-- Page \arabic{page} of {\eq@ExamLastPage} --}
```

```
\newcommand{\rheadSol}[1]{\def\eq@rheadSol{#1}}
\rheadSol{SOLUTIONS}
```

The `\runExamHeaderSol` distributes the three solution headers across the page.

```
\newcommand{\runExamHeaderSol}
    {\eq@lheadSol\hfill\eq@cheadSol\hfill\eq@rheadSol}
```

The solutions headers are inserted using the `\eq@solutionshook`, see the definition of `\exerSolnInput`.

```
\def\eq@solutionshook
{%
    \gdef\eqExamRunHead{\addtolength\textwidth{\oddsidemargin}%
    \noindent\hspace*{-\oddsidemargin}\makebox[\textwidth]
    {\runExamHeaderSol}}%
}
```

13.2 `\maketitle` definitions

`\maketitle` `\maketitledesign` `\altTitle` Standard LaTeX macro, but this time it is used to create the header at the top of the first page of the exam. Typically, consisting of two rows of info. (1) first row has course name, exam name, and a place for the student to put his/her name. (2) second row has date and instructor. May be redefined.

Modify the title by redefining `\maketitledesign`, the `\maketitle` command itself has LaTeX commands in it that should not be changed.

```
\newcommand\maketitledesign
{%
    \makebox[\textwidth]{\normalsize
        \shortstack[l]{\strut\websubject\\\@date}\hfill
        \shortstack[c]{\webtitle\\\strut\@altTitle}\hfill
        \shortstack[l]{\strut\eq@ExamName\\\webauthor}}%
}
\def\altTitle#1{\def\@altTitle{#1}}
\let\@altTitle\@empty
\def\eqemaketitle
{%
```

`\EQEcalculateAllTotals`: We don't actually calculate all totals, just some of them. We do calculate the grade total of all the `exam` environments in the document, we also calculate the percentage that each exam contributes to to the total. If `\maketitle` is not used, for whatever reason, this command should be calculated explicitly just after `\begin{document}`.

```
    \EQEcalculateAllTotals
    \begingroup
%    \addtolength\textwidth{\oddsidemargin}%
    \noindent%\hspace*{-\oddsidemargin}%
    \raisebox{.7in}[0pt][0pt]{\SubmitButton}%
    \maketitledesign
    \endgroup
}
```

If the `fortextbook` option is not taken, we set up the usual `\maketitle` definition. If the `useclassmaketitle` option is taken, we don't redefine either.

```
\ifeqfortextbook
    \@ifpackageloaded{web}{\let\maketitle\web@save@maketitle}{}%
\else
    \if@eqeuseclassmaketitle
        \@ifpackageloaded{web}{\let\maketitle\web@save@maketitle}{}%
    \else
        \let\maketitle\eqemaketitle
    \fi
\fi
```

13.3 The cover page definitions

`\eqexcoverpage` The `eqexam` package allows for the possibility of a cover page, if the `coverpage` option is taken.

`\placeCoverPageLogo` A simple command to insert a logo on the cover page. The logo can be used to cover the score in the next page, if the instructor places the score under the logo. Example of usage

```
\placeCoverPageLogo{5in}{-1.5in}{\includegraphics{nwfsc_logo}}
```

Working from the upper left corner, the first parameter is the amount to move to logo to the right, the second parameter is the amount to move the logo vertically. The Third parameter is the content; perhaps an `\includegraphics` command.

```
\newcommand\placeCoverPageLogo[3]{%
    \def\eqe@insertLogo{\hbox to0pt{%
        \hspace*{#1}\smash{\raisebox{#2}{#3}}\hss}}}
\let\eqe@insertLogo\relax
```

Define `\eqexcoverpage`, this command places `\eqe@insertLogo` and `\eqexcoverpagedesign` in a group. It is this command that gets executed when the user calls for the `coverpage` option.

```
\def\eqexcoverpage{%
    \begingroup
    \pagenumbering{roman}
    \eqe@insertLogo
    \eqexcoverpagedesign
    \endgroup\newpage
    \pagenumbering{arabic}
}
```

`\eqexcoverpagedesign` The `eqexam` package allows for the possibility of a cover page, if the `coverpage` option is taken. This macro can and should be redefined to fit your needs. The definition below is just a representative example.

`\coverpagesubject`
`\coverpageUniversityFmt`
`\coverpageSubjectFmt`
`\coverpageTitleFmt`

The following four commands are used with the cover page. The `\coverpagesubject` is used to provide a special subject for the cover page, different from `\websubject`. The others are used for formatting.

```
\newcommand{\coverpagesubject}[1]{\def\coverpage@subject{#1}}
\let\coverpage@subject\@empty
\def\eqexamsubject{\ifx\coverpage@subject\@empty\websubject
    \else\coverpage@subject\fi}
\newcommand{\coverpageUniversityFmt}[1]{%
    \def\eqex@coverpageUniversityFmt{#1}}
\coverpageUniversityFmt{\bfseries\large}
\newcommand{\coverpageSubjectFmt}[1]{%
    \def\eqex@coverpageSubjectFmt{#1}}
\coverpageSubjectFmt{\bfseries\large}
\newcommand{\coverpageTitleFmt}[1]{%
    \def\eqex@coverpageTitleFmt{#1}}
\coverpageTitleFmt{\bfseries\large}
\newcommand\cpCID[1]{\def\cp@@CID{#1}}
\let\cp@@CID\@empty
%\newcommand\coverpageCID[1]{\def\cpCID@argi{#1}}
\newcommand{\cpSetCIDWidth}[1]{\bgroup\setlength{\@tempdima}{#1}
    \xdef\cpCID@argi{\the\@tempdima}\egroup}
\def\cp@CID{\bgroup\settowidth\eq@tmplengthA{\cp@@CID}%
    \@tempdima=\cpCID@argi\relax
    \advance\@tempdima-\eq@tmplengthA
    \eqe@idinfohl{\makebox[\cpCID@argi][l]{\cp@@CID}}\egroup}
\cpSetCIDWidth{2.25in}
%\coverpageCID{2.25in}
```

\qeSumryVert \eqeSumryHoriz

When `coverpage` and `coverpagesumry` are used, an **Exam Record** is generated, just a summary may appear beneath the name and ID section of the cover page (`\qeSumryVert`) or to the right (`\eqeSumryHoriz`).

```
\def\eqeSumryVert{\def\eqe@SumryVert{\par\vfill}%
    \let\eqe@SumryHoriz\relax
    \def\eqe@@SumryVert{\vspace{\stretch{-1}}\bigskip}}
\def\eqeSumryHoriz{\def\eqe@SumryHoriz{\hfill}\let\eqe@SumryVert\relax
    \let\eqe@@SumryVert\relax}
```

The default is a horizontal orientation.

```
\eqeSumryHoriz
```

\useUIPartNames \useCustomPartNames

We can name the components of the **Exam Record** by page or by parts. For the case they are named by parts, there are three options: You can used the exam name (the default); you can use the friendly name of the exam (the optional argument), executing `\useUIPartNames` invokes this option; you can use custom names (useful if the friendly names are too long), execute the command `\useCustomPartNames` for this option.

```
\def\eqe@coverPageNaming{0}
\newcommand{\useUIPartNames}{\def\eqe@coverPageNaming{1}}
\newcommand{\useCustomPartNames}{\def\eqe@coverPageNaming{2}}
```

\customNaming

When there is custom naming (`\customNaming`), we need to provide the user with a way of defining these custom names. `\customNaming` provides that mechanism. The command takes two arguments, the first is ***name***, the exam name, the second is the associated *text* the text that is to appear in the **Exam Record**

```
\newcommand{\customNaming}[2]{\@namedef{userCustom#1}{#2}}
```

```
\newcommand{\cpSetNameAndIDWidth}[1]{\bgroup
    \setlength{\@tempdima}{#1}%
    \xdef\cp@SetNameAndIDWidth{\the\@tempdima}\egroup}
\cpSetNameAndIDWidth{.45\textwidth}
\newcommand{\cpEnclNameAndID}[1]{\def\cp@EnclNameAndID{#1}}
\let\cp@EnclNameAndID\@empty
\newcommand{\cpSetHghtFrstLn}[1]{\def\cp@HghtFrstLn{#1}}
\def\cp@HghtFrstLn{0pt}
\def\cp@setHghtFrstLn{\rule[\cp@HghtFrstLn]{0pt}{0pt}}
\newcommand\cpNameAndID{\noindent
    \edef\eqExamName@argii{\cp@SetNameAndIDWidth}%
    \edef\eqSID@argii{\cp@SetNameAndIDWidth}%
    \edef\cpCID@argi{\cp@SetNameAndIDWidth}%
    \cp@EnclNameAndID{\parbox[b]{\cp@SetNameAndIDWidth}{\parindent0pt
    \cp@setHghtFrstLn\eq@ExamName\\[2ex]
    \ifx\cp@@CID\@empty\eq@SID\else\cp@CID\fi\\[2ex]
    \ifx\use@email\eqe@YES\eq@Email\\[2ex]\fi
    \textcolor{\webauthor@color}{\webauthor}, \@date\vskip0pt}}}
```

Now we present the definition of \eqexcoverpagedesign.

```
\newcommand{\eqexcoverpagedesign}
{%
    \begingroup
    \parindent0pt
    \thispagestyle{empty}
%    \addtolength\textwidth{\oddsidemargin}
    \vspace*{.1\textheight}
    \noindent%\hspace*{-\oddsidemargin}%
    \makebox[\linewidth]{\parbox{\linewidth}%
        {\eqex@coverpageUniversityFmt
        \color{\webuniversity@color}%
        \centering\webuniversity}}
    \par\vspace{.1\textheight}
    \noindent%\hspace*{-\oddsidemargin}%
    \makebox[\linewidth]{\parbox{\linewidth}%
        {\eqex@coverpageSubjectFmt
        \color{\websubject@color}%
        \centering\eqexamsubject}}
    \par\vspace{\bigskipamount}
    \noindent%\hspace*{-\oddsidemargin}%
    \makebox[\linewidth]{\parbox{\linewidth}%
        {\eqex@coverpageTitleFmt
         \color{\webtitle@color}%
        \centering\webtitle}}
    \par\vspace{\stretch{1}}
    \optionalpagematter
    \par\vspace{\stretch{1}}\cpNameAndID
    \eqe@SumryHoriz\eqe@SumryVert
    \sumryAnnots
    \endgroup
}
```

The following are various local strings used in the **Exam Record**, the default is to use English words.

`\cpSumryHeader` The header that appears at the top of the box

`\cpSumryPts` The formatting for the number of points

`\cpSumryPage` The word for “Page”

`\cpSumryTotal` The word for “Total”

`\cpSumryGrade` The word for “Grade”

`\cpSetSumryWidth` The width of the summary box

`\cpUsefbox` Enclose in an `\fbox`? The default is yes

`\cpNofbox` Do not enclose in an `\fbox`

```
\newcommand{\cpSumryHeader}{\textbf{Exam Record}}
\newcommand{\cpSumryPts}{\,\text{pts}}
\newcommand{\cpSumryPage}{Page}
\newcommand{\cpSumryTotal}{Total:}
\newcommand{\cpSumryGrade}{Grade:}
\newcommand{\cpSetSumryWidth}[1]{\bgroup\setlength{\@tempdima}{#1}
    \xdef\cp@SetSumryWidth{\the\@tempdima}\egroup}
\cpSetSumryWidth{.5\textwidth}
\newcommand{\cpUsefbox}{\let\cp@Usefbox\fbox}
\cpUsefbox
\newcommand{\cpNofbox}{\let\cp@Usefbox\mbox}
\newcommand{\cprulelength}{1.5in}
```

A helper command used in `\cpSumybyparts`

```
\def\cp@IsertNaming#1{%
    \ifcase\eqe@coverPageNaming
    #1\or\@nameuse{userFriendly#1}\or
    \@nameuse{userCustom#1}\else#1\fi
}
```

`\cpSumrybyparts` The routine for building the summary box, where we list the statistics for each part.

```
\newcommand{\cpSumrybyparts}{%
  \eqe@@SumryVert\cp@Usefbox{%
  \ifx\cp@Usefbox\fbox
    \def\cp@sumryWdth{\cp@SetSumryWidth-2\fboxrule-2\fboxsep}\else
  \def\cp@sumryWdth{\cp@SetSumryWidth}\fi
  \begin{minipage}[b]{\cp@sumryWdth}\kern0pt
    \begin{flushleft}
```

```
      \setlength{\eqetmplengtha}{\cprulelength}%
      \edef\eqe@cprulelength{\the\eqetmplengtha}%
      \csarg\ifx{NumberOfParts}\relax
        \else
          \count2=0
          \medskip
          \cpSumryHeader\par\parskip\bigskipamount
          \def\\##1{\advance\count2by\csname ##1total\endcsname
          \underbar{\makebox[0pt][l]{%
          \cp@IsertNaming{##1}}\hspace*{\eqe@cprulelength}}%
          / $\eqe@ptsFmt{\csname ##1total\endcsname}%
          \cpSumryPts$\par}\thePartNames
        \fi
        \underbar{\makebox[0pt][l]{\cpSumryTotal}%
          \hspace{\eqe@cprulelength}}/
            $\eqe@ptsFmt{\the\count2\relax}\cpSumryPts$ \par
          \underbar{\makebox[0pt][l]{\cpSumryGrade}%
            \hspace{\eqe@cprulelength}}%
        \ifx\cp@Usefbox\fbox\medskip\fi
      \end{flushleft}\kern0pt
  \end{minipage}}\par
}
```

\cpSumrybypages The routine for building the summary box, where we list the statistics for each page.

```
\newcommand{\cpSumrybypages}{%
  \eqe@@SumryVert\cp@Usefbox{%
  \ifx\cp@Usefbox\fbox
    \def\cp@sumryWdth{\cp@SetSumryWidth-2\fboxrule-2\fboxsep}\else
  \def\cp@sumryWdth{\cp@SetSumryWidth}\fi
  \begin{minipage}[b]{\cp@sumryWdth}\kern0pt
    \begin{flushleft}
      \setlength{\eqetmplengtha}{\cprulelength}%
      \edef\eqe@cprulelength{\the\eqetmplengtha}%
      \csarg\ifx{NumberOfParts}\relax
        \else\medskip\count0=0\relax\count2=0\relax
          \cpSumryHeader\par\parskip\bigskipamount
          \ifsolutionsAtEnd
            \@ifundefined{eqExamQuesLastPage}{\def\NUMPAGES{0}}
              {\edef\NUMPAGES{\@nameuse{eqExamQuesLastPage}}}%
          \else
            \@ifundefined{eqExamLastPage}{\def\NUMPAGES{0}}
              {\edef\NUMPAGES{\@nameuse{eqExamLastPage}}}%
          \fi
            \@whilenum\count0<\NUMPAGES\relax\do{%
              \advance\count0by1\relax
              \@ifundefined{Page\the\count0total}{\count4=0\relax}
              {\count4=\@nameuse{Page\the\count0total}\relax}%
              \csarg\ifx{Page\the\count0spilltotal}\relax
              \else\advance\count4by\csname%
```

```
                Page\the\count0spilltotal\endcsname\relax\fi
              \advance\count2by\count4\relax\underbar{%
              \makebox[0pt][l]{\cpSumryPage~\the\count0}%
              \hspace*{\eqe@cprulelength}}/
                $\eqe@ptsFmt{\the\count4\relax}%
                \cpSumryPts$\csarg\ifx{Page\the\count0spilltotal}\relax
                \else\ $(\@nameuse{Page\the\count0spilltotal}%
                  \cpSumryPts+\@nameuse{Page\the\count0total}%
                  \cpSumryPts)$\fi\par
            }% while
          \fi
          \underbar{\makebox[0pt][l]{\cpSumryTotal}%
             \hspace{\eqe@cprulelength}}/
             $\eqe@ptsFmt{\the\count2\relax}\cpSumryPts$\par
          \underbar{\makebox[0pt][l]{\cpSumryGrade}%
             \hspace{\eqe@cprulelength}}%
          \ifx\cp@Usefbox\fbox\medskip\fi
      \end{flushleft}\kern0pt
  \end{minipage}}\par
}
```

If the author takes the coverpage option, \eqex@coverpage is set equal to \eqexcoverpagedesign, otherwise, it is set equal to \relax.

```
\AtBeginDocument{\eqex@coverpage}
```

13.4 Insert Points in Margins and Compute Page Totals

Here we attempt to place point values of a problem in the margins and to compute the page totals.

\probvalue This is a fundamental macro for keeping track of the points of the problem. It increments the counter eqpointvalue, which keeps a running total of the points of the current part of the exam, puts the value in the margins, and sets a mark, so that at the end of the page, we can compute the number of points on the current page. This macro is used in several situations, for example, in the problem environment, \manualcalcparts and in \autocaleparts. **Should not be redefined**.

Parameters

```
#1 = total points for this problem
#2 = 0 if total points, otherwise, #2 is the number of points each
     problem.
```

```
\def\@marktotalvalue{\global\let\eqe@innermarkpts\relax
    \if@bypasseqexamheading\else\def\eqe@marktxt{%
        \theeqpointvalue\@nameuse{eqExam}\theeq@numparts}%
    \ifx\endparts\endexercise@parts@tabular
        \xdef\eqe@innermarkpts{\noexpand\mark{\eqe@marktxt}}\else
        \mark{\eqe@marktxt}\fi\fi}
\def\probvalue#1#2{\addtocounter{eqpointvalue}{#1}%
    \ifdispl@yPoints\marginpoints{#1}{#2}\fi
    \@marktotalvalue}
```

`\widthtpboxes` is the width of the box in the margins that contains the points or totals. The design of the box keys off this width.

```
\newcommand{\widthtpboxes}{35pt}
```

`\marginboxdesign` is the basic box that encloses the points on the right, and the totals. This box may be redefined as desired, in which case, `\measurePtBoxHt` should be re-executed.

```
\newcommand{\marginboxdesign}[2][]{%
    \parbox{\widthtpboxes}{\tabcolsep=0pt\relax
        \begin{tabular}{|c|}\hline
            \vrule height15pt width0pt#1\\\hline
            \makebox[\widthtpboxes-2\fboxrule]{#2}\\\hline
        \end{tabular}%
    }%
}
```

`\eqleftmarginbox` This macro places the problem value in the left margin, can be redefined, if you dare.

Parameters

```
#1 = total points for this problem
#2 = 0 if total points, otherwise, #2 is the number of points each
     problem.
```

Currently, this macro is not used so I'll make it into verbatim text.

```
\newcommand{\eqleftmarginbox}[2]{\makebox[0pt][r]{%
    \setlength\tabcolsep{0pt}%
    \raisebox{-.5\height}[0pt][0pt]{%
        \marginboxdesign{\marginpointsboxtext{#1}{#2}}%
    }\hspace*{\marginparsep}\hspace*{\eqemargin}}%
}
```

Formatting the points

`\ptsLabel` `\eachLabel` `\pointsLabel` The following three convenience commands can be used to localize some of the strings to other languages.

```
\providecommand{\ptsLabel}[1]{\def\eqptsLabel{#1}}\ptsLabel{pts}
\providecommand{\ptLabel}[1]{\def\eqptLabel{#1}}\ptLabel{pt}
\newcommand{\eachLabel}[1]{\def\eqeachLabel{#1}}\eachLabel{ea.}
```

These next two are used by the `instructions` environment to display the points for the exam.

```
\newcommand{\pointsLabel}[1]{%
    \def\eqpointsLabel{#1}}\pointsLabel{points}
\newcommand{\pointLabel}[1]{%
    \def\eqpointLabel{#1}}\pointLabel{point}
```

pointsonleft or pointsonboth

\marginpointtext \leftmarginPtsTxt \leftmarginPtsEaTxt

Creates the text for \eqleftmargin to use. From the macro definition, if #2 is 0, then we write the points for the problem, else, we write the points each for the problem. \leftmarginPtsTxt is the formatting for the total points for the problem; while \leftmarginPtsEaTxt is the formatting for the {problem*}{<num>ea}-type problems.

\pointsAsText

We begin by giving the user a choice, typeset the points in math mode (the default) or typeset in the normal font.

```
\def\pointsAsText{\def\eqe@ptsFmt{\text}}
\let\eqe@ptsFmt\relax
\@onlypreamble\pointsAsText
```

\marginpointtext

determines, based on #2 whether the points are 'points each' or not, and calls either \leftmarginPtsTxt or \leftmarginPtsTxt, as appropriate.

```
\newcommand{\marginpointtext}[2]{\ifnum#2=0\leftmarginPtsTxt{#1}\else
    \leftmarginPtsEaTxt{#2}\fi
}
```

\leftmarginPtsTxt

displays points in the left margin: (#1$^{\text{pt}}$) or (#1$^{\text{pts}}$), where #1 is the number of points for this problem.

```
\newcommand{\leftmarginPtsTxt}[1]{(\small$\eqe@ptsFmt{#1}^{\text{%
    \ifnum#1=1\relax\eqptLabel\else\eqptsLabel\fi}}$)}
```

\leftmarginPtsEaTxt

displays 'points each' in the left margin: (#1$^{\text{pt}}_{\text{ea.}}$) or (#1$^{\text{pts}}_{\text{ea.}}$), where #1 is the number of points for each part of this problem.

```
\newcommand{\leftmarginPtsEaTxt}[1]{(\small$\eqe@ptsFmt{#1}_{\text{%
    \eqeachLabel}}^{\text{\ifnum#1=1\relax\eqptLabel\else
    \eqptsLabel\fi}}$)}
```

\eqleftmargin

Places the number of points (or points each) in the left margin. Can be redefined as desired. This macro is used when author chooses the pointsonleft or pointsonbothsides option. The text for the points is defined above, \marginpointtext.

```
\newcommand{\eqleftmargin}[2]{\makebox[0pt][r]{\marginpointtext{#1}{#2}%
    \setlength{\@tempdima}{\marginparsep+\eqemargin}%
    \hspace*{\@tempdima}}}
```

pointsonright or pointsonboth

\marginpointsboxtext

When the author selects pointsonright or pointsonbothsides, a box appears in the right margin containing problem totals, this is the text for the box.

```
\newcommand{\marginpointsboxtext}[2]{\small$\eqe@ptsFmt{#1}\,\text{%
    \ifnum#1=1\relax\eqptLabel\else\eqptsLabel\fi}$}
```

\eqrightmarginbox

When the author selects pointsonright or pointsonbothsides, a box appears in the right margin containing problem totals, this is the box that appears.

\insertPointsBoxPDF
\insertTotalsBoxPDF
These two commands are \leting them be either a form field (that the document author can fill in during online grading, or as \relax. It depends on whether the email option is taken or not.

```
\def\@insertPointsBoxPDF
{
    \def\fieldName{pointsgiven.\curr@quiz.page\thepage.%
        \theeqquestionnoi}%
    \calcTextField[\F\FHidden\BC{}\Q1\textColor{1 0 0 rg}]
        {\fieldName}{\widthtpboxes}{15pt}%
}
\def\@insertTotalsBoxPDF
{
    \def\fieldName{pagetotals.\curr@quiz.page\thepage}%
    \calcTextField[\F\FHidden\BC{}\Q1\textColor{1 0 0 rg}
        \AA{\AACalculate{AFSimple_Calculate("SUM",
            new Array("pointsgiven.\curr@quiz.page\thepage"));}}]
    {\fieldName}{\widthtpboxes}{15pt}%
}
\ifx\use@email y
    \let\insertPointsBoxPDF\@insertPointsBoxPDF
    \let\insertTotalsBoxPDF\@insertTotalsBoxPDF
\else
    \let\insertPointsBoxPDF\@empty
    \let\insertTotalsBoxPDF\@empty
\fi
```

The points box that appears in the left margin, may be re-defined, if you dare. At the bottom of the box goes the points for the problem, if the email option is used, in the top, a text field is inserted.

```
\newcommand{\eqrightmarginbox}[2]{\makebox[0pt][l]{%
        \setlength\tabcolsep{0pt}%
```

(2011/05/08) new calculation for the right point box.

```
        \setlength{\@tempdima}{\textwidth-\eqemargin+\marginparsep}%
        \hspace*{\@tempdima}%
        \raisebox{-.5\height}[0pt][0pt]{%
            \marginboxdesign[\insertPointsBoxPDF]%
                {\marginpointsboxtext{#1}{#2}}%
        }\hfil
    }%
}
```

At the end of each exam \end{exam} the part totals can be optionally displayed on the right. This is the box for doing that.

```
\newcommand{\eqeomarginboxright}[2]{\makebox[0pt][l]{%
        \setlength\tabcolsep{0pt}%
        \setlength{\@tempdima}{\textwidth+\marginparsep-\parindent}%
        \hspace*{\@tempdima}%
        \raisebox{-.5\height}[0pt][0pt]{%
            \marginboxdesign[\insertTotalsBoxPDF]%
```

```
                {\marginpointsboxtext{#1}{#2}}%
        }\hfil
    }%
}
```

At the end of each exam \end{exam} the part totals can be optionally displayed on the left. This is the box for doing that.

```
\newcommand{\eqeomarginboxleft}[2]{\noindent\makebox[0pt][r]{%
    \raisebox{-.5\height}[0pt][0pt]{%
        \marginboxdesign[\insertTotalsBoxPDF]%
            {\marginpointsboxtext{#1}{#2}}%
    }{\settowidth{\@tempdimb}{\eqe@hspannerPrb}%
    \setlength{\@tempdima}{\marginparsep+\eqemargin-\@tempdimb}%
    \hspace*{\@tempdima}}}\hfil
}
```

\measurePtBoxHt

We measure the height of our point/total boxes and store it in the length \eq@pointboxtotalheight, which is used to separate the boxes so they don't overlap.

```
\newcommand{\measurePtBoxHt}{\setbox\eq@pointbox\hbox{%
        \marginboxdesign{\marginpointsboxtext{00}{0}}%
    }\setlength\eq@pointboxtotalheight{.5\ht\eq@pointbox}%
    \xdef\halfHtPtBox{\the\eq@pointboxtotalheight}%
    \setlength\eq@pointboxtotalheight{\dp\eq@pointbox+\ht\eq@pointbox}%
    \xdef\totHtPtBox{\the\eq@pointboxtotalheight}%
    \xdef\dpPtBox{\the\dp\eq@pointbox}%
    \xdef\htPtBox{\the\ht\eq@pointbox}%
}
\measurePtBoxHt
```

A helper command to set both margin boxes.

```
\newcommand{\eqbothmargins}[2]{\eqleftmargin{#1}{#2}%
    \eqrightmarginbox{#1}{#2}}
```

The macro \probvalue, defined above, says

```
\mark{\theeqpointvalue\csname eqExam\endcsname\theeq@numparts}
```

so the \botmark contains the point value of this page and the part number for this page.

```
\def\lastparttotaled{0}
```

The botmark looks like 18\eqExam0, where 18 will be the total number of points accumulated for this exam part. We need to retrieve these number, I don't remember why I save them this way.

\parsetotals

\stripeqExam

The \parsetotals macro is called in \settotalsbox to retrieve the ongoing point values from \botmark, and returns two parameters, contained in \argi and \argiii. Whereas, \stripeqExam just retrieves the first parameter only.

```
\def\parsetotals#1\eqExam#2\end{\def\argi{#1}\def\argii{#2}}
\def\stripeqExam#1\eqExam{\def\argii{#1}}
```

\settotalsbox This is the box containing the page total, it may appear on the left or right side bottom corner.

```
\def\settotalsbox{%
    \expandafter\parsetotals\botmark\eqExam\end
    \ifx\argi\@empty\hfil
    \else\ifx\argii\@empty\hfil
        \else
            \expandafter\stripeqExam\argii
```

Set eqpointsthispage equal to \argi, which should be the accumulated total for this part of the test so far.

```
            \setcounter{eqpointsthispage}{\argi}%
```

This subtract eqpointsofar, which should be the total for this test part through the previous page. The difference is the number of points for this page.

```
            \addtocounter{eqpointsthispage}{-\value{eqpointsofar}}%
```

Now we set eqpointsofar to the new accumulated total \argi.

```
            \setcounter{eqpointsofar}{\argi}%
```

And save to the auxiliary file for later usage.

```
            \xdef\lastparttotaled{\argii}%
            \eqe@IWO\@auxout{\string\csarg\string
              \gdef{Page\thepage total}%
                {\theeqpointsthispage}}%
```

Finally, place the totals box.

```
            \totalsbox
        \fi
    \fi
}
```

\totalsboxtext This is the text that appears in the totals boxes

```
\newcommand\totalsboxtext{\small$
    \eqe@ptsFmt{\theeqpointsthispage}\,\text{%
    \ifnum\theeqpointsthispage=1\relax\eqptLabel\else
    \eqptsLabel\fi}$}
```

\eqevtranstotbox \eqevtranstotbox is a length that can be used to raise or lower the position of the total boxes in the left/right bottom corner. The default is 0pt.

```
\newlength\eqevtranstotbox
\setlength{\eqevtranstotbox}{0pt}
```

\totalsboxleft
\totalsboxright There are two totals boxes, one for the left and one for the right side. These can be redefined as desired.

```
\def\eqe@chkZeroTotals{\let\eqe@allow\eqe@YES
    \ifx\eqe@zeroTotalsAllowed\eqe@NO
    \ifnum\theeqpointsthispage=0 \let\eqe@allow\eqe@NO\fi\fi}
\newcommand{\totalsboxleft}{\eqe@chkZeroTotals
    \ifx\eqe@allow\eqe@YES
    \makebox[0pt][r]{\setlength\tabcolsep{0pt}%
        \raisebox{-\height+\eqevtranstotbox}[0pt][0pt]{%
```

```
            \marginboxdesign[\insertTotalsBoxPDF]{\totalsboxtext}%
        }\hspace*{\marginparsep}%
    }\fi\hfil
}
\newcommand{\totalsboxright}{\eqe@chkZeroTotals
    \ifx\eqe@allow\eqe@YES
    \makebox[0pt][l]{\setlength\tabcolsep{0pt}%
        \hspace*{\textwidth}\hspace*{\marginparsep}%
        \raisebox{-\height+\eqevtranstotbox}[0pt][0pt]{%
            \marginboxdesign[\insertTotalsBoxPDF]{\totalsboxtext}%
        }%
    }\fi\hfil
}
```

We begin the document by declaring the current part is part 0 with 0 points. This is the default, in case the author writes a document with no parts or points!

```
\ifeqe@nopoints\else\if@bypasseqexamheading\else
\AtBeginDocument{\mark{0\csname eqExam\endcsname0}}
\fi\fi
```

At the end of the document, we write out the number of parts for this test, and the names of the parts the author has given each part.

```
\AtEndDocument{%
    \eqe@IWO\@auxout{\string\csarg\string\gdef{NumberOfParts}%
        {\arabic{eq@numparts}}}%
    \eqe@IWO\@auxout{\string\csarg\string\gdef{thePartNames}%
        {\the\partNames}}%
    \clearpage\addtocounter{page}{-1}\writelastpage\relax
    \addtocounter{page}{1}%
}
```

`\theGrandTotal` When an exam has multiple parts, the total of each part is computed and the grand total is computed with `\theGrandTotal`.

```
\newcommand\theGrandTotal{%
    $\eqe@ptsFmt{\csname eqeGrandTotal\endcsname}$}
```

`\totalForPart` User access to the total for a exam environment, one argument, the name of the exam.

`\percentForPart` Attempts to calculate the percent of the total that the referenced exam (`#1`) contributes to the grand total.

```
\newcommand{\totalForPart}[1]{$\eqe@ptsFmt{\csname#1total\endcsname}$}
\def\tot@lForPart#1{\csname#1total\endcsname}
\newcommand{\percentForPart}[1]{$\eqe@ptsFmt
    {\csname#1percent\endcsname}$\%}
```

13.5 Computing Number of Points within a Segment of the Exam

We add some commands for calculating number of points in a segment of the exam. A segment being defined subset of consecutive problems within an exam.

We can define segments by placing markers (using `\placeMarkerHere`) between problems. We can then calculate the total number of points between markers.

The counter below is a scratch counter for making the calculations. We assume the `calc` package is loaded, it does the work for us.

```
\newcounter{markerCnt}
```

`\placeMarkerHere` Placed outside of any `problem`/`problem*` environment, this command takes one argument, the symbolic name for this marker. We write to the auxiliary file and define a command whose name is based on `#1`, and whose value is the cumulative total `\theeqpointvalue`.

```
\def\placeMarkerHere#1{%
    \eqe@IWO\@auxout{\string\csarg\string\gdef
        {#1SaveTotalHere}{\theeqpointvalue}}%
}
```

`\calcFromMarkers` Once the markers are in place, we can calculate the number of points defined between two such markers. The `\calcFromMarkers` takes three arguments, `#2` and `#3` are the symbolic names of the two markers. While, `#1` is an optional argument for formatting the calculation. The default is `\@markerTotalFmt`, defined below in `\markerTotalFmt`. The value of the total is `\themarkerCnt`.

```
\newcommand{\calcFromMarkers}[3][\@markerTotalFmt]{%
    \@ifundefined{#2SaveTotalHere}{}%
        {\@ifundefined{#3SaveTotalHere}{}{\setcounter{markerCnt}%
        {\@nameuse{#2SaveTotalHere}-\@nameuse{#3SaveTotalHere}}%
        \ifnum\value{markerCnt}<0\relax
        \setcounter{markerCnt}{-\value{markerCnt}}\fi#1}}%
}
```

`\markerTotalFmt` Used to set the global format of the marker totals. The value of the marker total is `\themarkerCnt`. The default follows the definition of `\markerTotalFmt`

```
\newcommand{\markerTotalFmt}[1]{\def\@markerTotalFmt{#1}}
\markerTotalFmt{\space(\themarkerCnt\space points)}
```

`\calcQsBtwnMarkers` `\Mrk1Start` `\Mrk1End` `\calcQsBtwnMarkers[`*Mrk2*`]{`*Mrk1*`}` does a number of things; primarily, it determines the range of the questions between the two marks. The names of the commands produced are all based in the first marker name *Mrk1*. `\`*Mrk1*`Start` is the first question number that follows the the placement of `\calcQsBtwnMarkers`. Similarly, `\`*Mrk1*`End` is the last question number between the two marks *Mrk1* and *Mrk2*.

`\Mrk1nQs` We also calculate `\`*Mrk1*`nQs`, the number of questions appearing between *Mrk1* and *Mrk2*.

```
\newcommand{\calcQsBtwnMarkers}[2][]{\setcounter{markerCnt}{0}%
    \def\eqe@argi{#1}%
```

`\`*Mrk1*`thisQnum` is the question number in effect at the point where the command `\calcQsBtwnMarkers` is inserted.

```
    \eqe@IWO\@auxout{\string\csarg\string\gdef
        {#2thisQnum}{\theeqquestionnoi}}%
```

\Mrk1`Start` is 0 if undefined.

```
    \@ifundefined{#2thisQnum}{\csarg\xdef{#2Start}{0}%
        \eqe@IWO\@auxout{\string\csarg\string\gdef
            {#2Start}{0}}%
    }{%
```

\Mrk1`Start` is the current question number plus 1.

```
        \setcounter{markerCnt}{\value{eqquestionnoi}+1}\csarg
        \xdef{#2Start}{\the\value{markerCnt}}%
        \eqe@IWO\@auxout{\string\csarg\string\gdef
            {#2Start}{\@nameuse{#2Start}}}%
    }
```

Using now both *Mrk1* and *Mrk2* we calculate the difference in the two. Result held in `markerCnt`. If there is no optional parameter, *Mrk2*, do not produce the \Mrk1`thisQnum`.

```
    \ifx\eqe@argi\@empty\else
    \@ifundefined{#2thisQnum}{\setcounter{markerCnt}{0}}%
        {\@ifundefined{#1thisQnum}{\setcounter{markerCnt}{0}}{%
        \setcounter{markerCnt}%
            {\@nameuse{#1thisQnum}-\@nameuse{#2thisQnum}}%
    }}%
    \fi
```

\Mrk1`nQs` is the number of questions that appear between *Mrk1* and *Mrk2*.

```
    \csarg\xdef{#2nQs}{\the\value{markerCnt}}%
    \eqe@IWO\@auxout{\string\csarg\string\gdef
    {#2nQs}{\the\value{markerCnt}}}%
```

\Mrk1`End` is 0 if undefined. If there is no optional parameter, *Mrk2*, do not produce the \Mrk1`End`

```
    \ifx\eqe@argi\@empty\else
        \@ifundefined{#1thisQnum}{\csarg\xdef{#2End}{0}%
          \eqe@IWO\@auxout{\string
                \csarg\string\gdef{#2End}{0}}%
        }{%
```

\Mrk1`End` is \Mrk2`thisQnum`.

