AcroTeX Software Development Team

The eqexam Package
part of the
AcroTeX eDucation Bundle

D. P. Story

Version 5.1.4, 2019/01/31
June 3, 2019

http://www.acrotex.net
dpstory@acrotex.net

Table of Contents

1 Forward 4

2 Introduction 4

3 Required and Optional Packages 6

4 Installing eqexam 6

5 Demonstration files 7

6 Page Layout Considerations 7

7 Building an Exam 8
- 7.1 The Preamble . . . 9
- 7.2 The exam Environment . . . 9
- 7.3 The problem and problem* Environments . . . 11
  - problem . . . 11
  - problem* . . . 14
    - \leadinitem . . . 16
    - \tableadin . . . 16
  - Page Breaking . . . 17
- 7.4 Fill-in Questions . . . 18
  - Short Fill-in Questions . . . 18
  - True/False Questions . . . 18
  - Long Fill-in Questions . . . 19
- 7.5 Multiple Choice . . . 20
  - Using \bChoices/\eChoices . . . 20
  - \sqForms versus \sqLinks . . . 21
  - Using Circles for Multiple Choice . . . 21
  - Using \proofingsymbol and friends . . . 22
- 7.6 Multiple Selection . . . 23
- 7.7 Randomizing Choices . . . 24
- 7.8 Labeling Choices . . . 26
- 7.9 Gizmos and Gadgets . . . 27
  - The workarea Environment . . . 27
  - The \placeAtxy Command . . . 28
  - The splitsolution Environment . . . 29

8 eqexam Options 30
- 8.1 Configuration Files . . . 36
- 8.2 The solutionsonly Option . . . 37
- 8.3 The vspacewithsolns Option . . . 37
- 8.4 The fortextbook Option and fortextbook Package . . . 37

9 The `online` and `email` Options **38**
9.1 The `email` Option . . . 38
- Installing `eqAttach.asp` . . . 41
- Setting up and Modifying the Script . . . 41
- Some Options . . . 42
- References . . . 42

10 Bells, Whistles and other Customizations **43**
10.1 Customizations . . . 43
- Course Info Commands . . . 43
- Changing the Title and Cover Page . . . 44
- Changing the Running Headers . . . 46
- Changing the Running Footers . . . 47
- Exam Strings . . . 48
- Localization of Strings . . . 49
- Customization of Color . . . 50

10.2 Creating Multiple Versions of Exam . . . 51
- The Original Version Scheme . . . 51
- New Version Control . . . 53

10.3 The Point and Totals Boxes . . . 56
10.4 The `eqComments` Environment . . . 57
10.5 The `\OnBackOfPage` Command . . . 58
10.6 The `\pushProblem` and `\popProblem` Commands . . . 59
10.7 `\qNewPage`, `\aNewPage`, and `\promoteNewPage` . . . 60
10.8 Support for Solution Sets from a Text . . . 60
10.9 Referencing Multiple Choice Questions . . . 61
10.10 Displaying Points between two Markers . . . 62
10.11 Extending the `\fillin` Command . . . 63
- `\fillineol`: Fill-in to end-of-line . . . 69

10.12 Vertical Space Fill Types . . . 70
- On the `flextended` option . . . 72
- Breaking across pages . . . 74
- Annotating a continuing problem with `\useFillerLines` . . . 74

10.13 Keep vertical space with `answerkey` . . . 75
10.14 Annotating a Continuing Problem with Parts . . . 76
10.15 The Exam Record . . . 78
10.16 Calculate problem range between two markers . . . 79

References **80**

1. Forward

For the past several years (this year is 2019), I've been writing a book titled,

AcroTEX eDucation System Tools: LATEX for interactive PDF documents.

The book [1] covers AeB, which includes the eforms package, and AeB Pro in *great detail* and includes many examples to illustrate concepts and techniques. Numerous new examples are available on the CD-ROM that accompanies the book.

During the time of the writing, each of the packages covered was examined, bugs were fixed, and many new and major features were created. Any new features developed in the course of writing the book are documented in the book; however, they are *not included in this documentation*. You can either buy the yet-to-be-submitted book sometime in the future, or discover the features by studying the DTX documentation of the program files. Sorry, it took me three years to write the book, I don't want to spend another year on this documentation. :-{}

Dr. D. P. Story

June 3, 2019

2. Introduction

In my classroom work at The University of Akron, I've been using a personal LATEX package, which is called eqexam, for creating my in-class tests, quizzes, homework assignments, and review documents (pre-tests/sample tests). In recent weeks—at the end of the Fall Semester, 2004, and prior to the Spring Semester, 2005, I have filled the mundane and boring days with work on eqexam, fixing and enhancing it quite a bit.

The eqexam package is a stand-alone for LATEX, but is also tightly integrated with the AcroTEX eDucation Bundle. eqexam will be distributed by itself, as well as a part of the AcroTEX Bundle. The integration with the AcroTEX Bundle gives it many of the online features that users of the Bundle are familiar with.

(Version 3.0 or later) The method of formatting an eqexam document has changed, each problem/problem* environment is now in a list environment, the eqequestions environment. This environment is not normally used by the document author, but its parameters may be redefined. The purpose of this reformatting, is to open up eqexam for use by other packages. Textbook authors can now, I hope, easily integrate eqexam into the custom book format being used.

Let's have an overview of the package, with suggestions for possible uses.

1. The first, and most obvious application of this package is to create a pExam or a pQuiz. (Here, the 'p' prefix refers to paper or pulp; thus, we can use eqexam to write paper Exams and/or pulp Quizzes). You can write the questions and the solutions, and publish (i.e., print the document on a printer) the exam/quiz with no solutions—ready to be taken in class—, or LATEX the source document with solutions listed after each question to create an answer key, for your personal use, or for the use by the class.

2. So much for pulp. Now on to 'e' (for electronic publication). In some of my classes, I put sample questions (review tests) on the web as PDF documents. In this case, you can create a PDF document without the solutions, and give the class time to solve the problems; then publish the document (in PDF on the web) with solutions. The solutions can appear immediately after the questions, or can be accumulated at the end of the document.

   In the case where the solutions are at the end of the document, you can add links from the question to the solution.

   Documents can be published with color (to enhance the on screen appearance) or can be published in black and white, meant to be printed. Or, you can do both: a screen version and a paper version.

3. By invoking the `online` option, the white space left for hand-written answers to the questions become Acroform multi-line text fields, multiple choice questions become radio buttons, and fill-in questions also become text fields. The student can bring up the exam, and take it at a computer (in a CBT[1] lab). After the student is finished, he/she can print out the exam, and submit it to the instructor for traditional grading.

4. Now, here is an exciting feature of the `eqexam` package, that of email submittal! This feature is not too useful for technical fields (i.e., mathematics related fields) that require students to enter special symbols, but for some academic disciplines (English, History, Sociology, Politics and Government, etc.) this feature could be quite exciting.[2]

   When you take the `email` option of `eqexam`, as with the `online` option, the white space left for hand-written answers to the questions become Acroform multi-line text fields, multiple choice questions become radio buttons, and fill-in questions also become text fields. Additionally, a button is automatically provided to submit by email the results of the test to the instructor. The results arrive at the instructor's mailer as an FDF attachment. The instructor can open the FDF and view in the originating PDF the responses given by the student.

   The instructor can print out the document and grade in a traditional way, or if the instructor has **Acrobat Pro** or **Standard**, the instructor can use mark-up annotations within the PDF, save a copy of the students test to a class folder, and email a copy of the students exam, marked up with grade.[3]

   If the exam is given for credit, it can be taken in a secure lab.

5. Perhaps a more reasonable application of this email submission feature of `eqexam` is the building and publication of surveys and questionnaires! Perhaps teacher

[1] Computer Based Testing.

[2] Of course, I am addressing now the some six people worldwide in these fields that use LaTeX and PDF! For you six, this feature is for you!

[3] Seems doubtful that anyone at this time has the expertise to do this! But it's available if anyone ever wants it.

evaluations! The environments of eqexam can be easily used to write surveys and questionnaires to solicit the opinion of a target population. Responses are emailed to the designated person, who can summarize them.

By the way, speaking of summarizing results, a new feature of **Acrobat Pro 7.0**, allows you to take a folder of FDF files, such as the ones created by email submission, and extract all form fields and place results to a comma-delimited file (.csv). This comma-delimited file can be opened by a spreadsheet program and manipulated. Cool.

6. (08/05/11) Version 3.0 of eqexam has a major option, fortextbook,[4] designed to support (U.S.) textbook authors. Documentation for this option is found in the doc/fortextbook folder. See also the series of blogs at the AcroTeX Blog.

3. Required and Optional Packages

The following packages that are not part of the normal LaTeX distribution are *required*:

1. calc: Used for calculation of the position of the marginal points.
2. pifont: Used when the proofread option is used to indicate the correct answers to multiple choice questions.
3. aeb-comment: Used to have optional content, useful for developing exams for multiple sections of the same class.[5]
4. multicol: Used to create questions in multi-column mode.
5. verbatim: Used to write solutions to the hard drive.

Additionally, the following packages may be used depending on the options chosen:

1. web: Used when the pdf, links, online or the email option is taken.
2. exerquiz: Used when the links, online or the email option is taken.

Of course, web and exerquiz, in turn, input a whole plethora of packages. Consult the documentation for the AcroTeX eDucation Bundle.

4. Installing eqexam

Create a folder in your latex search path named eqexam and place the package files eqexam.dtx, eqexam.ins, eqexam.def and any .cfg files. (If you have an acrotex folder, you can place the files there as well.)

Next, latex eqexam.ins to create eqexam.sty and eqalone.def. The other files (*.tex and *.pdf) can be placed anywhere.

[4] The fortextbook option is briefly described on page 35.

[5] aeb-comment is an older version of the comment package by Victor Eijkhout; it is distributed with the acrotex package.

The eqexam is a stand alone package that is tightly integrated with the AcroTEX Bundle. The file eqexam.def comes from the AcroTEX Bundle to provide the necessary support for many of the commands and environments defined in eqexam. The file eqalone.def are miscellaneous definitions that are needed for the stand-alone version. When you choose one of the options links, online or email, then Exerquiz is included in the package files. When you use one of these options you will need the most recent version of the AcroTEX eDucation Bundle, the one published concurrently with this package.

5. Demonstration files

- The original eqexam demonstration files are posted on the AcroTEX Blog. Throughout the manual, individual files are references and a link is provided to that resource. The source file is attached to all PDFs on the AcroTEX Blog website.
- Additional demonstration files developed after the original set are also available from the AcroTEX Blog. See the articles tagged as *eqexam-package*.

6. Page Layout Considerations

With Version 3.0, you can design your own page layout scheme, perhaps to conform to a book style. The following are some basics on formatting for eqexam.