```
        \csarg\xdef{#2End}{\@nameuse{#1thisQnum}}%
        \eqe@IWO\@auxout{\string\csarg\string\gdef
            {#2End}{\@nameuse{#2End}}}%
        }%
    \fi
}
```

`\markStartFor`
`\markEndtFor`
`\markNumQsFor`

We provide a user interface to the three macros defined above. The required parameter is a mark (a name that was used as the first argument of `\calcQsBtwnMarkers`).

```
\newcommand{\markStartFor}[1]{\@nameuse{#1Start}}
\newcommand{\markEndFor}[1]{\@nameuse{#1End}}
\newcommand{\markNumQsFor}[1]{\@nameuse{#1nQs}}
```

13.6 Useful Commands to Write Problems

13.6.1 The \placeAtxy command

\placeAtxy Use to place material within the solutions area that is visible to the student on the test document. Syntax:

```
\placeAtxy{x_dim}{y_dim}{content}
```

\placeAtxy goes immediately after \end{solution}

```
\newcommand{\placeAtxy}[3]{{%
    \par\nointerlineskip
    \ifdisplayworkarea
        \ifx\eq@insertverticalspace\eqe@YES
            \makebox[0pt][l]{\hspace*{-\parindent}\hspace*{#1}%
            \raisebox{#2}[0pt][0pt]{#3}}\fi
    \fi
}}
```

13.6.2 The workarea environment

As of this writing, 2012/12/09, this segment of code has been re-written. The splitsolution and panel environments have a new syntax. I've never gotten any feedback from user who use these environments, so I suspect the impact of these changes are minimal.

workarea
\workareasb A work area is an environment used when we display vertical space such as when the nosolutions and vspacewithsolns options are in effect. \workareasb is a save box used in the argument of lrbox to save the contents of the minipage.

```
\begin{workarea}[width]{depth}
...
\end{workarea}
```

```
\newsavebox{\workareasb}
```

\workareaVadj{⟨*skip*⟩} adjusts the vertical skip of the first line that appears in the workarea environment.

```
\def\workareaVadj#1{\bgroup\def\eqe@rgi{#1}\ifx\eqe@rgi\@empty
  \def\eqe@rgi{0pt}\fi\setlength{\@tempdima}{\eqe@rgi}%
  \xdef\workareaV@dj{\the\@tempdima}\egroup
} % dps1
\workareaVadj{3pt}
\def\workareaCmds#1{\def\@rgi{#1}\ifx\@rgi\@empty
  \let\work@reaCmds\relax\else
  \def\work@reaCmds{#1}}
\let\work@reaCmds\relax
\def\priorworkareaCmds#1{\def\@rgi{#1}\ifx\@rgi\@empty % dpsj6
  \let\priorWorkAreaCmds\relax\else
  \def\priorWorkAreaCmds{#1}\fi}
\let\priorWorkAreaCmds\relax
```

```
\newenvironment{workarea}[2][\linewidth]
{%
    \removelastparskip
    \edef\workareadepth{\if\currhideopt H0pt\else#2\fi}%
    \begin{lrbox}{\workareasb}\setlength{\eqetmplengtha}{#1}%
```

A change in support of `\fillTypeGrid`. When `\ifforceEqualCells` is true, we adjust the width of this `minipage` to conform to the shorter line produced by the grid.

```
    \ifx\makeVgrid\relax\else % dps2
      \ifforceEqualCells
        \ifx\fillLinesLineWidth\@empty\else
        \setlength{\eqetmplengtha}{\fillLinesLineWidth}\fi
      \fi
    \fi
    \begin{minipage}[b][\workareadepth][t]{\eqetmplengtha}%
```

Apply vertical skip `\workareaV@dj` here, set by `\workareaVadj`.

```
      \vglue\workareaV@dj
```

Adjust `\leftskip` when the problem is a lead-in.

```
      \@ifundefined{leadinIndentPrtSep}{}{\ifx\solutionparshape\@empty
        \else\leftskip\leadinIndentPrtSep\fi}\work@reaCmds
}{%
    \end{minipage}\end{lrbox}%
```

`4/5/11` Changed the logic here, so the `workarea` is available for the `vspacewithsolns` option.

```
    \ifeq@solutionsafter\else
    \par\ifdisplayworkarea
            \ifx\eq@insertverticalspace\eqe@YES
            \removelastparskip
            \vglue-\baselineskip
            \if\currhideopt H\else
                \noindent\strut\smash{\usebox{\workareasb}}%
            \fi\fi
    \fi\fi
}
```

13.6.3 The `splitsolution` environment

`splitsolution` This is a special solution environment designed for use with the `online` or `email` options, but it is compatible with all other options as well.

New Syntax

```
\begin{splitsolution}[width][depth]
\begin{panel}[l|r]
...
\end{panel}
\begin{solution}
...
\end{solution}
\end{splitsolution}
```

Old Syntax

```
\begin{splitsolution}{depth}
\begin{panel}[l|r]{width}
...
\end{panel}
\begin{solution}
...
\end{solution}
\end{splitsolution}
```

The `panel` writes its contents to a CUT file, then inputs it back in to get its dimensions, it places the contents in the box `\eqpanelbox`.

Below is an example of the use of the `splitsolution` environment (which is defined well below here).

```
\begin{problem}[5]
This is a question worth $5$ points.
\begin{splitsolution}
\begin{panel}\relax
\includegraphics[scale=.2]{fig1}
\end{panel}
\begin{solution}
This a really good  solution and worthy of a $5$ points.
\end{solution}
\end{splitsolution}
\end{problem}
```

Here, we include a graphic. We put it into a box, `\eqpanelbox` to get its dimensions. The `splitsolution` environment has one required argument, the total height of the solution. We then place the graphic in the `panel` environment, and write the solution in the usual way.

`splitsolution` We arrive at the `splitsolution` environment.

```
\def\splitsolution{%
    \@ifnextchar[{\splitsolutioni}%
        {\splitsolutioniii{\panelwidth}{\panelheight}}%
}
\def\splitsolutioni[#1]{%
    \@ifnextchar[{\splitsolutionii{#1}}
        {\splitsolutioniii{\panelwidth}{#1}}%
}
\def\splitsolutionii#1[#2]{%
    \splitsolutioniii{#1}{#2}%
}
\def\splitsolutioniii#1#2{%
    \@ifundefined{panel@write}{\newwrite\panel@write}{}%
    \gdef\ss@Argiii{#1}\gdef\ss@Argii{#2}%
```

Here, we `\let` `\solution` to `\@sssolution` and replace it with `\@ssSolution`

```
    \let\@sssolution\solution
    \let\end@sssolution\endsolution
    \let\solution\@ssSolution
    \let\endsolution\end@ssSolution
    \par\noindent\ignorespaces
```

The panel environment should appear next, just after the \begin{splitslution}

```
}
\def\endsplitsolution{%
    \ifeq@solutionsafter\solutionsafterSkip\fi % dpsj10
    \ifx\solutionparshape\@empty\else
        \advance\linewidth-\leadinIndentPrtSep\fi %%dps
    \edef\eqe@workwidth{\if\ss@Argi l\noexpand\ss@Argiii
        \else\noexpand\linewidth\fi}%
    \edef\eqe@workfill{\if\ss@Argi l\else\hfill\fi}%
    \ifeq@solutionsafter\else
      \begin{workarea}[\eqe@workwidth]{\eqedepth}%
        \let\panelGetDimen\relax%
        \eqe@workfill\input{panel\thepanel@cnt.cut}%
      \end{workarea}
    \fi\ifeq@solutionsafter\else\solutionsafterSkip\fi
}
```

\panelgap
\eqpanelbox
\panelwidth
\panelheight
panel@cnt

\panelgap is the separation between solution area and the panel. is a box to put things into, to measure their dimensions. The user accessible commands \panelwidth returns the width and \panelheight the height. There is a special counter panel@cnt to keep track of the panels generated.

```
\newcommand\panelgap{3pt}
\newsavebox{\eqpanelbox}
\newdimen\eqepanelwidth
\newdimen\eqepanelheight
```

\getDimSSPanel

The \getDimSSPanel is an internal command. It places it argument in the box \eqpanelbox and gets its dimensions. It save the dimensions as \eqepanelwidth and \eqepanelheight (the total height).

```
\newcommand{\getDimSSPanel}[1]{%
    \sbox{\eqpanelbox}{#1}%
    \eqepanelwidth=\wd\eqpanelbox
    \eqepanelheight=\ht\eqpanelbox
    \advance\eqepanelheight by\the\dp\eqpanelbox
}
```

\panelwidth
\panelheight

\panelwidth and \panelheight are aliases for the dimension registers. This is for compatibility with previous versions.

```
\def\panelwidth{\eqepanelwidth}
\def\panelheight{\eqepanelheight}
\newcounter{panel@cnt}
```

panel

environment goes inside the splitsolution environment. Permissible values of the first parameter are r and l, the default being l. This is the alignment parameter. The second argument is the width of the panel.

```
\def\panel{\def\eqe@panelir{\eqe@paneli[r]}%
```

```
    \@ifnextchar[{\eqe@paneli}
        {\@ifnextchar\relax{\expandafter\eqe@panelir\@gobble}%
            {\eqe@paneli[r]}}%
}
```

We continue the `panel` environment.

```
\def\eqe@paneli[#1]{%
    \gdef\ss@Argi{#1}\def\eqe@l{l}\def\eqe@r{r}%
    \ifx\ss@Argi\@empty\gdef\ss@Argi{r}\else
    \ifx\eqe@l\ss@Argi\else\ifx\eqe@r\ss@Argi\else
    \PackageError{eqexam}{Permissible arguments for panel are l and r}
    {Use l or r for the argument of panel.}\fi\fi\fi
    \stepcounter{panel@cnt}%
    \immediate\openout \panel@write panel\thepanel@cnt.cut
```

Manage the `h`, `H`, and global overrides.

```
    \if\currhideopt H%
        \eqe@IWO\panel@write{\vfill}%
        \immediate\closeout\panel@write
        \gdef\ss@Argii{0pt}%
    \else
        \if\currhideopt h\ifeq@solutionsafter\else
            \ifeq@globalshowsolutions\else
                \eqe@IWO\panel@write{\vfill}%
                \immediate\closeout\panel@write
                \gdef\ss@Argii{0pt}%
            \fi\fi
        \fi
    \fi
    \begingroup
```

Write the contents of this environment to the file `panel\thepanel@cnt.cut`.

```
    \let\verbatim@out\panel@write
```

We begin by writing `\panelGetDimen{%` to the CUT file

```
    {\lccode`C=`\%\lccode`P=`\{
        \lowercase{\eqe@IWO\verbatim@out{\string\panelGetDimen PC}}}%
```

Followed by the verbatim listing of the `panel` environment.

```
    \verbatimwrite
}
\def\endpanel{%
    \endverbatimwrite
```

After the verbatim write, we write `}` to close off the argument of `\panelGetDimen`.

```
    {\lccode`P=`\}\lowercase{\eqe@IWO\verbatim@out{P}}}%
    \immediate\closeout\panel@write
    \endgroup
```

The command `\eqePANEL` cut does all the work at the end of this environment. We first `\let` `\panelGetDimen` to `\getDimenSSPanel` and input the CUT file. This loads the panel contents into `\eqpanelbox`, and gets their dimensions. We set the value of \ss@Argiii which is the width parameter of the splitsolution environment. \mp@Width calculates the width.

```
    \gdef\eqePANELCUT{%
        \let\panelGetDimen\getDimSSPanel
        \input{panel\thepanel@cnt.cut}%
        \setlength{\eq@tmpdima}{\ss@Argiii}%
        \xdef\ss@Argiii{\the\eq@tmpdima}%
        \xdef\mp@Width{\ifeq@solutionsafter\linewidth
            \else\linewidth-\ss@Argiii-\panelgap\fi}%
    }\aftergroup\eqePANELCUT
}
```

\eqe@IW is a macro that either writes verbatim text (#1), or it gobbles the argument, depending on whether have solutionsafter, nosolutions, hidden or not hidden.

```
\long\def\eqe@IW#1{%
    \ifeq@solutionsafter
        \let\eqe@next\@empty
    \else
        \def\eqe@next{\eqe@IWO\verbatim@out{#1}}%
        \ifeq@nosolutions\else
            \if\currhideopt H%
                \gdef\ss@Argii{0pt}\let\eqe@next\@empty
            \else
                \if\currhideopt h%
                    \ifeq@globalshowsolutions\else
                        \gdef\ss@Argii{0pt}\let\eqe@next\@empty
                    \fi
                \fi
            \fi
        \fi
    \fi
    \eqe@next
}
```

The splitsolution environment \lets the solution environment to the internal @ssSolution environment. It performs various tasks then turns it over to the old solution environment, which has been \let to \@sssolution.

```
\newenvironment{@ssSolution}[1][\ss@Argii]%
{%
    \def\eqe@argi{#1}\ifx\eqe@argi\@empty\else
        \let\soln@keys@nLines\@empty
        \edef\panelheight{\the\panelheight}%
        \edef\panelwidth{\the\panelwidth}%
        \edef\temp@exp{\noexpand
            \setkeys*{soln@keys}{#1}}\temp@exp
        \ifx\soln@keys@nLines\@empty
            \edef\ss@Argii{\XKV@rm}%
        \else
            \@tempdima\wlVspace\relax
            \@tempdima=\soln@keys@nLines\@tempdima
            \edef\ss@Argii{\the\@tempdima}%
            \ifx\XKV@rm\@empty\else
                \if@equsedim\edef\ss@Argii{\XKV@rm}\fi\fi
        \fi
        \setlength{\eq@tmpdima}{\ss@Argii}%
%        \ifx\sameVspace\@empty\gdef\sameVspace{0pt}\fi
```

If the panelheight is greater than the requested height, we make the requested height equal to panelheight.

```
        \ifdim\eqepanelheight>\eq@tmpdima
            \eq@tmpdima\eqepanelheight\fi
        \xdef\ss@Argii{\the\eq@tmpdima}%
    \fi
```

We calculate the depth of the solutions; this is generally the values passed to use as the optional argument, and encapsuled by \ss@Argii.

```
    \xdef\eqedepth{%
        \ifvspacewithsolns\ss@Argii\else
            \ifeq@nosolutions\ss@Argii\else
            \ifeq@solutionsafter\ss@Argii\else0pt\fi\fi\fi}%
```

The width is either \linewidth or \linewidth-\ss@Argiii-\panelgap, where \ss@Argiii is the width of the panel.

```
    \xdef\mp@Width{\ifeq@solutionsafter\noexpand\linewidth\else
        \noexpand\linewidth-\ss@Argiii-\panelgap\fi}%
```

If its a left panel...

```
    \if\ss@Argi l%
        \ifeq@solutionsafter
            \def\eqe@lPanel{\parbox[b][\eqedepth][t]{\ss@Argiii}
```

Here, and elsewhere, we \let \panelGetDimen to \relax when we want to purely input the CUT file in the solution.

```
                {\let\panelGetDimen\relax
                \input{panel\thepanel@cnt.cut}\vfill}\hfill}%
            \def\eqe@rPanel{\@empty}%
        \else
```

For the left panel, we need to subtract \eqemargin

```
            \def\eqe@subleftgutter{-\string\eqemargin}%
            \def\eqe@lPanel{%
                \string\parbox[b][\string\eqedepth]{\ss@Argiii}%
                {\string\let\string\panelGetDimen\string\relax
                \string\input{panel\thepanel@cnt.cut}%
                \string\vfill}\string\hfill^^J}%
            \def\eqe@rPanel{\@empty}%
        \fi
    \else
```

If its a right panel...

```
        \ifeq@solutionsafter
```

```
            \def\eqe@lPanel{\@empty}%
            \def\eqe@rPanel{%
                \hfill\parbox[b][\eqedepth][t]{\ss@Argiii}
                {\let\panelGetDimen\relax
                \hfill\input{panel\thepanel@cnt.cut}\vfill}}%
        \else
```

For the right panel, we do not to subtract `\eqemargin` as above.

```
            \let\eqe@subleftgutter\@empty
            \def\eqe@lPanel{\@empty}%
            \def\eqe@rPanel{%
                \string\hfill\string\parbox[b][\string\eqedepth]%
                {\ss@Argiii}{\string\endgraf%
                    \string\noindent\string\hfill%
                    \string\let\string\panelGetDimen\string\relax
                    \string\input{panel\thepanel@cnt.cut}%
                    \string\vfill}}%
        \fi
    \fi
    \let\verbatim@out\ex@solns
    \par\ifeq@solutionsafter\smallskip\fi
```

Since everything is put a box, we set the `\linewidth`, and set `\solutionparshape` to `\@empty`.

```
    \ifx\solutionparshape\@empty\else
        \advance\linewidth-\leadinIndentPrtSep\fi %%dps
    \let\solutionparshape\@empty
    \noindent\minipage{\linewidth}%
    \if\ss@Argi l\noindent\parbox[b][\eqedepth][t]{\ss@Argiii}%
        {\vfill}\hfill\fi
    \minipage[b][\eqedepth][t]{\mp@Width}%
    {\lccode`C=`\%\lowercase{\eqe@IW{%
        \string\def\string\panelgap{\panelgap}%
        \string\setlength{\string\panelwidth}{\ss@Argiii}%
        \string\def\string\eqedepth
            {\ifeq@nosolutions\ss@Argii\else\ss@Argii\fi}C}}}%
    {\lccode`C=`\%\lowercase{\eqe@IW{%
            \string\strut\string\noindent%
%            \string\strut\par\string\nobreak\string\noindent%
                \string\hbox\space to\string\linewidth\bgroup^^J%
                \eqe@lPanel\string\minipage[b][\string\eqedepth][t]%
                {\string\linewidth\eqe@subleftgutter%
                    -\string\panelwidth-\string\panelgap}C
     }}}%
     \ifeq@solutionsafter
        \noindent\strut\hbox to\linewidth\bgroup
        \eqe@lPanel
        \minipage[b][\eqedepth][t]{\linewidth-\ss@Argiii-\panelgap}%
        \def\solutionsafterSkip{}%
     \fi
```

Here, we start the original definition of the solutions environment, which was \let to \@sssolution.

```
     \ifx\soln@keys@nLines\@empty
        \edef\passedss@Argii{\ss@Argii}\else
        \edef\passedss@Argii{\ss@Argii,nLines=\soln@keys@nLines}\fi
```

For a split solution, cannot allow align left

```
     \if\ss@Argi l\fillerLinesAlignDef\fi % dpsj10
```

I cannot get the filler lines to work when the anserkey is in effect. The next few lines locally turns off filler lines for this problem.

```
     \ifanswerkey % dpsj10
        \gdef\p@ssToFLs{\let\eq@insertverticalspace\eqe@NO
        \@eqlinedfillerfalse}\fi % dpsj10
    \eqe@flextendedfalse
    \expandafter\@sssolution\expandafter[\passedss@Argii]%
}{%
    \eqe@IW{\string\endminipage\eqe@rPanel\egroup}%
    \ifeq@solutionsafter\endminipage\eqe@rPanel\egroup\fi
    \end@sssolution
    \endminipage\endminipage
}
% Redefine the \cs{paragraph} command
%    \begin{macrocode}
\renewcommand{\paragraph}
    {\@startsection{paragraph}{4}{0pt}{12pt}{-3pt}{\bfseries}}
```

\defaultInstructions For the instructions environment, defined next, the default string for the instructions is \eq@default@Instructions, this command is defined using \defaultInstructions.

```
\newcommand{\defaultInstructions}[1]{\def\eq@default@Instructions{#1}}
\defaultInstructions{Instructions.}
```

instructions Each test, or a part of a test usually have instructions. This instruction environment is used in this purpose. Normally, the number of points for the part is displayed following the heading (the default is **Instructions.**). You can eliminate the total points from the instructions by taking the nosummarytotals option.

```
\newcommand{\beforeInstrSkip}{1ex}
\newcommand{\afterInstrSkip}{-0em}
\newcommand\hInstrSpace{\ }
\newcommand\styleInstr{\bfseries}
\newenvironment{instructions}[1][\eq@default@Instructions]{%
```

If \summaryPointTotal is zero, show no summary points.

```
    \@ifundefined{\thisexamlabel total}{\let\eq@nosummarytotals\eqe@YES}
    {\ifnum\summaryPointTotal=0 \let\eq@nosummarytotals\eqe@YES\fi}%
    \expandafter\def\expandafter\eq@argi\expandafter{#1}%
    \def\hsi{\hInstrSpace}%
```

We start a paragraph environment

```
    \@startsection{paragraph}{4}{\z@}%
    {\beforeInstrSkip\space\@plus1ex \@minus.2ex}%
    {\afterInstrSkip}{\normalfont\normalsize\styleInstr}*%
    {\textcolor{\@instructionsColor}{#1}\normalcolor%
    \ifx\eq@nosummarytotals\eqe@YES\else\ifx\eq@argi\@empty\else\hsi\fi
        {\normalfont\summaryTotalsTxt}\hsi\fi}\hskip-\lastskip
    \ifx\eq@nosummarytotals\eqe@YES\ifx\eq@argi\@empty\else\hsi\fi\fi
    \normalfont\normalsize\ignorespaces
}{\par\vskip\beforeInstrSkip\relax}
```

\summaryPointTotal
\summaryTotalsTxt
\nQuesInExam

\summaryPointTotalcontains the total points for the current exam; the command \summaryTotalsTxt defines its the formatting for the points. The command \nQuesInExam is the total number of questions in the exam.

```
\newcommand\summaryPointTotal{\csname\thisexamlabel total\endcsname}
\newcommand\summaryTotalsTxt{($\eqe@ptsFmt{\summaryPointTotal}\,\text{%
    \@ifundefined{\thisexamlabel total}{\eqpointsLabel}%
    {\ifnum\summaryPointTotal=1\relax\eqpointLabel\else
    \eqpointsLabel\fi}}$)}
\newcommand{\nQuesInExam}[1][\thisexamlabel]{%
    \def\eqe@argi{#1}%
    \ifx\thisexamlabel\@empty\ifx\eqe@argi\thisexamlabel
    \PackageError{eqexam}{The optional argument for
    \string\nQuesInExam\MessageBreak must be specified}{}\fi\fi
        \csname#1nQuestions\endcsname}
```

Make this definition for hyperref, so its anchors will be unique. Useful when there are multiple parts of the test.

```
\ifx\hyper@anchor\@undefined\else
    \renewcommand\theHeqquestionnoi
            {\curr@quiz.\theeqquestionnoi\eqe@fpmrk}
    \renewcommand\theHquizno{%
        \if\probstar*\curr@quiz.%
            \theeqquestionnoi.part\thepartno.\arabic{quizno}%
        \else
            \curr@quiz.%
                \theeqquestionnoi.\arabic{quizno}%
        \fi
    }
    \renewcommand\theHpartno{\curr@quiz.%
        \theeqquestionnoi.part\thepartno}
\fi
```

eqComments

Often, I want to make additional instructions between problems, you can use this comment environment.

```
\newcommand{\beforeCommentSkip}{1.25ex}
\newcommand{\afterCommentSkip}{-1ex}
\newcommand\hCommSpace{\ }
\newcommand\styleComm{\bfseries}
```

```
\newenvironment{eqComments}[1][\strut]{\removelastskip
    \def\eqe@argi{#1}\def\eqe@Strut{\strut}\def\hsc{\styleComm}%
    \@startsection{paragraph}{4}{\z@}%
    {\beforeCommentSkip\space\@plus1ex \@minus.2ex}%
    {\afterCommentSkip}{\normalfont\normalsize\styleComm}*%
    {\textcolor{\@eqCommentsColor}{#1}}%
    \ifx\eqe@argi\eqe@Strut\hskip\afterCommentSkip
    \else\ifx\eqe@argi\@empty\hskip\afterCommentSkip\else
    \hsc\strut\fi\fi\normalfont\normalsize
    \color{\@eqCommentsColorBody}\ignorespaces
}{\par\vskip\beforeCommentSkip\space\@plus1ex \@minus.2ex}
```

13.7 The exam Environment

Each part of the exam is enclosed in an exam environment. The environment is a customized version of the shortquiz environment.

\exambegdef Some definitions that are executed at the beginning of each exam environment.

```
\let\tb@beginexam@code\relax
\def\partialspillovertotals{0}
\def\exambegdef
{%
    \csname\thisexamlabel pagemark\endcsname
    \@ifundefined{partialtotalpg}{}{%
        \ifnum\partialtotalpg=\arabic{page}%
            {\count0=\partialspillovertotals
             \advance\count0by\partialtotaleoe
             \xdef\partialspillovertotals{\the\count0}%
             \eqe@IWO\@auxout{\string\csarg\string\gdef
                {Page\partialtotalpg spilltotal}{\partialtotaleoe}}%
            }%
        \fi
    }%
    \csarg\ifx{\thisexamlabel pageno}\relax
    \else
        \csarg\ifx{pagenofirstprob\thisexamlabel}\relax
        \else
            \csarg\ifnum{\thisexamlabel pageno}
                < \@nameuse{pagenofirstprob\thisexamlabel}%
            \else
                \csarg\ifx{\thisexamlabel pagemark}\relax
                \else
                    \ifnum\value{page}=%
                        \csname\thisexamlabel pageno\endcsname
                        \eqe@IWO\@auxout{\string\csarg\string\gdef
                            {\thisexamlabel pagemark}{\string\newpage}}%
                    \fi
                \fi
            \fi
        \fi
    \fi
    \setcounter{eqquestionnoi}{0}\setcounter{eqpointvalue}{0}%
    \setcounter{eqpointsofar}{0}\setcounter{eqpointsthispage}{0}%
    \setcounter{eq@count}{0}%
```

We wrote `\begin{eqequestions}` to the top of the solutions file (`\jobname.sol`.

```
    \writeBeginEqeQuestions
```

```
    \label{\thisexamlabel PageBegin}%
    \eqe@IWO\@auxout{\string\csarg\string\gdef
    {\thisexamlabel pageno}{\thepage}}%
%    \let\sq@priorhook\@empty
    \ifeqfortextbook
        \global\examenvtrue\tb@beginexam@code
    \fi
}
```

`\examenddef` Some definitions that are executed at the end of each exam environment. We place a totals box to report the total since the last page.

```
\def\tb@insmargmark{\ifisinstred\ifismarginans
    \insMidMarg{\mark{}}\fi\fi}
\def\eqe@adjForSepRule{\if\eq@parttotals\eqe@YES
    \eq@pointboxtotalheight\halfHtPtBox\relax
    \advance\eq@pointboxtotalheight\dpPtBox\relax
    \advance\eq@pointboxtotalheight\pointsmarginparpush
    \vspace*{\eq@pointboxtotalheight}\fi}
\def\eqe@afterexamsepcode{%
    \if\eq@parttotals\eqe@YES
        \@actionsAtPageBreak{\global\let\@spacetobreak1}%
            {\global\let\@spacetobreak0}%
        \ifx\@spacetobreak0\relax
            \bgroup\@tempdima=\pagetotal
            \advance\@tempdima\eq@pointboxtotalheight
            \ifdim\@tempdima>\pagegoal\aftergroup\newpage
            \else\ifnum\arabic{eq@count}>0\relax
                \removelastskip\vskip6pt\kern0pt
                \if\@reportpoints\eqe@One\else\@checkSpacing{0}\fi
                    \textcolor{\endexamtotal@color}%
                    {\eqeomarginbox{\arabic{eq@count}}{0}}\fi
                \ifx\eqx@separationrule\eqe@YES
                    \separationrule\eqe@adjForSepRule\fi
            \fi\egroup
        \fi
    \else\ifx\eqx@separationrule\eqe@YES
        \separationrule\eqe@adjForSepRule\fi
    \fi
}
\def\examenddef
{%
    \global\let\partialtotaleoe\relax
    \global\let\partialtotalpg\relax
    \global\let\afterexamsepcode\relax
    \csarg\ifx{NumberOfParts}\relax
    \else
        \ifnum\value{eq@numparts}<\NumberOfParts
            \setcounter{eq@count}{\value{eqpointvalue}}%
            \addtocounter{eq@count}{-\value{eqpointsofar}}%
            \xdef\partialtotaleoe{\arabic{eq@count}}%
            \xdef\partialtotalpg{\arabic{page}}%
```

See if there is enough room at the bottom of the page to place the end of exam totals and to start a new exam, if any. If not, forget it, and start a new page.

```
            \global\let\afterexamsepcode\eqe@afterexamsepcode
        \fi
    \fi
    \ifeqfortextbook\global\examenvfalse\fi
```

(2011/05/08) Just before the file is closed and input, we write the end of the eqequestions environment, \end{eqequestions}.

```
    \writeEndEqeQuestions
```

```
    \writetotalstoaux
    \addtocounter{page}{-1}%
    \writelastpage[\thisexamlabel]\addtocounter{page}{1}%
    \ifeqfortextbook\tb@insmargmark\fi
}
\def\@actionsAtPageBreak#1#2{%
    \bgroup\@tempdima\pagegoal\advance\@tempdima-\pagetotal
    \@tempdimb\@fvsizeskip\vsize
    \ifdim\@tempdima < \@tempdimb #1\else #2\fi\egroup
}
```

\separationrule For an exam with multiple parts, a separation rule is created, unless absorbed into a page break. The command \separationrule defines this separation rule, it can be redefined as desired.

```
\newcommand{\separationrule}{\makebox[\linewidth-\eqemargin][c]%
    {\rule{.67\linewidth}{.4pt}}}
```

(2011/05/08) This is a new environment that makes an exam into a list of problems. This is an attempt to expand the use of eqexam to LaTeX documents. We give control over the page layout so an eqexam document can be used within a textbook.

```
\def\eqe@hspannerPrb{\ }
\@ifundefined{ifwithinsoldoc}{\newif\ifwithinsoldoc\withinsoldocfalse}{}
\newcommand{\eqequestopsep}[1]{\def\eqeques@topsep{#1}}
\providecommand{\eqequesparsep}[1]{\def\eqeques@parsep{#1}}
\newcommand{\eqequesitemsep}[1]{\def\eqeques@itemsep{#1}}
\newcommand{\eqequeslistparindent}[1]{\def\eqeques@listparindent{#1}}
\eqequestopsep{0pt}
\eqequesparsep{0pt}
\eqequesitemsep{0pt}
```

```
\eqequeslistparindent{0pt}
\newif\iffirstitem
\def\eqeSolnItemMngt{\iffirstitem\global\firstitemfalse
    \let\eqe@next\@empty\else\let\eqe@next\item\fi\eqe@next\relax}
\newenvironment{eqequestions}{%
    \begin{list}{}{%
    \ifwithinsoldoc\let\solnItemMngt\eqeSolnItemMngt\fi
    \setlength{\labelwidth}{\eqemargin}%
    \setlength{\parsep}{\eqeques@parsep}%
    \setlength{\itemsep}{\eqeques@itemsep}%
    \setlength{\topsep}{\eqeques@topsep}%
    \setlength{\itemindent}{0pt}%
    \setlength{\listparindent}{\eqeques@listparindent}%dps%
    \ifwithinsoldoc\settowidth{\labelsep}{\eqe@hspannerSoln}\else
    \settowidth{\labelsep}{\eqe@hspannerPrb}\fi
    \setlength{\leftmargin}{\labelwidth}%
    }\ifwithinsoldoc\global\firstitemtrue\fi\item\relax}{\end{list}}
```

exam Each part of the exam is enclosed in an exam environment. The one required parameter is the name of the part, for example, ‘Part1’, ‘Part2’. These should be one word, no white spaces, just letters and possibly numbers.

```
\def\setDefaultfvsizeskip#1{\def\default@fvsizeskip{#1}%
    \def\@fvsizeskip{#1}}
\def\default@fvsizeskip{.3}
\edef\@fvsizeskip{\default@fvsizeskip}
\newcommand{\fvsizeskip}[1]{\def\@fvsizeskip{#1}}
\def\autoExamName{exam\the\value{eq@numparts}}
\def\nNumberOfP@rts{\csname NumberOfParts\endcsname}
\abovesqskip{}
\let\eqeWrtExamTitleToSolns\eqe@YES
\def\wrtExamTitleInSolns{\let\eqeWrtExamTitleToSolns\eqe@YES}
\def\noExamTitleInSolns{\let\eqeWrtExamTitleToSolns\eqe@NO}
\let\thisexamlabel\@empty
\let\isInExamEnv\eqe@NO
```

\RecordThisExamOff causes eqexam to place \iffalse and \fi around the solutions to this exam in the solution file. This makes the solution results of this exam invisible. \RecordThisExamOff sets an internal switch \ifDoNotRecordThisExam to true. This switch is set back to false at the end of the environment.

```
\newif \ifDoNotRecordThisExam \DoNotRecordThisExamfalse % dpsf02
\def\RecordThisExamOff{\DoNotRecordThisExamtrue} % dpsf02
\def\bIFFalseWrtSolns{\writeT@SolnFile{\protect\iffalse^^J}}
\def\eIFFalseWrtSolns{\writeT@SolnFile{\protect\fi^^J}}
```

The beginning of the exam environment.

```
\newenvironment{exam}[2][]
{%
  \ifDoNotRecordThisExam % dpsf02
    \expandafter\bIFFalseWrtSolns\fi
    \makeRoomForProb{\@fvsizeskip\textheight}{0}%
    \let\isInExamEnv\eqe@YES
    \stepcounter{eq@numparts}%
```

If #2 is empty, use \autoExamName

```
    \def\eqexamargii{#2}\ifx\eqexamargii\@empty
        \edef\eqexamargii{\autoExamName}\fi
    \xdef\thisexamlabel{\eqexamargii}\xdef\curr@quiz{\eqexamargii}%
    \def\eqexamargi{#1}\ifx\eqexamargi\@empty
        \edef\eqexamargi{\eqexamargii}\fi
    \expandafter\gdef\expandafter\thisUFexamlabel
    \expandafter{\eqexamargi}%
    \edef\eq@tmp{\the\partNames\string\\{\eqexamargii}}%
    \global\partNames=\expandafter{\eq@tmp}%
    \csarg\ifx{NumberOfParts}\relax\else
\ifx\eqeWrtExamTitleToSolns\eqe@YES
    \ifnum\nNumberOfP@rts=1\relax\else
    \def\eqe@argi{#1}\ifx\eqe@argi\@empty
        \eqe@writetoSolns{\eqexamargii}\eqe@writetoAux{\string
            \csarg\string\gdef{userFriendly\eqexamargii}{\eqexamargii}}
        \else
        \eqe@writetoSolns{#1}\eqe@writetoAux{\string
            \csarg\string\gdef{userFriendly#2}{#1}}
        \fi
    \fi\fi
\fi
    \exambegdef
    \edef\temp@Exp{\noexpand\shortquiz\sqstar[\eqexamargii]}\temp@Exp
%    \expandafter\shortquiz\sqstar[\eqexamargii]%
}{%
    \examenddef
```

```
    \vskip\eqeques@parsep\relax\kern0pt %dps88
    \endshortquiz
    \aftergroup\afterexamsepcode
    \par\penalty-100\vskip0pt
    \ifDoNotRecordThisExam % dpsf02
      \expandafter\eIFFalseWrtSolns\fi
  \global\DoNotRecordThisExamfalse
}
```

\EQEcalculateAllTotals

(4/22/11) Added the command \EQEcalculateAllTotals. The command is executed as part of the \maketitle command. If \maketitle is not used for some reason \EQEcalculateAllTotals can be executed just after \begin{document}.

```
\newcommand{\EQEcalculateAllTotals}{%
    \begingroup
```

We calculate the grand total of all the parts of the exam environments, and we define \eqeGrandTotal, which contains the total.

```
        \count\z@=0\relax
        \def\\##1{\csarg\ifx{##1total}\relax\else
            \advance\count\z@\csname##1total\endcsname
```

`\thePartNames` list all named exam environments in the document, e.g.,

```
\\{Part1}\\{Part2}...\\{LastPart}
```

```
        \fi}\csname thePartNames\endcsname
        \xdef\eqeGrandTotal{\the\count\z@}%
        \ifnum\eqeGrandTotal=0 \else
```

If there is a nonzero grandtotal, we move on to calculate the percentages.

```
        \def\\##1{\eqe@calc@percent{##1}}%
        \csname thePartNames\endcsname\fi
    \endgroup
}
```

`\eqe@calc@percent` We go through the parts listed in `\thePartNames` and create a calculation of the percentage for that part, and leave it in `\csname#1percent\endcsname`, which can be accessed through the `\percentForPart` command, for example `\percentForPart{<part_name>}` might expand to `45.6%`.

`\nPctDecPts` The number of decimal points to carry in the representation of the percentage.

```
\newcommand{\nPctDecPts}{1}
\def\eqe@calc@percent#1{\@ifundefined{#1total}{%
        \csarg\gdef{#1percent}{??}}{%
```

If the fp package is not loaded, we use register arithmetic, percentages are truncated to integers.

```
        \csarg\ifx{FPdiv}\relax
        \count2=\tot@lForPart{#1}%
        \edef\expGT{\csname eqeGrandTotal\endcsname}%
        \multiply\count2by100\relax\divide\count2by\expGT\relax
        \csarg\xdef{#1percent}{\the\count2 }\else
```

If the fp package is loaded, we use this package to calculate the percentage, accurate to one decimal place.

```
        \FPdiv{\eqe@pForPart}{\csname#1total\endcsname}%
            {\csname eqeGrandTotal\endcsname}%
        \FPmul{\eqe@pForPart}{\eqe@pForPart}{100}%
        \FPround{\eqe@pForPart}{\eqe@pForPart}{\nPctDecPts}%
        \csarg\xdef{#1percent}{\eqe@pForPart}\fi
    }%
}
```

```
\def\writetotalstoaux{\eqe@IWO\@auxout{\string
        \csarg\string\gdef{\thisexamlabel total}{\theeqpointvalue}}%
    \eqe@IWO\@auxout{\string\csarg
        \string\gdef{\thisexamlabel nQuestions}{\theeqquestionnoi}}%
}
\newcommand{\writelastpage}[1][]{\def\eqe@argi{#1}%
   \ifx\eqe@argi\@empty\else\label{#1PageEnd}\fi
   \eqe@IWO\@auxout{\string\csarg
        \string\gdef{eqExamLastPage}{\arabic{page}}}%
}
\def\exlabel{}
\def\sqlabel{}
\def\exsolafter{\textit{Solution}:}
\def\sqsolafter{\textit{Solution}:}
```

The exercise labels in the body of the text. (2015/02/27) changed the command `\exlabelformatwp` to incorporate other parameters.

```
%\def\exlabelformat{\textbf{\theeqquestionnoi.\ }}
\def\exlabelformat{\textbf{%
    \theeqquestionnoi\eqe@decPointPrb\eqe@hspannerPrb}}
\def\exlabelformatwp{\exlabelformat}
```

The exercise labels for solutions at the end of the document

```
\def\exsllabelformat
    {\string\makebox[0pt][r]{\string\textbf{%
        \theeqquestionnoi\eqe@decPointPrb\eqe@hspannerSoln}}}
\def\exsllabelformatwp
     {\string\makebox[0pt][r]{\string\textbf{%
        \theeqquestionnoi\eqe@decPointPrb\eqe@hspannerSoln}}%
        (\thepartno)\eqe@hspannerSoln}
\ifanswerkey
    \def\exrtnlabelformat{}
    \def\exrtnlabelformatwp{}
    \def\eq@sqslrtnlabel{}
\else
    \def\exrtnlabelformat{$\square$}
    \def\exrtnlabelformatwp{$\square$}
    \def\eq@sqslrtnlabel{$\square$}
\fi
\def\sqslrtnlabel{\eq@sqslrtnlabel}
```

(2010/08/21) Enable some localizations of strings

```
\newcommand{\exsectitletext}{Solutions to \webtitle}
\def\exsectitle{\normalsize\exsectitletext}
%\def\exsectitle{\normalsize\hspace*
%    {-\oddsidemargin}\exsectitletext}
\@ifpackageloaded{exerquiz}{%{Solutions to \websubject}
    \renewcommand{\exsecrunhead}{}}{\newcommand{\exsecrunhead}{}}
%\providecommand{\exsecrunhead}{Solutions to \websubject}%
\def\eq@sqslsectitle{}
\def\eq@sqslsecrunhead{}
\def\eq@sqsllabel{{\string\llap{\string\textbf{\theeqquestionnoi.\ }}}}
\def\eq@sqlabel{}
\let\include@quizsolutions\relax
\let\solnhspace\@empty
```

13.8 `problem` Environments

A single question is posed with the `problem` environment, and a question with multiple parts with the `problem*` environment.

`\fillin` This macro is used for fill-in type questions. The first argument is the length of the underline blank to leave to fill-in, the second argument is the correct answer.

```
\newcommand{\optsFillIn}[1]{\def\eqe@optsFillIn{#1}}
\let\eqe@optsFillIn\@empty
\newcommand{\fillin}[3][u]{%
    \ifx#1u\let\@fillinFmt\underbar
    \else\ifx#1b\let\@fillinFmt\relax
    \else\let\@fillinFmt\relax\fi\fi
    \ifeq@proofing
        \@fillinFmt{\makebox[#2]{%
            \strut\hfil\bfseries\color{red}#3\hfil}}%
    \else
        \@fillinFmt{\makebox[#2]{\strut\hfil}}%
        \@ifundefined{@quiz}{}{%
            \if\eq@online\eqe@YES\relax
                \ifeq@nosolutions
                    \ifeq@solutionsafter\else
                        \ifx\eq@insertverticalspace\eqe@YES\relax
                            \stepcounter{@cntfillin}%
                            \edef\fieldName{%
                                \if\probstar*eqexam.\curr@quiz.fillin.%
                                    \theeqquestionnoi.part\thepartno.%
                                    fi\the@cntfillin%
                                \else
                                    eqexam.\curr@quiz.fillin.%
                                    \theeqquestionnoi.fi\the@cntfillin%
                                \fi
                            }\makebox[0pt][r]{\textField[\BC{}
                                \presets{\eqe@optsFillIn}]{%
                                \fieldName}{#2}{11bp}}%
                        \fi
                    \fi
                \fi
            \fi
        }%
    \fi\space\ignorespaces}
```

`\TF` A specialized version of `\fillin` for True/False questions.

```
\newcommand\defaultTFwidth{30pt}
\newcommand\TF[2][\defaultTFwidth]{%
    \def\eqe@next{\fillin{#1}{#2}}%
    \ifdim\eq@extralabelsep=0pt\relax\else
        \if\probstar*\relax\if\exerwparts@cols0
            \def\eqe@next{\makebox[0pt][r]{%
                \fillin{#1}{#2}}\ignorespaces}%
    \fi\fi\fi
\eqe@next}
\def\fillinWidth#1{%
    \if\probstar*
```

```
        \settowidth{\eq@tmplengthA}{\normalfont\ }%
        \addtolength{\eq@tmplengthA}{#1}%
        \edef\eq@extralabelsep{\the\eq@tmplengthA}%
    \fi
}
\let\fillInFormatDefault\@empty
```

\Do<num> The following commands supports the optional argument \Do<num>. When I teach senior or graduate-level classes, I often give a problem with multiple parts (each of equal value) and ask them to "do 3 of the following 5" parts.

```
\def\eqe@DoWarning#1{\PackageWarning{eqexam}{\string#1\space
    requires the first argument\MessageBreak of problem* to be <num>ea}}
\def\eqe@DoNum{\textbf{??}\eqe@DoWarning{\DoNum}}
\def\eqe@nDoNum{\textbf{??}\eqe@DoWarning{\nDoNum}}
\let\DoNum\eqe@DoNum
\let\nDoNum\eqe@nDoNum
\def\makeDoNum#1{\xdef\nDoNum{#1}%
    \def\ifc@sewrap{\ifcase#1??\or}%
    \xdef\DoNum{\expandafter\ifc@sewrap\eqe@wordNums\else
    \eqe@wordNumbsError\fi}}
\def\makeOutOfNum#1{\xdef\nOutOfNum{#1}%
    \def\ifc@sewrap{\ifcase#1??\or}%
    \xdef\OutOfNum{\expandafter\ifc@sewrap\eqe@wordNums\else
    \eqe@wordNumbsError\fi}}
\def\eqe@OutOfNum{\textbf{??}\eqe@DoWarning{\OutOfNum}}
\def\eqe@nOutOfNum{\textbf{??}\eqe@DoWarning{\nOutOfNum}}
\let\OutOfNum\eqe@OutOfNum
\let\nOutOfNum\eqe@nOutOfNum
```

\eqe@wordNums is used to typeset the English word for the numbers (1–10). This command may be redefined to other languages.

```
\newcommand{\eqe@wordNums}{one\or two\or three\or
    four\or five\or six\or seven\or eight\or nine\or ten}
\newcommand{\eqe@wordNumbsError}{\noexpand\PackageError{eqexam}%
    {Number out of range, 1--10}%
    {Use a smaller number, or redefine the command
    \string\eqe@wordNums.}}
```

Added two hooks \priorPNPAction and \postPNPAction to \makeRoomForProb to allow some defined actions before and after a page break generated by \makeRoomForProb. . Examples of usage are found in \vspaceFillerLines and \eq@linesXPgs.

```
\let\priorPNPAction\relax % dps1
\let\postPNPAction\relax
\def\eqe@mkRoomPgBrk{\priorPNPAction\newpage\postPNPAction}
\def\makeRoomForProb#1#2{\par %\endgraf % dps 11/11/10
    \bgroup\@nobreakfalse\addpenalty{-500}%
    \setlength{\@tempdimb}{#1}%
    \@tempdima \pagegoal \advance \@tempdima -\pagetotal
    \ifdim \@tempdima<\@tempdimb\ifnum\col@number>\@ne\columnbreak
        \else\aftergroup\eqe@mkRoomPgBrk\fi\fi\egroup
    \ifnum\@reportpoints>1
        \ifx\eqe@prevProbZero\eqe@YES
            \if\eqe@isPts0\else\vskip-\halfHtPtBox\relax\fi
        \else
            \ifl@stDispl@yPoints
            \if\eqe@isPts0\@checkSpacing{1}\else
               \if#21 \@checkSpacing{0}\fi
            \fi\fi
        \fi
    \fi
}
```

\emitMessageNearBottom

The syntax for this command is

```
\emitMessageNearBottom*[vspace]{msg}
```

If there is less than vspace remaining on the page, a message, msg, is emitted. Then the optional * appears, a \newpage is also emitted just after the msg and \insertContAnnot is expanded just after \newpage.

```
\newcommand{\emitMessageNearBottom}{\@ifstar
        {\def\eqe@emnb{\ifnum\col@number>\@ne\columnbreak
            \else\newpage\fi\insertContAnnot}\eq@emitMessageNearBottom}
        {\let\eqe@emnb\relax\eq@emitMessageNearBottom}}
\newcommand{\eq@emitMessageNearBottom}[2][2\wlVspace]{%
    \eq@@emitMessageNearBottom{#1}{#2}}
\def\eq@@emitMessageNearBottom#1#2{\par
    \bgroup\@nobreakfalse\addpenalty{-500}%
    \setlength{\@tempdimb}{#1}%
    \@tempdima \pagegoal \advance \@tempdima -\pagetotal
    \ifdim \@tempdima<\@tempdimb #2 \eqe@emnb
    \xdef\eq@currProbStartPage{0}\fi\egroup} % dps4
```

\getSpaceLeftOnPage
\amtSpaceLeftOnPage

\getSpaceLeftOnPage calculates the amount of space left on the current page. It saves the calculation in the text macro \amtSpaceLeftOnPage.

```
\newcommand{\getSpaceLeftOnPage}{\par\bgroup
    \@tempdima \pagegoal \advance \@tempdima -\pagetotal
    \xdef\amtSpaceLeftOnPage{\the\@tempdima}\egroup}
```

\promoteNewPage

A simple variation on \makeRoomForProb designed for user use.

```
\newcommand{\pnpDflt}{\@fvsizeskip\textheight}
\newcommand{\promoteNewPage}[1][\pnpDflt]{%
    \makeRoomForProb{#1}{0}}
```

\pointsmarginparpush
\@checkSpacing

Used in \makeRoomForProb and elsewhere. \pointsmarginparpush adds a little more separation between point boxes. \@checkSpacing is the algorithm for adjusting the vertical spaces between problems when points are on the right.

```
\def\pointsmarginparpush{3pt}
```

```
\def\@checkSpacing#1{\bgroup\ifinner\else
    \@tempdima\lastPageTotal
    \@tempdimb\pagetotal
```

If \lastPageTotal is less than \pagetotal, continue

```
    \ifdim\@tempdima < \@tempdimb
```

Compute \pagetotal - \lastPageTotal

```
        \advance\@tempdimb by-\@tempdima
        \ifdim\@tempdimb < \eq@pointboxtotalheight
        \if#11%
            \eq@pointboxtotalheight\halfHtPtBox\relax
            \advance\eq@pointboxtotalheight\dpPtBox\relax
        \fi
            \@tempdima=\eq@pointboxtotalheight
            \advance\@tempdima\pointsmarginparpush\relax
            \advance\@tempdima by-\@tempdimb
            \vspace*{\@tempdima}%
         \fi
    \fi\fi
\egroup}
\def\@checkSpacingi{\bgroup
    \@tempdima = \lastPageTotal
    \@tempdimb = \pagetotal
    \ifdim\@tempdima < \@tempdimb
        \advance\@tempdimb by-\@tempdima
        \eq@pointboxtotalheight\halfHtPtBox\relax
        \advance\eq@pointboxtotalheight\dpPtBox\relax
        \ifdim\@tempdimb < \eq@pointboxtotalheight
            \@tempdima=\eq@pointboxtotalheight
            \advance\@tempdima\pointsmarginparpush\relax
            \advance\@tempdima by-\@tempdimb
            \vspace*{\@tempdima}%
         \fi
    \fi
\egroup}
```

\default@nbaselineskip
\nbaselineskip

is the default number of \baselineskips needed to place a new problem. While \nbaselineskip is the number of \baselineskips needed for a new problem (or problem*).

```
\newcommand{\setDefaultnbaselineskip}[1]{\def\default@nbaselineskip{#1}}
\setDefaultnbaselineskip{6}
\def\nbaselineskip#1{\def\@nbaselineskip{#1}}
\def\nbaselineskipReset{\edef\@nbaselineskip{\default@nbaselineskip}}
\nbaselineskipReset
```

problem

The problem is used to pose a single—non-multi-part—question. The optional argument is the number of points for this problem.

```
\def\eqe@gobbletoend#1\end{}
\def\eqe@grabarg#1\end{\def\numpoints{#1}}
\def\eqe@isPts{1}
```

We reset some exerquiz parameters for eqexam

```
\aboveexskip{3pt}\belowexskip{3pt}
\partstopsep{3pt}\partsitemsep{3pt}\partsparsep{0pt}
\rowsepDefault{3pt}\partstabcolsep{1.5pt}%
\partstabtopsep{3pt}\partstabrowsep{3pt}
\renewcommand\belowexsolnskip{{}}%
\let\isProbEnv\eqe@NO
\let\topofprobhook\relax
```

Normally, you don't leave a blank line between the beginning of problem and the statement of problem. If you're in the habit of doing that, we can absorb that blank space. See similar command above the definition of problem* for more information.

```
\def\eqe@p@gobnxtp@rDef{\@ifnextchar\par{\eqe@p@gobtop@rnext}{}}
\def\eqe@p@gobtop@rnext#1\par{}
\def\applyparfixesp{\let\eqe@p@gobnxtp@r\eqe@p@gobnxtp@rDef}
%\def\cancelparfixesp{\let\eqe@p@gobnxtp@r\relax}
\def\cancelparfixesp{\let\eqe@p@gobnxtp@r\ignorespaces}
```

The problem environment now begins.

```
\newcommand{\problem}[1][]{\lowercase{\def\@rgi{#1}}%
```

Support for h and H argument when it is the first argument.

```
    \ifx\@rgi\eqe@h
      \def\hidden@ttr{[#1]}\let\numpoints\@empty\else
      \let\hidden@ttr\relax\def\numpoints{#1}\fi
    \problem@cont}
\newcommand{\problem@cont}[1][]{\let\isProbEnv\eqe@YES
  \def\@rgi{#1}\ifx\hidden@ttr\relax % assume a number or empty
    \ifx\@rgi\@empty\else\def\hidden@ttr{[#1]}\fi\fi
    \if\eqe@isPts0\global\let\eqe@prevProbZero\eqe@YES\else
        \global\let\eqe@prevProbZero\eqe@NO\fi
    \global\thereissolutionfalse
    \def\numpointsEmpty{0}%
    \ifx\numpoints\@empty\def\numpoints{0}\def\numpointsEmpty{1}%
        \gdef\eqe@isPts{0}\else\gdef\eqe@isPts{1}\fi
    \makeRoomForProb{\@nbaselineskip\baselineskip}{\eqe@isPts}%
    \gdef\probstar{x}\let\afterlabelhskip\@empty
    \ifx\marginpoints\@empty\else
      \if\numpointsEmpty\eqe@One\let\marginpoints\@gobbletwo\fi
```

(2011/5/13) We add a * feature. When the author types [*3], it is a three point problem, but the value is expressed in-line, not in the margins.

```
      \def\@lti{\let\@isitstar\eqe@One\eqe@grabarg}%
      \def\@ltii{\let\@isitstar\eqe@Zero\eqe@gobbletoend}%
      \expandafter\@ifstar\expandafter\@lti
        \expandafter\@ltii\numpoints\end
      \if\@isitstar\eqe@One\addtocounter{eqpointvalue}{\numpoints}%
        \@marktotalvalue
        \ifdispl@yPoints
          \def\marginparafterhook{\PTs{\numpoints}\space}\else
          \def\marginparafterhook{\PTs*{\numpoints}\space}\fi\else
          \def\marginparpriorhook{\noindent
            \probvalue{\numpoints}{0}}\fi
    \fi
    \setcounter{eq@count}{\value{eqquestionnoi}}%
    \addtocounter{eq@count}{1}%
    \ifnum\value{eq@count}=1\relax
      \eqe@IWO\@auxout{\string
        \csarg\string\gdef{pagenofirstprob\thisexamlabel}{\thepage}}%
    \fi
```

\topofprobhook is a general purpose hook at the top of the problem environment, before the beginning of the nested eqequestions environment.

```
    \topofprobhook
    \begin{eqequestions}%
```

Begin exercise env. Insert h or H when it is the first argument through \hidden@ttr, then begin the exercise environment using eqquestionnoi.

```
    \edef\ctrld@exp@exercise{\noexpand
      \begin{exercise}[eqquestionnoi]\hidden@ttr}%
    \ctrld@exp@exercise\ignorespaces\eqe@p@gobnxtp@r}
```

end problem env. The code for the end of the problem environment.

```
\def\endproblem{\end{exercise}%
  \end{eqequestions}%
  \ifeqlocalversion\ifeqglobalversion
    \xdef\eqe@tmp{\noexpand\forVersion{\eq@selectedVersion}}%
    \aftergroup\eqe@tmp
  \fi\fi
  \global\eqlocalversionfalse
  \ifdispl@yPoints\global\l@stDispl@yPointstrue
  \else\global\l@stDispl@yPointsfalse\fi
  \global\let\eqe@fpmrk\@empty}
```

\PTs
\itemPTsTxt
\itemPTsFormated

When you specify \auto for the optional argument of the problem* environment, when each item must have the command \PTs to assign the value of that question. The \PTs has one optional star-parameter, and one required parameter. The required parameter is the number of points for this item, if the * is specified, then the point value is not typeset in the document.

The command \itemPTsTxt has one argument, the number of points for this item. This argument is passed from the \PTs command. You can redefine the way the points appear in the document using \itemPTsTxt. As separate command \itemPTsFormated is used to put parentheses around \itemPTsTxt. If the * option is taken with \PTs, then you are free to place \itemPTsTxt anywhere in the problem statement.

```
\newcommand\itemPTsTxt[1]{$\eqe@ptsFmt{#1}\,\text{%
    \ifnum#1=1\relax\eqptLabel\else\eqptsLabel\fi}$}
\newcommand\itemPTsEaTxt[1]{$\eqe@ptsFmt{#1}\,\text{%
    \ifnum#1=1\relax\eqptLabel\else\eqptsLabel\fi\space\eqeachLabel}$}
\newcommand{\itemPTsFormated}[1]{(#1)}
```

\obeyPTsStar
\ignorePTsStar

(2013/12/04) Added \obeyPTsStar and \ignorePTsStar. The former is the default. if \ignorePTsStar is in effect, \PTs* is treated as \PTs. Useful for assigning points to parts of a question, but not explicitly telling the student. when you compile for the key (with answerkey option), you can include \ignorePTsStar so the instructor can see the assigned points for each part.

```
\newif\ifObeyPTsStar \ObeyPTsStartrue
\def\obeyPTsStar{\global\ObeyPTsStartrue}
\def\ignorePTsStar{\global\ObeyPTsStarfalse}
\def\PTs{\leavevmode\@ifstar{\@PTs{*}}{\@PTs{x}}}
\def\@PTs#1#2{%
    \if\@reportpoints\eqe@Zero\ignorespaces\else
        \if\eqe@pointsPartsId\eqe@One
        \addtocounter{eqpointvalue}{#2}\@marktotalvalue
        \addtocounter{eq@count}{#2}\fi
```

\ignorePTsStar

If \ignorePTsStar is in effect (\ObeyPTsStarfalse), we gobble the \else and let the code flow through.

```
        \if#1*\ifObeyPTsStar\ignorespaces\else\expandafter\@gobble\fi
    \else
```

(2012/04/26) Wrapped \itemPTsFormated{\itemPTsTxt{#2}} as the argument of \eqe@movePTs. \eqe@movePTs does nothing by default, but may be redefined, for example, to place the value of each part on the margin. The default definition of \eqe@movePTs follows.

```
        \eqe@movePTs{\itemPTsFormated{\itemPTsTxt{#2}}}\fi
    \fi
}
\def\eqe@movePTs#1{#1}
```

\Do<num>

The \isItD@ tests to see if the next token is \Do, if yes, it marks it and calls \y@st@Do, which gets the argument if the \Do token.

```
\let\auto\relax
\def\isItD@{\@ifnextchar\Do{\let\yest@D@\eqe@YES\y@st@Do}
    {\let\yest@D@\eqe@NO\@gobblet@end}}
\let\yest@D@\eqe@NO
\let\D@Num\@empty
\def\y@st@Do\Do#1\end{\gdef\D@Num{#1}}
\def\@gobblet@end#1\end{}
```

Test the argument to see if it is the \auto token

```
\def\is@uto#1\auto#2\@nil{\def\eqe@argi{#1}\def\eqe@argii{#2}%
    \ifx\eqe@argii\@empty\let\isit@uto\eqe@Zero\else
        \let\isit@uto\eqe@One\fi
}
```

problem*

The problem* environment is used to pose a multi-part question. The parts environment is used to enumerate the parts.

We create a Id for the points specified by the first (and second) optional parameters: 0 (total points specified); 1 (\auto specified); 2 (points each specified); 4

(\Do second optional parameter); a value of \relax means no points specified (the default).

```
\let\eqe@pointsPartsId\relax
```

```
\def\leadinitemWarningStar{\PackageWarning{eqexam}
    {Using the star (*) in front of the points\MessageBreak
     designator is not allowed when there is a
     \MessageBreak\string\leadinitem}}
\def\tableadinWarningStar{\PackageWarning{eqexam}
    {Using the star (*) in front of the points\MessageBreak
     designator is not allowed when there is a
     \MessageBreak\string\tableadin}}
\let\isProbStarEnv\eqe@NO
\let\topofprobstarhook\relax
```

\applyparfixes
\cancelparfixes

Added \eqe@gobnxtpar to hopefully, eliminate the need to leave no pars (blank lines) between \begin{parblem*} and either \leadinitem or \tableadin. The default is to not affect the old behavior. You have to declare \applyparfixes to apply the fix; revert back to the old behavior with \cancelparfixes

```
\def\eqe@ps@gobtop@r#1\par{\eqe@isle@dinnext}
\def\eqe@isle@dinnext{\@ifnextchar\leadinitem{}{\vskip-\baselineskip}}
\def\restorele@dinpfixDef{\@ifnextchar\par{\eqe@ps@gobtop@r}{}}
\let\eqe@ps@gobnxtpar\restorele@dinfixDef
\def\applyleadinfix{\let\eqe@gobnxtpar\restorele@dinpfixDef}
\def\cancelleadinfix{\let\eqe@gobnxtpar\relax}
\def\applyparfixes{\PackageInfo{eqexam}
  {Applying paragraph fixes to problem\MessageBreak
  and problem* environments}\applyparfixesp\applyleadinfix}
\def\cancelparfixes{\PackageInfo{eqexam}
  {Cancelling paragraph fixes to problem\MessageBreak
  and problem* environments}\cancelparfixesp
  \cancelleadinfix}
```

The default behavior is to do nothing about blank lines.

```
\cancelparfixesp\cancelleadinfix
\let\neutralizeparfixes\cancelparfixes
```

We now begin the code for the problem* environment.

```
\csarg\def{problem*}{\let\isProbEnv\eqe@YES
    \let\isProbStarEnv\eqe@YES
    \global\thereissolutionfalse
    \@ifnextchar[{\pr@bl@m@star}{\pr@bl@m@star[]}}
\def\pr@bl@m@star[#1]{%
    \@ifnextchar[{\pr@blem@star{#1}}{\pr@blem@star{#1}[]}}
\def\pr@blem@star#1[#2]{%
    \if\eqe@isPts0\global\let\eqe@prevProbZero\eqe@YES\else
        \global\let\eqe@prevProbZero\eqe@NO\fi
    \def\numpoints{#1}\ifx\numpoints\@empty
        \gdef\eqe@isPts{0}\else\gdef\eqe@isPts{1}\fi
    \makeRoomForProb{\@nbaselineskip\baselineskip}{\eqe@isPts}%
```

Let the \Do commands to internal versions

```
    \let\DoNum\eqe@DoNum\let\nDoNum\eqe@nDoNum
    \let\OutOfNum\eqe@OutOfNum\let\nOutOfNum\eqe@nOutOfNum
```

```
%     \proofingsymbol{\ding{52}}%
    \gdef\probstar{*}%
    \gdef\pr@b@secondarg{#2}\setcounter{eq@count}{0}%
    \let\afterlabelhskip\@empty
    \global\let\probpointseach\@empty\def\numpoints{#1}%
```

Determine if the argument begins with *

```
    \@ifstar{\let\@isitstar\eqe@One\eqe@grabarg}%
        {\let\@isitstar\eqe@Zero\eqe@gobbletoend}#1\end
```

If this problem* environment has a \leadinitem, the * option is not allowed

```
    \if\@isitstar\eqe@One
        \@tempcnta=\theeqquestionnoi\relax
        \advance\@tempcnta1\relax
        \@ifundefined{leadinitem\thisexamlabel-\the\@tempcnta}{}
            {\leadinitemWarningStar\let\@isitstar\eqe@Zero}%
        \@ifundefined{tableadin\thisexamlabel-\the\@tempcnta}{}
            {\tableadinWarningStar\let\@isitstar\eqe@Zero}%
    \fi
```

Check for the \auto keyword . There are problems when the first two digits are the same, the old comparison would say that the first parameter is \auto, which is it not. Replace the old comparison with a more robust method.

```
    \expandafter\is@uto\numpoints\auto\@nil
    \if\isit@uto\eqe@One\let\eqe@pointsPartsId\eqe@One
        \global\let\probpointseach\relax
```

The author has requested \auto

```
        \def\eqe@next{\autocalcparts}%
    \else
        \ifx\pr@b@secondarg\@empty\else
        \let\eqe@pointsPartsId\eqe@Four
        \isItD@#2\end\fi
```

Not \auto so either <num>ea or <num>

```
        \def\eqe@next{\manualcalcparts{\numpoints}}%
    \fi\eqe@next
```

2012/11/30 added \leadinitem defined eqexam.def as a dummy command, which we the \let here to \eqe@leadinitem.

```
    \let\leadinitem\eqe@leadinitem
```

2015/05/31 added \tableadin defined eqexam.def as a dummy command, which we the \let here to \eqe@leadinitem.

```
    \let\tableadin\eqe@tableadin
```

\topofprobstarhook is a general purpose hook at the top of the problem* environment, before the beginning of the nested eqequestions environment.

```
    \topofprobstarhook
    \ifeqfortextbook
    \writeT@SolnFile{\protect\global
      \protect\frstProbNumShownfalse}\fi
    \begin{eqequestions}%
    \begin{exercise}[eqquestionnoi]*\eqe@gobnxtpar}%
```

```
\def\ftb@endprobstarCks{%
    \ifWithinANSGrp
        \PackageError{eqexam}{\string\bGrpANS\space is still open}
        {You need to match it with an \string\bGrpANS,
         or remove it.}%
    \fi
}
```

\endproblem* begins here.

```
\csarg\def{endproblem*}{%
    \eqe@IWO\@auxout{\string\csarg\string
      \gdef{nPartsThisProb\thisexamlabel.\theeqquestionnoi}%
        {\arabic{partno}}}%
    \ifx\probpointseach\@empty\else
    \ifx\probpointseach\auto
        \eqe@IWO\@auxout{\string\csarg\string
          \gdef{prob\thisexamlabel.\theeqquestionnoi}%
            {\theeq@count}}%
    \else
        \setcounter{eq@count}{\value{partno}}%
        \ifx\pr@b@secondarg\@empty\else
            \bgroup\toks0=\expandafter{\pr@b@secondarg}%
            \expandafter\isItD@\the\toks0 \end
```

If there is a \Do, we write this info to AUX.

```
            \ifx\yest@D@\eqe@YES\ifx\eqe@pointsEach\eqe@YES
                \eqe@IWO\@auxout{\string
                    \csarg\string\gdef{DoNumThisProb\thisexamlabel.%
                    \theeqquestionnoi}{\D@Num}}%
                \@tempcnta = \value{eq@count}%
                \advance\@tempcnta -\D@Num
                \global\advance\value{eq@count}-\@tempcnta
% 3.0k
                \@tempcnta=\value{partno}%
                \advance\@tempcnta -\D@Num
                \multiply\@tempcnta by\argi
                \addtocounter{eqpointvalue}{-\@tempcnta}%
            \fi\fi
            \egroup
        \fi
        \multiply\value{eq@count}\argi
        \eqe@IWO\@auxout{\string\csarg\string
          \gdef{prob\thisexamlabel.\theeqquestionnoi}{\theeq@count}}%
    \fi\fi
    \end{exercise}%
    \end{eqequestions}%
    \ifeqfortextbook\ftb@endprobstarCks\fi
    \ifeqlocalversion\ifeqglobalversion
        \xdef\eqe@tmp{\noexpand\forVersion{\eq@selectedVersion}}%
        \aftergroup\eqe@tmp
    \fi\fi
    \global\eqlocalversionfalse

    \@marktotalvalue
    \ifdispl@yPoints\global\l@stDispl@yPointstrue
    \else\global\l@stDispl@yPointsfalse\fi
}
```

\leadinitem Some authors, no me among them, want to post a question with parts (problem*) without an introductory sentence. Scandalous!

```
\begin{problem*}\relax
\leadinitem Prove that $4x^2\ge (x+1)^2$ for $x\ge1$.
\begin{solution} soln\end{solution}
\begin{parts}
\item Use the principle of mathematical induction to prove that
   $4^n\geq3n^2$ for all positive integers $n$.
\begin{solution} soln\end{solution}
\end{parts}
\end{problem*}
```

Below is my solution to this problem. There were changes in both eqexam and exerquiz (eqexam.def). Here are the listing of changes for my future reference.

- Defined \eqe@leadinitem, \eqe@@leadinitem, and \leadinitem below. The first one mostly tests whether we are permitted to use the \leadinitem command. If \solutionparshape is empty, the command may be used. (This gives a restriction of only one use per a single problem* environment.) Control is passed to \eqe@@leadinitem which does all the work.

- In the definition of \pr@blem@star (part of the startup for problem* we say \let\leadinitem=\eqe@leadinitem

- In the startup code for exercise@parts@list (exerquiz) lines containing \solutionparshape and \let\leadinitem\eq@leadinitemparts are found. Also, when the parts environment is complete, we restore the default definition of \leadinitem, \let\leadinitem\leadinitem@external.

- In the definition of \eqe@@leadinitem, we also define \solutionparshape to help control the paragraphs for this delicate problem of a lead-in item. Its normal definition is \@empty, and this fact used to detect whether the document author is trying to use two \leadinitems in one problem* environment.

- At the beginning of the definition of the command `\@exercise` (in exerquiz we `\let` `\solutionparshape` to `\@empty`.
- So too does `\solutionparshape` appear in the definition of `\vspaceFmt` (exerquiz).
- `\solutionparshape` use used to correctly format in the `solutionsafter`, found in `\solnexer@@@woparts` (exerquiz).

As can be seen, this simple feature is not so simple.

Set the default value of `\solutionparshape`.

```
\let\solutionparshape\@empty
```

`\leadinitem` This is the beginning of `\leadinitem`, at least the the active definition within the `problem*` environment, and outside the `parts` environment.

```
\def\eqe@leadinitem{%
    \ifx\solutionparshape\@empty
```

`\solutionparshape` empty means `\leadinitem` hasn't been used yet, so we can use it here by passing control to `\eqe@@leadinitem`, otherwise, we toss an exception.

```
        \def\eqe@next{\eqe@@leadinitem}%
            \ifeqfortextbook\writeT@SolnFile{\protect\bpartsmrk}\fi
        \else
        \def\eqe@next{\PackageError{eqexam}
        {The \string\leadinitem\space command may
        only be used\MessageBreak once per problem* environment}
        {Remove all but one of the \string\leadinitem\space
        commands.}}%
    \fi\eqe@next
}
\newif\ifisleadin \isleadinfalse
\let\isparshapeExpanded\eqe@NO
\def\eqe@@leadinitem{\if\itsforleadinitem\eqe@NO
    \setcounter{partno}{0}\fi\refstepcounter{partno}%
```

After initializing the counter, we make various calculations

```
    \eq@initializeContAnnot
    \isleadintrue\let\isitleadin\eqe@YES
```

We signal a `\leadinitem` by writing a special command to the AUX file, this is used to disallow the use of the `*` option of points.

```
    \eqe@writetoAux{\string\csarg\string
      \gdef{leadinitem\thisexamlabel-\theeqquestionnoi}{}}%
    \settowidth{\eq@tmpdima}{\normalfont\parts@indent\eqe@prtsepPrb}%
    \xdef\leadinIndentPrtSep{\the\eq@tmpdima}%
    \setlength{\eq@tmpdima}{\eqemargin+\eq@tmpdima}%
    \xdef\leadinIndent{\the\eq@tmpdima}%
    \settowidth{\eq@tmplength}{\parts@indent}%
    \edef\partsleadinIndent{\the\eq@tmplength}%
    \setlength{\eq@tmplength}{\linewidth-\leadinIndentPrtSep}%
    \edef\leadinIndentLength{\the\eq@tmplength}%
```

Modify `\@listii` from core LaTeX: Increase `\leftmarginii` and `\linewidth` appropriately.

```
\@ifundefined{@listii@SAVE}{\global\let\@listii@SAVE\@listii}{}%
    \expandafter\def\expandafter\@listii\expandafter{\@listii@SAVE
     \advance\leftmarginii\leadinIndentPrtSep\relax
      \leftmargin\leftmarginii \labelwidth\leftmarginii
      \advance\labelwidth-\labelsep
%      \advance\linewidth\leadinIndentPrtSep\relax
    }%
```

The `parshape` for the first paragraph of the `\leadinitem`.

```
    \parshape=2 \eqemargin \linewidth \leadinIndent \leadinIndentLength
```

Here is the definition of `\solutionparshape`, we use the shape of subsequent paragraphs.

```
    \xdef\solutionparshape{%\noexpand\linewidth\leadinIndentLength
        \noexpand\parshape=1 \leadinIndent\space\leadinIndentLength}%
```

After the current paragraph, we execute `\solutionparshape`, then empty out `\everypar`. Seems to work.

```
    \global\let\isparshapeExpanded\eqe@NO
    \everypar{\solutionparshape
        \global\let\isparshapeExpanded\eqe@YES\everypar{}}%
```

These next lines were taken from `\eq@item@common` in exerquiz. The command `\eq@insertContAnnot` was removed from the `\eq@item@common` code, it is not needed here and caused trouble. We `\let` `\eq@item` to `\eq@leadin@item` to continue the flow. `\@ckhide` checks for an optional argument (`h` or `H`) and sets switches as appropriate.