The following two commands appear in eqexam, the first sets some basic page parameters.

```
\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}
}
```

The second command calculates values for \textwidth and \textheight based on the the settings of the first command.

```
\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}
}
```

When the package option `usecustomdesign` *is not taken,* then the two commands `\eqeSetExamPageParams` and `\eqExamPageLayout` are executed immediately after the above definitions. These are the original parameters used by `eqexam`, designed to yield a maximum text body in which to typeset an exam. The margins are set at 1 inch, the `\topmargin` is raised up, all to maximize space.

Now, if the package option `usecustomdesign` is specified, the commands `\eqeSetExamPageParams` and `\eqExamPageLayout` are *not executed*, the package designer can either do a `\renewcommand` for these two commands in the preamble with custom values inserted (and execute `\eqeSetExamPageParams` and `\eqExamPageLayout`), or the designer may use another package to set the page layout parameters (or take the default of the class being used). In the latter case, neither `\eqeSetExamPageParams` nor `\eqExamPageLayout` should be executed.

The following commands directly effect how the problems are displayed within an `eqexam` environment.

```
\eqexammargin{\normalsize\normalfont\bfseries00.\ }
```

The command `\eqexammargin` is a convenient way of specifying the `\labelwidth` as set by the `eqequestions` environment (see below). The command uses `\settowidth` to set the `\eqemargin` length. The `\eqemargin` may also be set directly with `\setlength`. `\eqexammargin` can be executed anytime between exam environments (or even between problems, though this is not a intuitive option). Normally it is executed once for the entire document; but may be executed multiple times to change margins.

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

This command sets the width of the boxes that appear in the right margin when one of more of the options `pointsonright`, `pointsonboth`, `totalsonleft`, `totalsonright`, are used. These boxes are used for exams, and not relevant for problem sets of textbooks. Normally, this parameter is not redefined.

```
\newenvironment{eqequestions}{%
    \begin{list}{}{%
        \setlength{\labelwidth}{\eqemargin}%
        \setlength{\topsep}{3pt}\setlength{\parsep}{0pt}%
        \setlength{\itemindent}{0pt}\setlength{\itemsep}{3pt}%
        \setlength{\leftmargin}{\labelwidth}%
        \settowidth{\labelsep}{\ }%
    }\item\relax}{\end{list}}
```

This environment is opened at the beginning of a `problem` (`problem*`), and closed at the end of these environments.

7. Building an Exam

In this section, we outline the steps to create an exam using the `eqexam` package. Consult the sample exams for additional examples.

7.1. The Preamble

Of course, we begin with the standard article class, and the eqexam package:

```
\documentclass{article}
\usepackage[options]{eqexam}
```

The *options* are discussed in section 8. Next comes a exam identification information:

```
\title[T1]{Test 1}
\subject[C1]{Calculus I}
\author{D. P. Story}
\keywords{Calculus I, Section 004}
\university{%
      THE UNIVERSITY OF AKRON\\
    Mathematics and Computer Science
}
\date{\thisterm, \the\year}
\duedate{October 17, 2005}
```

The \title, \subject, \author and \date are the same as is used in the web package. These are used by the standard LaTeX macro to create the heading line of the first page of the exam, and are used in the running headers.

The \title, \subject have optional first arguments, where you can list a shorted version of the title or the subject. The shortened versions, if present, are used in the running headers.

The \keywords is used when you publish your exam in PDF and you use the pdf option (or online, links, email). The value of the argument of \keywords appears in the keywords field of the document info dialog.

When you take the coverpage option, the value of \university is used, along with some of the others on the cover page.

I've also defined a keyword of \duedate, this might be useful when using eqexam to create homework assignments with a due date, or just to record the date of the exam. The argument of \duedate fills the text macro \theduedate. So that if you say \duedate{05/31/06}, the macro \theduedate will expand to '05/31/06'.

Beginning with version 1.6, \thisterm is defined. The academic year of many American universities are divided into semesters (or terms); Fall, Spring, and Summer. The command \thisterm takes the current date and determines if it is the Fall, Spring or Summer Semester. For example, if the date of the compile is October 17, 2005, then \thisterm, \the\year expands to 'Fall, 2005'. This command is useful with the \date command.

The command \thisterm can be redefined to conform to the terms of the document author's university. See the definition in eqexam.dtx, copy and modify it.

7.2. The exam Environment

An exam is contained within the exam environment.

One of the things that I do in my courses, especially for the final exam, is to have a two-part exam. Typically, the first part is worth 100 points and covers the new material

not already tested; the second part is usually a 50 point review. I grade these two parts separately and record them separately. Therefore, an eqexam test may contain one or more exam environments.[6]

After the preamble, we then say

```
\begin{document}

\maketitle

\begin{exam}[Part I.]{Part1}

\begin{instructions}[Part I.]
Solve each of the problems without error. If you make an error,
points will be subtracted from your total score.
\end{instructions}
...
...
...
\end{exam}

\begin{exam}[Part II.]{Part2}

\begin{instructions}[Part II.]
The following is a short review of previously mastered material.
\end{instructions}
...
...
...
\end{exam}
\end{document}
```

After the \begin{document} and standard \maketitle, we begin an exam by opening an exam environment.

```
\begin{exam}[friendly_name]{exam_name}
...
\end{exam}
```

This environment has two arguments: the first optional, the second required. The first argument is a user friendly name (used when the solutions are listed at the end of the document when there are multiple exam environments); the second required argument is the name of the of the exam, Part1 or Part2, for example. This argument is used to build the names of the PDF Acroform field names. This argument should consist of letters and numbers only. You can use the command \autoExamName for the *exam_name*; this command will name each exam environment exam1, exam2, exam3, etc.

Following the opening of the exam, typically, the instructor would have some instructions, this is the purpose of the instructions environment. It has one optional argument, heading text for the instructions; if this optional parameter is not provided, then

[6] Remember, this was originally a personal package, meant to suit my own needs.

the default word is used, the default word is determined by `\defaultInstructions`, its default definition is

```
\defaultInstructions{Instructions.}
```

Following this label, the total number of points for this part is inserted, unless the `nosummarytotals` option is taken.

- The optional argument of the `instructions` environment has a color associated with it, and is visible when you compile the document with the `forcolorpaper` option. This color can be set by the command `\instructionsColor`; this command takes a single argument, a named color:

```
\instructionsColor{blue}
```

The above is the default definition.

At this point, you would insert your questions. Following the listing of all the questions (and optionally, their solutions), you finish up by closing out the `exam` environment.

Repeat, if additional parts to the exam are desired. Finally, finish off the document with `\end{document}`.

- You must `latex` your document *three times* to be sure all points have been properly calculated.

7.3. The `problem` and `problem*` Environments

All questions are posed using the `problem` and `problem*` environments. The former is for a single question, the latter is for a question with multiple parts.

- **`problem`**

The `problem` encloses a single question; the question itself may contain special constructs such as one or more fill-in the blanks.

The syntax for `problem` is

```
\begin{problem}[num|*num|empty][h|H]
⟨Statement of question, which may contain special constructs⟩
...
...
\begin{solution}[vspace,nLines=n]
...
...
\end{solution}
\end{problem}
```

The environment takes two optional arguments. The first argument *num* is the number of points for this problem, for example, if we want to have a 5 point question, we would begin the environment like so, `\begin{problem}[5]`; on the other hand, if we say `\begin{problem}`, the problem has no points associated with it. If you specify points weight for a problem, the points appear in the margins (when one of the option

pointsonleft, pointsonright, or pointsonboth is specified); if the * form is specified (*num), the point weight appears "in-line," just after the problem number; thus, typesetting a problem with the specification \begin{problem}[*5] yields

1. (5 pts) …

This is useful when the problems are put into a two-column format; the problems in the right-hand column do not have the margin to hold the points, in this case, we place the points "in-line."

The problem is actually a redefined exercise environment, as defined in exerquiz. The second parameter is inherited from the exercise environment. The second argument can optionally be an h or a H.

Use h if you do not want the solution to appear at the end of document (when you do not use the nosolutions or the solutionsafter options); the solution, however, will appear if the solutionafter option is specified.

For the H argument, the solution will not appear at the end of the document (just as in h), nor will it appear if you specify the solutionsafter option.

To make things work correctly, if you do not want to have points for a question and want to hide the solution, use '[]' (empty brackets with no spaces) for the first argument.

```
\begin{problem}[][H]
($5$ Points Extra Credit) Solve this problem for extra credit.
\begin{solution}
This solution will not appear in all cases, unless the second
parameter is eliminated or is changed to h, in the latter case,
the solution appears just for \texttt{solutonsafter}.
\end{solution}
\end{problem}
```

Here, this problem has no points that will be added into the total number of points for the test.

The solution environment encloses the solutions. This environment is optional. The environment takes at most two optional parameters, *vspace* and nLines=*n*. The *vspace* parameter is a length that determines the amount of vertical space to leave for the student to work the problem. The nLines=*n* specification signals eqexam to leave *n* lines of vertical space; each line is \wlVspace in height. (For more information on \wlVspace, read about the linegap key in Section 10.12.) This vertical space is *created only* when the document author takes the nosolutions or vspacewithsolns option. For example,

nLines=n explained

```
\begin{problem}[10]
Do this problem.
\begin{solution}[2in]
This is the solution.
\end{solution}
\end{problem}
```

This defines a 10 point problem and leaves 2 inches of vertical space following the problem statement for the student to respond. The vertical space is generated provided the `nosolutions` or `vspacewithsolns` option has been taken.

Be aware that the solution environment searches for its optional parameter, and will expand macros looking for a left bracket ([). In documents where the optional parameter is not used; this can lead to problems in compiling. For example, if you say, `\begin{solution} \textbf{My solution:}...`, the command `\textbf` will be expanded prematurely and result in 'My solution' not appearing in bold. Similarly, if you write `\begin{solution} \begin{equation}...` can lead to compilation stopping. Suggested workarounds:

- Supply empty brackets: `\begin{solution}[]`
- Use `\relax`: `\begin{solution}\relax\textbf{...}`. The `\relax` should not be on the line by itself.

  ```
  \begin{solution}\relax  % Not this
  \textbf{...}
  ...
  ```

  The above causes an unwanted newline. The next two examples show the "correct" method.

  ```
  \begin{solution}\relax\textbf{...}  % correct
  ...
  ```

  The `\relax` appearing on the second line.

  ```
  \begin{solution}
  \relax\textbf{...}  % correct
  ...
  ```

If you have no need for the vertical space in your document and putting in these workarounds is too much trouble, you can use a global solution. Use `\noSolnOpt` to globally turn off the check for the option parameter by the `solution` environment; `\ckSolnOpt` turns on parameter checking (the default). To summarize:

```
\ckSolnOpt  % turn on checking for the optional argument (the default)
\noSolnOpt  % turn off checking for the optional argument
```

Place either of these two commands between problems to turn off (or back on) the parameter checking.