```
    \let\eq@item\eq@leadin@item\eqp@rtc@lcm@rk
    \def\currhideopt{x}\eq@hidesolutionfalse\eq@nolinkfalse
    \@ifnextchar[{\@ckhide}{\eq@item}%
}
```

The final step. If the solution is hidden, we do not write the header.

```
\def\eq@leadin@item{\eq@ckglobalhide\ifeq@hidesolution\else
    \global\let\eqExerSolnHeader\eq@@writeexheaderlist\fi
    \ifeq@nosolutions\eq@nolinktrue\fi
    \ifeq@solutionsafter\eq@nolinktrue\fi
    \unskip\noindent\makebox[\partsleadinIndent]{\eqexlisttabheader}%
    \eqe@prtsepPrb\ignorespaces
}
```

`\tableadin` `\tableadin` is `\let` to `\eqe@tableadin` within the `problem*` env. The switch `\if@tableadinitem` is defined in exerquiz.

```
\def\eqe@tableadin{\@tableadinitemtrue\eqe@writetoAux{\protect
  \csarg\protect\gdef{tableadin\thisexamlabel-\theeqquestionnoi}{}}%
}
```

`\pushProblem`
`\popProblem`

There may be an occasion when a multi-part question needs to be broken between parts. use the `\pushProblem` and `\popProblem` for this purpose. The push saves the counter value, and ends the `parts` environment. The pop restarts the `parts`, and resets the parts counter.

In the example below, we have our parts in a `multicols` environment, we `\pushProblem`, close `multicols`, `\popProblem` and continue with the multi-parts in single column.

```
\item Compute $\lim_{x\to2^{\text{$-$}}} f(x)$
\begin{solution}[1in]\end{solution}
\pushProblem
\end{multicols}
\popProblem
\item What value(s) of $c$ make the function $f$
continuous at $x=2$?
\begin{solution}[.5in]\end{solution}
\end{parts}
```

```
\newcommand\pushProblem{\xdef\nlastItem{\arabic{partno}}\end{parts}}
```

(2013/05/30) Adding an optional parameter to `\popProblem` to match the option argument of the `parts` environment. This enables you to push a `list` environment and push a `tabular` environment.

```
\newcommand\popProblem[1][]{%
    \def\@argi{#1}\ifx\@argi\@empty
    \def\eqe@bParts{\begin{parts}}\else
    \def\eqe@bParts{\begin{parts}[#1]}\fi
    \eqe@bParts\setcounter{partno}{\nlastItem}}
```

```
\def\lastPageTotal{0pt}
\def\marginparafterhook{\xdef\lastPageTotal{\the\pagetotal}}
```

`\manualcalcparts`

is the command calculates points when the argument is *not* `\auto`. The macro `\prob@Arg` determines if the points argument passed is of the form `<num>ea`.

```
\def\prob@Arg#1ea#2\end{\def\argi{#1}\def\argii{#2}}
```

Now begin `\manualcalcparts`; `#1` is the number of points, which may be of the form `<num>ea`, or just `<num>`.

```
\let\eqe@pointsEach\eqe@NO
\def\manualcalcparts#1{%
    \expandafter\prob@Arg#1ea\end
    \ifx\argii\@empty\edef\numpoints{#1}%
        \let\eqe@pointsEach\eqe@NO
```

Argument form is `<num>` assumed: Total points specified, we should ignore any `\PTs` commands.

```
        \let\eqe@pointsPartsId\eqe@Zero
    \else
```

Argument form is `<num>ea` assumed: Again we should ignore any `\PTs` commands.

```
        \gdef\probpointseach{x}\let\eqe@pointsPartsId\eqe@Two
        \let\eqe@pointsEach\eqe@YES
```

Points each specified

```
    \setcounter{eq@count}{\value{eqquestionnoi}}%
    \addtocounter{eq@count}{1}\csarg
     \ifx{prob\thisexamlabel.\theeq@count}\relax
        \def\numpoints{\argi}\else
        \def\numpoints{\expandafter
            \csname prob\thisexamlabel.\theeq@count\endcsname}%
    \fi
```

```
    \ifx\yest@D@\eqe@YES
        \setcounter{eq@count}{\value{eqquestionnoi}}%
        \addtocounter{eq@count}{1}%
        \@ifundefined{nPartsThisProb\thisexamlabel.\theeq@count}
            {\makeOutOfNum{0}\makeDoNum{0}}{%
            \expandafter\makeOutOfNum{%
            \csname nPartsThisProb\thisexamlabel.%
                \theeq@count\endcsname}%
            \expandafter\makeDoNum{%
            \csname DoNumThisProb\thisexamlabel.%
                \theeq@count\endcsname}}%
    \fi
\fi
```

If `\marginpoints` is `\@empty`, the author has chosen the `nopoints` option or used the `\NoPoints` command.

```
\ifx\marginpoints\@empty
```

No points for this exam

```
\else
    \ifx\argi\@empty
```

If `\argi` is empty, no points were specified, so we `\let` `\marginpoints` to `\@empty`

```
        \let\marginpoints\@empty
    \else
```

Points are displayed in margins or inline

```
        \ifx\argii\@empty
```

Total points specified

```
            \ifx\marginpoints\@empty\else
                \if\@isitstar\eqe@One
```

Points to appear “in-line” rather than in the margins

```
                    \addtocounter{eqpointvalue}{#1}%
                    \@marktotalvalue\ifdispl@yPoints
                    \def\marginparafterhook{\itemPTsFormated{%
                        \itemPTsTxt{\numpoints}}\space}\fi
                \else
```

Points appear in the margins

```
                    \def\marginparpriorhook{\noindent
```

```
                            \probvalue{\numpoints}{0}}%
                    \fi
                \fi
            \else
```

Points each specified

```
                \ifx\marginpoints\@empty\else
                    \if\@isitstar\eqe@One\ifdispl@yPoints
```

Points to appear “in-line” rather than in the margins

```
                        \def\marginparafterhook{%
                        \itemPTsFormated{\itemPTsEaTxt{\argi}}\space}\fi
                    \else
```

Points appear in the margins

```
                        \ifdispl@yPoints
                            \def\marginparpriorhook{\noindent
                            \marginpoints{\numpoints}{\argi}}\fi
                    \fi
```

Don’t remember why I named this command the way I did, but it is use to pass the number of points, when there is a `\leadinitem`.

```
                    \edef\eqp@rtc@lcm@rk{\noexpand
                    \addtocounter{eqpointvalue}{\argi}%
                        \noexpand\@marktotalvalue}%
                \fi
            \fi
        \fi
    \fi
    \ifnum\value{eq@count}=1\relax
        \eqe@IWO\@auxout{\string\csarg\string
            \gdef{pagenofirstprob\thisexamlabel}{\thepage}}%
    \fi
}
```

`\autocalcparts` is the command that computes the total points when the author specifies `\auto` as the optional argument of `problem*`. The commands `\acp@mpah` and `\acp@mpph` were recently (2012/04/21) separated out to allow for additional customization, without re-defining the whole of `\autocalcparts`.

```
\def\acp@mpah{\itemPTsFormated{\itemPTsTxt{\numpoints}}\space}
\def\acp@mpph{\noindent\marginpoints{\numpoints}{0}}
\def\autocalcparts{%
    \setcounter{eq@count}{\value{eqquestionnoi}}%
    \addtocounter{eq@count}{1}%
    \csarg\ifx{prob\thisexamlabel.\theeq@count}\relax
        \def\numpoints{0}% assume zero points until we get the total
    \else
        \edef\numpoints{\@nameuse{prob\thisexamlabel.\theeq@count}}%
    \fi
%    \addtocounter{eqpointvalue}{\numpoints}\@marktotalvalue
```

```
    \ifx\marginpoints\@empty\else
        \ifdispl@yPoints
            \if\@isitstar\eqe@One
```

If we have `*\auto`, the total is to appear inline.

```
                \def\marginparafterhook{\acp@mpah}\else
```

Otherwise, the total will appear in the margin.

```
                \def\marginparpriorhook{\acp@mpph}\fi
        \fi
    \fi
    \setcounter{eq@count}{0}%
}
```

`\forproblem` `\foritem` `\forleadinitem`

When typing solutions from assigned problems in a textbook, the problems assigned are not consecutive. You can set the problem number before the problem environments by using the `\forproblem` command. The one required argument is the problem number: `\forproblem{10)`.

```
\let\eqe@fpmrk\@empty
\newcommand{\forproblem}[1]{\def\eqe@fpmrk{-\the@exno}%
    \setcounter{eqquestionnoi}{#1 - 1}}
```

A similar comment for `\item`. These are useful for making out solution sets to homework assignments where problems are assigned from the textbook and you want to give a solution to problem 12, part (b), An example of usage is

```
\forproblem{12}
\begin{problem*}
Factor each.
\begin{parts}
   \foritem{b} $ x^2 + 2x + 1 = ( x + 1 )^2 $
   \item       $ x^2 - x - 2 = ( x - 2 )( x + 1) $ % this is part (c)
   \foritem{e} $ x^2 + 7x + 10 = ( x - 2 )( x + 7) $
   \item ...   % this will be part (f)
\end{parts}
\end{problem*}
```

```
\newcommand{\foritem}[2][]{%
    \setcounter{partno}{0}\def\@rgi{#1}\ifx\@rgi\@empty
        \def\eqe@nextitem{\item}\else
        \def\eqe@nextitem{\item[#1]}\fi
    \foritem@cont{#2}}
\newcommand{\forleadinitem}[2][]{%
    \setcounter{partno}{0}\def\@rgi{#1}\ifx\@rgi\@empty
        \def\eqe@nextitem{\leadinitem}\else
        \def\eqe@nextitem{\leadinitem[#1]}\fi
    \foritem@cont{#2}}
```

(2017/01/04) Modified `\foritem@cont` to allow for numbering parts.

```
\def\foritem@cont#1{\ifuseNumForParts
    \edef\fliPartNo{#1}\setcounter{partno}{#1-1}\else
    \lowercase{\def\eq@selectedItem{#1}}%
    \let\eq@initLoop\eqe@NO
    \loop
        \stepcounter{partno}\expandafter
        \if\alph{partno}\eq@selectedItem
            \let\eq@initLoop\eqe@YES\fi
    \ifx\eq@initLoop\eqe@NO\repeat
    \edef\fliPartNo{\the\c@partno}%
    \addtocounter{partno}{-1}\fi\let\itsforleadinitem\eqe@YES
    \eqe@nextitem}
```

The command \eqe@insertContAnnot attempts to insert a string just prior to a part, if that part begins a new page. To get it right, it promotes a new page using the default of .25in. The optional parameter allows you to insert a new value; this may be needed to get the string \annotContStr placed properly. The commands

\annotContStr The string that is typeset by the \eqe@insertContAnnot command.

\acvspace User access to changing the vertical spacing \promoteNewPage uses within \eqe@insertContAnnot.

\resetacvspace Resets the vertical spacing back to its default.

```
\newcommand{\annotContStr}{%
    \textbf{Problem~{\eqeCurrProb} continued\strut}}
\newcommand{\acvspace}[1]{\def\ic@vspace{#1}\setlength\iacvspace{#1}}
```

\ic@vspacedefault is the default vertical spacing used by \eqe@insertContAnnot

```
\newlength\iacvspace
\newcommand{\ic@vspacedefault}{1in} % changed .25in to 1in 2012/12/04
\newcommand{\resetacvspace}{\setlength\iacvspace{\ic@vspacedefault}}
\resetacvspace
```

\eqe@insertContAnnot promotes a new page, and if the current page is different than the starting page, \eq@currProbStartPage, we insert \annotContStr.

```
\def\@nnotContStrSkip{\vskip3pt}
\newcommand{\eqe@insertContAnnot}[1][\iacvspace]{\promoteNewPage[#1]%
  \ifnum\arabic{page}>\eq@currProbStartPage
    \xdef\eq@currProbStartPage{\arabic{page}}%
    {\settowidth{\eq@tmplength}{\parts@indent\eqe@prtsepPrb}%
    \xdef\eqe@partsIndent{\the\eq@tmplength}}%
    \ifwithinparts
```

\eq@item is the list definition of \item which has been \let to \eq@item within the parts environment.

```
      \@ifundefined{eq@item@latex}{\item[]\hspace*{-\eqemargin}%
      \ifx\solutionparshape\@empty\else\hspace*{-\eqe@partsIndent}\fi}
      {\eq@item@latex[]\hspace*{-\eqemargin}%
      \hspace{-\eqe@partsIndent}}\else
```

If placed between problem sets, we indent as appropriate.

```
      \hspace*{-\eqemargin}\fi
    \annotContStr\@nnotContStrSkip % dpsj20
  \fi
```

```
}
\edef\eq@currProbStartPage{\arabic{page}}
```

\turnContAnnotOff
\turnContAnnotOn

Turn off and on this feature. The default is off.

```
\newif\ifcont@nnot \cont@nnotfalse
%\newcommand{\@gobbloptone}[1][]{}
\newcommand{\turnContAnnotOff}{\global\cont@nnotfalse
    \global\let\eq@insertContAnnot\relax}
\newcommand{\turnContAnnotOn}{\global\cont@nnottrue
    \global\let\eq@insertContAnnot\eqe@insertContAnnot}
\turnContAnnotOff
```

Used for manually inserting annot

```
\def\insertContAnnot{\eq@insertContAnnot}
```

\aNewPage
\qNewPage

A simple command for inserting \newpage, only if the \answerkey option has been taken.

```
\newcommand\aNewPage{\ifanswerkey\newpage\fi}
\newcommand\qNewPage{\ifanswerkey\else\newpage\fi}
```

\OnBackOfPage

In an effort to make maximum use of the paper, I sometimes ask the students to solve the problem on the back of a page. The following command is an automated instruction. Generally, we work on the back of the previous page, unless we are on page 1, in this case we work on the back of page 1.

```
\newcounter{backofpage}
\newcommand\bopText{on the back of page~\boPage}
\newcommand\bopCoverPageText{on the back of the cover page}
\newcommand\OnBackOfPage[1][\bopText]{%
    \refstepcounter{backofpage}\label{bop\thebackofpage}
    \begingroup
    \csarg\ifx{r@bop\thebackofpage}\relax
        \def\boPage{??}\else
        \edef\eqe@temp{\csname r@bop\thebackofpage\endcsname}%
        \ifx\hyper@anchor\@undefined
            \edef\boPage{\expandafter\@secondoftwo\eqe@temp}\else
            \edef\boPage{\expandafter\@secondoffive\eqe@temp}\fi
        \c@eq@count\boPage
        \advance\c@eq@count-1\relax
```

If on page 1, we work on the back of page 1, otherwise, we work on the back of the previous page.

```
        \edef\boPage
        {%
            \ifx\eqex@coverpage\relax
                \ifnum\value{eq@count}=0
                    1%
                \else
                    \the\value{eq@count}%
                \fi
            \else
                \ifnum\value{eq@count}=0
                    -1%
                \else
                    \the\value{eq@count}%
                \fi
            \fi
        }%
    \fi
    \csarg\ifx{r@bop\thebackofpage}\relax#1\else
    \ifnum\boPage=-1\relax\bopCoverPageText\else#1\fi\fi
    \endgroup
}
```

14 Vertical Space Filling Options

When the `nosolutions` or the `vspacewithsolns` is used, a vertical space is generated by the `solution` environment. Previously, this has just been a vertical white space, now, we provide the ability to fill the space with horizontal rules of different types. Below is the implementation of this.

14.1 General filler lines commands and controls

`\useFillerLines` When used, the vertical space is written with lines (rules, dashes, dots, grids).

```
\newcommand{\useFillerLines}{\ifx\vspaceFiller % dpsj8
  \vspaceFillerLines\else\@eqlinedfillertrue
  \let\vspaceFiller\vspaceFillerLines\fillTypeDefault
\fi}
```

`\useFillerDefault` Resets the vertical space to the original white space.

```
\newcommand{\useFillerDefault}{\@eqlinedfillerfalse
    \let\vspaceFiller\vspaceFillerDefault
    \let\eqWriteLine\hfill}
```

`\fillTypeHRule` Writes the line as a solid line (`\hrulefill`).

```
\newcommand{\fillTypeHRule}{\let\eqWriteLine\eqWriteLineFill
  \let\makeVgrid\relax}
```

`\fillTypeDots` Writes the line as a dotted line (`\dotfill`).

```
\newcommand{\fillTypeDots}{\let\eqWriteLine\eqWriteLineDots
  \let\makeVgrid\relax}
```

`\fillTypeDashLine` Writes the line as a dotted line (`\eqdashrulefill`).

```
\newcommand{\fillTypeDashLine}{\let\eqWriteLine\eqWriteLineDashFill
  \let\makeVgrid\relax}
```

\fillTypeBlankLine Fills the line

```
\newcommand{\fillTypeBlankLine}{\let\eqWriteLine\eqWriteLineBlankFill
  \let\makeVgrid\relax}
```

\fillTypeDefault Resets fill type back to the default, `\hrulefill`.

```
\newcommand{\fillTypeDefault}{\let\eqWriteLine\eqWriteLineFill
  \let\makeVgrid\relax}
```

\fillTypeGrid Fills the space with a grid, horizontal and vertical lines.

```
\newcommand{\fillTypeGrid}{\ifx\vspaceFiller\vspaceFillerLines
  \let\eqWriteLine\eqWriteLineFill\let\makeVgrid\eqe@makeVgrid\fi}
```

\eqWriteLineColor The color of the rule to use.

```
\newcommand{\eqWriteLineColor}[1]{\def\eq@WriteLineColor{#1}}
\eqWriteLineColor{gray}
```

\eqWLSpacing The line spacing between the rules.

```
\newlength\wlVspace
\newcommand{\eqWLSpacing}[1]{\setlength\wlVspace{#1}}
\eqWLSpacing{14.0pt}
```

Fill Types. We have three types of line fill: `\hrulefill`, `\dotfill`, and a custom rule `\eqdashrulefill`. The commands three `\eqWriteLineFill`, `\eqWriteLineDots`, and `\eqWriteLineDashFill` implements these three types. They are `\let` to `\eqWriteLine`, which is used in `\vspaceFillerLines`.

```
\newcommand{\eqWriteLineFill}{%
  \textcolor{\eq@WriteLineColor}{\leaders
    \hrule height \flfboxrule\hfill}}
```

The first box is used in `\eqdotrulefill`, whereas the second two are used to split off content using `\vsplit`. These are active when the options flextended and answerkey are in force, and when `\useFillerLines` and `\turnflanskeyOn` have been expanded.

```
\newbox\eqe@tempbox
\newbox\eqe@nskeyflsplit
\newbox\eqe@nskeyfltop
\def\eqdotrulefill{\leavevmode
  \cleaders\hb@xt@ .44em{\copy\eqe@tempbox\hss}\hfill} % dps16
%  \cleaders\hb@xt@ .44em{\unhcopy\eqe@tempbox\hss}\hfill} % dps8
\newcommand{\eqWriteLineDots}{%
  \textcolor{\eq@WriteLineColor}{\eqdotrulefill}}
\def\eqdashrulefill{\leavevmode
  \cleaders\hb@xt@ .44em{\rule{.22em}{\flfboxrule}\hss}\hfill\kern\z@}
\newcommand{\eqWriteLineDashFill}{%
  \textcolor{\eq@WriteLineColor}{\eqdashrulefill}}
%\newcommand{\eqWriteLineBlankFill}{\hfill}
\newcommand{\eqWriteLineBlankFill}{\vphantom{\hrulefill}}
```

Vertical counterparts to those above, excepting blank fill.

```
\newcommand{\eqWriteLineVFill}{\leaders\vrule width\flfboxrule\vfill}
\def\eqdotruleVfill{\cleaders\vbox to .44em{\vss
  \hbox to0pt{\hss.\hss}}\vfill}
\newcommand{\eqWriteLineVDots}{\eqdotruleVfill}
\def\eqdashruleVfill{\cleaders\vbox to .44em{\vss
  \hbox to0pt{\hss\rule{\flfboxrule}{.22em}\hss}}\vfill}
\newcommand{\eqWriteLineDashVFill}{\eqdashruleVfill}
```

\usenLineDimen \useVspaceDimen When the solution environment contains both an nLines and a dimension, by default, the dimension is used; however, if \usenLineDimen is specified, we force the use of the nLines specification. Switch back to the default using \useVspaceDimen.

```
\newif\if@equsedim \@equsedimtrue
\newcommand{\useVspaceDimen}{\@equsedimtrue}
\newcommand{\usenLineDimen}{\@equsedimfalse}
\newif\if@eqalignfilllinestoleft\@eqalignfilllinestoleftfalse
```

\fillerLinesAlignDef \fillerLinesOnLeftMargin Use \fillerLinesAlignDef and \fillerLinesOnLeftMargin to adjust the alignment of the rule lines for this feature.The first is the default, the second one aligns rule lines to the left margin.

```
\newcommand{\fillerLinesOnLeftMargin}{\@eqalignfilllinestolefttrue}
\newcommand{\fillerLinesAlignDef}{\@eqalignfilllinestoleftfalse}
\newcommand\priorPageBreakMsg[1]{\def\priorP@geBre@kMsg{#1}}
\let\priorP@geBre@kMsg\@empty
\newcommand{\flPageBreakMsg}[1]{%
    \priorPageBreakMsg{\emitMessageNearBottom[\iacvspace]
    {\eqfititin{{\Large\strut}#1}}}%
}
```

The command \makeVgrid is an internal macro that is let to either \relax, as below, or to \eqe@makeVgrid. The latter creates a grid of fill lines.

```
\let\makeVgrid\relax %dps1
```

Key-values for fill lines We define the eqefillLines xkeyval family, the keys are set through the command \setFillLinesFmt. We define nine keys numbers, numbersep, color, topline, gridtype, fltype, align, outlineonly, bgonly, and bgcolor. The first two are used to number the fill lines, the color is to color the fill lines, the last two concerning the grid type filler lines.

numbers=⟨none|left|right⟩ This is a numbers is a choice key with possible values of none, left and right. It places a line numbering to the left or right of the line, or no line number is used if none is selected.

```
\define@choicekey+{eqefillLines}{numbers}[\val\nr]%
  {none,left,right}[none]{\ifcase\nr\relax
    \let\eqe@numLinesL\relax\let\eqe@numLinesR\relax\or
    \let\eqe@numLinesL\eqe@@numLinesL\let\eqe@numLinesR\relax\or
    \let\eqe@numLinesL\relax\let\eqe@numLinesR\eqe@@numLinesR
  \fi
}{\PackageWarning{aeb}{Bad choice for numbers, permissible values
  are none, left, and right. Try again}}
\let\eqe@numLinesL\relax\let\eqe@numLinesR\relax
```

`numbersep=`⟨*dimen*⟩ When `numbers` is either `left` or `right`, the separation between the line and the number is `numbersep`. The default value is 2pt.

```
\define@key{eqefillLines}{numbersep}[2pt]{%
  \setlength{\eqetmplengtha}{#1}%
  \edef\eqe@numbersep{\the\eqetmplengtha}}
\def\eqe@numbersep{2pt}
```

`color=`⟨*color-spec*⟩ The `color` key paints the lines the specified color, for example, `color=red!20`. It simply defines `\eq@WriteLineColor` to be this color.

```
\define@key{eqefillLines}{color}[]{\def\eq@WriteLineColor{#1}}
```

`topline=`⟨`true|false`⟩ When `\fillTypeGrid` is in effect, `topline=true` writes an additional line above the top most line, this line is not numbered.

```
\define@boolkey{eqefillLines}{topline}[true]{}
\KV@eqefillLines@toplinefalse
```

`gridtype=`⟨`line|dash|dots`⟩ The `gridtype` key determines the line style for the grid, choices are `line` (the default), `dash`, and `dots`. This key also sets the line style when the `outlineonly` key is specified.

```
\let\eqe@usedeffboxrule\eqe@NO % dps20
\let\gridtypeselected\@empty % dps26
\define@choicekey+{eqefillLines}{gridtype}[\val\nr]%
  {line,dash,dots}[line]{\edef\gridtypeselected{\val}% dps26
  \ifcase\nr\relax
    \let\gridHLineFill\eqWriteLineFill
    \let\gridVLineFill\eqWriteLineVFill
  \or
    \let\gridHLineFill\eqWriteLineDashFill
    \let\gridVLineFill\eqWriteLineDashVFill
  \or
    \let\gridHLineFill\eqWriteLineDots
    \let\gridVLineFill\eqWriteLineVDots
    \let\eqe@usedeffboxrule\eqe@YES %dps20
  \fi
}{\PackageWarning{aeb}{Bad choice for numbers, permissible values
  are line, dash, and dots. Try again}}
\let\gridHLineFill\eqWriteLineFill
\let\gridVLineFill\eqWriteLineVFill
```

`fltype=`⟨`line|dash|dots|blank`⟩ The `fltype` key determines the type of line style for the horizontal rendering. This key simply executes the various command versions.

```
\define@choicekey+{eqefillLines}{fltype}[\val\nr]%
  {line,dash,dots,blank,grid}[line]{\ifcase\nr\relax
    \fillTypeHRule\or
    \fillTypeDashLine\or
    \fillTypeDots
    \let\eqe@usedeffboxrule\eqe@YES\or % dps20
    \fillTypeBlankLine\or
    \fillTypeGrid\fi
}{\PackageWarning{aeb}{Bad choice for numbers, permissible values
   are line, dash, dots, blank, and grid. Try again}}
```

`align=`⟨`left|default`⟩ Another convenience key, `align` simply executes the command versions.

```
\define@choicekey+{eqefillLines}{align}[\val\nr]%
  {default,left}[default]{\ifcase\nr\relax
    \fillerLinesAlignDef\or
    \fillerLinesOnLeftMargin\fi
}{\PackageWarning{aeb}{Bad choice for numbers, permissible values
   are line, dash, dots, blank, and grid. Try again}}
\define@choicekey*{eqefillLines}{equalcells}[\val\nr]%
  {true,false}[true]{\ifcase\nr\relax
    \equalCellSizesOn\or
    \equalCellSizesOff\fi
}
```

`outlineonly*=`⟨`true|false`⟩ outlines the work area only, does not create horizontal or vertical lines, other than the lines to outline the region. Obeys the `gridtype` key.

```
\define@boolkey{eqefillLines}{outlineonly}[true]{}
\KV@eqefillLines@outlineonlyfalse
\define@choicekey{eqefillLines}{outlineonly*}{true,false}[true]{% dps26
  \@nameuse{KV@eqefillLines@outlineonly#1}%
  \def\eqefillLines@outlineonlystar{#1}}%
\let\eqefillLines@outlineonlystar\@empty
```

`bgonly*=`⟨`true|false`⟩ colors the work area with a color of your choice, see `bgcolor` below.

```
\define@boolkey{eqefillLines}{bgonly}[true]{} % dps11
\KV@eqefillLines@bgonlyfalse
\define@choicekey{eqefillLines}{bgonly*}{true,false}[true]{% dps26
  \@nameuse{KV@eqefillLines@bgonly#1}%
  \def\eqefillLines@bgonlystar{#1}}
\let\eqefillLines@bgonlystar\@empty
```

`bgcolor=`⟨*color-spec*⟩ The color to be used when `bgcolor` is in force. If no color has been assigned, the color will be white.

```
\define@key{eqefillLines}{bgcolor}[]{\def\eqe@BGColor{#1}} % dps11
\let\eqe@BGColor\@empty
```

`\fillLinesNumFmt{`⟨*cmds*⟩`}` is used to specify the format for line numbers. Within the ⟨*cmds*⟩ argument, use `#1` to indicate the placement of the number. For example, the code `\fillLinesNumFmt{\textbf{#1}}` puts the line numbers in bold font.

```
\def\fillLinesNumFmt#1{\def\eqe@fillLinesNumFmt##1{\def\flnum{##1}#1}}
\fillLinesNumFmt{\flnum} % set to default values
\define@key{eqefillLines}{numfmt}[]{\fillLinesNumFmt{#1}} % dps21
\define@key{eqefillLines}{linegap}[14pt]{\def\fl@wlspacing{#1}} % dps21
\def\fl@wlspacing{14pt}
```

`\setFillLinesFmt{`⟨*KV-pairs*⟩`}` Set the format of the fill lines, use the keys from the `eqefillLines` defined and described above. Changes outside a group are global. An example of the usage is given here: `\setFillLinesFmt{numbers=left,color=red}`.

Other important notes: To produce horizontal lines only, use `fltype=line|dash|dots`, to produce blank space, use `fltype=blank` and to produce a grid `fltype=grid`.

```
\newcommand{\setFillLinesFmt}[1]{\def\@rgi{#1}%
  \let\eqe@usedeffboxrule\eqe@NO %dps20
  \let\gridtypeselected\@empty
  \ifx\@rgi\@empty
    \setkeys{eqefillLines}{numbers,numbersep,color,
      gridtype,fltype,align}%
  \else
    \let\is@outlineonly\eqe@NO\let\is@bgonly\eqe@NO
    \ifKV@eqefillLines@outlineonly\let\is@outlineonly\eqe@YES\fi
    \ifKV@eqefillLines@bgonly\let\is@bgonly\eqe@YES\fi
    \setkeys{eqefillLines}{#1}%
    \eqWLSpacing{\fl@wlspacing}%
    \ifx\makeVgrid\relax
      \KV@eqefillLines@toplinefalse
%      \forceEqualCellsfalse % dps21
      \equalCellSizesOff % dpsj15
      \ifx\is@bgonly\eqe@YES % carryover from prev prob
        \ifKV@eqefillLines@bgonly
          \KV@eqefillLines@bgonlyfalse
        \else
          \PackageWarning{eqexam}{You must specify fltype=grid for
          bgonly to work.\MessageBreak
          Ignoring this key for now}%
          \KV@eqefillLines@bgonlyfalse
          \let\is@bgonly\eqe@NO
        \fi
      \else % not continuation
        \ifKV@eqefillLines@bgonly
          \PackageWarning{eqexam}{You must specify fltype=grid for
          bgonly to work.\MessageBreak
          Ignoring this key for now}%
          \KV@eqefillLines@bgonlyfalse
          \let\is@bgonly\eqe@NO
        \fi
      \fi
% same for outline
      \ifx\is@outlineonly\eqe@YES % carryover from prev prob
        \ifKV@eqefillLines@outlineonly
          \KV@eqefillLines@outlineonlyfalse
        \else % dps18
          \PackageWarning{eqexam}{You must specify fltype=grid for
          outlineonly to work.\MessageBreak
          Ignoring this key for now}%
          \KV@eqefillLines@outlineonlyfalse
          \let\is@outlineonly\eqe@NO
        \fi
      \else % not continuation
        \ifKV@eqefillLines@outlineonly
          \PackageWarning{eqexam}{You must specify fltype=grid for
          outlineonly to work.\MessageBreak
          Ignoring this key for now}%
          \KV@eqefillLines@outlineonlyfalse
          \let\is@outlineonly\eqe@NO
        \fi
      \fi
      \let\gridHLineFill\eqWriteLine
      \let\gridVLineFill\eqWriteLine
    \else
      \ifx\gridtypeselected\@empty % dps26
        \let\gridHLineFill\eqWriteLineFill
        \let\gridVLineFill\eqWriteLineVFill
      \fi
    \fi
  \fi
}
```

The next two definitions place line numbers on the left and right, respectively.

```
\def\eqe@@numLinesL{\ifKV@eqefillLines@topline\ifnum\@tempcntb>0  % dps5
  \smash{\llap{\tiny
    \eqe@fillLinesNumFmt{\the\@tempcntb}\hglue\eqe@numbersep\relax}}\fi
  \else\smash{\llap{\tiny
    \eqe@fillLinesNumFmt{\the\@tempcntb}\hglue\eqe@numbersep\relax}}\fi}
\def\eqe@@numLinesR{\smash{\rlap{\tiny
  \hglue\eqe@numbersep\relax\eqe@fillLinesNumFmt{\the\@tempcntb}}}}
```

The `\fillLinesLineWidth` is a special command used for adjusting the length of the horizontal lines, and is used only when `\fillTypeGrid` is in effect and when `\forceEqualCells` is true.

```
\let\fillLinesLineWidth\@empty
```

A switch to signal that a grid has broken across pages. This is only set to true when the `topline=true` is specified through `\setFillLinesFmt`.

```
\newif\ifgridpgbrk\gridpgbrkfalse % dps7
```

`\vspaceFillerLines`{⟨*dimen*⟩} This is an internal command, though we declare is publicly. It writes the horizontal lines (lines, dots, dashes) when the `solution` environment uses a ⟨*dimen*⟩ to specify space (as opposed to using the `nLines` key).

```
\newif\ifwriteVertic@lFLines \writeVertic@lFLinestrue % dps16
\def\fl@set@nnotContStrSkip{\def\@nnotContStrSkip{\vskip6pt}} % dpsj23
\let\p@ssToFLs\relax
\newif\iffl@firstpass % dps27
\newcommand{\vspaceFillerLines}[1]{\begingroup\offinterlineskip %dps1
  \global\fl@firstpasstrue % dps27
  \fl@set@nnotContStrSkip   % dpsj20
  \p@ssToFLs\if@eqlinedfiller
    \def\eqe@next{\vspaceFillerLines@i{#1}}\else
    \expandafter\def\expandafter\eqe@next
    \expandafter{\expandafter\endgroup\p@ssToFLs}\fi
    \global\let\p@ssToFLs\relax\eqe@next}
\def\vspaceFillerLines@i#1{\advance\wlVspace-\flfboxrule % dps25
  \ifx\eqe@usedeffboxrule\eqe@YES\flfboxrule=.4pt\fi  % dps20
  \ifKV@eqefillLines@outlineonly\KV@eqefillLines@toplinetrue %dps10
    \KV@eqefillLines@bgonlyfalse % dps11
%    \forceEqualCellsfalse % dpsj15
    \equalCellSizesOff\let\makeVgrid\eqe@makeVgrid\fi %dps10
  \ifKV@eqefillLines@bgonly\KV@eqefillLines@toplinetrue
    \KV@eqefillLines@outlineonlyfalse\flfboxrule=.4pt % dps20
%    \forceEqualCellsfalse % dpsj15
    \equalCellSizesOff\let\makeVgrid\eqe@makeVgrid\fi %dps11
  \parindent0pt\relax\parskip0pt
  \@tempdima\wlVspace\eqetmplengtha0pt
  \@tempcnta=0 \@tempcntb=1
```

\eqe@fillwidth is the width of the line, it is \linewidth plus a fudge factor, which is determined dynamically by \eqe@wrtLineKernal.

```
  \def\eqe@fillwidth{\linewidth+\eqetmplengtha}%
```

If we are creating a grid (\fillTypeGrid), and \forceEqualCells is true, we modify the length of the line to make equal cell sizes. \forceEq@lCells determines the value of \fillLinesLineWidth.

```
  \eqe@wrtLineKernal
  \setbox\eqe@tempbox\hbox{\hb@xt@0pt{\hss.\hss}}\ht\eqe@tempbox.33pt
    \dp\eqe@tempbox0pt % dps16
  \ifx\makeVgrid\relax
%    \forceEqualCellsfalse % dpsj15
    \equalCellSizesOff
  \else % dps2
    \let\eqWriteLine\gridHLineFill
    \forceEqu@lCells % dps2
    \ifx\flEqu@lLineWidth\@empty\else
      \linewidth=\flEqu@lLineWidth\relax\fi
    \ifKV@eqefillLines@bgonly\else
      \ifKV@eqefillLines@topline\bgroup\@tempcntb=0% dps5
        \makebox[0pt][l]{\eqe@x
          \makebox[\eqe@fillwidth]{\eqWriteLine}}\egroup
    \fi\fi % dps5
  \fi
  \settowidth{\eqetmplengthb}{\eqe@decPointPrb}\def\vfGo@l{#1}%
  \ifx\makeVgrid\relax\else
    \def\priorPNPAction{\xdef\fl@nRows{\the\@tempcnta}% dps27
      \if@eqalignfilllinestoleft\eqe@x\fi % dpsj12
      \makeVgrid\priorP@geBre@kMsg\global\fl@firstpassfalse}% dps4
    \def\postPNPAction{\global\gridpgbrktrue % dps27
    \setlength{\@tempdima}
```

```
      {(\vfGo@l+\wlVspace)-\depthtodate}% dps25
    \edef\vfGo@l{\the\@tempdima}\@tempdima=\wlVspace\relax
    \@tempcnta=0\relax}% dps18
  \fi
  \@whiledim\@tempdima<\vfGo@l\relax\do
    {\edef\depthtodate{\the\@tempdima}%
    \ifx\eqe@insert@more@content\relax\ifx\makeVgrid\relax
      \priorP@geBre@kMsg\fi\fi %dps15
    \eq@insertContAnnot\vskip\wlVspace\eqe@x
```

Draw the horizontal line: a rule, dotted line, dashed line

```
    \ifKV@eqefillLines@bgonly\vglue\flfboxrule\else % dpsj11 dpsj21
      \ifKV@eqefillLines@outlineonly\vglue\flfboxrule\else %dps10
         \makebox[0pt][l]{\eqe@numLinesL
            \makebox[\eqe@fillwidth]{\eqWriteLine}\eqe@numLinesR}\fi
    \fi
    \advance\@tempcnta1\relax\advance\@tempcntb1\relax
    \edef\depthtodate{\the\@tempdima}%
    \addtolength{\@tempdima}{\wlVspace}%\par dps25
```

At the conclusion of the `\do` loop, we insert `\makeVgrid`, which is either `\relax` or is `\eqe@makeVgrid`, the latter is a complicated command to draw vertical lines across the horizontal ones, and to perform other tasks.

```
  }\xdef\fl@nRows{\the\@tempcnta}\makeVgrid % dps27
  \ifx\makeVgrid\relax\else %dps15
```

When there is insufficient space to accommodate the content, we place a small blank box in the lower left corner, put a information message in the log, and reset the color.

```
    \ifvoid\eqe@nskeyflsplit\else\PackageWarning{eqexam}
      {Some material from a solution environment does not\MessageBreak
       appear. Increase space allotted}{\normalcolor
       \llap{\smash{\rule{5pt}{5pt}}}}\aftergroup
       \reset@color\fi % dpsj24
  \fi
```

We conclude by adding some vertical space to better align the `workarea` environment. When a fill type is used, the space created by not be exactly ⟨*dimen*⟩ the dimension specified.

```
  \setlength{\@tempdima}{\vfGo@l-\depthtodate}\vskip\@tempdima
% \vskip\flfboxrule\vskip3pt
  \dimen0=\sameVspace\relax
  \dimen2=\fboxrule\advance\dimen0by\dimen2
  \xdef\sameVspace{\the\dimen0 }\endgroup
  \global\flfrstsplittrue
}
```

`\eq@linesXPgs` Does the same as `\vspaceFillerLines` but is used instead of that command when the optional argument for `solution` specified a value for `nLines`. Code comments are the same as those just given for `\vspaceFillerLines`.

```
\def\eq@linesXPgs{\@tempdimb\wlVspace % dpsj20
  \advance\@tempdimb-\flfboxrule
  \@tempdimb=\soln@keys@nLines\@tempdimb
  \advance\@tempdimb\flfboxrule
  \expandafter\vspaceFillerLines\expandafter{\the\@tempdimb}}
```

\eqe@wrtLineKernal, based on a case analysis, sets the register \eqetmplengha and makes some horizontal glue adjustments as well.

```
%\def\eq@linesXPgs{\@tempdimb\wlVspace
%  \@tempdimb=\soln@keys@nLines\@tempdimb
%  \expandafter\vspaceFillerLines\expandafter{\the\@tempdimb}}
\def\eqe@wrtLineKernal{\let\eqe@x\relax\let\eqe@y\relax  % dpsj9
  \ifx\solutionparshape\@empty % not lead-in
    \if@eqalignfilllinestoleft % align on left
      \setlength{\eqetmplengtha}{\labelwidth+\eqemargin}%
      \setlength{\eqetmplengtha}
        {\eqetmplengtha+\eqetmplengthb}%
      \ifwithinparts
        \ifx\istabularexer\eq@YES  % tabular mode
          \setlength{\eqetmplengtha}{\prtsIndntSep}% dps27
          \eq@tmplengthA\eqetmplengtha
          \def\eqe@x{\hglue-\prtsIndntSep}% dps27
          \ifKV@eqefillLines@outlineonly\let\eqe@y\eqe@x\fi
        \else
          \settowidth{\eqetmplengtha}{\eqe@prtsepPrb}%
          \setlength{\eqetmplengtha}{\eqetmplengtha+\widthOfParts}% dps9
          \setlength{\eq@tmplengthA}{\eqetmplengtha}% dps9
          \def\eqe@x{\hspace*{-\eqetmplengtha}\hglue-\flfboxrule}% dps27
        \fi
      \else % not parts, still align left
        \def\eqe@x{\hspace*{-\eqemargin}}%
          \eqetmplengtha\eqemargin
          \eq@tmplengthA\eqemargin
      \fi
    \else % not lead-in, not align on left
      \ifwithinparts
        \setlength{\eqetmplengtha}{\labelwidth}%
        \setlength{\eqetmplengtha}
          {\eqetmplengtha+\eqetmplengthb}%
        \ifx\istabularexer\eq@YES  % tabular mode
          \eqetmplengtha0pt
          \eq@tmplengthA\eqetmplengtha % dps27
        \else %dps2
          \def\eqe@x{\hglue-\flfboxrule}% dps4
          \eqetmplengtha0pt
          \eq@tmplengthA\eqetmplengtha
        \fi
      \else % not parts, not align left
        \eqetmplengtha0pt
        \eq@tmplengthA\eqetmplengtha
      \fi
    \fi
  \else % lead-in
    \if@eqalignfilllinestoleft
      \setlength{\eqetmplengtha}{\leadinIndentPrtSep}% dps27
      \eq@tmplengthA\eqetmplengtha
    \else
      \eqetmplengtha0pt
      \eq@tmplengthA\eqetmplengtha
    \fi
  \fi
}
```

\equalCellSizesOn
\equalCellSizesOff

set the switch `\ifforceEqualCells` to true, while `\equalCellSizesOff` sets the same switch to false. Additionally, the latter command resets `\fillLinesLineWidth` to its default value of `\@empty`. It is therefore important not to use the switch directly, but to use these two convenience commands.

```
\newif\ifforceEqualCells
\def\equalCellSizesOn{\global\let\flEqu@lLineWidth\@empty
    \global\forceEqualCellstrue}
\def\equalCellSizesOff{\global\let\flEqu@lLineWidth\@empty
    \global\forceEqualCellsfalse}
\equalCellSizesOff
```

`\forceEqu@lCells` determines the value of `\fillLinesLineWidth`. It successively adds `\wlVspace` until it exceeds `\linewidth`, it then subtracts off one `\wlVspace` to get the value for `\fillLinesLineWidth`.

```
\def\forceEqu@lCells{\bgroup
  \ifforceEqualCells
    \dimen6\wlVspace % reduced from \vspaceFillerLines
    \advance\dimen6\flfboxrule
    \dimen2=\z@
    \dimen4\linewidth
    \advance\dimen4by-\flfboxrule
    \if@eqalignfilllinestoleft % dps8
      \advance\dimen4\eq@tmplengthA
    \fi
    \@tempcnta0
    \@whiledim \dimen2 < \dimen4 \do{%
      \advance\@tempcnta1
      \advance\dimen2by\dimen6
    }%
    \advance\@tempcnta-1
    \advance\dimen2-\dimen6
    \advance\dimen2\flfboxrule
    \advance\dimen2by-\eq@tmplengthA
    \xdef\flEqu@lLineWidth{\the\dimen2}%
  \fi
\egroup}
\let\eqe@insert@more@content\relax
```

```
\def\gobtodot#1.#2\@nil{\def\intPrt{#1}\def\fracPrt{#2}}
```

\eqe@makeVgrid A command that is executed when the document author expands \fillTypeGrid. The command itself draws vertical rule lines, which intersect with horizontal rule lines to form a grid. When \fillTypeGrid is active, \makeVgrid is \let to \eqe@makeVgrid. \makeVgrid appears twice in each of the macros \vspaceFillerLines and \eq@linesXPgs. The first instance of \makeVgrid draws the vertical line, assuming there is no page break. If there is a page break, the second instance of \makeVgrid continues to draw vertical lines with the correct height and number. To make the grid break across a page, the low level command \makeRoomForProb was modified; now prior and post actions (\priorPNPAction and \priorPNPAction) can be defined.

The \dbMrk is used to debug this portion of the code, view the debugging marks by setting \ifeqedb to true.

```
\def\dbMrk#1{\textsf{\footnotesize#1}}
\@ifundefined{ifeqedb}{\newif\ifeqedb \eqedbfalse}{}
```

\fillerCustomBg{⟨*alt-bg*⟩} Add a custom background feature, only available when the key bgonly is active. The argument ⟨*alt-bg*⟩ uses #1 and #2 to refer to the width and height of the work area. Use \resetFillerCustomBg to reset the method of coloring the background to its default method.

\resetFillerCustomBg

```
\def\fillerCustomBg#1{\def\fillerBgIm@ge##1##2{#1}}
\def\resetFillerCustomBg{\let\fillerBgIm@ge\@empty}
\resetFillerCustomBg
```

\gridIndentAdj{⟨*dimen*⟩} is an adjustment in the horizontal positioning of the vertical lines of the grid. Under normal conditions, \gridIndentAdj is not needed. If the exam environment is embedded in environments that change some basic parameters, the alignment of the grid may not be right, in which case you can use \gridIndentAdj to shift the grid horizontially.

```
\def\gridIndentAdj#1{\def\@rgi{#1}\ifx\@rgi\@empty
  \let\gridIndent@dj\relax\else % dps4
  \setlength{\@tempdima}{#1}\edef\gridIndent@dj{\the\@tempdima}\fi}
\let\gridIndent@dj\relax
\def\eqe@makeVgrid{\begingroup
  \@tempcnta=\fl@nRows\relax
  \advance\wlVspace\flfboxrule % dps25
  \setlength{\@tempdima}{\eqe@fillwidth}%
  \setlength{\@tempdimb}{\wlVspace}%
  \edef\eqe@tmp{\strip@pt\@tempdimb}%
  \expandafter\gobtodot\eqe@tmp.\@nil
  \divide\@tempdima by\intPrt\relax
  \dimen2=\@tempdima
  \dimen4=1pt
  \advance\dimen2 by 0.5\dimen4
  \divide\dimen2 by \dimen4
  \multiply\dimen2 by \dimen4
  \edef\eqe@tmp{\strip@pt\dimen2}%
  \expandafter\gobtodot\eqe@tmp.\@nil
```

```
\advance\@tempcnta-1\relax % dps21
\multiply\@tempdimb\@tempcnta
\edef\eqe@HOfVrule{\the\@tempdimb}% dps7
\ifKV@eqefillLines@topline % dps21
  \advance\@tempdimb\wlVspace
  \advance\@tempdimb\flfboxrule
  \edef\eqe@HOfVruleFrstRow{\the\@tempdimb}%
  \def\eqe@setVRule{\vbox
    to\eqe@HOfVruleFrstRow\relax{\gridVLineFill}}% dps8
\else
  \def\eqe@setVRule{\vbox
    to\eqe@HOfVrule\relax{\gridVLineFill}}% dps8
\fi
\eqe@tempcnta=0
\dimen2=\wlVspace
\count4=\intPrt\relax
\setlength{\dimen4}{\eqe@fillwidth}%
\multiply\dimen2by\count4
\@whiledim \dimen2 > \dimen4 \do{%
  \advance\count4by-1\relax
  \edef\intPrt{\the\count4 }%
  \dimen2\wlVspace
  \multiply\dimen2by\count4
}%
\dimen2\wlVspace
\if@eqalignfilllinestoleft % dps27
  \let\fl@oSmash\smash\let\fl@iSmash\relax
\else
  \let\fl@iSmash\smash\let\fl@oSmash\relax
\fi
\fl@oSmash{%
\makebox[0pt][l]{\makebox[\eqe@fillwidth][l]{\fl@iSmash{% dps27
  \ifx\gridIndent@dj\relax\else
    \hglue\gridIndent@dj\relax\fi
  \eqe@y\ifeqedb\rlap{\dbMrk{D}}\fi % dpsj9 \eqe@y (fudge)
  \rlap{\ifeqedb\rlap{\dbMrk{B}}\fi
  \ifwriteVertic@lFLines % dps16
    \setlength{\@tempdima}{\eqe@fillwidth}%
    \ifKV@eqefillLines@bgonly % dps11
      \ifx\fillerBgIm@ge\@empty
        \ifx\eqe@BGColor\@empty\else
          \rlap{\textcolor{\eqe@BGColor}
            {\rule{\eqe@fillwidth}{\eqe@HOfVruleFrstRow}}}\fi
      \else
        \rlap{\fillerBgIm@ge{\eqe@fillwidth}{\eqe@HOfVruleFrstRow}}\fi
    \else
      \dimen2=\wlVspace
      \dimen0=0pt
      \@whilenum\eqe@tempcnta<\count4\do{\rlap{\hglue\dimen0
        \ifKV@eqefillLines@outlineonly %dps10
```

```
            \ifnum\eqe@tempcnta=0 \else\let\gridVLineFill\vfill\fi\fi
          \textcolor{\eq@WriteLineColor}{\eqe@setVRule % dps27
          \ifeqedb\rlap{\advance\eqe@tempcnta by1\relax\scriptsize
             \raisebox{1pt}{\kern1pt\the\eqe@tempcnta}}\fi}}%
          \advance\dimen0by\dimen2
          \advance\eqe@tempcnta by1
        }% dps27
      \fi % dps11
      \advance\dimen2by\flfboxrule
      \ifKV@eqefillLines@bgonly\else % dps11
        \rlap{\hglue\@tempdima
          \textcolor{\eq@WriteLineColor}{\eqe@setVRule
          \ifeqedb\rlap{\thinspace\dbMrk{R}}\fi}}%
        \ifKV@eqefillLines@outlineonly %dps10
          \makebox[0pt][l]{\makebox[\eqe@fillwidth]{\eqWriteLine}}%dps10
        \fi
        \ifKV@eqefillLines@topline% dps7
          \ifgridpgbrk\@tempdima\eqe@HOfVruleFrstRow\relax % dps21
            \raise\@tempdima\hbox{\makebox[0pt][l]{%
              \makebox[\eqe@fillwidth]{\eqWriteLine}}}%
              \global\gridpgbrkfalse
          \fi
        \fi
      \fi
      \ifx\eqefillLines@outlineonlystar\@empty\else % dps26
        \global\let\eqefillLines@outlineonlystar\@empty
        \global\KV@eqefillLines@outlineonlyfalse
      \fi
      \ifx\eqefillLines@bgonlystar\@empty\else % dps26
        \global\let\eqefillLines@bgonlystar\@empty
        \global\KV@eqefillLines@bgonlyfalse
      \fi
    \fi % ifwriteVertic@lFLines
    \eqe@insert@more@content
    }% rlap
  }% smash (fl@iSmash)
}}}% makeboxes and \fl@oSmash
  \endgroup
}% \eqe@makeVgrid
```

In preparation for the `flextended` option, we define several commands and one environment that emit a package warning if the `flextended` option is not taken.

```
\def\turnfl@nskeyMsg{\PackageWarning{eqexam}
  {You must first take the flextended\MessageBreak
   option for this command to have any\MessageBreak effect}}
\def\turnflanskeyOn{\turnfl@nskeyMsg}
\def\priorw@Msg{\PackageWarningNoLine{eqexam}
  {The priorworkarea environment does nothing\MessageBreak
   without the flextended option. All such content\MessageBreak in these
   environments are absorbed}\global\let\priorw@Msg\@empty}
```

```
\newenvironment{priorworkarea}{\priorw@Msg
    \setbox\eqe@nskeyflsplit\vbox\bgroup}
    {\egroup\setbox\eqe@nskeyflsplit\box\voidb@x}
\let\turnflanskeyOff\turnflanskeyOn
\newif\ifflfrstsplit \flfrstsplittrue
\newif\ifeqe@flnosolns \eqe@flnosolnsfalse
\def\turnflnosolnsOn{\eqe@flnosolnstrue\turnflanskeyOn}
\def\turnflnosolnsOff{\eqe@flnosolnsfalse\turnflanskeyOn}
%\let\turnflnosolnsOn\turnflanskeyOn
%\let\turnflnosolnOff\turnflanskeyOn
\newdimen\flfboxrule \flfboxrule=.4pt
\def\tweakBreakPoint#1{\def\@rgi{#1}\ifx\@rgi\@empty
  \gdef\twe@kBre@kPoint{0pt}\else
  {\setlength{\@tempdima}{#1}%
    \xdef\twe@kBre@kPoint{\the\@tempdima}}\fi}
\def\twe@kBre@kPoint{0pt}
```

When flextended option is not loaded, we make a minimal definition of \eq@b@ddCodeSpecial. This redefinition is needed when bgonly is in effect.

```
\def\eq@b@ddCodeSpecial#1{\if@eqlinedfiller
    \ifKV@eqefillLines@bgonly\leavevmode
        \vskip-\baselineskip\kern\lineskip\fi\fi#1}
```

\flSeparateCutNames For debug, or whatever reason, \flSeparateCutNames gives each flx cut file a distinct name.

```
\def\flSeparateCutNames{\def\fl@CutName{flx\fl@Cnt-\jobname.cut}}
⟨/package⟩
⟨*flextended⟩
```

14.2 Utility commands used with the flextended option

```
\def\fls@vebaselinelineskip{% dps22
  \xdef\flbaselineskip{\the\baselineskip}}
\def\fl@Cnt{0}
\def\fl@CutName{flx-\jobname.cut}
\def\flwriteexsol@fter{% dpsj23
  \ifx\exsolafter\@empty
    \immediate\write\verbatim@out{\string\ignorespaces}% dpsj23
  \else
    \immediate\write\verbatim@out{\string\exsolafter
      \string\space\string\ignorespaces}%
  \fi
}
\def\fl@getcontent#1{\begingroup
  {\count0=\fl@Cnt\relax
    \advance\count0by1\relax
    \xdef\fl@Cnt{\the\count0 }}%
%  \let\save@wlog\wlog\let\wlog\@gobble
%  \let\save@message\message\let\message\@gobble
  \global\let\verbatim@out\CommentStream
  \immediate\openout\verbatim@out=#1
  \ifwithinparts
    \ifx\istabularexer\eq@YES
      \immediate\write\verbatim@out{\string\hfuzz\string\hsize}%
      \immediate\write\verbatim@out{\string\minipage[t]{\string\hsize}}%
      \ifanswerkey
        \immediate\write\verbatim@out{\string\expandafter
          \string\noindent\string\eqSolnExCmds}% dps23
        \flwriteexsol@fter
      \else
        \immediate\write\verbatim@out{\string\expandafter
          \string\noindent\string\priorWorkAreaCmds
          \string\ignorespaces}% dps23
      \fi
    \else
      \ifanswerkey\flwriteexsol@fter\fi
    \fi
  \else
    \ifanswerkey\flwriteexsol@fter\fi
  \fi
  \verbatimwrite
}
\def\fl@vsplitandplace#1{%
  \iffl@firstpass
  \global\setbox\eqe@nskeyflsplit\color@vbox
    \normalbaselines
    \predisplaypenalty=-50
    \postdisplaypenalty=-50
    \setlength{\hsize}{\eqe@fillwidth}%
    \expandafter\noindent % dpsj23
    \ifanswerkey\expandafter\eqSolnExCmds\else
      \expandafter\priorWorkAreaCmds\fi
    \ignorespaces\input{#1}\fls@vebaselinelineskip
    \color@endbox\global\fl@firstpassfalse\fi
  \@tempdima\eqe@HOfVrule\relax
```

I'm getting a black box when there is one line left on the second page and there is less than one line of text. The following three lines attempt to get that last line to appear when there is enough room for it.

```
  \advance\@tempdima\flfboxrule
  \advance\@tempdima\lineskip
  \ifflfrstsplit\else\advance\@tempdima\wlVspace\fi
  \advance\@tempdima\maxdepth
  \advance\@tempdima by\twe@kBre@kPoint\relax
  \eq@tmplengthB\@tempdima % target height
  \splittopskip\wlVspace
  \splitmaxdepth\maxdepth
  \advance\splitmaxdepth by\twe@kBre@kPoint\relax
  \setbox\eq@pointbox=\copy\eqe@nskeyflsplit
  \setbox\eqe@nskeyfltop=\vsplit\eq@pointbox to \@tempdima
```

```
\setbox\eqe@nskeyfltop=\vbox{\unvbox\eqe@nskeyfltop}%
\ifvoid\eq@pointbox\else
  \setlength{\@tempdimb}
      {\eq@tmplengthB-\dp\eqe@nskeyfltop}%
  \ifdim\@tempdimb<0pt
    \@tempdimb=-\@tempdimb
    \ifdim\@tempdimb>\maxdepth
      \advance\@tempdima-\flbaselineskip\relax
    \else
    \fi
  \else
\setbox\eq@pointbox=\copy\eqe@nskeyflsplit
\advance\@tempdima by\flbaselineskip\relax
\setbox\eqe@nskeyfltop=\vsplit\eq@pointbox to \@tempdima
\setbox\eqe@nskeyfltop=\vtop{\unvbox\eqe@nskeyfltop}%
\setlength{\@tempdimb}{\eq@tmplengthB-\dp\eqe@nskeyfltop}%
  \ifdim\@tempdimb<0pt % too much
      \advance\@tempdima-\flbaselineskip\relax
    \else
    \fi
  \fi
\fi
\splittopskip\wlVspace
\lineskip1pt
\setbox\eq@pointbox=\copy\eqe@nskeyflsplit
\global\setbox\eqe@nskeyfltop=\vsplit\eqe@nskeyflsplit to \@tempdima
\global\setbox\eqe@nskeyfltop\vtop{\unvbox\eqe@nskeyfltop\vskip0pt}%
\ifvoid\eqe@nskeyflsplit
  \ifeqedb\llap{Bot}\fi
  \ifeqedb\raise\eqe@HOfVrule\llap{Top}\fi
  \ifeqedb\llap{V\qquad}\fi
  \ifflfrstsplit\global\flfrstsplitfalse
    \ifeqedb\llap{FP\hskip.5in}\fi
    \rlap{\raisebox{\eqe@HOfVrule+\flfboxrule+\lineskip}
      {\vtop{\vsize=\eqe@HOfVrule\relax
      \unvbox\eqe@nskeyfltop\vfil}}}\else
    \ifeqedb\llap{SP\hskip.5in}\fi
    \rlap{\raisebox{\eqe@HOfVrule+\flfboxrule+\wlVspace+\lineskip}
      {\vtop{\vsize=\eqe@HOfVrule\relax
      \unvbox\eq@pointbox\vfil}}}%
      \global\setbox\eqe@nskeyflsplit\box\voidb@x
  \fi
\else
  \ifeqedb\llap{Bot}\fi
  \ifeqedb\raise\eqe@HOfVrule\llap{Top}\fi
  \ifeqedb\llap{nV\qquad}\fi
  \ifflfrstsplit\global\flfrstsplitfalse
  \ifeqedb\llap{FP\hskip.5in}\fi
    \rlap{\raisebox{\eqe@HOfVrule+\flfboxrule+\lineskip}
      {\vtop{\vsize=\eqe@HOfVrule\relax
```

```
        \unvbox\eqe@nskeyfltop\vfil}}}\else
      \ifeqedb\llap{SP\hskip.5in}\fi
      \rlap{\raisebox{\eqe@HOfVrule+\flfboxrule+\wlVspace+\lineskip}
        {\vtop{\vsize=\eqe@HOfVrule\relax
```

To have the filler lines/grid with prior and solution content to space more than one page, we unbox \eqe@anskeyfltop but do not void it.

```
        \unvbox\eqe@nskeyfltop\vfil}}}%
%        \unvbox\eq@pointbox\vfil}}}%
%        \global\setbox\eqe@nskeyflsplit\box\voidb@x
    \fi
  \fi
  \setbox\eq@pointbox\box\voidb@x
% \gdef\twe@kBre@kPoint{0pt}% dpsj21
}
```

14.3 Filler lines with answerkey option

When using the flextended option and the \turnflanskeyOn command, we always use the grid filler type. In the case of the basic filler types of line, dash, dots, and blank, the code threads passes to \makeVgrid, but in the case of the answerkey option, \makeVgrid writes no vertical lines. Hopefully, the spacing between the statement of the question and the solution are then consistent.

```
\let\turnfl@nskeyOnOff\relax
```

\turnflanskeyOn The \turnflanskeyOn command turns *on* the feature of superimposing the solution over the filler lines. Does nothing unless the answerkey option is in force.

```
\def\turnflnosolnsOn{\eqe@flnosolnstrue}
\def\turnflnosolnsOff{\eqe@flnosolnsfalse}
\def\turnflanskeyOn{%
  \def\turnfl@nskeyOnOff{%
    \ifanswerkey
      \ifx\makeVgrid\relax % already in grid mode, do nothing
        \ifwriteVertic@lFLines\KV@eqefillLines@toplinefalse\fi % dps16
        \writeVertic@lFLinesfalse
        \let\makeVgrid\eqe@makeVgrid % dpsj5
      \fi % dps17
      \let\eqe@insert@more@content\eqe@insertSolns % dpsj5
      \eqe@flextendedtrue % dps16
      \fillTypeGrid % dps16
      \global\let\eq@insertverticalspace\eq@YES
      \global\vspacewithsolnstrue
    \fi
  }%
}
```

\turnflanskeyOff The \turnflanskeyOff command turns *off* the feature of superimposing the solution over the filler lines. Does nothing unless the answerkey option is in force.

```
\def\turnflanskeyOff{%
  \def\turnfl@nskeyOnOff{%
    \ifanswerkey
```

```
      \writeVertic@lFLinestrue
      \eqe@flextendedfalse
      \let\eq@insertverticalspace\eq@NO
      \@eqlinedfillerfalse
      \global\let\eqe@insert@more@content\relax
      \global\let\eq@insertverticalspace\eq@YES
      \ifvspacewithsolns\else
      \global\vspacewithsolnsfalse\fi
      \global\displayworkareafalse
    \fi
  }%
}
\turnflanskeyOff
```

```
\let\p@ssToSolns\relax % dps30
\def\eqe@insertSolns{\clubpenalty=50
  \widowpenalty=50 \vbadness=10000
  \fl@vsplitandplace{s\fl@CutName}}
\def\eq@b@ddCodeSpecial#1{\global\gridpgbrkfalse % dpsj15
  \global\let\eq@e@ddCodeSpecial\relax
  \gdef\eqe@flnexti{#1}% dpsj5
  \if@eqlinedfiller
    \ifKV@eqefillLines@bgonly\leavevmode
        \vskip-\baselineskip\kern\lineskip\fi % dpsj11
    \ifKV@eqefillLines@outlineonly\leavevmode
        \vskip-\baselineskip\kern\lineskip\fi
    \ifanswerkey %\turnfl@nskeyOnOff % dps16
      \ifeqe@flextended
        \ifcont@nnot
          \ifx\makeVgrid\relax
            \gdef\p@ssToFLs{\turnfl@nskeyOnOff
              \let\eqe@insert@more@content\eqe@insertSolns
              \writeVertic@lFLinesfalse
              \KV@eqefillLines@toplinefalse
              \let\makeVgrid\eqe@makeVgrid
            }%
          \else
            \gdef\p@ssToFLs{\turnfl@nskeyOnOff
              \let\eqe@insert@more@content\eqe@insertSolns
              \writeVertic@lFLinestrue}%
          \fi
          \gdef\eqe@flnexti{\turnfl@nskeyOnOff
            \global\fl@firstpasstrue % dpsj5
            \p@ssToSolns
            \global\gridpgbrkfalse
            \eqe@setStartSolns
```

We are with a good two deep, but we need to pass \vspacewithkeyOff, so we save its current state and declare a global call to \vspacewithkeyOff. The state is restored by \eq@e@ddCodeSpecial.

```
            \xdef\kdvsp@SAVE{\ifkeepdeclaredvspacing
                true\else false\fi}%
            \gdef\kdvsp@Restore{\@nameuse
              {keepdeclaredvspacing\kdvsp@SAVE}%
              \gdef\twe@kBre@kPoint{0pt}\global
              \let\kdvsp@Restore\relax}% dpsj21
          \expandafter\global\vspacewithkeyOff
          \vbadness=10000
          \fl@getcontent{s\fl@CutName}}%
          \gdef\eq@e@ddCodeSpecial{\endverbatimwrite\endgroup
%            \kdvsp@Restore % dpsj20
            \ifwithinparts\ifx\istabularexer\eq@YES
            \immediate\write\verbatim@out{\string\endminipage}\fi\fi
            \immediate\closeout\verbatim@out
            \global\let\eq@e@ddCodeSpecial\relax
          }%
        \else
          \PackageWarningNoLine{eqexam}{For solutions to appear
            using filler lines\MessageBreak
            (with flextexded and answerkey options),\MessageBreak
            you must first expand \string\turnContAnnotOn\MessageBreak
            prior to line \the\inputlineno.\MessageBreak
            Switching to \string\useFillerDefault}%
          \useFillerDefault
          \let\eq@insertverticalspace\eq@NO
        \fi
      \fi
    \fi
  \fi
  \eqe@flnexti
}% dpsj4
```

14.4 Filler lines with nosolutions option

Just as we can supply filler lines for the answerkey option, we can do the same for the nosolutions option and superimpose content on the filler lines.

priorworkarea Place the priorworkarea environment *prior to* the solution environment. It has not arguments, but gets dimensions from the solution environment that follows.

```
\def\eqe@priorw@content{%
      \clubpenalty=50 % dps15
      \widowpenalty=50
      \vbadness=10000
      \fl@vsplitandplace{p\fl@CutName}%
}%
\renewenvironment{priorworkarea}{\par
  \ifx\solutionparshape\@empty\else % dps28
    \pushEnvir
        \everypar{}\if@eqalignfilllinestoleft\else
```

```
       \parshape=1 \leadinIndent \linewidth\fi
    \popEnvir
  \fi
  \global\let\p@ssToFLs\relax
  \global\let\p@ssToSolns\relax
  \vspacewithkeyOff
  \ifeq@nosolutions\else\sloppy\fi
  \def\eqe@flnexti{\eqSavedComment}%
  \ifeqe@flnosolns\ifdisplayworkarea
    \def\eqe@flnexti{\fl@getcontent{p\fl@CutName}}\fi\fi % dpsj21
  \gdef\eqe@flnextii{\endeqSavedComment}%dps11
  \ifeqe@flnosolns\ifdisplayworkarea
    \gdef\eqe@flnextii{\endverbatimwrite\endgroup
    \ifwithinparts\ifx\istabularexer\eq@YES
      \immediate\write\verbatim@out{\string\endminipage}\fi\fi
    \immediate\closeout\verbatim@out}\fi\fi
  \eqe@flnexti
}{%
  \eqe@flnextii
  \ifcont@nnot
    \ifeqe@flnosolns
      \ifdisplayworkarea
        \ifx\eq@insertverticalspace\eqe@YES
          \ifx\makeVgrid\relax
            \gdef\p@ssToFLs{%
              \let\eqe@insert@more@content\eqe@priorw@content
              \writeVertic@lFLinesfalse
              \KV@eqefillLines@toplinefalse
              \let\makeVgrid\eqe@makeVgrid}%
          \else
            \gdef\p@ssToFLs{%
              \let\eqe@insert@more@content\eqe@priorw@content
              \writeVertic@lFLinestrue}%
          \fi
          \gdef\kdvsp@Restore{\gdef\twe@kBre@kPoint{0pt}\global
            \let\kdvsp@Restore\relax}% dpsj21
        \fi
      \fi
    \else
      \global\setbox\eqe@nskeyflsplit\box\voidb@x
      \global\setbox\eqe@nskeyfltop\box\voidb@x
    \fi
  \else
    \PackageWarningNoLine{eqexam}{For priorworkarea to appear
      using filler lines\MessageBreak
      (with flextexded and nosolutions options),\MessageBreak
      you must first expand \string\turnContAnnotOn\MessageBreak
      prior to line \the\inputlineno.\MessageBreak
      Switching to \string\useFillerDefault}%
    \gdef\p@ssToSolns{\useFillerDefault}%
  \fi
}
```

```
% end of segment
⟨/flextended⟩
⟨*package⟩
```

Set the eqexam page style, if not otherwise indicated.

```
\ifeqfortextbook\else
    \if@bypasseqexamheading\else
\pagestyle{eqExamheadings}\fi\fi
```

```
⟨/package⟩
⟨*textbook⟩
\ProvidesFile{eqtextb.def}
 [2016/01/18 v4.6 Cmds used by the fortextbook option (dps)]
```

15 Concerning the `fortextbook` option

What are my goals/desired features? Modern (U.S.) textbooks—at least the ones I'm familiar with—consist of some or all of the following resources:

- **Student Edition**: Answers to odd-numbered problems appear in the back of the text.

  We need to have a scheme where odd-numbered problems, under suitable options, are compiled. **Goal:** It does not need to be restricted to odd-numbered, however, need to latex only those problems that meet the "include" criteria.

  For chapter review problem sets, odd-number problems are have solutions in the back of the book.

  For chapter quizzes, odd-numbered (optionally all) solutions are in the back of the book.

- **Instructor Edition**: Answers to all problems appear in the back of the book. Answers may also appear in the body of the text, in the margins of the text, or immediately after the statement of the problem. If the answer is too long, there is a cross-reference to the solution in the appendix.

  Some publishers I've seen have wide margins where additional material can be inserted (historical sketches, instructor notes, pictures, etc.). In these margins, the answers to the problems can appear. Other publishers put answer immediately following the questions. The latter is easy to do; just have a macro, say `\ANS{$12.5$}`, which only expands when the "instructor" option is used.

- **Student Solution Manual**: Contains solutions to all odd-problems, as well as any review problems and chapter quizzes. Some publishers include all solutions to chapter quizzes.

  A solution manual is a separate publication. This document would be created by latexing one or more of the auxiliary file (`.sol`) These files might have to be edited before the final compile. We include only the solutions that meet the include criteria (i.e., odd-numbered ones).

  The current features of `eqexam` is what is needed here. The authors need only include solutions to each problem in a `solution` environment. Now, I realize that often times the authors create the solutions, but someone else, possibly a grade student or contractor, solves the problems. In the latter case, the authors would probably not like to turn over the source files to the one solving and typesetting the problems.

- **Instructor Solution Manual**: Contains solutions to all problems, review problems, chapter quizzes.

  Similar comments for the instructor solution manual.

Some other thoughts by a contributor:

- Often there is a diagram or graphic within the problem – this has some figure caption and after the running counter of the figure it is named the NUMBER of the problem (cross-reference to the problem number)
- Often the probs with soln are setup in two-column style.
- Often the PROBLEM NUMBER has a special formatting (not only bold and black), maybe with a colorframebox around or some special formatting from the author
- Of course I have seen in some EXAMPLES that there is a wide margin to put in additional graphics etc. setup in two-side style – wide left margin on even pages, wide right margins on odd pages. Here as well are captions setup and cross-references.

15.1 Setting options with \textbookOpts

`\textbookOpts` We set up a command for setting the options for the `fortextbook` option.

`marginans` `inlineans` `marginsonleft`

```
\define@boolkey{eqe@tbopts}[is]{instred}[true]{}
\define@boolkey{eqe@tbopts}[is]{studented}[true]{%
    \ifisstudented
```

`ssols` `lsols` Now let’s try to filter out the even-numbered problems for the student edition.

```
    \tbfilterOutEvenNums
```

The above command is normally `\let` to `\@gobble`.

```
    \fi
}
```

This code is executed in \exambegdef, the start up code of the exam environment. This enables problems with fill-ins, true/false, or multiple choice, to have the answer appear in the space provided.

```
\def\tb@beginexam@code{%
    \ifisinstred\answerkeytrue\eq@proofingtrue\fi}
```

\eqEXt{\theeqquestionnoi} and \endeqEXt\tok1\tok2 enclose each solution, \tbfilterOutEvenNums redefines \eqEXt to gobble everything, when the page number is even, through \endeqEXt and the two tokens it follows. This leaves only the odd-numbered problems.