▶ See 'eqexam Options' on page 30 for more details on the two options `nosolutions` and `solutionsafter`.

Optional arguments of `solution` environment. The `solution` environment takes at most two optional arguments *vspace* and `nLines=`*n*. If both are specified, by default the *vspace* parameter is used. The command `\usenLineDimen` changes the preference to the line specification; `\useVspaceDimen` switches the preference back to the *vspace* dimension.

- **problem***

This environment is used when you want to ask a multi-part question, a series of related questions that are to be treated as a group.

The syntax is

```
\begin{problem*}[num|⟨num⟩ea|\auto|empty][\Do⟨do_num⟩]
Do each of the following problems, and be quick about it.
\begin{parts}

\item[h|H] The first question.
\begin{solution}[vspace,nLines=n]
This is the solution to the first problem.
\end{solution}

\item[h|H] The second question.
\begin{solution}[vspace,nLines=n]
This is the solution to the second problem.
\end{solution}

\end{parts}
\end{problem*}
```

The problem* environment takes two optional parameters, the first one takes one of four values:

num When the value of the first parameter is a number, this represents the total number of points for this multi-part question. Here, the instructor does not specify the weight of each part.

**num* The points appear "in-line" rather than in the margin.

⟨*num*⟩ea When you specify a number followed by 'ea' (which is short for each). Thus, '[5ea]' signifies that each part of this problem has weight of 5 points.

*⟨*num*⟩ea The points appear "in-line" rather than in the margin.

\auto If the value of the first parameter is \auto, then the total number of points is calculated automatically from the points defined by the \PTs macro. The \PTs would be placed following \item of each part that is to be given points. For example:

*\auto The points appear "in-line" rather than in the margin.

```
\begin{problem*}[\auto]
Do each of the following problems, and be quick about it.
\begin{parts}

\item\PTs{3} The first question.
\begin{solution}[1.5in]
This is the solution to the first problem.
\end{solution}
```

```
\item\PTs{4} The second question.
\begin{solution}[3in]
This is the solution to the second problem.
\end{solution}

\end{parts}
\end{problem*}
```

This defines a 7 point problem.

`empty` You need not specify any points at all. In this case do not include this first parameter, in which case, the second parameter is not used, so don't include it either.

Now for a description of the second parameter the `[\Do⟨do_num⟩]` parameter. In my senior- or graduate-level classes, I sometimes ask a questions with multiple parts. As part of the instructions for that problem I write, "Do exactly three of the following five problems." These questions are usually proof-type problems, and they can choose their best three to grade. In this context, all parts of the problem must be of the same weight; the weight of each is ⟨*num*⟩ of the `[⟨num⟩ea]`.

This is what `[\Do⟨do_num⟩]` does. When you specify `\Do3`, then only the points of 3 of the problems are added into the exam total. This second parameter is only checked if the first parameter is `[⟨num⟩ea]`. For example, specifying

```
\begin{problem*}[5ea][\Do3]
```

creates a 15 point question. This assumes there are 3 or more parts to this question.

By the way, there are two macros that are defined when the `\Do` is used, they are `\DoNum` and `\OutOfNum`; these expand to the (English) word for the number of problems to do, and the (English) word for the total number of problems. For example, if there were five parts to the problem below,…

```
\begin{problem*}[5ea][\Do3]
Solve exactly \textit{\DoNum} of the following {\OutOfNum}
problems. ....
\end{problem*}
```

The instructions would read, "Solve exactly *three* of the following five problems." These macros can be easily redefined to reflect other languages. The numbers themselves are contained in the two macros `\nDoNum` and `\nOutOfNum`.

▶ `parts` and `\item`: For a multi-part problem (`problem*`), the actual problems are enclosed in a `parts` environment, and each question is posed as an `\item` of that `list` environment. The command `\item` takes the `[h|H]` optional argument. As in the case of the `problem` environment, `h` prevents the solution from appearing at the end of the document (but it appears with `solutionsafter`), and `H` removes the solution in all cases.

\leadinitem When using the problem* environment, there is an introductory sentence that sets up the multi-part problem set. For various reasons, some authors have asked to be able to pose multi-part questions without the introductory sentence. This is harder request than it sounds, but now there is the \leadinitem command. Study the code below.

```
\begin{problem*}[\auto]
\leadinitem\PTs{3} The first question.
\begin{solution}[1.5in]
This is the solution to the first problem.
\end{solution}

\begin{parts}
\item\PTs{4} The second question.
\begin{solution}[3in]
This is the solution to the second problem.
\end{solution}
...
\end{parts}
\end{problem*}
```

There is no introductory sentence. The problem starts off with \leadinitem\PTs{3} The first question; this problem is stated outside of the parts environment. The rest of the parts to this problem are listed, as usual, from within the parts environment. Only one \leadinitem is allowed per problem* environment.

The results of this code is viewed as follows, when typeset.

(10 pts) **1.** (a) The first question.

(b) The second question.

...

The general syntax for \leadinitem is the same as that of the \item command within the parts environment; \leadinitem[h|H], h prevents the solution from appearing at the end of the document (but it appears with solutionsafter or with answerkey), and H removes the solution in all cases.

\tableadin There is a tabular version of the \leadinitem command just discussed. Consider the following code:

```
\autotabOn
\begin{problem*}[\auto]
\tableadin
\begin{parts}[2}
\item\PTs{4} The first question.
\begin{solution}[1.5in]
This is the solution to the first problem.
\end{solution}
%
\item\PTs{4} The second question.
```

```
\begin{solution}[3in]
This is the solution to the second problem.
\end{solution}
...
\end{parts}
\end{problem*}
```

The results of this code is viewed as follows, when typeset.

(10pts) **1.** (a) The first question. (b) The second question.

…

- **Page Breaking**

The `exam`, `problem` and `problem*` environments use a (simple) page breaking algorithm to move a problem (or the beginning of an exam) to the next page.

If an `exam` environment begins at the lower third of the page, it is moved to the next page. You can influence this page break by using `\fvsizeskip` just before the beginning of the `exam` environment, like so,

```
\fvsizeskip{.4}
```

`\fvsizeskip` takes a decimal number between 0 and 1. In the example above, the environment will move to a new page if it begins in the lower `.4\textheight` of the page. The default value is .3.

There is a similar algorithm for `problem` and `problem` but is measured as a multiple of `\baselineskip`. If you place

```
\nbaselineskip{8}
```

just before a problem that appears near the bottom of the page, then it will be moved to the next page if it is within `8\baselineskip` of the bottom. The default for this command is 6.

The following are strategies for fitting the maximum number of questions on the minimum number of pages.

1. **Moving:** Rearrange the order of the questions, if a problem can't fit entirely on a page, you can exchange or move a shorter problem to that place, and move the longer problem to another page.
2. **Tweaking:** Modify the space defined by the `solutions` environment to fit a problem on the page that is below it.
3. **Placing work on back:** Using the `\OnBackOfPage` command, page 58, you can direct the student to answer the question on the back of another page, and thus, little space is needed to follow that question.
4. **Working on separate sheets:** Of course, for some types of exams, the exam just contains the questions, and the students answer the questions on separate sheets of paper. For this, you can use the `nospacetowork` option.

7.4. Fill-in Questions

In this section we cover the various fill-in constructs.

- **Short Fill-in Questions**

For a question requiring one or more short fill-in responses, eqexam has the \fillin command, the syntax is

```
\fillin[u|b]{width}{answer}
```

The first optional parameter determines whether the fill-in is underlined '[u]' or not '[b]', the default it to underline the fill-in. The second is the amount of horizontal space you want to leave for the student to write in the response. The third argument is the correct answer. This correct answer will appear when you compile the document with the answerkey option.

▶ An example of \fillin.

```
\begin{problem}[5]
It is well known that \fillin{1in}{Newton} and \fillin{1in}{Leibniz}
are jointly credited as the founders of modern calculus.
\begin{solution}
It is well known that \underbar{Newton} and \underbar{Leibniz}
are jointly credited as the founders of modern calculus.
\end{solution}
\end{problem}
```

▶ When you choose the online or email option, \fillin generates a text field.

When the usexkv option, and if the xkeyval package is available on the system, eqexam extends the capability and control of \fillin. See 'Extending the \fillin Command' on page 63.

- **True/False Questions**

True and false questions are, of course, just a special case of fill-in. A special command is available for true/false:

```
\TF[width]{answer}
```

The required parameter, *answer*, is the correct answer (e.g., 'T' or 'F'). The macro creates an underlined blank space the width of which is *wide*. When the *width* is *not specified*, \defaultTFwidth (default 30pt) is used (and this value can be redefined).

The \TF command behaves differently from the generic \fillin command. Suppose you want to create a multi-part question (using problem*) consisting entirely of true/false questions. When an \item leads off with the \TF there are two possible formatting options: This one:

(a) _____ Isaac Newton is considered to be one of the founders of Calculus.

or this one:

(a) _____ Isaac Newton is considered to be one of the founders of Calculus.

The first alignment is the default. To get the second alignment, you need to set the value of `\fillinWidth` to the common width value of the `\TF` fields. For example:

```
\fillinWidth\defaultTFwidth
```

When `\fillinWidth` is set to a positive length (the common width of the `\TF` field), the second alignment above is created.

```
\begin{problem*}[3ea]
\textit{True} or \textit{False}.

\fillinWidth\defaultTFwidth

\begin{parts}

    \item \TF{T} It is well known that Isaac Newton and
    Gottfried Leibniz are jointly credited as the founders
    of modern calculus.
    ...
    \item ...
    ...
\end{parts}
\end{problem*}
```

- **Important:** The example above demonstrates the correct placement of `\fillinWidth`, just outside the `parts` environment, before it has the time to set up the paragraph shape of the environment.

  The change is only local to that `parts` environment. The `\fillinWidth` command goes outside a `parts` environment, and can cause strange results if executed within a `parts` environment. Setting it to a *`width`* value other than the common width of the `\TF` fields will also create bad formatting.
- Just use `\fillinWidth` as illustrated in the above example.
- When you choose the `online` or `email` option, `\TF` generates a text field.

• Long Fill-in Questions

There is no special command for a longer response question, just leave enough vertical white space for the student to respond, for example,

```
\begin{problem}[5]
Do this problem
\begin{solution}[1.5in]
That's how you do it!
\end{solution}
\end{problem}
```

The above example leaves 1.5 inches of vertical space to do the work.

- When you choose the `online` or `email` option, this vertical space is changed into a multi-line text field.