```
\newcommand{\tbfilterOutEvenNums}{%
```

exerquiz changed \eqEXt to two variables, so we make the same change here

```
    \def\eqEXt##1##2{\ifodd##1\let\eqe@next\relax\else
    \def\eqe@next{\gobbletoEndEXt}\fi\eqe@next}%
}
\newcommand{\tballowAllNums}{%
```

exerquiz changed \eqEXt to two variables, so we make the same change here

```
%     \let\eqEXt\@gobble
    \let\eqEXt\@gobbletwo
    \let\endeqEXt\relax
}
\define@boolkey{eqe@tbopts}[is]{marginans}[true]{}
\define@boolkey{eqe@tbopts}[is]{inlineans}[true]{}
\define@boolkey{eqe@tbopts}[]{marginsonleft}[true]
```

If margins are always on left, we turn off switching of margin notes as placed by \marginpar, and use \reversemarginpar to get them on the left.

```
    {\@mparswitchfalse\reversemarginpar}
\define@boolkey{eqe@tbopts}[show]{ssols}[true]{}
\define@boolkey{eqe@tbopts}[show]{lsols}[true]{%
    \ifshowlsols\let\tb@soln@choice\tb@showlsols\fi}
```

The default settings are true for studented and false for instred.

```
\newcommand{\textbookOpts}[1]{\setkeys{eqe@tbopts}{#1}%
```

We do not allow both instred and studented to be true.

```
    \ifisinstred\global\isstudentedfalse\else
        \ifisstudented\global\isinstredfalse
    \fi\fi
```

Added this part in in case \textbookOpts comes after \marparboxwidth.

```
    \ifdim\tbmarparboxwidth=1sp\else
    \expandafter\tbMakeFinalCalcs\fi
}
```

As mentioned above, the default settings are true for studented and false for instred.

```
\isstudentedtrue
\isinstredfalse
\ismarginansfalse
\isinlineansfalse
```

The command is available only in the preamble.

```
\@onlypreamble{\textbookOpts}
```

\turnOffMarAnsOnAnsInline
\turnOnMarAnsOffAnsInline
\toggleInstrAns

These three command may not be useful in the creation of a textbook, but you never know, I used them in my demo doc `fortextbook.tex` to turn off and on the display of the answers (change margin to inline, change inline to margin, and toggle margin and inline).

```
\newcommand{\turnOffMarAnsOnAnsInline}{%
    \global\ismarginansfalse\global\isinlineanstrue
    \insMidMarg{\global\ismarginansfalse
        \global\isinlineanstrue}%
}
\newcommand{\turnOnMarAnsOffAnsInline}{%
    \global\ismarginanstrue\global\isinlineansfalse
    \insMidMarg{\global\ismarginanstrue
        \global\isinlineansfalse}%
}
\newcommand{\toggleInstrAns}{%
    \ifisinstred\ifismarginans
        \global\ismarginansfalse\global\isinlineanstrue
        \insMidMarg{\global\ismarginansfalse
            \global\isinlineanstrue}%
    \else
        \global\ismarginanstrue\global\isinlineansfalse
        \insMidMarg{\global\ismarginanstrue
            \global\isinlineansfalse}%
    \fi\fi
}
```

15.2 Macros to display answers/shortsolns

In this section, we develop some commands to display answers or short solutions. These would appear if `instred=true`, in-line, or in the margins.

\ANS

Let us begin by creating a simple macro for saving an answer. The answer is displayed "in-line." No verbatim-type text allowed, no unbalanced braces unless escaped. `\ANS` displays the answer if the `instred` option of the `eqe@tbopts` family, i.e., by executing

\bGrpANS
\eGrpANS

Two macros used to group answers in the margins.

```
\newif\ifWithinANSGrp\WithinANSGrpfalse
\newif\ifftb@isANSListOpen\ftb@isANSListOpenfalse
\newcommand{\bGrpANS}{%
    \if\probstar*\else
        \PackageError{eqexam}{Use of \string\bGrpANS\space
        only applies\MessageBreak to the problem* environment}{Please
        remove this \string\bGrpANS.}%
    \fi
    \ifWithinANSGrp
```

```
        \global\WithinANSGrpfalse
        \let\tb@next\relax
        \PackageError{eqexam}{\string\bGrpANS\space already open}
        {You issued an earlier \string\bGrpANS,
         but did not close it.}%
    \else
        \global\WithinANSGrptrue
        \global\ftb@isANSListOpenfalse
        \def\tb@next{\ANS}%
    \fi
    \tb@next
}
\newcommand{\eGrpANS}{%
    \if\probstar*\else
        \PackageError{eqexam}{Use of \string\eGrpANS\space
        only applies\MessageBreak to the problem* environment}{Please
        remove this \string\eGrpANS.}%
    \fi
    \ifWithinANSGrp
        \global\WithinANSGrpfalse
        \def\tb@next{\ANS}%
     \else
        \let\tb@next\relax
        \PackageError{eqexam}{\string\eGrpANS\space already closed}
        {You've issued two consecutive \string\eGrpANS\space
         commands,\MessageBreak either remove this one
        or the previous one.}%
     \fi
    \tb@next
}
```

`\ANS` begin by checking to see if there is a star that follows the command, this is used for inline answers. If `*` is present, we do not put the answer inline, but will put it in the margins if the option call for it.

```
\newcommand{\ANS}{\@ifstar{\let\tb@istart=1\tb@ANS}
    {\let\tb@istart=0\tb@ANS}}
```

(10/13/2011) The following is the original definition of `\tb@ANS` before the creation of the commands `\bGrpANS` and `\bGrpANS`. We keep this to revert to this definition if this new feature causes problems.

```
\newcommand{\tb@ANS}[1]{%
    \ifisinstred
        \ifisinlineans\if\tb@istart0\ANSFmt{\theeqquestionnoi}{#1}\fi\fi
        \ifismarginans
            \edef\eqe@prehold{\noexpand\par\kern0pt\noindent
                \if\probstar*%
                   \noexpand\begin{eqeList}[\tb@wparts@len]{%
                   \noexpand\eqedsplyOnlyFrst{\theeqquestionnoi}%
                   {\thepartno}\noexpand\eqe@hspannerMrg
```

```
                    \noexpand\makebox[\noexpand\tbmrgpartwdth]%
                         {\noexpand\tb@mrgPartFmt{\thepartno}}}%
                 \else
                    \noexpand\begin{eqeList}%
                     {\noexpand\tb@mrgDigitFmt{%
                         \theeqquestionnoi\eqe@decPointMrg}}%
                 \fi
             }\expandafter\insMidMarg%
                 \expandafter{\eqe@prehold#1\end{eqeList}}%
         \fi
     \fi
 }
```

`\ftb@defineInsSpan` is used when there is an optional argument for `\ANS`. It formats the range of parts, for example, (a)–(c). This macro can be redefined, I suppose, to meet the needs of the author.

```
\def\ftb@defineInsSpan#1{\def\ftb@argi{#1}\ifx\ftb@argi\@empty
    \def\ftb@InsSpan{}\else\ftb@spanPrts{#1}%
    \def\ftb@InsSpan{\noexpand\hspace{-\labelsep}%
    \noexpand\textcolor{MRGPARTcolor}{--}\noexpand
    \makebox[\noexpand\tbmrgpartwdth]{\noexpand
    \tb@mrgPartFmt{\ftb@EndSpanPrts}}\eqe@hspannerMrg}\fi
}
```

`\ftb@spanPrts` calculates the letter of the end of the range. `#1` is passed by `\ANS` (`\tb@ANS`, actually). For example if we have `\ANS[2]{...}`, `#1=2`.

```
\def\ftb@spanPrts#1{{%
    \advance\value{partno}by#1\relax
    \xdef\ftb@EndSpanPrts{\thepartno}}%
}
```

`\ftb@EqeListPrtsFmt` is the internal formatting used within the `eqeList` for the part letter.

```
\def\ftb@EqeListPrtsFmt{\noexpand
    \makebox[\noexpand\tbmrgpartwdth]{\noexpand
    \tb@mrgPartFmt{\thepartno}}\nobreak
}
```

`\ftb@OpenEqeListPrts` opens an `eqeList` environment, and displays the question number (optionally) and the part number.

```
\def\ftb@OpenEqeListPrts{\noexpand
   \begin{eqeList}[\tb@wparts@len]{\noexpand
   \eqedsplyOnlyFrst{\theeqquestionnoi}%
   {\thepartno}\noexpand\eqe@hspannerMrg\ftb@EqeListPrtsFmt}%
}
```

`\ftb@CloseEqeList` closes the `eqeList` after inserting `\qe@prehold` and the content, `#1`.

```
\def\ftb@CloseEqeList#1{\expandafter\insMidMarg%
    \expandafter{\eqe@prehold#1\end{eqeList}}%
}
```

`\grpANSDelimiter` delimits the parts when `\bGrpANS`/`\eGrpANS` is used. May be redefined.

```
\newcommand{\grpANSDelimiter}{\textcolor{MRGPARTcolor}{,}\space}
```

\tb@ANS does the main work of \ANS.

```
\newcommand{\tb@ANS}[2][]{%
    \ifisinstred
        \ifisinlineans
            \if\tb@istart0\ANSFmt{\theeqquestionnoi}{#2}\fi
        \fi
        \ifismarginans
            \ftb@defineInsSpan{#1}%
```

We create the code that we will introduce into `\insMidMarg`, this will be introduced prior to `#2`.

```
            \edef\eqe@prehold{%
                \if\probstar*%
```

If this question is one with parts...

```
                    \ifftb@isANSListOpen
```

If the list is already open (`\ifftb@isANSListOpen`), we just add content to the `eqeList` environment.

```
                       \ftb@EqeListPrtsFmt\noexpand\eqe@hspannerMrg
                    \else
```

If the list is not open, we start the `eqeList` environment in the usual way, this also includes the case where `\bGrpAns` is not uses, which is normally the case.

```
                        \noexpand\par\kern0pt\noindent
                        \ftb@OpenEqeListPrts\ftb@InsSpan
                    \fi
                \else
```

This is a question without parts.

```
                   \noexpand\begin{eqeList}%
                    {\noexpand\tb@mrgDigitFmt{%
                    \theeqquestionnoi\eqe@decPointMrg}}%
                \fi
            }%
```

We have finished constructing `\eqe@prehold`. We next set `\ftb@isANSListOpen` to `true`, if `\WithinANSGrp` is `true`.

```
            \ifWithinANSGrp\global\ftb@isANSListOpentrue\fi
```

If we are within an open group, we emit `\insMidMarg` with the `\eqe@prehold`, followed by `#2`, and a comma-space combo, but *we do not close* the `eqeList` environment.

```
            \ifWithinANSGrp
                \expandafter\insMidMarg\expandafter
                    {\eqe@prehold#2\grpANSDelimiter}%
            \else
```

This is the normal case, we insert `\eqe@prehold`, `#2`, and close the `eqeList` environment.

```
                \expandafter\insMidMarg%
```

```
            \expandafter{\eqe@prehold#2\end{eqeList}}%
            \global\ftb@isANSListOpenfalse
         \fi
      \fi % \ifismarginans
   \fi %\ifisinstred
}
```

End (10/13)

eqeList An environment used to format the answers in the margins, when marginans is in effect.

```
\newenvironment{eqeList}[2][\tb@woparts@len]{\begin{list}{#2}{%
   \def\argi{#1}\setlength{\labelwidth}{#1}%
   \ifx\argi\tb@wparts@len
   \settowidth{\labelsep}{\eqe@prtsepMrg}\else
   \settowidth{\labelsep}{\eqe@hspannerMrg}\fi
   \setlength{\leftmargin}{\labelwidth+\labelsep}%
   \setlength{\parskip}{0pt}\setlength{\partopsep}{0pt}%
   \setlength{\topsep}{1pt}\setlength{\parsep}{0pt}%
   \setlength{\itemindent}{0pt}\setlength{\itemsep}{3pt}%
}\item\relax}{\end{list}}
```

Formatting Answers and Solutions

\mrgDigitFmt Format of the digit (and the decimal point) for the answers in the margins.

```
\mrgDigitFmt{\textbf{#1}}
\mrgPartFmt{\textbf{(\hfil#1\hfil)}}
\setMarIndents[\bfseries\normalsize\normalfont]{00}{(d)}
```

```
\newcommand{\mrgDigitFmt}[1]{\def\tb@mrgDigitFmt##1{#1}}
\mrgDigitFmt{#1}
```

\mrgPartFmt Format of the part (including possibly the parentheses), example give above.

```
\definecolor{MRGPARTcolor}{named}{black}
\newcommand{\mrgPartFmt}[1]{\def\tb@mrgPartFmt##1{#1}}
\mrgPartFmt{\textcolor{MRGPARTcolor}{(\hfil#1\hfil)}}
```

ANScolor The default color of the answers that appear in the margins or inline.

```
\definecolor{ANScolor}{rgb}{0,0,.8}
```

\ANSFmt The command that sets the format, may be redefined as needed. Used in the \ANS command above.

```
\newcommand{\ANSFmt}[2]{\textcolor{ANScolor}{#2}}
```

We have two environments that we use in three different situations:

- eqequestions environment: (1) Used to control the display of the probset environment within the body of the textbook; (2) used to control the display of the solutions “in the back of the book.”

- eqeList environment: Used for displaying answers in the margin of the book, when the appropriate options allow it.

We want to be able to manipulate some of the parameters of these three situation, independently of each other. There are several issues, setting what I have been calling the gutter width, and the display of the problem numbers.

We define four commands for each of the three situations described above. The names have a pattern to them, and similarly named commands have the same use.
The numbering of the problems has the pattern: dd.␣(a)␣ We provide convenience commands to give these internal macros values

\prbDecPt \prbPrtsep \prbNumPrtsep

Basic parameters for the problems in the body of the text.

```
\def\eqe@decPointPrb{.}     % decimal point of prob number
\def\eqe@prtsepPrb{\ }      % prob with parts, space after part
\def\eqe@hspannerPrb{\ }    % space after prob number
\providecommand{\prbDecPt}[1]{\def\eqe@decPointPrb{#1}}
\providecommand{\prbPrtsep}[1]{\def\eqe@prtsepPrb{#1}}
\providecommand{\exPrtsep}[1]{\def\eqe@prtsepPrb{#1}}
\providecommand{\prbNumPrtsep}[1]{\def\eqe@hspannerPrb{#1}}
```

\solDecPt \solPrtsep \solNumPrtsep

Basic parameters for the problems in the solution sets.

```
\def\eqedecPointSoln{.}    % decimal point of prob number
\def\eqe@prtsepSoln{\ }     % prob with parts, space after part
\def\eqe@hspannerSoln{\ }   % space after prob number
\newcommand{\solDecPt}[1]{\def\eqedecPointSoln{#1}}
\newcommand{\solPrtsep}[1]{\def\eqe@prtsepSoln{#1}}
\providecommand{\solNumPrtsep}[1]{\def\eqe@hspannerSoln{#1}}
```

\mrgDecPt \mrgPrtsep \mrgNumPrtsep

Basic parameters for the problems in the margins.

```
\def\eqe@decPointMrg{.}     % decimal point of prob number
\def\eqe@prtsepMrg{\ }      % prob with parts, space after part
\def\eqe@hspannerMrg{\ }    % space after prob number
\newcommand{\mrgDecPt}[1]{\def\eqe@decPointMrg{#1}}
\newcommand{\mrgPrtsep}[1]{\def\eqe@prtsepMrg{#1}}
\newcommand{\mrgNumPrtsep}[1]{\def\eqe@hspannerMrg{#1}}
```

\setMarIndents

Sets some dimensions used by the eqeList environment. \tb@woparts@len is calculated and is used as the default gutter width in eqeList. \tb@wparts@len is used for the gutter width for the gutter width when there is a problem with parts. Finally, \tbmrgpartwdth in \ANS and is used for the width of a \makebox that enclosed the part letter.

```
\newcommand{\setMarIndents}[3][\normalsize\normalfont]{{%
    \settowidth{\@tempdima}{#1#2\eqe@decPointMrg}%
    \xdef\tb@woparts@len{\the\@tempdima}%
    \settowidth{\@tempdima}%
        {#1#2\eqe@decPointMrg\eqe@hspannerMrg#3}%
    \xdef\tb@wparts@len{\the\@tempdima}%
    \settowidth{\@tempdima}{#1#3}%
    \xdef\tbmrgpartwdth{\the\@tempdima}%
}}
\setMarIndents{00}{(d)}
```

\setSolnIndent

Used to set the some parameters used by eqequestions, in the solutions file.

```
\newdimen\solnGutter
\newcommand{\setSolnIndent}[3][\normalsize\normalfont\bfseries]{%
    {\settowidth{\@tempdima}{#1#2\eqedecPointSoln\eqe@hspannerSoln}%
    \global\solnGutter\@tempdima
    \settowidth{\@tempdima}{#1#3}%
    \xdef\tbsolnpartwdth{\the\@tempdima}%
}}
\setSolnIndent{00}{(d)}
```

\setSolnMargins

This command is written to the solution file, and expanded when that file is input back in. If \solnGutter is not 0pt, we set the length of \eqemargin using the current value; otherwise, we use the value determined by \setSolnIndent, above.

```
\renewcommand{\setSolnMargins}[1]{%
    \ifdim\solnGutter=0pt \setlength\eqemargin{#1}\else
    \setlength\eqemargin{\solnGutter}\fi\ignorespaces}
%\newcommand{\defaultSolnIndent}{\gdef\solnGutter{0pt}}
\newcommand{\defaultSolnIndent}{\global\solnGutter=0pt}
\defaultSolnIndent
```

\prbNumFmt
\solWoPrtsFmt
\solWPrtsFmt

We redefine \exlabelformat, \exsllabelformat, and \exsllabelformatwp. They are defined in such a way as to simply their modification through a series of simple formatting commands. The defaults are

```
\prbNumFmt{\textbf{#1}}
\solWoPrtsFmt{\textbf{#1}}
\solWPrtsFmt{\textbf{#1}}{(\hfil#2\hfil)}
```

```
\renewcommand{\exlabelformat}{%
    \tbprbNumFmt{\theeqquestionnoi\eqe@decPointPrb}}
```

\prbNumFmt is the format for the number of the problems in the body of the text. The argument #1 is a symbolic argument for the question number.

```
\newcommand{\prbNumFmt}[1]{\def\tbprbNumFmt##1{#1\eqe@hspannerPrb}}
\prbNumFmt{\textbf{#1}}
```

Redefine \exsllabelformat, and \exsllabelformatwp

```
\renewcommand{\exsllabelformat}{\string\tbsolWoPrtsFmt{%
    \theeqquestionnoi\string\eqedecPointSoln}}
\renewcommand{\exsllabelformatwp}{\string\tbsolWPrtsFmt%
    {\string\eqedsplyOnlyFrst{\theeqquestionnoi}{\thepartno}}%
    {\thepartno}%
}
```

\solWoPrtsFmt is the format for the number of the problems in the solution set. The argument #1 is a symbolic argument for the question number. \solWPrtsFmt is the format for a problem with parts in the solution file. #1 is symbolically the question number, and #2 is a symbolic for the part letter.

```
\newcommand{\solWoPrtsFmt}[1]{\def\tbsolWoPrtsFmt##1{%
    \makebox[0pt][r]{#1\eqe@hspannerSoln}}}
\solWoPrtsFmt{\textbf{#1}}
\newcommand{\solWPrtsFmt}[2]{\def\tbsolWPrtsFmt##1##2{%
    \makebox[0pt][r]{#1\eqe@hspannerSoln}%
    \makebox[\tbsolnpartwdth][l]{#2}\eqe@prtsepSoln%
}}
\solWPrtsFmt{\textbf{#1}}{(\hfil#2\hfil)}
```

An alternate definition for \solWPrtsFmt, used by \hangSolWPrtsFmt.

```
\newcommand{\solWPrtsFmt@hang}[2]{%
    \def\tbsolWPrtsFmt##1##2{%
    \makebox[0pt][r]{#1\eqe@prtsepSoln%
    \makebox[\tbsolnpartwdth][l]{#2}\eqe@hspannerSoln}%
}}
```

\hangSolWPrtsFmt The command takes two arguments, the same as \solWPrtsFmt. When this command is executed in the preamble, we get hanging indentation for problems with parts.

```
\let\bpartsmrk\relax
\let\epartsmrk\relax
\newcommand{\hangSolWPrtsFmt}[2]{%
```

At the beginning and ending of a parts environment, we begin and end a special eqepartsquestions environment, designed to give the desired indentation.

```
    \def\prior@parts@hook{%
        \ifisleadin\else
        \writeT@SolnFile{^^J\protect\bpartsmrk}\fi}%
    \def\post@parts@hook{%
        \writeT@SolnFile{\protect\epartsmrk^^J}}%
    \def\bpartsmrk{\global\firstitemtrue\begin{eqepartsquestions}}%
    \def\epartsmrk{\end{eqepartsquestions}\global\firstitemfalse}%
```

We must also redefine \solWPrtsFmt by letting it to \solWPrtsFmt@hang, then executing it using the parameters passed.

```
    \let\solWPrtsFmt\solWPrtsFmt@hang
     \solWPrtsFmt{#1}{#2}%
}
%\@onlypreamble\hangSolWPrtsFmt
```

Define some switches, token registers, and boxes for managing the answers and marginal notes.

```
\newif\ifexamenv \examenvfalse
\newif\iffirstemit \firstemittrue
\newtoks\txtbkt@ks \txtbkt@ks={}
\newtoks\txtbkt@ksi \txtbkt@ksi={}
\newbox\txtbkb@xb@t
\newbox\txtbkb@xt@p
```

```
\newbox\txtbkb@xh@ld
\let\tbTopMargin\relax
\let\tbBotMargin\relax
\long\def\tb@addtoTopMargin#1{\txtbkt@ksi={#1}%
    \edef\eqe@tmphold{\the\txtbkt@ksi\the\txtbkt@ks}%
    \global\txtbkt@ks=\expandafter{\eqe@tmphold}%
}
\newcommand{\tb@addtoMargin}[1]{%
    \edef\eqe@tmphold{\the\txtbkt@ks}%
    \global\txtbkt@ks=\expandafter{\eqe@tmphold#1}%
}
```

As my first attempt, let's create two comment environmets to be used within the `solution` environment.

```
\begin{solution}
\begin{ssol}
   <short solution/answer>
\end{ssol}
\begin{lsol}
   <long solution>
\end{lsol}
\end{solution}
```

The control of these environments are made through

`lsol` Place full (or long) solutions in this environment.

`\tb@showlsols`

```
\newcommand{\tb@showlsols}{\includecomment{lsol}\excludecomment{ssol}}
```

`ssol` Place short solutions in this environment.

`\tb@showssols`

```
\newcommand{\tb@showssols}{\includecomment{ssol}\excludecomment{lsol}}
```

The default is to show the short solutions.

```
\let\tb@soln@choice\tb@showssols
%\let\tb@sols@choice\tb@showssols
```

15.3 Marginal Matter.

There are three levels in the margins:

1. **Top level**: This is a command `\tbTopMargin` with may be redefined between pages. It should have the width of the `\parbox` that contains all the content of the margin, this width is `\tb@marparboxwidth` (`\oddsidemargin-\marginparsep`). The content *must* be unbreakable across pages. The content of `\tbTopMargin` will appear *on every page subsequent to its definition.*

`\clearTopMargin` We can clear the top level using the following command

```
  \newcommand{\clearTopMargin}{\global\let\tbTopMargin\relax}
```

Clearing will take effect on the following page.

`\setTopMargin` As a convenience macro, we can create top margin content. Redefinitions will appear on the next page from where the definition was made.

```
  \newcommand{\setTopMargin}[1]{%
      \gdef\tbTopMargin{\color@begingroup\normalcolor
          #1\color@endgroup}%
      \gdef\tbSaveTopMargin{\color@begingroup\normalcolor
          #1\color@endgroup}}
  \let\tbTopMargin\@empty
  \let\tbSaveTopMargin\@empty
```

2. **Middle level**: The middle level is the most interesting. You write to it using `\insMidMarg`. Normally, this is text. If there is too much text, it will be split off and placed in the middle level of the next page. The command `\ANS` also writes to the middle level when the `instred` and `marginans` options are taken.

3. **Bottom level**: This is similar to the top level, but on the bottom. The command is named `\tbBotMargin` and follows the same rules as `\tbTopMargin`. Again, the content of `\tbBotMargin` will appear *on every page subsequent to its definition.*

`\clearBotMargin` We can clear the bottom level using the following command

```
  \newcommand{\clearBotMargin}{\global\let\tbBotMargin\relax}
```

Clearing will take effect on the following page.

`\setBotMargin` As a convenience macro, we can create bottom margin content. Redefinitions will appear on the next page from where the definition was made.

```
  \newcommand{\setBotMargin}[1]{\gdef\tbBotMargin{#1}%
      \gdef\tbSaveBotMargin{#1}}
  \let\tbBotMargin\@empty
  \let\tbSaveBotMargin\@empty
```

```
  \newcommand{\restoreLastTopMargin}{\expandafter\setTopMargin
      \expandafter{\tbSaveTopMargin}}
  \newcommand{\restoreLastBotMargin}{\expandafter\setBotMargin
      \expandafter{\tbSaveBotMargin}}
```

`\insMidMarg` `\insMidMarg` is a `\parbox` that will hold the material in the margin.

`\MarParBoxFmt` The formatting for the marginal `\parbox`

```
\newcommand{\MarParBoxFmt}{\normalsfcodes
    \normalfont\normalsize\normalbaselines\parindent0pt
    \vbadness\@Mi \hbadness5000 \tolerance9999
    \parskip0pt\raggedright %\spaceskip=0pt\xspaceskip=0pt
    \setlength{\linewidth}{\tbmarparboxwidth}%
}
```

`\tbmarparboxwidth` The width of the margin box. Initial value of `1sp`, if the user does not reset the value, it is a package error.

```
\newlength\tbmarparboxwidth
\setlength\tbmarparboxwidth{1sp}
```

`MidMargcolor` The default color of text of the middle level

```
\definecolor{MidMargcolor}{rgb}{0,0,.8}
\newcommand{\midMargFmt}[1]{%
    \def\tb@midMargFmt{\normalfont\normalsize\normalcolor#1}}
\midMargFmt{\color{MidMargcolor}}
```

`\eqe@MarParBox` This is the actual `\parbox` that holds the marginal material. I have two versions of this box, the first one has height `\textheight`, the second one has height `\textheight+\footskip`.

```
%\def\eqe@MarParBox#1{\parbox[b][\textheight][t]%
%    {\tbmarparboxwidth}{\color{MidMargcolor}#1}}
\def\eqe@MarParBox#1{\lower\footskip\hbox{%
    \leavevmode\parbox[b][\textheight+\footskip][t]%
    {\tbmarparboxwidth}{\tb@midMargFmt#1}}} %
```

Finally, we get to the `\insMidMarg`, this is used to write to the middle level.

```
\newcommand{\insMidMarg}[1]{%
    \let\eqe@margininsert\@empty
    \expandafter\tb@addtoMargin\expandafter{\eqe@margininsert#1}%
}
```

`\tbPreMarginHeader` Executed prior to the marginal heading

`\tbPostMarginHeader` Executed after the marginal heading

`HEADERcolor` Default color of a marginal header

`\cngMargHeadColorTo` Changes the marginal header to a named color

`\resetMargHeadColor` Reset the marginal header color to the default, `HEADERcolor`

`\tbMarginHeaderFmt` Formatting for a marginal header. Format the marginal header, the default is `HEADERcolor` in bold

```
\newcommand{\tbPreMarginHeader}{\par\penalty0 \kern3pt}
\newcommand{\tbPostMarginHeader}{\par\nobreak}
\definecolor{HEADERcolor}{named}{black}
\newcommand{\cngMargHeadColorTo}[1]{\insMidMarg{\gdef\tb@MHC{#1}}}
\newcommand{\resetMargHeadColor}{\insMidMarg{\gdef\tb@MHC{HEADERcolor}}}
\resetMargHeadColor
\newcommand{\tbMarginHeaderFmt}[1]{\textcolor{\tb@MHC}{\textbf{#1}}}
```

`\insMargHead` Used to insert a general marginal heading into the middle level. The optional parameter allows you to set a mark.

\insProbHead Used to insert a marginal heading for a problem set into the middle level. The optional parameter allows you to insert a mark, the default mark is #1 \tbcontinued.

```
\newcommand{\insMargHead}[2][]{% dps
    \insMidMarg{\tb@marginHeader{#1}{#2}}}
\newcommand{\insProbHead}[2][]{%
    \def\tb@argi{#1}\ifx\tb@argi\@empty
        \protected@xdef\currProbHead{#2 \tbcontinued}\else
        \protected@xdef\currProbHead{#1}\fi
    \ifisinstred\ifismarginans
        \insMidMarg{\tb@marginProbHeader{#1}{#2}}\fi\fi
}
\newcommand{\tb@marginProbHeader}[2]{%
    \def\tb@argi{#1}\ifx\tb@argi\@empty
    \tb@marginHeader{#2 \tbcontinued}{#2}\else
    \tb@marginHeader{#1}{#2}\fi
}
\newcommand{\tb@marginHeader}[2]{\tbPreMarginHeader
    \tbMarginHeaderFmt{#2}\def\tb@argi{#1}\ifx\tb@argi\@empty
    \mark{#2}\else\mark{#1}\fi\tbPostMarginHeader
}
```

\tbcontinued The continue annot that appear when a problem set flows over to the next page.

```
\newcommand{\tbcontinued}{(cont.)}
```

\tbplaceMargins Redefine this macro to set the locations of the margins we are writing to.

```
\newcommand{\marparboxwidth}[1]{%
    \setlength\tbmarparboxwidth{#1}%
    \setlength{\marginparwidth}{\tbmarparboxwidth}%
    \tbMakeFinalCalcs
}
\@onlypreamble\marparboxwidth
\newcommand{\chkmarginboxwidth}{%
    \ifdim\tbmarparboxwidth=1sp \PackageError{eqexam}%
    {You have not set the value of\MessageBreak
        \string\marparboxwidth}%
    {Define the \string\marparboxwidth\space command}\fi
}
```

\ifmarginsonleft A Boolean switch, if true, all margins are on the left; otherwise, they alternate

```
\newif\ifmarginsonleft \marginsonleftfalse
```

\tbSetupForMargins We compute \oddsidemargin, \evensidemargin, and \textwidth

```
\newcommand{\tbSetupForMargins}{%
    \ifmarginsonleft
        \setlength{\oddsidemargin}{\tbmarparboxwidth+\marginparsep}%
        \setlength{\evensidemargin}{\oddsidemargin}%
        \setlength{\textwidth}{\paperwidth-2in-\oddsidemargin}%
    \else
```

```
        \setlength{\oddsidemargin}{0pt}%
        \setlength{\evensidemargin}{\tbmarparboxwidth+\marginparsep}%
        \setlength{\textwidth}{%
            \paperwidth-2in-\oddsidemargin-\evensidemargin}%
    \fi
}
```

\tbplaceMargins We calculate the coordinates of the lower left hand corner of the margin \parbox depending on the value of \ifmarginsonleft.

```
\newcommand{\tbplaceMargins}{{%
    \setlength{\@tempdima}{%
        \paperheight-1in-\topmargin-\headheight-\headsep-\textheight}%
    \xdef\@evenlly{\strip@pt\@tempdima}%
    \xdef\@oddlly{\@evenlly}%
    \setlength{\@tempdima}{1in}%
    \xdef\@evenllx{\strip@pt\@tempdima}%
    \ifmarginsonleft\else
        \setlength{\@tempdima}{1in+\textwidth+\marginparsep}\fi
    \xdef\@oddllx{\strip@pt\@tempdima}%
}}
```

\tbMakeFinalCalcs Executed by \marparboxwidth

```
\newcommand{\tbMakeFinalCalcs}{%
    \tbSetupForMargins
    \tbplaceMargins
}
```

\tbminskipbtnlayers \tbminskipbtnlayers is the minimum skip between layers (top, middle, bottom) Executed by \marparboxwidth

```
\newlength\tbminskipbtnlayers
\setlength{\tbminskipbtnlayers}{6pt}
```

```
\newif\ifiscarryover \iscarryoverfalse
```

carryoverFmt is a work-around for the color problem experienced with carry over text: Suppose there is a change of color of the text on the previous page, the carry over text will naturally be colored the default color, MidMargcolor. To continue the text with the same color as the one the previous page, we enclose the text in the carryoverFmt environment.

```
\newenvironment{carryOverFmt}[1]{#1\let\tb@carryoverFmt\@empty
    \c@rryoverFmt{#1}}{}
```

This command is called by the carryOverFmt environment. it takes its argument, which is a change in color or style, and defines \tb@carryoverFmt, which will be executed on the next page.

```
\def\c@rryoverFmt#1{%
    \ifx\tb@carryoverFmt\@empty
        \global\let\tb@carryoverFmt\@empty
        \xdef\tb@co@page{\thepage}%
        \gdef\tb@carryoverFmt{\ifnum\thepage>\tb@co@page\relax
        #1\global\let\tb@carryoverFmt\@empty\fi}%
    \fi
}
\let\tb@carryoverFmt\@empty
```

`\tb@insertCarryOver` takes its argument, that is always `\unvbox\txtbkb@xb@t}`, and if there is any carryover content, will insert its argument followed by a copy, `\tb@rest@reMarginFmt` of the default margin format. This seems to work for recovering from a change of text or style over a page boundary.

```
\let\tb@rest@reMarginFmt\relax
\def\tb@insertCarryOver#1{%
    \let\tb@rest@reMarginFmt\relax
    \ifiscarryover\ifx\tb@carryoverFmt\@empty\else
        \let\tb@rest@reMarginFmt\tb@midMargFmt
        \tb@carryoverFmt\fi\fi
    #1 \tb@rest@reMarginFmt
}
```

`\eqe@tb@shipout`

We define the shipout to the margins.

Bug: When I use graphicxsp, embed the picture (such as a logo), and use that picture as the `\setTopMargin`, the shipout routine is executed twice for each page. I haven't figured out what causes this, but here is a work around. We record the most recent page number, if it equals the page number of the last iteration of `\eqe@tb@shipout`, we do nothing; otherwise, execute the shipout code.

```
\newif\iftb@shipoutPermitted \tb@shipoutPermittedtrue
\newcommand{\turnOnFTBShipout}{\global\tb@shipoutPermittedtrue}
\newcommand{\turnOffFTBShipout}{\global\tb@shipoutPermittedfalse}
```

`\turnOnFTBShipout`
`\turnOffFTBShipout`

`\turnOnFTBShipout` turns on the shipout, the default, and `\turnOffFTBShipout` turns it off.

```
\newcommand{\eqe@tb@shipout}{\iftb@shipoutPermitted
    \ifnum\arabic{page}=\tblastpageshipped
    \let\tb@so@next\relax\else
    \xdef\tblastpageshipped{\arabic{page}}%
    \def\tb@so@next{\eqe@tb@ship@ut}\fi
    \expandafter\tb@so@next\fi
}
\def\tblastpageshipped{-100}
```

Here is the actual shipout code for writing to the margins.

```
\newcommand{\eqe@tb@ship@ut}{%
    \fboxsep=0pt\setlength{\unitlength}{1pt}%
    \global\setbox\txtbkb@xb@t=\vbox\bgroup
         \color@begingroup
         \hsize=\tbmarparboxwidth
         \vsize=\textheight
         \MarParBoxFmt
         \csname tbTopMargin\endcsname
         \vskip\tbminskipbtnlayers
\set@typeset@protect
         \the\txtbkt@ks
         \color@endgroup\vfil
    \egroup
    \global\setbox\txtbkb@xt@p=\vsplit\txtbkb@xb@t to\textheight
    \ifvoid\txtbkb@xb@t\global\iscarryoverfalse
    \else\global\iscarryovertrue\fi
```

We have three levels the top (`\tbTopMargin`), the bottom (`\tbBotMargin`), and the middle (`\txtbkt@ks`). `\tbTopMargin` is no problem but `\tbBotMargin` requires some special attention.

```
    \ifx\tbBotMargin\relax\else
```

If `\tbBotMargin` is not `\relax`, we begin by putting `\tbBotMargin` into a `\vbox` under the same assumptions, and get its height.

```
        \bgroup\setbox2=\vbox{%
            \color@begingroup\normalcolor
            \hsize=\tbmarparboxwidth\kern0pt
            \MarParBoxFmt\csname tbBotMargin\endcsname
            \color@endgroup
            \kern0pt
        }%
```

We reduce `\textheight` by the height of `\tbBotMargin`

```
        \dimen0=\textheight
        \advance\dimen0-\ht\txtbkb@xh@ld
        \advance\dimen0-\tbminskipbtnlayers
```

We split off the top material by this amount, the new bottom is in `\txtbkb@xt@p` the new top is in `\box0`

```
        \setbox0=\vsplit\txtbkb@xt@p to \dimen0
```

The new bottom (which will overflow to the next page) is the content we clipped off bottom of `\txtbkb@xt@p` and the original overflow material still in `\txtbkb@xb@t`.