7.5. Multiple Choice

For multiple choice questions, we use the `answers` environment. If the `online` or `email` option is taken, the choices are made into radio button fields so that *only one alternative* can be chosen. When multiple selections are permitted, the `manswers` environment can be used, see 'Multiple Selection' on page 23.

```
\begin{problem*}[\auto] Answer each of the following.
\begin{parts} %\sqLinks
    \item\PTs{5} In what year did Columbus sail the ocean blue?
    \begin{answers}{4}
    \Ans0 1490 &\Ans0 1491\\
    \Ans1 1492 &\Ans0 1493
    \end{answers}
    \item\PTs{6} In what year did Columbus sail the ocean blue?
    \begin{answers}{1}
    \Ans0 1490
    \Ans0 1491
    \Ans1 1492
    \Ans0 1493
    \end{answers}
\end{parts}
\end{problem*}
```

Note: No solutions are given for this problem.

▶ Because the labels and values of the alternatives are based on the alphabet, the number of alternatives is restricted to twenty-six.

The `answers` environment is borrowed from `exerquiz` and operates the same way. The one required argument is the number of columns to be used in displaying the alternative answers. If the number of columns is 1, a `list` environment is used, otherwise a `tabular` environment is used.

In the first item in the example above, we specify 6 columns, and must use tabular notation (separate columns with '&') and end rows with '\\'. The second item in the example above uses 1 column, the tabular notation is not needed, or used.

The `\Ans` macro is used to designate which alternative is the correct answer (1 for correct, 0 for not correct).

- **Using `\bChoices/\eChoices`**

Beginning with Version 1.3, an alternate style of specifying the alternatives is defined. A new pair of commands are defined: `\bChoices` and `\eChoices`. These two enclose the alternatives like so:

```
\begin{exam}{Exam1}
\begin{instructions}
Select the ``best'' answer and darken the corresponding oval on
your scantron sheet.
\end{instructions}
\begin{problem}[5] In what year did Columbus sail the ocean blue?
```

```
    \begin{answers}{3}
    \bChoices
        \Ans0 1490\eAns
        \Ans0 1491\eAns
        \Ans1 1492\eAns
        \Ans0 1493\eAns
    \eChoices
    \end{answers}
\end{problem}
\begin{problem}[5] In what year did Columbus sail the ocean blue?
    \begin{answers}{1}
    \bChoices
        \Ans0 1490\eAns
        \Ans0 1491\eAns
        \Ans1 1492\eAns
        \Ans0 1493\eAns
    \eChoices
    \end{answers}
\end{problem}
\end{exam}
```

Notice that the set of alternatives are the same, and are specified in exactly the same way; the first question, however, is a tabular environment with 6 columns (the argument of 6 of the `answers` environment), the second question is a list environment (since the argument `answers` environment is 1). Notice also that '&' and '\\' are not used, and that each alternative is terminated by `\eAns`.

The `\bChoices` and `\eChoices` are creatures of the `exerquiz` package, and are fully documented in the reference for the AcroTEX Bundle.

- **`\sqForms` versus `\sqLinks`**

There are two styles of multiple choice: (1) enumerate the alternatives using letters; (2) enumerate the alternatives using boxes (that the student would check or fill-in). The default is (1), but you can change the default to (2) by using the `useforms` option. This styles can be locally changed by specifying the `\sqLinks` or `\sqForms` commands. In the above example, the `\sqLinks` command is commented out, but shows the correct position for it to change to style (1), which I am calling "links". Within a multi-part, multiple choice set of questions, you can change one item to "links" and the next to "forms." Changes are local as long as you place the commands, `\sqLinks` or `\sqForms` within an environment (`parts`, `problem`, or `problem*`).

- **Using Circles for Multiple Choice**

Then the package option `allowcirc4mc` is used, the font standard LaTeX font `lcircle10` is loaded at which point `eqexam` can use it to create circles, instead of rectangles, to indicate the parts in a multiple choice (MC) question. To use circles in a multiple choice question, execute the command,

```
\useCircForMC
```

prior to the MC question.

To return to the use of rectangles, execute the command,

```
\useRectForMC
```

prior to the MC question.

Both commands have a local context. If expanded inside a group, the definition going into the group will hold on exit from the group.

- **Using `\proofingsymbol` and friends**

By default, a check mark (✔) is used to indicate which of the alternatives in a MC problem is correct; however, there are other "proofing symbols" that can be used. Below are two additional suggested proofing symbols.

`\useCheckForProof`	Check ✔, the default
`\useCrossForProof`	Cross ✘, alternative to check
`\useCircForProof`	Circle ●, appropriate with `\useCircForMC`

All of these user friendly commands are based on the `\proofingsymbol` command. For example, the definitions of `\useCheckForProof` and `\useCrossForProof` are,

```
\newcommand{\useCheckForProof}{\symbolchoice{check}%
    \proofingsymbol{\ding{52}}}
\newcommand{\useCrossForProof}{\symbolchoice{cross}%
    \proofingsymbol{\raisebox{-1pt}
        {\rlap{\kern-1pt\Large\ding{56}}}}}
```

Both definitions use the `pifont` package to create the symbols. Note that some adjustment of size and position is used for the cross symbol.

The command `\symbolchoice` is defined in the `eforms` package and does nothing in `eqexam` unless either `online` or `email` options are taken. From the `eforms` manual, possible values for `\symbolchoice` are `check`, `circle`, `cross`, `square`, `diamond`, and `star`. The `\proofingsymbol` command is for marking the multiple choices when either the `answerkey` or `vspacewithsolns` option is taken. The choice of `\proofingsymbol` is 'LaTeXed' into the document. The `\proofingsymbol` may be used to create other proofing symbols, as desired.

Summary. Currently, there only two geometric shapes used for multiple choice, rectangles (the default) and circles. To Shift between these two types, use `\useRectForMC` and `\useCircForMC`, respectively. Accompanying the choice for geometric shape for MC is the symbol used to make the choice/correct answer. When the `answerkey` or `vspacewithsolns` option is used, the correct answer is marked using a symbol, current choices are `\useCheckForProof`, `\useCrossForProof` and `\useCircForProof` (used with `\useCircForMC`).

When the `vspacewithsolns` is used, solutions are written to the back of the document and markup up as they are with the `answerkey` option. To get the answers in the solutions section to have the same choices, you must write to the solutions file using `\writeToSolnFile`. Below is an example.

```
\useCircForMC\useCircForProof
\writeToSolnFile{\protect\useCircForMC\protect\useCircForProof}
\begin{problem}[5] In what year did Columbus sail the ocean blue?
    \begin{answers}{3}
    \bChoices
        \Ans0 1490\eAns
        \Ans0 1491\eAns
        \Ans1 1492\eAns
        \Ans0 1493\eAns
    \eChoices
    \end{answers}
\end{problem}
```

Any multiple choice question that follows will also draw circles for multiple choice questions, and mark them with a filled circle. To shift back to the default, expand the following commands prior the next question.

```
\useRectForMC\useCheckForProof
\writeToSolnFile{\protect\useRectForMC\protect\useCheckForProof}
...
```

7.6. Multiple Selection

When writing a multiple choice question for which more than one alternative is permitted, use the `manswers` environment (multiple answers). The distinction between the `answers` and `manswers` environments is lost when publishing to paper, but becomes important with the `online` and `email` options.

Use the `manswers` environment in the same way you use `answers`, except code in more than one correct answer. For example,

```
\begin{problem}[5]
Which of the following are primary colors?
\begin{manswers}{6} % specify tabular with 6 columns
    \bChoices
        \Ans1 Blue\eAns
        \Ans0 Green\eAns
        \Ans1 Yellow\eAns
        \Ans0 Orange\eAns
        \Ans1 Red\eAns
    \eChoices
\end{manswers}
\begin{solution}
Yes, red, blue and yellow are primary colors.
\end{solution}
\end{problem}
```

You can use the `\bChoices`/`\eChoices` pair to specify the alternatives, or you can use the standard tabular notation. As with the `answers` environment an argument of `1` specifies a list environment. See 'Multiple Choice' on page 20 for more examples on the use of the `\bChoices`/`\eChoices` pair.

7.7. Randomizing Choices

Beginning with version 1.7 of eqexam, the choices of a multiple choice/selection question can be randomized. The random.tex macro file by Donald Arseneau is used for this purpose.

The randomization is only allowed if the allowrandomize option of eqexam is used; otherwise, no randomization can occur.

The randomization is only defined for choices listed between the pair \bChoices and \eChoices. The \bChoices command now takes two optional key-value arguments:

- nCols=⟨*num*⟩: The number of columns to create, as described. You can also use the old style by specifying just ⟨*num*⟩. Thus, \bChoices[nCols=2] and \bChoices[2] are equivalent.
- random=⟨true|false⟩: Specify this option if you want the choices to be randomized. You can use the key word random instead of random=true. For example, the following commands all will randomize the choices, \bChoices[random] or \bChoices[nCols=2,random] or \bChoices[2,random=true]. The default is to not randomize the choices.

The following is an example of the random option of \bChoices.

```
\begin{problem}[5]
Try to guess the correct answer.
    \begin{answers}{3}
    \bChoices[nCols=2,random]
        \Ans0 1 a choice\eAns
        \Ans1\label{eq} 2 another choice\eAns
        \Ans0 3 still another choice\eAns
        \Ans0 4 another\eAns
        \Ans0 5 incoming\eAns
        \Ans0 6 more choices\eAns
        \Ans0 7 another still\eAns
        \Ans0 8 too many\eAns
        \Ans0 9 choices\eAns
    \eFreeze
        \Ans0 10 None of these\eAns
    \eChoices
    \end{answers}
\end{problem}
```

Note the presence of the command \eFreeze. Any of the items listed after \eFreeze are not randomized, and are placed at the end of the list. So, for the example above, the first nine items will be randomized, whereas, the last item (None of these) will placed at the end of the list.

Additionally, there are five other commands that support the randomization feature.

```
\saveRandomSeed
\inputRandomSeed
```

A pseudo-random sequence of numbers requires an initial seed value. The `random.tex` macro file creates, by default, a seed value based on the data and time (the number of minutes since midnight); consequently, after every minute, the random sequence will change. By setting the value of the count register `\randomi`, as in `\randomi=24`, the document author can also set the initial seed of the pseudo-random sequence.

The command `\saveRandomSeed` will write the last seed used in the source file to an auxiliary file (`\jobname_ran.sav`), while the command `\inputRandomSeed` inputs the seed stored in the `\jobname_ran.sav` back into the beginning of the source file. These two commands should be placed in the preamble.

By invoking both of these commands, a new pseudo-random sequence will be generated each time the source file is latexed.