```
        \global\setbox\txtbkb@xb@t=\vbox{%
            \unvbox\txtbkb@xt@p\unvbox\txtbkb@xb@t}%
```

We then patch everything together the new top is in `\txtbkb@xt@p` the new top is in `\@tempboxa` followed by `\tbBotMargin` (in `\box\txtbkb@xh@ld`).

```
        \global\setbox\txtbkb@xt@p=\vbox{\unvbox0
            \vfill\vskip\tbminskipbtnlayers
            \vfil\unvbox2\relax}\egroup
    \fi
    \ifodd\value{page}%
        \put(\@oddllx,\@oddlly){%
            \eqe@MarParBox{\unvbox\txtbkb@xt@p}}\else
        \put(\@evenllx,\@evenlly){%
            \eqe@MarParBox{\unvbox\txtbkb@xt@p}}\fi
```

We see if there is any carry over, if yes, we insert into `\txtbkt@ks` for use on the next page, along with a heading, if any.

```
    \global\txtbkt@ks={}\ifvoid\txtbkb@xb@t\else
```

We test whether these is a `\splitbotmark`, if yes, then we will insert it at the top of the next page with formatting.

```
    \if!\splitbotmark!\global\let\tb@sbm@exp\relax\else
        \xdef\tb@sbm@exp{\noexpand\tbPreMarginHeader
            \noexpand\tbMarginHeaderFmt{\splitbotmark}%
            \noexpand\tbPostMarginHeader
            \noexpand\par\kern3pt}%
    \fi
```

Here is the content that will be carried over to the next page, we insert a `\splitbotmark` if it is non-empty (`\tb@tmp@exp`).

```
        \global\txtbkt@ks=\expandafter{\tb@sbm@exp
        \tb@insertCarryOver{\unvbox\txtbkb@xb@t}}%
    \fi
}
```

`\insertpageifcarryover` This macro is use to generate a blank page if there is carry over from the previous page. It is place just after the exercises, and before a new chapter of section. The optional argument allows you to insert something into the new page, if one is automatically created. The default is `\null`.

```
\newcommand{\insertpageifcarryover}[1][\null]{%
```

We begin by starting a new page, the shipout routine of previous page will be initialized and can then get an accurate result for `\ifiscarryover`.

```
    \newpage
```

If there is carryover, we create a new page by inserting a content into the page. If there is no carry over, we do now insert any content, and the page will not be created.

```
    \ifiscarryover\def\eqeifnext{\csname iftrue\endcsname}%
    \PackageInfo{eqexam}{Carry over of content in margin
    from page \thepage.\MessageBreak Creating a blank page}\else
    \def\eqeifnext{\csname iffalse\endcsname}\fi\eqeifnext#1\fi}
```

`\setFullWidthHeader` Makes the running header full width.

```
\newcommand{\setFullWidthHeader}{%
    \setlength{\@tempdima}{%
        \evensidemargin+\tbmarparboxwidth+\marginparsep}%
    \edef\@headoffset{\the\@tempdima}%
    \def\@evenhead{\makebox[0pt]{\makebox[0pt][l]
        {\thepage}\hspace{\@headoffset}}\hfil\slshape\leftmark}%
    \ifmarginsonleft
        \def\@oddhead{\makebox[0pt]{\makebox[0pt][l]
            {\slshape\rightmark}\hspace{\@headoffset}}\hfil\thepage}%
    \else
        \def\@oddhead{{\slshape\rightmark}\hfil\makebox[0pt]
```

```
            {\hspace{\@headoffset}\makebox[0pt][r]{\thepage}}}%
    \fi
}
```

15.4 In support of solutions at end of document and chapter

A feature that may not be used much is to have solutions at the end of each chapter.

\chaptersolutions If \tb@EndOfChapterExercises is executed, and \chaptersolutions is placed between chapters, we can generate solutions at the end of the chapters, instead of at the end of the book. \chaptersolutions is \let to \relax unless \tb@EndOfChapterExercises is executed. In this case \chaptersolutions inputs the the .sol file, then then opens it

```
\newif\ifchapterexercises \chapterexercisesfalse
\let\chaptersolutions\relax
\def\tb@EndOfChapterExercises{%
    \let\include@solutions@chapter\include@solutions
    \def\includeexersolutions{%
        \include@solutions@chapter
          \global\let\include@solutions\relax
    }%
```

\chaptersolutions is redefined from \relax. Input current solutions, close stream, open stream.

```
    \def\chaptersolutions{%
        \includeexersolutions
        \immediate\closeout\ex@solns
        \newwrite \ex@solns \global\let\quiz@solns\ex@solns
        \immediate\openout \ex@solns \jobname.sol
        \ifvspacewithsolns\writeAllAnsAtEnd\fi
    }%
}
```

```
\def\writeallsolutions{\let\chaptersolutions\relax}
```

\exercisesAtEndOfChapter If you want solutions at the end of each chapter, you'll have to execute this command in the preamble. See \initChapAfterSolns for an example of usage.

afterChapSolns This comment environment is a convenience for placing content between chapters.

```
\excludecomment{afterChapSolns}
%\includecomment{solnsAtEnd}
```

(2014/05/08) There is a problem with nested comment environments when solnsAtEnd contains within it the \includeexersolutions command, especially when multicols is used. The fix seems to redefine things so that solnsAtEnd writes to a different CUT file

```
\def\NewCommentCutFile{\def\CommentCutFile{solnsAtEnd.cut}}
\def\RestoreCommentCutFile{\def\CommentCutFile{comment.cut}}
\@ifundefined{BeforeIncludedComment}{%
\long\def\solnsAtEndcomment
 #1{\message{Special comment '#1'}%
    \csarg\def{#1}{\endgroup  \message{Processing '#1' comment.}%
                   \NewCommentCutFile\SetUpCutFile
                   % #2 before SetUp, so we can do renaming.
          \message{Comment '#1' writing to \CommentCutFile.}%
          \ProcessComment{#1}}%
    \csarg\def{After#1Comment}{\immediate\closeout\CommentStream
        \RestoreCommentCutFile\input{solnsAtEnd.cut}\relax}%
    \CommentEndDef{#1}}
}{\long\def\solnsAtEndcomment
 #1{\message{Special comment '#1'}%
    \csarg\def{After#1Comment}{\immediate\closeout\CommentStream
    \RestoreCommentCutFile\input{solnsAtEnd.cut}\relax}%
    \csarg\def{#1}{\NewCommentCutFile\BeforeIncludedComment\relax
          \ProcessComment{#1}}%
    \CommentEndDef{#1}}
 }
\solnsAtEndcomment{solnsAtEnd}
\newcommand{\exercisesAtEndOfChapter}{%
    \ifeq@nosolutions\else
        \typeout{^^J!!!!!Executing in chapter solutions!!!!!^^J}
        \chapterexercisestrue\tb@EndOfChapterExercises
        \ifchapterexercises
        \csarg\let{solnsAtEnd}\@gobble
        \excludecomment{solnsAtEnd}%
        \csarg\let{AftersolnsAtEndComment}\relax
        \includecomment{afterChapSolns}\else
        \excludecomment{afterChapSolns}\fi
    \fi
}
\@onlypreamble\exercisesAtEndOfChapter
```

15.5 Modifying and restoring the Layout

The book may need a wide page format and use multi-columns to display homework sets, or solutions at the end if the book.

`\setFullWidthLayout` A command to set the page layout for the solutions in the back of the book. Typically, we do away with the wide margins. We also save the current values of the parameters we are changing so we can restore them later.

```
\newcommand{\setFullWidthLayout}{%
    \saveBasicLayoutParams
    \setlength{\oddsidemargin}{0in}%
    \setlength{\evensidemargin}{\oddsidemargin}%
    \setlength{\textwidth}{\paperwidth-2in}%
    \setlength{\linewidth}{\paperwidth-2in}%
    \setlength{\columnseprule}{0pt}%
    \def\@evenhead{\thepage\hfil\slshape\leftmark}%
    \def\@oddhead{{\slshape\rightmark}\hfil\thepage}%
}
```

fullwidthtext When `\setFullWidthLayout` is in effect, we have the problem of writing text. Originally, I used a `\parbox` with width of `\linewidth`, but this has it problems when breaking across pages. We have instead an environment for writing; the list environment obeys the current `\linewidth`, which is set to `\paperwidth-2in`, this latter value may not always be correct (especially when the margins are smaller than 2in.

```
\newenvironment{fullwidthtext}{%
\begin{list}{}{%
    \setlength{\labelwidth}{0pt}\setlength{\labelsep}{0pt}%
    \setlength{\itemindent}{0pt}\setlength{\itemsep}{0pt}%
    \setlength{\topsep}{0pt}\setlength{\parsep}{0pt}%
    \setlength{\listparindent}{\parindent}%
    \setlength{\leftmargin}{0pt}\setlength{\rightmargin}{0pt}
}\item\relax}{\end{list}}
```

\restorePageLayout Restore the last saved page parameters.

```
\newcommand{\restorePageLayout}{\newpage
    \setlength{\oddsidemargin}{\tb@osms}
    \setlength\evensidemargin{\tb@esms}
    \setlength{\textwidth}{\tb@tws}
    \setlength{\linewidth}{\tb@lws}
    \setlength{\columnseprule}{\tb@csr}
}
```

Used by `\setFullWidthLayout` just before the page layout parameters are changed.

```
\newcommand{\saveBasicLayoutParams}{%
    \xdef\tb@osms{\the\oddsidemargin}%
    \xdef\tb@esms{\the\evensidemargin}%
    \xdef\tb@tws{\the\textwidth}%
    \xdef\tb@lws{\the\linewidth}%
    \xdef\tb@csr{\the\columnseprule}%
}
```

\initChapAfterSolns Initializes the environment when solutions appear after each chapter. Example of usage, taken from fortextbook.ltx,

```
\begin{afterChapSolns}
\initChapAfterSolns
\section{Solutions to Chapter Exercises}
\begin{fullwidthtext}
We present short solutions to the problems.
We present short solutions to the problems.
We present short solutions to the problems.
We present short solutions to the problems.
\end{fullwidthtext}
\bigskip
```

```
\begin{multicols}{2}\forceNoColor
\chaptersolutions
\end{multicols}
\restoreFromChapAfterSolns
\end{afterChapSolns}
```

```
\newcommand{\initChapAfterSolns}{\newpage
    \clearTopMargin\clearBotMargin
    \setFullWidthLayout
}
```

\restoreFromChapAfterSolns Restores the saved parameters at the end of the chapter solutions, see above for an example.

```
\newcommand{\restoreFromChapAfterSolns}{\newpage
    \restorePageLayout\setFullWidthHeader
}
```

```
% End of textbook segment
⟨/textbook⟩
⟨*package⟩
```

15.6 We shipout in support of fortextbook

We shipout \eqe@tb@shipout to be placed in the margins on every page.

```
\ifeqfortextbook
\AtBeginDocument{\tb@soln@choice
    \ifeqwritetomargins\chkmarginboxwidth
    \AddToShipoutPicture{\eqe@tb@shipout}\fi}
\fi
⟨/package⟩
⟨*textbook⟩
```

15.7 Modify eqequestions environment

We adjust the eqequestions environment to minimize spacing between problems.

```
\eqequestopsep{0pt}
\eqequesparsep{0pt}
\eqequesitemsep{0pt}
\eqequeslistparindent{0pt}
\renewenvironment{eqequestions}{%
    \begin{list}{}{%
    \ifwithinsoldoc\let\solnItemMngt\eqeSolnItemMngt\fi
    \setlength{\labelwidth}{\eqemargin}%
    \setlength{\parsep}{\eqeques@parsep}%
    \setlength{\itemsep}{\eqeques@itemsep}
    \setlength{\topsep}{\eqeques@topsep}%
    \setlength{\itemindent}{0pt}%
    \setlength{\listparindent}{\eqeques@listparindent}%
    \ifwithinsoldoc\settowidth{\labelsep}{\eqe@hspannerSoln}\else
    \settowidth{\labelsep}{\eqe@hspannerPrb}\fi
    \setlength{\leftmargin}{\labelwidth}%
    }\ifwithinsoldoc\global\firstitemtrue\fi\item\relax}{\end{list}}
```

`eqepartsquestions` This environment is used in the SOL file with problems with parts to hang indent the solutions with parts.

```
\newcommand{\eqepquestopsep}[1]{\def\eqepques@topsep{#1}}
\newcommand{\eqepquesparsep}[1]{\def\eqepques@parsep{#1}}
\newcommand{\eqepquesitemsep}[1]{\def\eqepques@itemsep{#1}}
\eqepquestopsep{\eqeques@itemsep}
\eqepquesparsep{\eqeques@parsep}
\eqepquesitemsep{\eqeques@itemsep}
\newenvironment{eqepartsquestions}{%
    \begin{list}{}{%
    \settowidth{\labelwidth}{\eqe@prtsepSoln\hspace{\tbsolnpartwdth}}
    \setlength{\parsep}{\eqepques@parsep}%
    \setlength{\itemsep}{\eqepques@itemsep}%
    \setlength{\topsep}{\eqepques@topsep}%
    \setlength{\itemindent}{0pt}%
    \settowidth{\labelsep}{\eqe@hspannerSoln}
    \setlength{\leftmargin}{\labelwidth}%
    }\item\relax}{\end{list}}
```

15.8 Modifications for solutions page

`\gobbletoEndEXt` `\eqExtArg` is a command to gobble all content from the current position `\eqEXt` down to `\endeqEXt`. In the solutions file ends with `\par\medskip`, which we gobble up too. We define `\eqExtArg` to `\thequestionno` so we can use the problem number to filter out the even-problems.

```
\long\def\gobbletoEndEXt#1\endeqEXt{\@gobbletwo}
\def\eqExtArg{\theeqquestionnoi}
```

`\autoInsSolns` When creating a book, we can manually create a chapter and insert the solutions, of we can automatically have it done. The manual method is the default, emit `\autoInsSolns` in the preamble to have the solutions inserted automatically. Use `\InputExrSolnsLevel` to tune the section level (`eqexam.def`).

```
\if\load@exerquiz\eqe@NO\DoNotFitItIn\fi
\let\fillInFormatDefault\@empty
\def\fbInsSolnsStyle{\def\exerSolnsHeadnToc{}}
\def\autoInsSolns{\let\fbInsSolnsStyle\relax}
\AtBeginDocument{\fbInsSolnsStyle}
\renewcommand{\exerSolnInput}{%
    \global\let\webnewpage\relax
    \ifsolutionsonly\else\immediate\closeout\ex@solns\fi
    \ifeq@nosolutions\else\newpage % 2012-03-14
        \iftherearesolutions\eq@solutionshook\eqsolutionshook
            \ifsolutionsonly\else\newpage\eq@solutionshook
            \eqsolutionshook\fi
            \ifx\webnewpage\relax
```

```
                \gdef\webnewpage{\global\let\webnewpage\newpage}%
            \fi
            \priorexsectitle\exerSolnsHeadnToc\priorexslinput
            \InputIfFileExists{\jobname.sol}{}{\PackageWarning{exerquiz}
            {!!! Solutions to exercises not found}}%
        \fi
    \fi
}
```

\eqedsplyOnlyFrst The default listing of a problem with multiple parts is to typeset <num>. (<part>). Here, we do not typeset the number after the first time.

```
\setcounter{partno}{1}\edef\firstPartLtr{\thepartno}
```

???? 6/2/11 When part (a) is hidden we need to generate the questions number for the the first non-hidden part. Created \iffrstProbNumShown to help but it not working yet.

```
\newif\iffrstProbNumShown\frstProbNumShownfalse
\def\tb@insertDecPoint{\ifwithinsoldoc\eqedecPointSoln\else
    \eqe@decPointMrg\fi}
\newcommand{\eqedsplyOnlyFrst}[2]{\def\thisPart{#2}%
    \ifx\thisPart\firstPartLtr\global\frstProbNumShowntrue
        \tb@mrgDigitFmt{#1}\tb@insertDecPoint\else
        \iffrstProbNumShown\tb@GenProbNum{#1}\else
        \global\frstProbNumShowntrue\tb@mrgDigitFmt{#1}%
        \tb@insertDecPoint\fi\fi\global\eqeGenProbNumfalse
}
```

\displayProbNumOnce If a part is carried over to the next page, it may be necessary to manually force the display of the first digit.

```
\insMidMarg{\displayProbNumOnce}
```

```
\newif\ifeqeGenProbNum \eqeGenProbNumfalse
\newcommand{\displayProbNumOnce}{\global\eqeGenProbNumtrue} %
%\def\tb@GenProbNum#1{\ifeqeGenProbNum#1\eqe@decPointMrg\else
\def\tb@GenProbNum#1{\ifeqeGenProbNum#1\tb@insertDecPoint\else
    \phantom{#1\tb@insertDecPoint}\fi}%
```

```
\def\sq@priorhook{\medskip}
```

Adjustments of spacing between problems \eqexerskip, and the check for enough room for the next problem.

```
\def\default@fvsizeskip{.1}
```

The skip prior to the beginning of an exercise

```
\aboveexskip{0pt}
```

The skip after the end of an exercise

```
\belowexskip{0pt}
```

The skip in the solutions file following an exercise OR a part of an exercise The text of this command should be a single token, that's why I've enclosed it in braces. (There is a `\@gobbletwo` that gobbles it up for the `studented` option.)

```
\renewcommand\belowexsolnskip{{}}
```

We remove the `\mark` from this definition, see original definition in `eqexam.def`

```
\renewcommand\exerSolnHeader[3]{%
    \ifeqforpaper\else\webnewpage\fi%\par
    \noindent\@ifundefined{hypertarget}
    {#3}{\hypertarget{#2}{#3}\relax}\solnhspace
}
```

This causes the `eqexam` environment to write the user friendly name of the exam even if there is only one exam.

```
\def\nNumberOfP@rts{0}
```

15.9 Some Convenience/Formatting Commands

`\preExamSolnHead`
`\examSolnHeadFmt`
`\postExamSolnHead`

These are redefinitions of commands defined in `eqexam`, They control the vertical spacing before and after a heading in the solutions at the end of the book, as well as the formatting.

```
\renewcommand{\preExamSolnHead}{\medbreak\noindent}
\renewcommand{\examSolnHeadFmt}[1]{\textbf{#1}}
\renewcommand{\postExamSolnHead}{\smallskip}
```

`\wrtChapSolnHead`

Writes a chapter heading to the solution file, usage,

```
\wrtChapSolnHead{The New {\eqexam}}
```

```
\newcommand{\wrtChapSolnHead}[1]{%
    \writeT@SolnFile{%
    \protect\preChapSolnHead
    \protect\chapHeadSolnFmt{\protect\ftbFmtChapter{\thechapter}#1}%
    \protect\postChapSolnHead
}}
```

`\preChapSolnHead`
`\chapHeadSolnFmt`
`\postChapSolnHead`

Same as above, except for chapter headings.

```
\newcommand{\preChapSolnHead}{\bigbreak\noindent}
\newcommand{\chapHeadSolnFmt}[1]{{\large\textbf{#1}}}
\newcommand{\postChapSolnHead}{\medbreak}
```

`\ftbFmtChapter`

This command may (optionally) insert the chapter number into the chapter title passed to `\wrtChapSolnHead`. The default is to pass the chapter name ("Chapter") and chapter number. If you say `\let\ftbFmtChapter\@gobble`, the chapter name and number will not appear. You can redefine this command as desired.

```
\newcommand{\ftbFmtChapter}[1]{\chaptername\space#1.\space\ignorespaces}
```

In the solution manual, all these chapter commands may be redefined like so

```
\let\preChapSolnHead\relax
\let\chapHeadSolnFmt\chapter
```

```
\let\ftbFmtChapter\@gobble
\let\postChapSolnHead\relax
```

In fact, let's make this into a command.

`\convertChapHeadToChapters` In the solutions manual, the chapter headings will become chapters of the manual, rather than just a bold faced heading.

```
\newcommand{\convertChapHeadToChapters}{%
    \let\preChapSolnHead\relax
    \let\chapHeadSolnFmt\chapter
    \let\ftbFmtChapter\@gobble
    \let\postChapSolnHead\relax
}
```

`\probSet` A simple command to announce the problem set.

```
\subsection*{\probSet{\thesection}}
```

See also the definition for the `probset` environment below.

```
\newcommand{\probSet}[1]{Problem Set #1}
```

`\annotPage` Use to annotation the page number onto a solution heading, for example,

```
\begin{exam}[\thesection. Another Section\annotPage]{\autoExamName}
```

or using the `probset` environment defined below

```
\begin{probset}{{\thesection} Setting the page layout\annotPage}
```

```
\newcommand{\annotPage}{\protect\annotThePage{\thepage}}
\newcommand{\annotThePage}[1]{\space(page\protect~#1)}
```

15.10 The probset and example environments

We define two environments based. The first (`probset`) is based on the `exam` environment; the second (example) is based on the `exercise` environment.

`probset` A convenience environment, it is the `exam` environment, renamed, with different arguments. `#1` is the heading that will appear in the margins, and `#2` is the heading to appear in the back of the book.

```
\def\noProbHeader{NPH}
\newenvironment{probset}[2][\probSet{\thesection}]{%
    \exam[#2]{\autoExamName}\ifx#1\noProbHeader\else
    \protected@edef\ftb@tmp@exp{\noexpand\insProbHead{#1}}%
    \ftb@tmp@exp\fi}{\endexam}
```

```
\newcounter{exampleno}[section]
\renewcommand{\theexampleno}{\arabic{section}.\arabic{exampleno}}
```

Note: The counter is `exampleno` and is designed to show the section number and example number, and to reset at each section. To change the definition of

\theexampleno to reflect the chapter number followed by the example number, and resetting at the beginning of each new chapter, the following code is needed.

```
\usepackage{remreset}
\@removefromreset{exampleno}{section}
\renewcommand{\theexampleno}{\arabic{chapter}.\arabic{exampleno}}
\@addtoreset{exampleno}{chapter}
```

\examplenoname is the label placed on the example.

```
\newcommand{\examplenoname}{Example}
```

example A simple example environment, based on the exercise environment.

```
\newenvironment{example}{%\medskip
    \belowexskip{\medskipamount}\aboveexskip{\medskipamount}%
    \makeRoomForProb{\@nbaselineskip\baselineskip}{}%
    \renewcommand\exlabel{Example}%
    \renewcommand\exlabelformat{\textbf{\exlabel~\theexampleno.}}%
    \let\eq@fititin\eqfititin
    \renewcommand\exrtnlabelformat{$\square$}%
    \def\eqexheader@wrapper{\eqexheader}%
    \SolutionsAfter
    \begin{exercise}[exampleno]}{\end{exercise}}
```

example* An example environment with parts.

```
\newenvironment{example*}{%\medskip
    \belowexskip{\medskipamount}\aboveexskip{\medskipamount}%
    \makeRoomForProb{\@nbaselineskip\baselineskip}{}%
    \renewcommand\exlabel{Example}%
    \renewcommand\exlabelformat{\textbf{\exlabel~\theexampleno.}}%
    \let\eq@fititin\eqfititin
    \renewcommand\exrtnlabelformat{$\square$}%
    \def\eqexheader@wrapper{\eqexheader}%
    \SolutionsAfter
    \begin{exercise*}[exampleno]}{\end{exercise*}}
```

We set some parameters, to values better suited for the option.

```
\setDefaultfvsizeskip{.1}
\nbaselineskip{4}
```

15.11 Commands in support of Solution Manuals

Generally, the solution manual source file should have the same packages as the source file for the book itself, perhaps with a few exceptions, but definitely the eqexam package is required with its fortextbook option.

At this time, we provide only two commands, these are \ftbInputBookAux and \ftbInputSolnFiles.

\ftbInputBookAux This command is used to input the auxiliary files of the master source file. It takes one argument, the name of the master source file (myBook.ltx or myBook.tex). If the extension is not present, it is assumed to be .tex.

```
\newcommand{\ftbInputBookAux}[1]{%
    \filename@parse{#1}\@ifundefined{filename@ext}%
        {\def\filename@ext{tex}}{}%
    \xdef\tbBaseName{\filename@base}%
    \xdef\tbSourceFile{\filename@base.\filename@ext}%
```

In the next 4 lines, we save `\@writefile` and `\@setckpt`, and `\let` them to `\@gobbletwo`. We restore their definitions after we input the aux files. We include the aux files of the source file in case there are some cross references in the solution files, or the body of the text would like to refer back to the original book. (Seems unlikely.)

```
    \let\save@writefile\@writefile
    \let\@writefile\@gobbletwo
    \let\save@setckpt\@setckpt
    \let\@setckpt\@gobbletwo
    \makeatletter
    \InputIfFileExists{\tbBaseName.aux}{%
        \PackageInfo{eqexam}
            {Inputting auxiliary files of\MessageBreak\tbSourceFile}%
        }{%
        \PackageError{eqexam}
            {Auxiliary files for \tbSourceFile\space were not found}
            {Compile the source file \tbSourceFile\space
                three times\MessageBreak%
                to create the required auxiliary files.}%
        }%
    \makeatother
```

The solution files really shouldn't have a label, but if we do we'll save the LaTeX definition of `\label`, and `\let` it two `\@gobble`. Within the body of the solutions, the command `\ftblabel` may be used to cross reference, if needed.

```
    \global\let\ftblabel\label
    \let\label\@gobble
    \let\@writefile\save@writefile
    \let\@setckpt\save@setckpt
}
\@onlypreamble\ftbInputBookAux
```

`\restorelabel` `\gobblelabel` These two are used to restore the usual definition of `\label`, and to cancel it out by letting it to `\@gobble`.

```
\newcommand{\restorelabel}{\global\let\label\ftblabel}
\newcommand{\gobblelabel}{\let\label\@gobble}
```

`\ftbInputSolnFiles` In the body of the text, place `\ftbInputSolnFiles` in vertical mode. This will input the `.sol` file of the master source document. The optional argument is the name of the solution file. The default name is `\tbBaseName.sol`, where `\tbBaseName` was defined in `\ftbInputBookAux`. If no extension is specified, an extension of `.sol` is assumed. The original `.sol` may have changed its name, if someone renamed it (to keep it from being overwritten). The solution file may be editing (by hand) as needed.

```
\newcommand{\ftbInputSolnFiles}[1][\tbBaseName.sol]{%
    \filename@parse{#1}\@ifundefined{filename@ext}%
        {\def\filename@ext{sol}}{}%
    \xdef\tbBaseName{\filename@base}%
    \xdef\tbSourceFile{\filename@base.\filename@ext}%
    \InputIfFileExists{\tbBaseName.sol}{%
        \PackageInfo{eqexam}
            {Inputting solutions file \tbBaseName.sol\MessageBreak}%
        }{%
        \PackageError{eqexam}
            {Solutions file for \tbSourceFile\space was not found}%
            {Compile the source files three times}%
        }%
}

% End of textbook segment
⟨/textbook⟩
⟨*ftbsty⟩
```

16 fortextbook Style File

One person said it would be nice to separate eqexam from the `fortextbook` option, and have fortextbook as a `separate` style (package). Rather than spending tens of hours separating them I create a simple “wrapper” package, which simply calls eqexam with the `fortextook` option along with all the recommended options.

Usage:

```
\documentclass[twoside,letterpaper]{book}
\usepackage[fleqn]{amsmath}
\usepackage{fortextbook}
...
```

Below is the style.

```
\NeedsTeXFormat{LaTeX2e}
\ProvidesPackage{fortextbook}
 [2012/03/14 v1.0 A fortextbook Package (dps)]
\DeclareOption*{\PassOptionsToPackage{\CurrentOption}{eqexam}}
\ProcessOptions
\RequirePackage[%
    ftbsolns,fortextbook,usecustomdesign,
    forcolorpaper,noseparationrule,usexkv
]{eqexam}
```

In support of this style, I’ve also defined `\NoSolutions` to compile the document without creating the solutions at the end of the file (this reduces the amount if IO when compiling). I’ve also defined a special option `nocustomdesign` which cancels out the `usecustomdesign` option.

```
% End of ftbsty segment
⟨/ftbsty⟩
⟨*package⟩
```

Input `eqtextb.def`**.** Back in the main package, we choose this point to input the `fortextbook` code (`eqtextb.def`) if the `fortextbook` option is taken.

```
\edef\ftbInputEqTextb{\ifeqfortextbook\noexpand
    \InputIfFileExists{eqtextb.def}{}{}\fi}
\ftbInputEqTextb
```

17 xkeyval Extensions

We load this material if xkeyval exists, and if the document author has specified the `usexkv` option.

2014/12/19 Now, the `usexkeys` is on by default.

```
%\IfFileExists{xkeyval}{%
%    \if\eq@usexkeys\eqe@YES\RequirePackage{xkeyval}\else
%    \endinput\fi}{\endinput}
```

We redefine selected commands if the user has specified the `usexkv` option.

New options for `\fillin`

`underline` Underline the fillin

`u,b` Legacy parameters, underlines (`u`) or leaves a blank space (`b`)

`boxed` Boxes in the response region

`boxpretext` When boxed is use, use this to insert text in front of the answer, for example, `x=`

`boxsize` When boxed is taken, use boxsize to set the size of the box; permissible choices are tiny, scriptsize, footnotesize, small, normalsize, large, Large, LARGE, huge, Huge

`align` Align the answer within the response region, permissible values are `l`, `c`, `r`.

`color` The color of the response (named color)

`format` Special formatting for the answer, the default is `\bfseries`

`enclosesoln` This Boolean key only takes effect when the `boxed` key is used, and when either the `nosolutions` or the `vspacewithsolns` option is taken. When these conditions are met, a box is created around the solution (the third parameter of `\fillin`); the solution is enclosed in a `\phantom` so it is not seen, but the dimensions of the solution are used. This key allows you to create a box or arbitrary dimension.

fitwidth The fitwidth option uses the natural width of the answer to create the fillin when the answerkey option is in effect; otherwise it uses the second parameter #2.

parbox The parbox parameter may be used to create a multiline \fillin box. The value of parbox is the same as the first three parameters of the LaTeX command \parbox, e.g., parbox={[t][.5in][t]}. The value needs to be enclosed in braces.

hiddenbox When the boxed option is used, this option resets the \fbox parameters to 0pt, making the box "hidden."

Below are the xkeyval definitions of the keys recognized by \fillin. Add some logic to the underline key, now it is equivalent to the b key.

```
\define@boolkey{eqFillin}{underline}[true]{}
\define@key{eqFillin}{u}[]{\KV@eqFillin@underlinetrue}
\define@key{eqFillin}{b}[]{\KV@eqFillin@underlinefalse}
\define@boolkey{eqFillin}{boxed}[true]{}
\define@key{eqFillin}{boxpretext}[]{\def\eq@fillintext{#1}}
\let\eq@fillintext\@empty
```

If the user just says parbox,... the value of \eq@fillinparbox is \relax. If parbox does not appear in the option list, \eq@fillinparbox has a default value of \@empty. In this way, we can distinguish between parbox with the empty value, and parbox not present at all.

```
\define@key{eqFillin}{parbox}[\relax]{\def\eq@fillinparbox{#1}}
\let\eq@fillinparbox\@empty
\define@key{eqFillin}{hiddenbox}[]{%
    \def\eq@fillinhiddenbox{%
        \setlength{\fboxrule}{0pt}\setlength{\fboxsep}{0pt}}}
\let\eq@fillinhiddenbox\@empty
\define@boolkey{eqFillin}{enclosesoln}[true]{}
\define@choicekey+{eqFillin}{boxsize}{tiny,scriptsize,footnotesize,%
    small,normalsize,large,Large,LARGE,huge,Huge}[normalsize]{%
    \def\eq@eqFillin@boxsize{\text{\csname#1\endcsname\strut}}%
}{\PackageWarning{eqexam}{Bad choice for boxsize, permissible values
    are tiny, scriptsize, footnotesize, small, normalsize,
    large, Large, LARGE, huge and Huge. Try again}}
\def\eq@eqFillin@boxsize{\text{\normalsize\strut}}

\define@key{eqFillin}{fboxsep}[3pt]{\def\eq@fillin@fboxsep{#1}}
\def\eq@fillin@fboxsep{3pt}
\define@choicekey+{eqFillin}{fontsize}{tiny,scriptsize,footnotesize,%
    small,normalsize,large,Large,LARGE,huge,Huge}[normalsize]{%
    \def\eq@eqFillin@fontsize{\csname#1\endcsname}%
}{\PackageWarning{eqexam}{Bad choice for boxsize, permissible values
    are tiny, scriptsize, footnotesize, small, normalsize,
    large, Large, LARGE, huge and Huge. Try again}}
\def\eq@eqFillin@fontsize{\ifmmode\else\normalsize\fi}
\define@key{eqFillin}{color}[\eq@fillinColor]{\edef\eq@fillin@color{#1}}

\define@choicekey+{eqFillin}{align}[\val\nr]%
    {l,r,c}[\eq@eqFillin@align@default]{%
    \def\eq@eqFillin@align{#1}%
    \ifcase\nr\relax
        \def\eqe@align@hfill{}\or
        \def\eqe@align@hfill{\hfill}\or
        \def\eqe@align@hfill{\hfil}\fi
    }{%
    \PackageWarning{eqexam}{Bad choice for align, permissible values
    are l, r, and c. Try again}}
\let\eqe@align@hfill\relax
```

`defaultalign` is used to change the values of the default macros `\eq@eqFillin@align@default` and `\eqe@align@hfill@default` for the `align` key together.

```
\define@choicekey+{eqFillin}{defaultalign}[\val\nr]{l,r,c}[c]{%
    \def\eq@eqFillin@align@default{#1}%
    \ifcase\nr\relax
        \def\eqe@align@hfill@default{}\or
        \def\eqe@align@hfill@default{\hfill}\or
        \def\eqe@align@hfill@default{\hfil}\fi
    }{%
    \PackageWarning{eqexam}{Bad choice for defaultalign, permissible
    values are l, r, and c. Try again}}
\setkeys{eqFillin}{defaultalign=c}
```

`\fillInFormatDefault` is the default fill-in format

```
\renewcommand{\fillInFormatDefault}{\normalfont}
\define@key{eqFillin}{format}[\fillInFormatDefault]{%
    \def\eq@fillin@format{#1}}
\def\eq@fillin@format{\bfseries}
\def\eqe@fbox@corr#1{#1-2\fboxsep-2\fboxrule}
\define@boolkey{eqFillin}{fitwidth}[true]{} %
```

`boxcmd` The `boxcmd` key is used to define a boxing command. The default is either `\boxed` or `\fbox`. You can say `boxcmd={\fboxcolor{blue}{yellow}}` to obtain a box with a blue frame and yellow background.

```
\@ifundefined{boxed}{%
        \def\eq@fillin@defaultbox{\fbox}%
        \def\eq@fillin@boxcmd{\fbox}%
    }{%
        \def\eq@fillin@defaultbox{\boxed}%
        \def\eq@fillin@boxcmd{\boxed}%
    }
```

`ulcmd` The `ulcmd` key is used to define a underlining command. The default is either `underline` or `underbar`. You can say `ulcmd=underline` to obtained an underline using `\underline`.

```
\define@key{eqFillin}{boxcmd}%
    [\eq@fillin@defaultbox]{\def\eq@fillin@boxcmd{#1}}
\@ifundefined{underbar}{%
        \def\eq@fillin@defaultul{underline}%
        \def\eq@fillin@ulcmd{\underline}%
    }{%
        \def\eq@fillin@defaultul{underbar}%
        \def\eq@fillin@ulcmd{\underbar}%
    }
```

(2017/01/28) Added custom underline option, the author must define `\ulcustom` and specify `ulcmd=custom`.

```
\edef\ulcustom{\expandafter\noexpand\eq@fillin@ulcmd}
\edef\temp@exp{\noexpand
\define@choicekey+{eqFillin}{ulcmd}
    {underbar,underline,custom}[\eq@fillin@defaultul]}
\temp@exp{\def\@rgi{#1}\def\eq@custom{custom}\ifx\@rgi\eq@custom
    \def\eq@fillin@ulcmd{\@nameuse{ulcustom}}\else
    \def\eq@fillin@ulcmd{\@nameuse{#1}}\fi
}{\PackageWarning{aeb}{Bad choice for ulcmd, permissible values
   are underbar and underline. Try again}}
```

`lift`
`autolift`
`addtoautolift`

(2013/02/16) `lift` lifts (actually lowers) the underline by the amount specified. `autolift` measures the depth of the content and lifts (actually lowers) the underline by that amount. `addtoautolift` works with `autolift` to add the specified amount to the amount of lift as calculated by `autolift`.