Assuming a `\jobname_ran.sav` has already been created, by invoking the command `\inputRandomSeed` only (and not `\saveRandomSeed`), the seed already saved will be used for every subsequent compiling of the source document. Using the same seed is necessary in two situations:

1. When the document contains one or more `\label` commands, using the same seed gives you the same sequence every time you latex the document. This will give the auxiliary files a chance to come up to date so that any referencing of the label will be accurate.

2. When creating an exam with randomization that has several versions, which later you publish the solutions to, it is important that the randomization for the document is the same as that for the solution document. By using `\inputRandomSeed` (and not `\saveRandomSeed`), you should get the same sequence for the solution document (unless you modify the source file, adding or removing questions that have randomization).

Things to look for: If `eqexam` is not rearranging the order of the choices as you expect it to, it could be that `eqexam` is reading an old `.sav` file. Either delete that file in your source folder, or comment out `\inputRandomSeed` in your document.

`\useRandomSeed{`*num*`}`

You may have several sections of the same class take the exam with the questions rearranged for each. Save the seed value used by `eqexam` to randomize the choices (open the `.sav` and copy and paste line you see into your document, for example, it could read `\randomi=132088850`. Then use `\useRandomSeed` to use that seed value for that class, for example

```
\useRandomSeed{132088850}   % 11:00 class
% \useRandomSeed{634952429} % 12:30 class
```

Of course comment out `\inputRandomSeed`.

```
\turnOnRandomize
\obeyLocalRandomize
```

The command \turnOnRandomize overrides all local settings of \bChoices and causes all choice lists to be randomized. While \obeyLocalRandomize returns control to the local settings. For example,

```
\turnOnRandomize
...
\bChoices
    \Ans...\eAns
    \Ans...\eAns
    ...
\eChoices
```

will cause the choice list to be randomized, even though the random option was not specified. Whereas, in this code

```
\turnOnRandomize
...
\obeyLocalRandomize
,,,
\bChoices
    \Ans...\eAns
    \Ans...\eAns
    ...
\eChoices
```

the choices will not be randomized, because the random option was not specified; or they will be randomized if the random option is used.

Limitations: There are natural limitations on the use of \bChoices and \eChoices and consequently, there are limitations on the randomization. The content between \Ans and \eAns cannot have any verbatim text. This is usually not a problem for mathematical content, but could be a limitation for computer science where questions about syntax may be posed. I have in mind a work-around, but haven't pursued the problem as of yet.

7.8. Labeling Choices

The \bChoices command has a label key, \bChoices[label=⟨*label*⟩], used to specify a (unique) label for the current set of choices. When a label is specified, eqexam creates commands that save the label of each correct answers (for multiple choice/multiple selection problems), and saves the answer text for each correct answer. These can be read back into the document using some user-interface commands: \useSavedAlts, \useSavedAns, \useSavedAltsAns, and \useSavedNumAns.

The demo file for this feature is named test03.tex, download test03.pdf from the AcroTeX Blog website. The source file is attached to the PDF.

```
\useSavedAlts[num]{label}
\useSavedAns[num]{label}
\useSavedAltsAns[num]{label}
\useSavedNumAns{label}
```

The optional argument is useful *only if* the \bChoices appears in a manswers environment where there are more than one selectable answer. The required parameter is the value of the label key.

\useSavedAlts{*label*} expands to the *label* of the correct answer(s). For example \useSavedAlts{*label*} might expand to (c); if there are multiple answers, it might expand to (a), (c), a comma-delimited list of labels of the (correct) answers. For multiple selection, \useSavedAlts[2]{*label*} might expand to (c), the label of the second correct answer.

\useSavedAns{*label*} expands to the *text* of the correct answer(s). As an example, \useSavedAns{*label*} might expand to $y = x^3$; if there are multiple answers, it might expand to $y = x^3$, $y = -x^3$, a comma-delimited list of the text of the (correct) answers. For multiple selection, \useSavedAns[2]{*label*} might expand to $y = -x^3$, the text of the second correct answer.

\useSavedAltsAns{*label*} combines the two previous commands. Again, for example, \useSavedAltsAns{*label*} might expand to (c) $y = x^3$. When there are multiple answers, it expands to a comma delimited of labels and text. As with the other two commands, the optional argument can be used to pick off a particular choice.

\useSavedNumAns{*label*} is the number of correct answers in the current list of choices.

7.9. Gizmos and Gadgets

I have a couple of crazy gizmos that you can use.

- **The workarea Environment**

For a mathematics test, we often pose a question that needs to be worked out. Vertical space is created by the solutions environment, and appears when the nosolutions or vspacewithsolns option is used; however, often we want to mark up this vertical space with additional instructions, a diagram or a figure. The problem is how can the author write over the provided white space. For this, eqexam provides the workarea environment. The syntax is:

```
\begin{workarea}[width]{depth}
...
⟨Material that will overwrite the solutions vertical space⟩
...
\end{workarea}
```

This environment is placed immediately *after* the solutions environment, and the value of its parameter should be the same as the optional parameter at the beginning the solutions environment (\begin{solutions}[*depth*]). The optional *width* parameter is the width of the work area, which is \linewidth by default. The required *depth* parameter is the depth of the work area and it should match the optional parameter of the solutions environment, directly above it.

```
\begin{problem}[3]
This is a question.

\begin{solution}[2in]
This is the solution, let's hope it's correct, or I would be
embarrassed to no end.
\end{solution}

\begin{workarea}{2in}
\textit{Hint}: Think long and hard before answering.
\par\vfill\hfill\setlength{\fboxsep}{2mm}
\fbox{Answer:\fillin[n]{1in}{The correct answer.}}
\end{workarea}
\end{problem}
```

When the `nosolutions` option is taken, the `solutions` leaves 2 inches of white space. The `workarea` environment that follows also specifies 2 inches, and the content of this environment will overlap the white space. (The student would then work around the written material.) Here, we give a hint, and leave an answer box (a fill-in) for the student to insert his/her answer.

When the `nosolutions` is not specified, the vertical space is not provided, and the `workarea` does nothing. If `solutionsafter` is specified, that space is replaced by the provided solution.

- **The `\placeAtxy` Command**

The `\placeAtxy` command is another device that I've used to place a block of text or a graphic on top of the vertical space created by the `solutions` environment when the `nosolutions` or `vspacewithsolns` option is in effect.

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

The first two arguments are the x and y coordinates (with dimensions) of where the *content* is to be placed. If this command is placed below the `solutions` environment, then the origin is the lower left corner of the solutions box.

The following example, places the frame box Place a graph here (roughly) one inch up and one inch shifted to the right, measured from the bottom left corner of the `solutions` environment (when the `nosolutions` option is in effect). As with `workarea`, `\placeAtxy` does nothing if the `nosolutions` option has not been taken.

```
\begin{problem}[3]
This is a question.
\begin{solution}[2in]
This is the solution, let's hope it's correct, or I would be
embarrassed to no end.
\end{solution}
\placeAtxy{1in}{1in}{\framebox{Place a graph here}}
\end{problem}
```

The `\placeAtxy` command can also be used in combination with the `workarea` environment.

- **The `splitsolution` Environment**

I developed this environment to solve a problem with the `online` and `email` options. The white space created by the `solution` environment is converted into text fields (PDF form fields). If the `workarea` environment or the `\placeAtxy` command is used to place content on the white space, the student will be in the position of having to type on top of this content. (See the demo file `test01.pdf` for an illustration of this.)

Therefore, it was necessary to have a way to separate the space reserved for the text field, and the additional content you might want to appear in this white space area. The `splitsolution` environment is my solution to this problem.

As of 2012/12/10, a new syntax has been implemented for the `splitsolution` and `panel` environments. Below is a side-by-side comparison of the new syntax and the old.

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}
```

There has not been any feedback to this feature, so I am confident that this change has little impact on users. Both *width* and *height* are optional arguments for the new syntax of `splitsolution`. If there are no optional arguments, the default values of `\panelwidth` and `\panelheight`; these are automatically measured. If only one optional parameter is given, it is interpreted as *height* (and the *width* is taken to be `\panelwidth`). The default value of the optional parameter for the `panel` environment is now `r` rather than `l`.

▶ Consider the following example.

```
1  \begin{problem}[7]
2  This is a question worth $7$ points.
3  \begin{splitsolution}
4  \begin{panel}\relax
5  \includegraphics[scale=.2]{fig1}
6  \end{panel}
7  \begin{solution}
8  This a really good  solution. I hope this solution is correct or I
9  will be totally embarrassed to no end. Even if it is wrong, maybe
10 the students will appreciate my tremendous effort. You can see
11 from the figure that the solution is obvious.
12 \end{solution}
13 \end{splitsolution}
14 \end{problem}
```

Note the use of `\relax` in line (4). The first object in the panel environment is a command. To prevent the command from expanding prematurely, place a `\relax` as above.

This will give you the default parameter of `r` and prevent expansion. The use of `\relax` is only needed when there is a command immediately following the opening of the `panel` environment; otherwise, just `\begin{panel}` should work correctly. The optional argument can always be specified, `\begin{panel}[r]`; this too would prevent the premature expansion of any command that immediately follows.

The `panel` environment takes its contents and writes it verbatim to a CUT file, then inputs it back in (at the end of the `panel` environment), and places its contents in the box `\eqpanelbox` where it takes it measurements of `\panelwidth` and `\panelheight` (the total height).

The `splitsolution` environment *must* enclose two other environments: The `panel` and the `solutions` environments, *in that order*.

The `panel` environment comes first and takes optional argument. The optional parameter has takes a value of 'r' (the default) or 'l'. The `r` (resp., `l`) option means the panel is to appear on the right (resp., left) of the solution (or vertical white space).

After the `panel` environment comes the `solutions` environment. The optional parameter of this environment need not be specified, as it gets its value from the `splitsolution` parameter.

There is a small gap of `3pt` (the default) inserted between the panel and the solution. The value of this gap is contained in the `\panelgap` command,

```
\newcommand\panelgap{3pt}
```

which can be redefined.

▶ The depth (the default is `\panelheight`) that you specify as the parameter of the `splitsolution` environment needs to be large enough to accommodate your typeset solution; otherwise, the solution will overlap the next problem. This is because, unlike the solutions inside a `solution` environment (but not in a `splitsolution` environment) are typeset in a `minipage` with a specified depth.

To extend the height of the solution, use the following method.

```
\begin{splitsolution}[\panelheight+1in]
...
\end{splitsolution}
```

This sets the total height to be the natural height of the panel plus 1 inch.

8. eqexam Options

The options documented here are entered as optional arguments of the eqexam package:

```
\usepackage[optionals]{eqexam}
```

The optional arguments can also be introduced through `exambuilder.cfg`, the configuration file. Create a text file with the name of `exambuilder.cfg` and create the line shown below.

```
\ExecuteOptionsX{optionals}
```

Place exambuilder.cfg in the folder of the source file and not on the LaTeX search path.

- The eqexam package has numerous options, some inherited from web, some from exerquiz, and a number of new ones.

forpaper Take this option when you want to create a black and white paper version of your test.

forcolorpaper Take this option when you want to have a nice colorful paper version, or are publishing on the web in PDF. See 'Customization of Color' on page 50.

nosolutions This is the normal option taken when you are printing a test for distribution to a class of students. When this option is taken, vertical space is generated by the solutions environment based on the value of its optional parameter. This leaves room for the student to solve/answer the question.

nohiddensolutions If you use the h optional parameter for problem or \item, the solution will not be listed (at the end of the document) *when you do not specify* nosolutions; but solutions will be typeset for the solutionsafter option. This option will override this feature.

noHiddensolutions Normally, when you use the H optional parameter for problem or \item, the solution will not be listed when you use the nosolutions or solutionsafter options for eqexam. This option will override this feature.

solutionsafter Causes solutions to appear following the statement of the problem.

When the solutionsafter is in effect, the word *Solution:* is typeset at the beginning of the solutions. The command \renameSolnAfterTo can be used for conveniently changing the solutionsafter label, for example, executing the command \renameSolnAfterTo{\textbf{Proof:}} prior to a solution environment changes the label to **Proof:**; \renameSolnAfterTo{} produces no label. These changes will be local to the group in which they are made, or global of there they are not made in a group.

The command \resetSolnAfterToDefault sets the label text back to the default. The default label is \textit{Solution}:.

preview The bounding boxes are shown when this option is taken, provided the online or email option is chosen. See the description of these two options below.

proofing Using this option will cause the correct answer for multiple choice questions to be marked with a check mark; the correct answers for fill-in questions (\fillin or \TF) are also shown.

The answerkey option, described below, executes the proofing and solutionsafter options.

- The following options are unique to the eqexam package.

pointsonleft The points for the problem are displayed in the left margin.

`pointsonright` The points for the problem are on the left margin.

`pointsonboth` Points are displayed in both margins.

`nopoints` Causes points not to be displayed, or calculated. Useful for writing documents that do not have points, such as a questionnaire.

`totalsonleft` The totals for each page can be displayed at the bottom left corner of each page using this option.

`totalsonright` The totals for each page can be displayed at the bottom right corner of each page using this option.

`nototals` Use this option if you don't want any totals at the bottom of the page.

`noparttotals` When multiple `exam` environments appear on the same page, they are separated by a horizontal rule. The page total for the closing `exam` environment is inserted into the margin on the same line as the horizontal rule. This option turns off the insertion of the page total for the closing `exam` environment.

There are two commands that can be used for local control of this feature, they are `\eoeTotalOff` and `\eoeTotalOn`. When an `exam` ends near the bottom of one page, the new exam will begin on the next page, this results in the horizontal rule being generated with the end of exam totals, and the totals at the bottom as well. If these two numbers are the same, then you can turn off the end of exam total using `\eoeTotalOff`. Use this command just above `\end{exam}` and the changes will be local to that exam part.

`parttotalsonright` Place the part totals in the right margin, this is the default.

`parttotalsonleft` Place the part totals in the left margin.

`nosummarytotals` When you use the `instructions` environment, the total points for `exam` are displayed following the instruction heading. Using this option turns off this feature.

`noseparationrule` When the document has multiple `exam` environments, a separation rule is placed between them. This option turns off that feature.

The design of the separation rule may be modified by the document author by redefining `\separationrule`, its definition is given below:

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

`coverpage` Some instructors like to have a cover page for their exams, use this option to create a cover page. Use the `\eqexcoverpagedesign` command to design your own cover page.

`coverpagesumry` is a companion to the `coverpage` option, `coverpagesumry` takes one of three values: `bypages`, `byparts`, `none`.

`coverpagesumry=bypages` If `bypages` is chosen, an "Exam Record" appears on the cover page. See the left-hand figure in Figure 1. A page total appears on each line. Note "Page 3," in the figure; the total there is "37 pts (12 pts + 25 pts)." This means that there are 37 points on page 3; on this page the first `exam` environment ended and a second `exam` environment begins, there are 12 points on page 3 from the first `exam` environment, and 25 points on that page from the second `exam` environment.

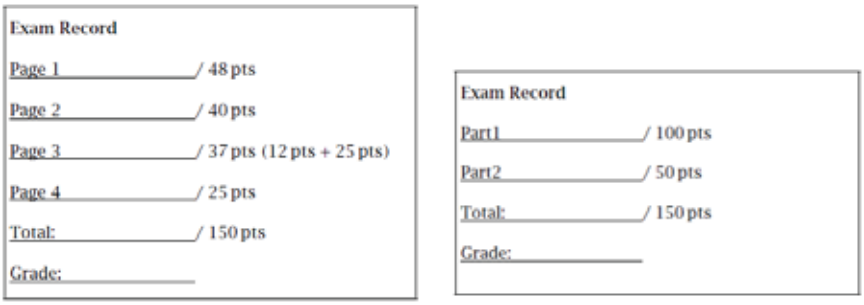


Figure 1: Exam Record

`coverpagesumry=byparts` If `byparts` is chosen, an "Exam Record" appears on the cover page that lists the number of points per part. (Each exam environment is considered here a "part.") See the right-hand figure in Figure 1.

`coverpagesumry=none` If this option is chosen (the default), no "Exam Record" is generated. If the key `coverpagesumry` does not appear in the option list of `eqexam`, no "Exam Record" is written.

See 'The Exam Record' on page 78 for more details on this topic.

`nospacetowork` When the `nosolutions` option is taken, the `solutions` environment leaves vertical space in which to respond to the question. Use this option to override this behavior.

The command `\SpaceToWork` causes the white space to be created again, and the `\NoSpaceToWork` turns it off again. Use these two commands to turn on and off the creation of vertical spaces in different parts of your exam.

`answerkey` This is a convenience option equivalent to `proofing` and `solutionsafter`. Useful for creating an "answer key" with answers and solutions displayed.

`solutionsonly` Using this option, it is possible to obtain a typeset document consisting of only the solutions to `eqexam` document. See 'The `solutionsonly` Option' on page 37 for further details.

`vspacewithsolns` An unusual feature requested by a user for homework assignments. This option is a combination of `nosolutions` (where vertical space is left by the

solutions environment to respond to the question), but the solutions are written the \jobname.sol and input back in at the end of the document.

This option is incompatible with answerkey, nosolutions, and solutionsafter, so when vspacewithsolns is used, it "cancels" these other options.

The command \showAllAnsAtEnd is inserted at the top of the \jobname.sol file and when \jobname.sol are input back in, the command \showAllAnsAtEnd is executed. The definition of \showAllAnsAtEnd is

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

Basically, this turns on all the switches that correspond to the answerkey option. This command may be redefined to suite your purposes.

☛ When this option is taken, the switch \ifvspacewithsolns may be used to bring in alternate content.

See 'The vspacewithsolns Option' on page 37 for further details.

ftbsolns An alias for vspacewithsolns.

useforms Multiple choice questions have two forms, (1) the choices are labeled using letters (a), (b), (c), etc.; or (2) using a rectangular fill box. The default is (1). The useforms switches the default to (2). You can use the commands \sqLinks and \sqForms to change back and forth between these two types within the exam document. Using one of these commands outside a problem environment will globally change the default, from within, it will only change the default locally.

flextended When this option is taken, additional code is input to support filler lines, refer to Section 10.12 for details; in particular, read 'On the flextended option' on page 72.

myconfig If this option is taken, eqexam looks for the configuration file eqexam.cfg. This configuration file is input at the end of the package, and can be used to redefine, for language localization purposes, any of the (text) macros described in this manual. See the section 'Customizations' on page 43 for a partial listing of macros that can be redefined and placed in eqexam.cfg.

myconfigi…myconfigvi Six additional options for inputting a configuration file. If you take one of these options, eqexam inputs the corresponding configuration file eqexami.cfg…eqexamvi.cfg.

cfg Syntax: cfg=⟨*basename*⟩. If this option is taken, eqexam looks for a file named ⟨*basename*⟩.cfg and is input.

For one of my recent classes, I wrote many standard handouts documents: first day handout, assignment documents, homework assignments, review documents, test documents, and in-class notes. Each document-type had its own eqexam format (configuration file, eqexami.cfg...eqexamiv.cfg. It got confusing to keep track of all these configuration files. At which point I decided to add a *named* configuration scheme. If you use the key cfg in the option list cfg=firstday, eqexam will look for a file named firstday.cfg

obeylocalversions An option put in to give greater control over versions. Perhaps you have a eqexam file that has questions with multiple versions. You would like to pick and choose the versions to be used. In this case, using obeylocalversions will cause eqexam to obey any \selectVersion commands embedded in the document.

allowrandomize Use this option to randomize the multiple choice/selection questions. See 'Randomizing Choices' on page 24 for details.

showgrayletters When showgrayletters is used, multiple choice questions will have a gray capital letter A, B, C, etc. underneath it. This letter can then be referred to in the text or the solution using the \REF command.

See 'Referencing Multiple Choice Questions' on page 61 for more information.

usexkv When this option is used, and the document author has the xkeyval package on his/her system, there is a re-definition of the \fillin command. For more information, see 'Extending the \fillin Command' on page 63.

allowcirc4mc Allows to use of circles (created by the lcircle10 font) for multiple choice questions. See 'Using Circles for Multiple Choice' on page 21 for details.

- The next option concerns a major new feature, available in eqexam, version 3.0 or later.

fortextbook This option is designed support textbook authors. See 'The fortextbook Option and fortextbook Package' on page 37 for a greater explanation.

- The next two apply to files that have several versions in them, these were defined for use by the AeB Exam Builder utility,[7] but they are available to the document author.

max The value of max, max=⟨*N*⟩, is a positive integer. The value of max is the number of versions for this document. This option executes \numVersions{⟨*N*⟩} at the end of the package.

rendition This is a key-value pair. rendition=⟨*alpha*⟩, where ⟨*alpha*⟩ letter corresponding to the version that is to be typeset. At the end of the package, the command \forVersion{⟨*alpha*⟩} is executed.

- The next four options require the AcroTeX Bundle, and all of its required packages, such as hyperref, their use implies you are going to publish the document as a PDF.

[7] http://www.acrotex.net/builders/

`pdf` This option doesn't do much, it brings in the `web` package, which in turn, places the values of the keywords (`\title`, `\author`, `\subject`, etc.) into the Document Description dialog of the PDF.

`links` This option brings in both `web` and `exerquiz`. When you do not use a solutions option (`nosolutions` and `solutionafter`), the solutions appear at the end of the document. When the `links` option is used, links from the questions to the solutions are created. Unless you use a "paper option" (`forpaper` and `forcolorpaper`), each solution is on a different page, making a document with a lot of pages. When you also specify a paper option, the solutions are separated by a `\medskip`.

`online` The `online` option implies the previous two options, but does more. When this option is taken, and the `nosolutions` option is specified, PDF forms are created: multiple choice questions become radio button fields; fill-in questions become text fields, and the vertical space created by the `solutions` environment become multi-line text fields.

This may be a useful option for an exam taken in a CBT[8] lab, where the students can type in their responses and when finished, print the document to a lab printer to hand in.

`email` This option implies the `online` option, in addition, adds a submit button to the upper left corner of the first page of the exam. The student can take the test in a CBT lab, then submit the results to the instructor via email.

See the section 'The `online` and `email` Options' on page 38 for additional details of these last two options.

▶ When any one of the four options above are taken, a driver needs to be specified as well, the choices are…

`dvips` For users of dvips, the dvi-to-postscript converter.

`dvipsone` For users of the Y&YTEX System, such as myself.

The drivers pdftex, luatex, and xetex are automatically detected and need not be specified as a driver option.

The driver names are passed on to hyperref and to eforms[9] for the proper creation of links and form fields.

8.1. Configuration Files

The eqexam looks for two configuration files, they are `web.cfg` and `eqexam.cfg`.

The first one `web.cfg` may be already present on your hard drive if you use the AcroTEX Bundle. Typically, desired default driver option is placed in here, for example, `web.cfg` might contain the single line,

[8] Computer Based Testing
[9] A component of AcroTEX Bundle.