```
\define@key{eqFillin}{lift}[-1sp]{\def\eq@fillin@lift{#1}}
\define@boolkey{eqFillin}{autolift}[true]{} %
\define@key{eqFillin}{addtoautolift}[0pt]{%
    \def\eq@fillin@addtoautolift{#1}}
```

The macro `\eqe@getiiiOpts` is based on early parsing code of `\parbox`. It picks up three optional parameters and saves their values under the commands `\eqe@opts@argi`, `\eqe@opts@argii`, `\eqe@opts@argiii`. We are interested in `\eqe@opts@argiii`, which specifies the depth of the `\parbox`. If the `boxed` option is taken, we reduce the value of `\eqe@opts@argiii` by `2\fboxsep+2\fboxrule` so that the height will be exactly as specified. The macro `\eqe@getiiiOpts` is used with the `parbox` option of `\fillin`. The macro `\eqe@getiiiOpts` has syntax:

`\eqe@getiiiOpts[`*`pos`*`][`*`height`*`][`*`inner-pos`*`]`

```
\def\eqe@getiiiOpts{%
    \@ifnextchar[%]
    \i@eqe@getiiiOpts
    {\iii@eqe@getiiiOpts{c}{\relax}[s]}}
```

Get *`pos`*

```
\def\i@eqe@getiiiOpts[#1]{%
    \@ifnextchar[%]
    {\ii@eqe@getiiiOpts{#1}}%
    {\iii@eqe@getiiiOpts{#1}{\relax}[s]}}
```

Get *`height`*

```
\def\ii@eqe@getiiiOpts#1[#2]{%
    \@ifnextchar[%]
    {\iii@eqe@getiiiOpts{#1}{#2}}%
    {\iii@eqe@getiiiOpts{#1}{#2}[#1]}}
```

Get *inner-pos*

```
\def\iii@eqe@getiiiOpts#1#2[#3]{%
    \def\eqe@opts@argi{#1}%
    \def\eqe@opts@argii{#2}%
    \def\eqe@opts@argiii{#3}}
```

Redefine the `\fillin` command

`\fillin` Re-worked `\fillin` to have xkeyval in the optional first parameter. The syntax is illustrated below.

```
\fillin[
      underline=true|false,u,b,boxed=true|false,boxpretext=<text>,
      align=l|r|c,boxsize=\tiny|..\normalsize|\large|...|\Huge,
      color=<namedcolor>,format=<\bfseries|\ttfamily|\Large|whatever>
]{<width>}{<ans>}
```

`\setfillinDefaults` `\setfillinDefaults` sets any default options for `\fillin` the document author wants.

```
\newcommand{\setfillinDefaults}[1]{\def\eqe@argi{#1}\ifx\eqe@argi\@empty
    \let\eqe@setfillinDefaults\@empty\else
    \def\eqe@setfillinDefaults{#1}\fi}
\let\eqe@setfillinDefaults\@empty
```

`\fillineol` The `\fillineol` command is used to extend the `\fillin` box or line to the end of the line.

`\fillineol*{⟨phrase⟩}[⟨opts⟩]{⟨ans⟩}`

Normally, `\fillineol` is placed at the beginning of a line. The command measures the length of {⟨*phrase*⟩}, subtracts this from `\linewidth` and uses this value as the width of the underlying `\fillin`. When the star-form is used, the collectbox package is required; in the star-form, ⟨*phrase*⟩ can contain verbatim text.

```
\AtBeginDocument{\@ifpackageloaded{collectbox}
  {\let\eqe@cb\eqe@YES}{\let\eqe@cb\eqe@NO}}
\def\fillineolTooLongMsg{\PackageWarning{eqexam}{%
  The 'phrase' you are measuring is longer than\MessageBreak
  \string\linewidth. Changing length to 0pt in hopes you\MessageBreak
  can fix things}}
\def\fillineolNoCBMsg{\PackageWarning{eqexam}{%
  For the \string\fillineol*\space form, the
    collectbox package\MessageBreak
  is required, but not loaded at this time.\MessageBreak
  Switching to \string\fillineol\space in hopes you
    load collectbox\MessageBreak
  next time or you remove the star-option if not\MessageBreak
  really needed}}
\newcommand\fillineol{\@ifstar{\ifx\eqe@cb\eqe@YES
  \let\eqe@next\cbfillineol\else
  \def\eqe@next{\fillineolNoCBMsg\fillineol@i}\fi
  \eqe@next}{\fillineol@i}}
\newcommand{\fillineol@i}[1]{\bgroup\setbox\z@\hbox{#1}\fillineol@ii}
\newcommand\fillineol@ii[2][]{%
  \setlength\eqetmplengthb{\linewidth-\wd\z@}%
  \ifdim\eqetmplengthb<0pt\eqetmplengthb0pt\fillineolTooLongMsg\fi
  \unhbox\z@\fillin[#1]{\eqetmplengthb}{#2}\egroup}
```

Allow the third argument to have verbatim text as well.

```
\newcommand\cbfillineol{\bgroup\collectboxto{\@tempboxa}
  {\setlength{\eqetmplengthb}{\linewidth-\wd\@tempboxa}%
  \ifdim\eqetmplengthb<0pt\eqetmplengthb0pt\fillineolTooLongMsg\fi
    \unhbox\@tempboxa\cbfillineol@ia}}
\newcommand\cbfillineol@ia[1][]{\def\@rgi{#1}\cbfillineol@ib}
\newcommand\cbfillineol@ib{\collectboxto{\@tempboxa}%
  {\cbfillineol@i[\@rgi]}}
\newcommand\cbfillineol@i[1][]{\fillin[#1]{\eqetmplengthb}%
  {\unhbox\@tempboxa}\egroup}
```

\fillin We finally begin the \fillin command.

```
\renewcommand{\fillin}[3][]{\begingroup
```

\ifsp@expand is defined in spdef package. This is a version if \ifsp that expands correctly in an \edef.

```
    \expandafter\let\expandafter\ifsp\csname ifsp@expand\endcsname
```

Get the keys indicated by the document author.

```
    \setkeys{eqFillin}{boxsize,underline=true,boxed=false,%
    boxpretext,color,format,enclosesoln=false,fitwidth=false,lift,%
    autolift=false,addtoautolift}%
    \ifx\eqe@setfillinDefaults\@empty\else
        \protected@edef\eq@temp@exp{\noexpand
        \setkeys{eqFillin}{\eqe@setfillinDefaults}}%
        \eq@temp@exp
    \fi
    \protected@edef\eq@temp@exp{\noexpand\setkeys{eqFillin}{#1}}%
    \eq@temp@exp
    \fboxsep\eq@fillin@fboxsep\relax
```

If boxed, we turn underlining off

```
    \ifKV@eqFillin@boxed\KV@eqFillin@underlinefalse\fi
```

Put \eq@fillin@lift to -2sp as a signal not to use the lift value in the case the author specified autofill and lift.

```
    \ifKV@eqFillin@autolift\def\eq@fillin@lift{-2sp}\fi
```

Get the second parameter.

```
    \edef\eqe@argii{#2}\ifx\eqe@argii\@empty\else
    \ifdim\eqe@argii=0pt\let\eqe@argii\@empty\fi\fi
```

We reset \fboxrule and \fboxsep as needed.

```
    \eq@fillinhiddenbox
```

If the document author uses the `hiddenbox` option, this option assumes the `boxed` option as well so we'll set `\KV@eqFillin@boxedtrue` to signal the `boxed` option.

```
    \ifx\eq@fillinhiddenbox\@empty\else
        \KV@eqFillin@boxedtrue\fi
```

If the `parbox` option is taken, we define the third parameter to be wrapped in a `\parbox`.

```
    \ifx\eq@fillinparbox\@empty\def\eqe@argiii{#3}\else
```

If `parbox` is specified, we make `align=l` the default.

```
        \ifx\eqe@align@hfill\relax
        \def\eq@eqFillin@align{l}\def\eqe@align@hfill{}\fi
```

If `parbox` is specified, we get its three optional parameters so we can manipulate the width parameter.

```
        \expandafter\eqe@getiiiOpts\eq@fillinparbox\relax
```

Now, if this is to be `boxed`, we reduce the height of the box (`\boxed` increases the height by `2\fboxrule+2\fboxrule`

```
        \ifKV@eqFillin@boxed
```

`\eqe@opts@argii` has a value of `\relax` if the document author did not specify a height for the box.

```
            \expandafter\ifx\eqe@opts@argii\relax\else
            \edef\eqe@opts@argii{\expandafter
                \eqe@fbox@corr\expandafter{\eqe@opts@argii}}\fi
        \fi
```

We need to feed `\parbox` the parameters it expects, so, if the height parameter is not given, we just pass the first argument; otherwise, we pass all three parameters.

```
        \edef\eqe@parboxOptArgs{[\eqe@opts@argi]%
            \expandafter\ifx\eqe@opts@argii\relax\else
            [\eqe@opts@argii][\eqe@opts@argiii]\fi}%
```

Now we build the third parameter, `\eqe@argiii`.

```
        \def\eqe@argiii{\expandafter\parbox\eqe@parboxOptArgs{\eqe@bw}%
```

We insert `\eqe@align@hfill`, which is synchronized to the value of the `align` key to move the `\parbox` contents to left aligned, centered, or right aligned. `\eqe@align@hfill` will only be effective if `#3` is enclosed in a narrower box.

```
            {\eqe@align@hfill\ifKV@eqFillin@boxed\eq@fillintext\fi#3}}%
    \fi
```

If `\eqe@align@hfill` is still equal to `\relax`, give it the default value.

```
    \ifx\eqe@align@hfill\relax
        \def\eq@eqFillin@align{c}%
        \edef\eqe@align@hfill{\eqe@align@hfill@default}\fi
    \ifmmode\let\@eqmath\ensuremath\else\let\@eqmath\text\fi
```

We re-calculate the width of the formatted box

```
    \ifx\eq@fillinparbox\@empty
        \ifx\eqe@argii\@empty
```

If no parbox option and if the second argument is empty, we set width based on the natural width of #3

```
            \settowidth{\eqetmplengthb}{\@eqmath{\eq@eqFillin@fontsize
            \eq@fillin@format\ifKV@eqFillin@boxed\eq@fillintext\fi
            \eqe@argiii}}%
            \ifKV@eqFillin@boxed
                \setlength{\eqetmplengthb}{%
                \eqetmplengthb+2\fboxsep+2\fboxrule}%
            \fi
        \else
```

If #2 is nonempty, we use this value.

```
            \setlength{\eqetmplengthb}{#2}%
        \fi
    \else
```

parbox option with empty second argument, use \linewidth. for width

```
        \ifx\eqe@argii\@empty
            \setlength{\eqetmplengthb}{\linewidth}%
            \PackageWarning{eqexam}{Parameter \#2
                is empty with parbox option,\MessageBreak
                using \string\linewidth\space for width%
            }%
        \else
```

parbox option with second argument, use #2 for width

```
            \setlength{\eqetmplengthb}{#2}%
        \fi
    \fi
```

Return \ifsp to its default definition.

```
    \expandafter\let\expandafter\ifsp\csname ifsp@default\endcsname
```

Save the final calculated width as \eqe@bw.

```
    \edef\eqe@bw{\the\eqetmplengthb}%
```

Set the underline option, . . .

```
    \ifKV@eqFillin@underline\let\@fillinFmt\eq@fillin@ulcmd
    \else\let\@fillinFmt\relax\fi
```

however, if parbox is specified, we remove the underlining, if any.

```
    \ifx\eq@fillinparbox\@empty\else
        \ifKV@eqFillin@underline\let\@fillinFmt\relax
%        \ifx\@fillinFmt\underbar\let\@fillinFmt\relax
        \PackageInfo{eqexam}{Removing underline option, not permissible
        \MessageBreak with parbox option}%
    \fi\fi
```

Build the \fillin box. After the preliminaries, we create the requested answer field. We begin by building the answer field for the case of \ifeq@proofing is true (which occurs when the answerkey is used.

```
    \ifeq@proofing
        \ifKV@eqFillin@fitwidth
```

If the fitwidth option is taken, we measure the width of the box. Ignored when the parbox option is used.

```
            \settowidth{\eqetmplengthb}{\@eqmath{\eq@fillin@format
                \ifx\eq@fillinparbox\@empty\ifKV@eqFillin@boxed
                    \eq@fillintext\fi\fi\eqe@argiii}}%
```

If boxed, we increase the width by 2\fboxsep+2\fboxrule; when content is \boxed, the dimensions are reduced.

```
            \ifKV@eqFillin@boxed
                \setlength{\eqetmplengthb}{%
                \eqetmplengthb+2\fboxsep+2\fboxrule}%
            \fi
            \edef\eqe@bw{\the\eqetmplengthb}%
        \fi
```

We build the fill-in field for the case of boxed.

```
        \ifKV@eqFillin@boxed
            \ifmmode\let\@eqmath\ensuremath\else\let\@eqmath\text\fi
%            \mbox{\eq@fillin@format\ensuremath{\boxed{%
            \setbox\z@\hbox{\@eqmath{\eq@fillin@boxcmd{%
%                \eq@fillin@format\ensuremath{\eq@fillin@boxcmd{%
            \eq@eqFillin@boxsize
            \@fillinFmt{\eq@eqFillin@fontsize % dpsf0214
                \ifKV@eqFillin@boxed
                    \edef\eqe@bw{\eqe@fbox@corr{\eqe@bw}}%
                \fi
```

When the boxed option is taken, we adjust the width of the \makebox to get the desired width #2.

```
                \makebox[\eqe@bw][\eq@eqFillin@align]{\strut
                \@eqmath{\eq@fillin@format\color{\eq@fillin@color}%
                \ifx\eq@fillinparbox\@empty
                    \eq@fillintext\fi\eqe@argiii}}%
                }% end \@fillinFmt
            }}}% end \mbox
            \setlength{\@tempdima}{\ht0+\dp0}%
            \xdef\fillinTotalHeight{\the\@tempdima}%
            \mbox{\unhbox\z@}%
        \else
```

The content is not to be boxed.

```
            \ifmmode\let\@eqmath\ensuremath\else\let\@eqmath\relax\fi
            \setbox\z@\hbox{\eq@eqFillin@fontsize % dpsf0214
```

A value of -1sp signals that lift has not been specified, nor has autofill been specified. So we use the usual code for underlining.

```
            \ifdim\eq@fillin@lift=-1sp\relax
            \@fillinFmt{\makebox[\eqe@bw][\eq@eqFillin@align]{\strut
                \color{\eq@fillin@color}%
                \@eqmath{\eq@fillin@format\eqe@argiii}}}%
            \else
```

(2014/02/16) The autolift or lift keys are specified. If autofill, put contents in a box. Set \@tempdima to \dp2+\eq@fillin@addtoautolift, the latter normally has a value of 0pt unless specified on the option list.

```
            \ifKV@eqFillin@autolift
                \setbox2=\hbox{\@eqmath{\eq@fillin@format\eqe@argiii}}%
                \setlength{\@tempdima}{\dp2+\eq@fillin@addtoautolift}%
                \edef\eq@fillin@lift{\the\@tempdima}\fi
```

(2014/02/16) We lower by an amount of \eq@fillin@lift, but raise the context by that same amount.

```
            \raisebox{-\eq@fillin@lift}{\@fillinFmt{%
                \makebox[\eqe@bw][\eq@eqFillin@align]{\strut
                \color{\eq@fillin@color}%
                \raisebox{\eq@fillin@lift}{\@eqmath{\eq@fillin@format
                    \eqe@argiii}}}}}%
            \fi
            }\setlength{\@tempdima}{\ht0+\dp0}%
            \xdef\fillinTotalHeight{\the\@tempdima}%
            \mbox{\unhbox\z@}%
        \fi
    \else
```

We begin the case of not \ifeq@proofing, that is, the document author is not compiling with the answerkey option.

```
        \ifKV@eqFillin@boxed
%            \mbox{\eq@fillin@format\ensuremath{\boxed{%
%            \eq@fillin@format\ensuremath{\eq@fillin@boxcmd{%
            \setbox\z@\hbox{\eq@eqFillin@fontsize % dpsf0214
              \@eqmath{\eq@fillin@format
                \eq@fillin@boxcmd{\eq@eqFillin@boxsize\ifeq@nosolutions
                \rlap{\@eqmath{\eq@eqFillin@fontsize % dpsf0214
                  \eq@fillintext}}\fi
            \ifvspacewithsolns % dpsf0214 \eq@eqFillin@fontsize
              \rlap{\@eqmath{\eq@eqFillin@fontsize\eq@fillintext}}\fi
            \@fillinFmt{%
```

We do a similar thing if proofing is not active (nosolutions is taken).

```
                \makebox[\eqe@fbox@corr{\eqe@bw}]{%
                    \ifKV@eqFillin@enclosesoln\phantom
                    {\setlength\eqetmplengthb{\eqe@bw-2\fboxsep}%
                        \edef\eqe@bw{\the\eqetmplengthb}%
                        \@eqmath{\eqe@argiii}}\else
                    \strut\hfill\fi
                }%
            }%end \@fillinFmt
            }}}% end \hbox
            \setlength{\@tempdima}{\ht0+\dp0}%
            \xdef\fillinTotalHeight{\the\@tempdima}%
            \mbox{\unhbox\z@}%
        \else
```

This is the case where the field is *not* boxed.

```
            \setbox\z@\hbox{\eq@eqFillin@fontsize % dpsf0214
            \ifdim\eq@fillin@lift=-1sp\relax
                \@fillinFmt{\makebox[\eqe@bw]{%
%                    \phantom{\@eqmath{\eqe@argiii}}
                    \strut\hfil}}%
            \else
                \ifKV@eqFillin@autolift
                    \setbox2=\hbox{\@eqmath{\eq@fillin@format
                        \eqe@argiii}}%
                    \setlength{\@tempdima}%
                        {\dp2+\eq@fillin@addtoautolift}%
                    \edef\eq@fillin@lift{\the\@tempdima}\fi
                \raisebox{-\eq@fillin@lift}{\@fillinFmt{%
                    \makebox[\eqe@bw]{\raisebox{\eq@fillin@lift}
                        {\phantom{\@eqmath{\eq@fillin@format
                            \eqe@argiii}}\strut\hfil}}}}%
            \fi
            }\setlength{\@tempdima}{\ht0+\dp0}%
            \xdef\fillinTotalHeight{\the\@tempdima}%
            \mbox{\unhbox\z@}%
        \fi
```

Online Code. If the `quiz` environment is defined, and the user has asked for `online` option we build a text field.

```
        \@ifundefined{@quiz}{}{%
            \if\eq@online\eqe@YES\relax
                \ifeq@nosolutions
                    \ifeq@solutionsafter\else
                        \ifx\eq@insertverticalspace\eqe@YES\relax
```

OK, we get this far if we choose `online` (or higher) and if `nosolutions` (which includes the `vspacewithsolns` option). We require `\eq@insertverticalspace` to be `y`. This last value is the default (`\SpaceToWork`).

```
                            \stepcounter{@cntfillin}%
                            \edef\fieldName{%
                                \if\probstar*eqexam.\curr@quiz.fillin.%
                                    \theeqquestionnoi.part\thepartno.%
                                    fi\the@cntfillin%
                                \else
                                    eqexam.\curr@quiz.fillin.%
                                    \theeqquestionnoi.fi\the@cntfillin%
                                \fi
                            }\ifx\eq@fillinparbox\@empty
```

If the `parbox` option is not taken, we build a text field with height `11bp`

```
                            \raisebox{-1bp}{\makebox[0pt][r]{%
                            \textField[\BC{}\presets{\eqe@optsFillIn}]
                            {\fieldName}{#2}{\fillinTotalHeight}}}\else
```

If the user has taken the parbox option, then the text field becomes a multiline field, with height equal to the requested height.

```
                            \setlength{\@tempdima}%
                                {\eqe@opts@argii+2\fboxrule+2\fboxsep}%
                            \raisebox{-1bp}{\makebox[0pt][r]{%
                            \textField[\BC{}\presets{\eqe@optsFillIn}
                            \Ff{\FfMultiline}]{\fieldName}{#2}
                            {\fillinTotalHeight}}}\fi
                        \fi
                    \fi
                \fi
            \fi
        }%
```

(2018/02/02) Removed \space\ignorespaces from \fillin

```
    \fi\endgroup}
```

\TF The \TF command depends on \fillin, so we make the appropriate changes.

```
\renewcommand\TF[2][\defaultTFwidth]{%
    \def\eqe@next{\fillin[underline]{#1}{#2}}%
    \ifdim\eq@extralabelsep=0pt\relax\else
        \if\probstar*\relax\if\exerwparts@cols0
            \def\eqe@next{\makebox[0pt][r]{%
                \fillin[underline]{#1}{#2}}\ignorespaces}%
    \fi\fi\fi\eqe@next
}
```

This marks the end of the eqexam package. dps

```
⟨/package⟩
```

18 Index

Numbers written in italic refer to the page where the corresponding entry is described; numbers underlined refer to the code line of the definition; numbers in roman refer to the code lines where the entry is used.

19 Change History

1.6a (2006/01/22)

- `\selectVersion`: added `\selectVersion` command, also the command was fixed `\forVersion` so that it can be changed within the document, added switches to control new selection. 28

1.6b (2006/04/03)

- `\forVersion`: Changed the alphabet environments `A`, `B`, etc. due to a conflict with `\S`, the control sequence for the `amsmath` package for a section: `\S` expands to §. Names changed to `verA`, `verB`, etc. 26

1.6b (2006/3/12)

- `\popProblem`: Added `\pushProblem` and `\popProblem` to grant the ability to interrupt a parts environment for, for example, a multicolumn environment. 91

1.6c (2006/05/02)

- `\forVersion`: At the end of the package, set the initial value of `\selectVersion` to `\selectVersion{26}` and set the initial value of `\forVersion` to `\forVersion{A}`. 28

2.0a (2010/05/06)

- `\rheadeqe`: Changed the definitions of `\lhead`, `\chead`, and `\rhead` so they don't clash with the fancyhdr package. If fancyhdr is not loaded at the time eqexam is loaded, we `\let` the old names to the new names. Therefore, when fancyhdr is loaded first use the new definitions. 44

4.4a (2015/06/01)

- `problem`: Added the hook `\topofprobstarhook` 83

(2015/06/01)

- `problem*`: Added the hook `\topofprobstarhook` 86

(2015/06/02)

- `problem*`: Added `\@marktotalvalue` which seems to solve a problem with `multicols`. Sometimes the totals are not correct, they do not 'migrate' out as they should. 88

(2015/07/21)

- General: add `\let\CT@arc@\relax` 13

v1.6e (2006/05/07)

- General: Added a custom option feature. Just before the options are processed, the tex compiler looks for the file `eqecus.opt`. This file should contain one or more custom options. 12
- `\eqEndExamTotalColor`: Added easy user access to various colors, `\proofingsymbolColor`, `\instructionsColor`, `\eqCommentsColor`, `\authorColor`, `\titleColor`, `\universityColor` and `\subjectColor` 21

v1.6f (2006/10/24)

- `\forVersion`: Fixed a bug in the `\eqe@initializeMultiVersions` command, made sure that any already defined comment environments are set to relax. 26

v1.6g (2006/11/29)

- `\itemPTsFormated`: Added `\itemPTsTxt` and `\itemPTsFormated` to work with `\PTs`. Also added a `*` option, to `\PTs`, in this case the points are not typeset. 83

v1.6h (2007/01/24)

- `\forleadinitem`: Added `\forproblem`, `\foritem`, `\aNewPage` 94
- `\qNewPage`: Added `\aNewPage` and `\qNewPage` 96

v1.6i (2007/09/18)

- `\altTitle`: Added the command `\altTitle` as an alternate title for the exam document. This alternate title appear centered under the title of the document. 46

v1.7a (2007/12/10)

- `\separationrule`: Added `\separationrule` so user's can redesign the separation rule that is created between two parts of an exam. 73

v1.7b (2007/21/07)

- General: Added a `solutionsonly` option 9
- `\encloseProblemsWith`: Added `\encloseProblemsWith` to support the `solutionsonly` option 21

v1.7c (2008/08/21)

- General: Added the `showgrayletters` option to eqexam (ported from exerquiz) 10

v1.8 (2008/11/02)

- `\placeMarkerHere`: Added a set of commands `\placeMarkerHere`, `\calcFromMarkers`, and `\markerTotalFmt` to enable the calculation of totals of segments of the exam. 59

v1.9a (2009/28/09)

- `\coverpageTitleFmt`: Added these various Fmt commands for coverpage. 47

v1.9b (2009/09/29)

- `\placeCoverPageLogo`: Added `\placeCoverPageLogo` to insert a logo on the cover page. 47

v1.9f (2009/10/06)

- `\endshortTitleText`: Modified `\longTitleText`, `\shortTitleText` to have an optional argument (A–Z;a–z). You can select a particular title from a list of titles. If no optional argument is passed, then the title determined by `\forVersion` is used. 23

v2.0 (2010/03/05)

- General: Added exambuilder.cfg for use by AeB Exam Builder, to pass the values of the options max and rendition to eqexam. 12
  - Switched over to `xkeyval`, added max and rendition to be consistent with the renditions package, though we don't use the rendition package itself. `eqexam` has a more extensive renditions system already. Introduced this mostly for use AeB Exam Builder. 8

v2.0c (2011/01/11)

- `problem*`: Changed `\@next` to `\eqe@next`. There was conflict in the use of this command with one of the float environments. When user used the table environment inside the problem* environment, the compiled stopped because `\@next` was overwritten. 84

v2.0d (2011/03/04)

- General: Added the `vspacewithsolns` option . . . 7

v2.0e (2011/03/07)

- `\useFillerLines`: Added the feature of filling the vertical space with ruled lines of different types. This feature is available for paper options and for `nosolutions` and `vspacewithsolution` options. 97

v2.0h (2011/04/14)

- `\fillin`: Modified the calculation of the width of `\fillin`, the width of enclosing box now equals the requested width 156

v2.0i (2011/04/17)

- General: Added the switch `\ifdisplayworkarea` to better control when the work area is to be displayed. 10

v2.0j (2011/04/19)

- `enclosesoln`: added the `enclosesoln` key to `eqFillin` family. 148

v2.0k (2011/04/29)

- `\vspacewithkeyOff`: Added user interface to the switch `\ifkeepdeclaredvspacing`, which is defined in `eqexam.def`/`exerquiz`. 20

v2.0l (2011/05/05)

- `\firstPageOfExam`: Returns the page number of the beginning of the exam with a given name. 43
- `\lastPageOfExam`: Returns the page number of the end of the exam with a given name. . . . 43

v2.0n (2011/05/13)

- `\promoteNewPage`: A simple variation on `\makeRoomForProb` designed for user use. . . 80
- `problem`: Added `*<num>` to signal in-line display of points. 82

v3.0f (2011/08/13)

- `\ftbInputBookAux`: Added `\ftbInputBookAux` to support solution manual 145
- `\ftbInputSolnFiles`: Added `\ftbInputSolnFiles` to support solution manual 147

v3.0g (2011/08/15)

- `\hangSolWPrtsFmt`: Use this to use "hanging indentation" for the parts for problems with parts in the solutions file. 128

v3.0h (2011/08/17)

- General: 2011/08/17 v3.0h Added the `vspacewithsolns` option 7

v3.0i (2011/08/18)

- `fitwidth`: added the `fitwidth` key to `eqFillin` family. 149

v3.0l (2011/08/22)

- `\annotContStr`: Defined `\eqe@insertContAnnot` and related commands 95

v3.0n (2011/09/18)

- `probset`: Added an `\edef` in case `\thesection` does not get expanded early enough to display correctly in the margins. 144

v3.0o (2011/09/20)

- `\cngMargHeadColorTo`: Added `\cngMargHeadColorTo` and `\resetMargHeadColor` to make it easier to change the color of the header globally, or just once. 131

v3.0p (2011/09/22)

- General: Added `\NoSolutions` to be executed in preamble, needed with the fortextbook package. 20
  - Added the wrapper package fortextbook. . . . 147

v3.0q (2011/14/22)

- `\eGrpANS`: Added `\bGrpANS` and `\eGrpANS` 121

v3.0s (2012/01/01)

- `\eqexammargin`: Moved a copy of `\eqe@spannerSoln` out of the `ftbsty` to the package section, its needed here as well. . . . 18

v3.0t (2012/25/01)

General: Added four more CFG files are the request of a user. 5

v3.0u (2012/09/03)

General: Added the cfg option for inputting a custom config file. 5

v3.0v (2012/03/14)

\textbookOpts: Include \tbMakeFinalCalcs at end of \textbookOpts 120

v3.0w (2012/03/27)

parbox: Added parbox 149

hiddenbox: Added hiddenbox options 149

v3.0x (2012/04/03)

hiddenbox: Added \eqe@align@hfill to align property. used to set position of content when parbox is used. 149

v3.0y (2012/04/20)

\altTitle: Moved \EQEcalculateAllTotals from the bottom to the top of \eqemaketitle. In case the author wants the grand total of the exam in the title, we need to make all calculations before \maketitledesign. 46

v3.1 (2012/05/16)

\calcQsBtwnMarkers: Added \calcQsBtwnMarkers 59

v3.1a (2012/05/21)

problem*: Using a more robust method of detecting the presence of \auto 86

v3.1b (2012/06/18)

General: Added \eqe@auto@chk@drivers 7

New requirement ifpdf 4

v3.1c (2012/07/30)

ulcmd: Added ulcmd to \fillin 150

boxcmd: Added boxcmd to \fillin 150

v3.1d (2012/09/29)

General: Added \ifeqexamCFG and cseqexamCFG; these are used by ATB to attached the configuration file. 5

v3.1e (2012/11/10)

General: Changed default for \proofingsymbol . 41

\makeAnsEnvForSolnsAtEnd: Removed hard-wired \proofingsymbol 13

problem*: Removed hard-wired \proofingsymbol 86

v3.1f (2012/11/27)

\leadinitem: Created \leadinitem command for a lead in question for the parts* environment. 88

v3.2 (2012/12/10)

splitsolution: Rewrote splitsolution and related code to change the syntax. 62

v3.2c (2013/02/23)

General: Added \selectedMC at end of package . 13

v3.3 (2013/04/07)

lsols: exerquiz changed \eqEXt to two variables, so we make the same change here. 120

v3.3a (2013/05/03)

\leadinitem: Changed name of \eq@writeexheader to \eqExerSolnHeader 90

v3.5a (2014/12/19)

General: Made usexkeys as default 8

v3.6a (2015/01/31)

\forleadinitem: Added \eqe@fpmrk to add to the uniqueness of \theHeqquestionnoi. . . . 94

v3.7 (2015/02/11)

\selectVersion: Worked on the logic 28

v3.7a (2015/02/26)

\eqe@tb@shipout: Added a new Boolean switch \iftb@shipoutPermitted to turn off the shipout. Two convenience commands in support are also added, \turnOffFTBShipout and \turnOffFTBShipout. 134

v3.8 (2015/03/17)

\leadinitem: Spacing is not correct with standard itemize and enumerate environments, this is a fix. 90

v4.1 (2015/04/23)

General: Moved this segment of code to \manualcalcparts, it is taken from the main problem* env. 92

v4.4 (2015/05/24)

exam: Finish with a \vskip to clear the last item or problem. 75

v4.5 (2015/11/10)

\fillin: Added \setfillinDefaults 152

v4.5a (2015/11/10)

hiddenbox: Added some logic to the underline key 149

v4.6 (2016/01/18)

\eqe@tb@shipout: Removed \vfil between unboxing, this allows correct breaking of margin content across pages. 135

v4.6a (2016/06/04)

General: Added \ifequsecolor to easily distinguish between forpaper and forcolorpaper options. 9

v4.6b (2016/10/02)

- `\showAllAnsAtEnd`: Include `\eqTopOfSolnPage` from exerquiz to support copying question to solution page. 13

v4.6c (2017/01/28)

- `ulcmd`: added custom underline option 151

v4.7 (2017/01/04)

- `\forleadinitem`: Modified `\foritem@cont` to allow for numbering parts 94

v4.9 (2017/03/16)

- General: Updated `eqexam` to reflect changes in `eqexam.def`. 38

v4.9a (2017/04/05)

- `\leadinitem`: Added a test for solutions after within lead-in item 90

v4.9b (2017/10/10)

- `\eqEndExamTotalColor`: Provides color for check boxes when a PDF-related option is not taken 22

v4.9c (2017/11/03)

- `\fillTypeGrid`: Added `\fillTypeGrid` 98

v5.0 (2017/11/08)

- General: Added `\setFillLinesFmt` 102
  - Added hooks `\priorPNPAction` and `\postPNPAction` to `\makeRoomForProb` 79
  - Added support for `\fillTypeGrid`, added keys numbers, numbersep, and color as well 99
  - Added switch `\ifgridpgbrk` 103
  - Added vertical rules 99
- `workarea`: Added `\workareaVadj` 61
  - Changes in `workarea` to support `\fillTypeGrid` 62
  - Within `workarea`, adjust `\leftskip` when the problem is a lead-in type 62

v5.0a (2017/11/14)

- General: Placed `\eqe@makeVgrid` in a group, `\count0` was leaking out. 108

v5.0c (2017/11/18)

- General: Added the outlineonly, bgonly, and bgcolor keys to the eqefileLines family. 101

v5.0d (2017/11/19)

- General: Added a custom background feature 108

v5.0g (2017/12/05)

- General: Added filler lines with `nosolutions` option 116
  - Added option `flextended`. 7
- `priorworkarea`: Added the `priorworkarea` environment 116

v5.0o (2017/12/26)

- General: Added `\chngToNoSolns` 20

v5.1.1 (2018/12/05)

- General: Added `\eqprior` 14

v5.1.2 (2018/12/05)

- General: Use version 3.2 of comment style under the name of aeb-comment 16

v5.1.3 (2018/12/13)

- General: `\ifdisplayworkarea` conditionally defined; defined `\displayworkareaOn` and `\displayworkareaOff`. Consistent with exerquiz. 10
- `workarea`: added solutions-after condition to displaying `workarea` 62
- `\postExamSolnHead`: Write to solution file if not solutions-only 42

v5.1.4 (2019/01/31)

- General: `\let \comment` and `\endcomment` to `\relax` we use the definition of the comment environment given in verbatim. 16

v5.1.5 (2019/10/28)

- General: Define `\eq@Radio@driver` in case eqexam.def/exerquiz still use `\eq@RadioCheck@driver` 36
  - Insert `\bWebCustomize` and `\eWebCustomize` to conform with web dated 2019/10/23. 12
  - Replace `\eq@RadioCheck@driver` by `\eq@Radio@driver` 36

v5.1.6 (2019/10/29)

- `\fillin`: Allow the third argument of `\fillineol*` to have verbatim text as well. 153

v5.1.7 (2019/12/17)

- General: Added `\autoInsSolns` 141

v5.1.8 (2020/01/06)

- General: Define `\aebc@end` as a work around for aeb-comment 16

v5.1a (2018/01/21)

- `problem`: added `\eqe@p@gobnxtp@r` 82
- `problem*`: added `\eqe@gobnxtpar` 85

v5.1b (2018/01/24)

- `\vspaceFillerLines`: Added 105

v5.1c (2018/01/31)

- General: Added additional lines to get a better break point 112

v5.1d (2018/02/02)

- `\fillin`: removed `\space\ignorespaces` from `\fillin` 159
- `exam`: Added `\RecordThisExamOff` 74

v5.1e (2018/02/09)

- `problem`: Support for h and H argument 82, 83 v5.1f (2018/02/09)

`problem`: modify problem env to analyze two optional args 82

v5.1g (2018/02/14)

`hiddenbox`: Added `fontsize` and `fboxsep` keys 149

v5.1h (2018/02/18)

`\fillin`: Added `\mtoeol` 152

Added test, 0pt same as empty width 153

v5.1i (2018/02/19)

`\fillin`: Changed name to `\fillineol` and changed order of arguments 152

v5.1j (2018/04/15)

`problem`: Change comparison to `\ifx`, `\if` give incorrect result when points are of the form 11, 22, 33, etc. 82