```
\ExecuteOptions{dvips}
```

for users of the dvips application for converting .dvi files to .ps file. The drivers supported by eqexam are listed in the previous section.

The second configuration file, eqexam.cfg, is input at the end of the package, provided the document author takes the myconfig option. Use this file to redefine some of the commands described in 'Customizations' on page 43, and elsewhere, to customize eqexam. An obvious use for this is to have a language customization of the package, input through eqexam.cfg.

If you place eqexam.cfg in the LaTeX search path, these customization will be global to all documents that specify the myconfig option. If is is placed in the source document folder (which is not in the LaTeX search path) the changes are local to all documents developed in that folder.

8.2. The solutionsonly Option

With this option, it is possible to obtain a listing of only the solutions in an eqexam source file. A possible application of this feature is if you publish homework or practice test questions, you can later publish the solutions to them.

The demo file for this option is eqex_solnsonly.pdf, available from the AcroTeX Blog website. The source file is attached to the PDF. Within the source file, you will find detailed instructions for how to create a solutions-only file.

8.3. The vspacewithsolns Option

With option is a combination of nosolutions (where vertical spaces are left for extended response questions, and multiple choice and fill-in the blank are left, well, blank) and compiling the document with no options at all, in this case the solutions appear at the end of the document.

To summarize, when vspacewithsolns is used, the test section is left blank for the student to fill in, but at the end of the document are the solutions. I've recently used this option to compile an old test (from a previous semester) and publish it on the web. The student can try solving the old test, with the solutions at the end of the document.[10]

An alias for this option is the option ftbsolns.

The demo file for this feature is named hw02.pdf. See also the file test03.pdf, which demos both the solutionsonly option and the vspacewithsolns option.

8.4. The fortextbook Option and fortextbook Package

This option is designed for authors of textbooks. The fortextbook option defines the probset environment—used to create problems sets in the textbook—as a re-purposing of the exam environment. When the text is compiled with the studented option (student edition), only odd-numbered solutions/answers are written to the end of the document;

[10] Another option is to first publish your old exam with the nosolutions option, then, after a suitable time, publish the same document with the solutionsonly option.

when the `instred` option (instructor edition) is used, all solutions/answers are written to the end of the document, there are options for annotating the book with the answers in the margins or following the questions (instructor edition).

Documentation for this option is found in the `doc/fortextbook` folder. See the file `fortextbook.pdf` and its source file `fortextbook.ltx`.

To use the `fortextbook` option, a whole panoply of options are needed,

```
\usepackage[%
    fortextbook,ftbsolns,usecustomdesign,
    forcolorpaper,noseparationrule,usexkv
]{eqexam}
```

Accompanying the `eqexam` is a simple wrapper package called `fortextbook`, which basically calls `eqexam` with all the above options. So, the textbook author needs only to specify,

```
\usepackage{fortextbook}
```

The documentation for this option is quite extensive and is available in the separate document `fortextbook.pdf`. This document is a short "textbook" that illustrates and documents the features of this option.

See the **AcroTEX Blog** for several articles on the `fortextbook` option, beginning with the first article *The `fortextbook` option, Part 1, The Instructor Edition*. A listing of all articles of the `eqexam` package may be obtained by following this link.

The full series of articles on the `fortextbook` option may be found under the fortextbook tag at the blog site.

9. The `online` and `email` Options

When you use the `online` option, all fields created by the `\fillin` command, and this includes `\TF`, are converted into text fields, and the white space created by the `solutions` environment is converted to a multi-line text field. The fields manifest themselves when the document is viewed within the Adobe Reader, or any other PDF viewer that supports form fields.

This may be a useful option to the few people out there who are not in a technical field that requires specialized symbols to respond to a question. An exam created by the `online` option can be filled out online, printed, and submitted to the course instructor, perhaps within a lab setting.

There are other applications, such as creating a course survey, or a questionnaire of some type the students can fill out and submit. The `email` option may be more appropriate for these applications.

9.1. The `email` Option

When you pass the `email` option to `eqexam`, this does everything the `online` option does, in addition, it creates a "Submit" button that appears in the top-left margin of the exam (it does not appear on the cover page), and is placed there by the `\maketitle`

command, that normally goes just after the opening of the document environment, \begin{document}.

The forms button is all setup to submit to the server-side script, eqAttach.asp, an active server page using vbscript as its scripting language. This script, when properly installed and functional, receives the form data generated by the document and attaches it to an email, which it sends off to the designated destination. Before discussing how to install and use eqAttach.asp, let me cover some commands that controls this button as well as options for changing what is sent to the server-side script.

When you take the email option, you need to supply a minimum of two pieces of information: the path to the server-side script eqAttach.asp and the email address of the person the results are to be sent. The command \SubmitInfo is used to supply this info, for example,

```
\SubmitInfo{http://localhost/scripts}{dpstory@uakron.edu}
```

This command takes two arguments, the first is the URL to the server-side folder that contains eqAttach.asp, the second argument is the email address of the recipient of the email. (You can have multiple recipients by separating the address by an comma.)

After the student submits the responses to the questions, an email is sent to the recipient (the instructor, perhaps). When the recipient receives the email, s/he can save the FDF attached file (containing the student responses) to a folder on the local hard drive. At least for a Windows machine when you open the FDF, the PDF will be fetched and the student data will be populated into the form fields.

Once this is done, the instructor can either save the populated file to the hard drive for later processing (the Acrobat application needed for this step) or print it to a printer for grading by hand.

If the instructor has Acrobat, s/he can use the markup capability of Acrobat to grade the electronic version of the test, and return the electronic version, with markup, to the student.

Below is the subject and message body of a “typical” submittal for the student “John Q. Student”.

Message Subject:

```
Exam Results: Test 1 of U. S. History
```

Message Body:

```
Exam Information:
    Course Name: U. S. History
    Exam: Test 1
    Student: John Q. Student
    TimeOfQuiz: 1/19/2005 12:07:56 PM
```

The FDF is attached.

The following commands can be used to modify the email message.

- \EmailCourseName is used to specify the 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.

```
\EmailCourseName{\websubject} % the default
```

Important: When you use TEX formatting in the subject, such as

```
\subject{\bfseries Calculus 1}
```

and you are using the email option, it will be necessary to use \EmailCourseName to redefine the subject, e.g., \EmailCourseName{Calculus 1}, to avoid possible TEX compile errors, or to prevent TEX primitives being a part of your email!

- \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.

```
\EmailExamName{\webtitle} % the default
```

Important: If you use some TEX formatting in the title, such as

```
\title{\bfseries Test 1}
```

and you are using the email option, it will be necessary to use \EmailExamName to redefine the title, e.g., \EmailExamName{Test 1}, to avoid possible TEX compile errors, or to prevent TEX primitives being a part of your email!

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

```
\EmailSubject{} % the default
```

In this case eqAttach.asp inserts the standard one.

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

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

To change the email subject we would put the following command in the preamble:

```
\EmailSubject{Another Set of Cool Results}
```

- \ServerRetnMsg The server script (eqAttach.asp) returns a message acknowledging the receipt of the data, this command allows the document author to customize the return message. The default definition is:

```
\ServerRetnMsg{}
```

In this case eqAttach.asp inserts the standard one, "Exam results successfully sent to your instructor!".

To change the return message to something more meaningful, put this command in the preamble, for example,

```
\ServerRetnMsg{Your responses to the \\TeX Survey have been
    received, thank you!}
```

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

```
\SubmitButtonLabel{Submit} % the default
```

- **Installing eqAttach.asp**

On the server side, in order for eqAttach.asp to run correctly, Microsoft Internet Information Server (IIS), version 4.0 or greater, is needed. The script eqAttach.asp needs to be placed where ASP scripts have execute permissions.

The eqAttach.asp uses the *Acrobat FDF Toolkit*[11], version 6.0. Follow the directions for installation contained in the accompanying documentation.

Install eqAttach.asp in a folder (perhaps called Scripts) designated to execute scripts. If you don't have such a folder, then the following steps explain how to create a virtual directory through IIS that points to this folder.

1. Create a new folder on the system (Scripts, for example). Its recommended location is inside the Inetpub folder.
2. Place eqAttach.asp in this newly created folder.
3. In the MMC snap-in for IIS, create a virtual directory by right-clicking on the Default Web Site and selecting New > Virtual Directory.
4. Type "Scripts" (or whatever the name of the folder you created in Step 1) as the alias for the virtual directory, and then link it to the physical directory you created in Step 1.
5. Make sure that "Script execution" privileges are enabled. If not, enable them.

- **Setting up and Modifying the Script**

On the server side, in order for eqAttach.asp to run correctly, Microsoft Internet Information Server (IIS), version 4.0 or greater, is needed. The script eqAttach.asp should be placed where ASP scripts have execute permissions. There are two methods of sending e-mail:

1. CDONTS: This method (which is commented out by default) can be used on an NT server. Uncomment if you want to use CDONTS, and comment out the CDOSYS code lines that follow.
2. CDOSYS: This can be run on a Win2000 or WinXP server.

The script needs to be modified appropriate to your server, in particular, search down in eqAttach.asp for the configuration line

```
eqMail.Configuration.Fields.Item
    ("http://schemas.microsoft.com/cdo/configuration/smtpserver")
        = "mySMTP"
```

replace mySMTP with your SMTP server.

[11] Currently located at the Acrobat Family Developer Center.

- **Some Options**

The default behavior of eqAttach.asp is to return a message to the document that indicates the receipt of the data, this message is"Exam results successfully sent to your instructor!" The message, as explained earlier, can be changed using the \ServerRetnMsg, like so

```
\ServerRetnMsg{Your TeX survey results have been received, thank you.}
```

Now, if for whatever reason you don't want this confirmation message to return to the document for display in alert box, you can sent the silent as part of the query string. For example, if

```
\SubmitInfo{http://myWebSite/scripts/eqAttach.asp?silent\#FDF}
    {myname@mymailprovider}
```

placed in the preamble of your document specifies the path to the script, silent mode, and the email address of the recipient of the form data.

Another other feature of eqAttach.asp that can be changed through the query string is the /F key-value pair of the FDF sent out in email. The value of this key is the path to the document that sent the FDF, it may be a url (an address on the Internet) or it could be a file specification of a local hard drive. If you specify nopath in the query string, like so

```
\SubmitInfo{http://myWebSite/scripts/eqAttach.asp?nopath\#FDF}
    {myname@mymailprovider}
```

then eqAttach.asp strips out the file path and leaves only the file name.

▶ This is what I did with the tex_survey.tex source file. I placed tex_survey.pdf in a LaTeX Survey folder on my desktop. As the emails came in, I saved the FDF attachments to this folder. By (double) clicking on the FDF, tex_survey.pdf, which is in the same folder, opened and the form data populated the fields from whence they were sent. It worked well for me.

If you don't use the nopath option, when you click on an FDF file you've received by email, your browser opens and the PDF on the Internet is brought into the browser and the form data populates the form fields, …at least on a Windows machine. :-)

- **References**

The following links were used as a reference in the development of the Email.asp script.

- CDOSYS:
  - Invision Portal Tutorial: CDOSYS email tutorial
  - MSDN: CDO for Windows 2000. The IMessage Interface. (Use MIE to view this page.)
  - ASP 101 Sending Email Via an External SMTP Server Using CDO
- CDONTS
  - Juicy Studio The ASP CDONTS Component
  - DevASP Sending Mail from ASP with CDONTS.NewMail Object

10. Bells, Whistles and other Customizations

10.1. Customizations

We enumerate some commands for changing the default design of eqexam.

- **Course Info Commands**

eqexam has several commands for the student to provide some identification information.

- \eqexamName. This command defines the macro \eq@ExamName that creates the underlined space for the student to enter his/her name, and also defines the text box form field, in the case the online or email options are taken. There are two (design) parameters for \eqexamName

```
\eqexamName[eforms_opts]{width}
```

The first optional parameter can be used to modify the appearance of the text field, see the eForms documentation for details. The second parameter is the width of the field. The default definition is

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

Here, the text field that will be generated (when online or email is specified) will be a required field. The total width of the space provided is 2.25 inches.

The command \examNameLabel controls the label to be used for this name field. It takes one parameter, the label to be used for the name field; the default definition is \examNameLabel{Name:}.

- \eqSID. This command defines the macro \eq@SID that creates the underlined space for the student to enter his/her student Identification number (SID), and also defines the text box form field, in the case the online or email options are taken. There are two (design) parameters for \eqSID

```
\newcommand\eqSID[eforms_opts]{width}
```

The first optional parameter can be used to modify the appearance of the text field, see the eForms documentation for details. The second parameter is the width of the field. The default definition is

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

Here, the text field that will be generated (when online or email is specified) will be a required field. The total width of the space provided is 2.25 inches.

The command \examSIDLabel controls the label used for this SID field. It takes one parameter, the label to be used for the name field; the default is \examSIDLabel{SID:}.

- \eqEmail. This command defines the macro \eq@Email that creates the underlined space for the student to enter his/her student email address, and also defines the text box form field, in the case the online or email options are taken. There are two (design) parameters for \eqEmail

```
\newcommand\eqEmail[eforms_opts]{width}
```

The first optional parameter can be used to modify the appearance of the text field, see the eForms documentation for details. The second parameter is the width of the field. The default definition is

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

Here, the text field that will be generated (when `online` or `email` is specified). The total width of the space provided is 2.25 inches.

The command `\examEmailLabel` controls the label to be used for this email field. It takes one parameter, the label to be used for the name field; the default definition is `\examEmailLabel{Email:}`.

- **Changing the Title and Cover Page**

`\maketitle`. The main heading that appears at the top of the first page of the exam is created by the LaTeX (redefined) command `\maketitle`. The `\maketitle` has some code to place the email button in the top margin, followed by the expansion of the command `\maketitledesign`, whose definition is

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

This command can be redefined using `\renewcommand` to suite your needs, for example,

```
\makeatletter
\renewcommand\maketitledesign
{%
    \makebox[\textwidth]{\normalsize
        \shortstack[l]{\strut\websubject\\\webauthor, \@date}\hfill
        \shortstack[l]{\webtitle\\\strut\@altTitle}\hfill
        \shortstack[l]{\strut\eq@ExamName\\\eq@SID}}%
}
\makeatother
```

This code adds in a field for the student to enter his/her student Id, here we enclose the code in a `\makeatletter`/`\makeatother` because this redefinition occurs in the preamble, and the code has an '@' in it.

Command elements that are appropriate to the redefinition are `\maketitledesign` are…

`\websubject` This is the course name, as determined by the `\subject` command.

`\webtitle` This is the exam name as determined by the `\title` command.

`\altTitle` An additional text field that is placed below `\webtitle`.

`\@date` This is the date as determined by the `\date` command.

`\eq@ExamName` This is the name field for the student to enter his/her name, as defined by default or redefined by `\eqexamName`, see 'Course Info Commands' on page 43.

`\eq@SID` This is the student ID field for the student to enter his/her ID, as defined by default, or redefined by the command `\eqSID`, see 'Course Info Commands' on page 43.

`\eq@Email` This is the student email field for the student to enter an email address, as defined by default, or redefined by `\eqEmail`, see 'Course Info Commands' on page 43.

`\theduedate` This is a text macro defined by the `\duedate` command. For example, setting `\duedate{03/10/05}` defines `\theduedate` so that it expands to `03/10/05`. May be useful when redefining `\maketitledesign` for a homework assignment page.

`\eqexcoverpagedesign`. When the `coverpage` option is taken, a default cover page appears unless it is redefined. The `eqexam` package provides `\eqexcoverpagedesign` to design your own cover page. The default cover page uses the

`\websubject` This is the course name, as determined by the `\subject` command.

`\webtitle` This is the exam name as determined by the `\title` command

`\webuniversity` This is the value set by the `\university` command, given in the preamble.

`\@date` This is the date as determined by the `\date` command.

`\eq@ExamName` This is the name field for the student to enter his/her name, as defined by default or redefined by `\eqexamName`, see 'Course Info Commands' on page 43.

`\eq@SID` This is the student ID field for the student to enter his/her ID, as defined by default, or redefined by the command `\eqSID`, see 'Course Info Commands' on page 43.

`\eq@Email` This is the student email field for the student to enter his/her email address, as defined by default, or redefined by `\eqEmail`, see 'Course Info Commands' on page 43.

Copy the definition of `\eqexcoverpagedesign` from `eqexam.dtx` and modify as desired. Place the new definition in the preamble (enclosed between `\makeatletter` and `\makeatother`) or in a custom style file. No special support for this design is offered, because a cover page can be designed in so many different ways.

Another command associated with the `coverpage` option,is the `\placeCoverPageLogo`, a simple command used 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.

- **Changing the Running Headers**

There are two running headers, one header for the exam itself, and another when the solutions are shown at the end of the document.

- Running Header for Exam. The commands `\lheadeqe`, `\cheadeqe` and `\rheadeqe` are used for defining the left, right, and center running headers. **Note:** these commands have been recently renamed, originally they were named `\lhead`, `\chead` and `\rhead`, but this conflicts with the `fancyhdr` package. If `fancyhdr` has not been loaded by the time `eqexam` is loaded, the eqexam definitions for `\lhead`, `\chead` and `\rhead` still hold. Generally, it is recommended that the new command be used, `\lheadeqe`, `\cheadeqe` and `\rheadeqe`.

1. `\lheadeqe{`*text*`}`

   Changes the left header text of the running header. This command defines an internal macro `\eq@lhead` that actually contains the text. The default is

   ```
   \lheadeqe{\shortwebsubject/\shortwebtitle}
   ```

2. `\cheadeqe{`*text*`}`

   Changes the center header text of the running header. This command defines an internal macro `\eq@chead` that actually contains the text. The default is

   ```
   \cheadeqe{-- Page \arabic{page}\space of \nPagesOnExam\space--}
   ```

3. `\rheadeqe{`*text*`}`

   Changes the right header text of the running header. This command defines an internal macro `\eq@rhead` that actually contains the text.

   The default is `\rhead{\eq@ExamName}`.

If you want to redesign the layout of the running header, here is the macro that the above components fill.

```
\newcommand\runExamHeader{\eq@lhead\hfill\eq@chead\hfill\eq@rhead}
```

- Running Header for Solutions. The components of the running header for the solutions pages occur, as above, on the left, center and right of each header are defined by the commands `\lheadSol`, `\cheadSol` and `\rheadSol`.

1. `\lheadSol{`*text*`}`

Changes the left header text of the running header. This command defines an internal macro `\eq@lheadSol` that actually contains the text. The default is

```
\lheadSol{\shortwebsubject/\shortwebtitle}
```

2. `\cheadSol{`*text*`}`

   Changes the center header text of the running header. This command defines an internal macro `\eq@cheadSol` that actually contains the text. The default is

   ```
   \cheadSol{-- Page \arabic{page}\space of \nPagesOnExam\space--}
   ```

3. `\rheadSol{`*text*`}`

   Changes the right header text of the running header. This command defines an internal macro `\eq@rheadSol` that actually contains the text. The default definition is `\rheadSol{SOLUTIONS}`.

If you want to redesign the layout of the running header, here is the macro that the above components fill.

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

- **Changing the Running Footers**

The default set up of `eqexam` is to use no running footers; actually, that's not quite right. `eqexam` places the command `\settotalsbox` in the footer; this command is the one that places the totals boxes, when requested.

`eqexam` defines three commands for the footer,

```
\lfooteqe{text}
\cfooteqe{text}
\rfooteqe{text}
```

where the *text* is placed at the left, center, and right of the running footer. The default for each is empty text. These three comprise the definition of `\runExamFooter`

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

The `\settotalsbox` and `\runExamFooter` then appear in the definition of `\@oddfoot` in the definition of the `eqExamheadings` page style.

```
\renewcommand{\@oddfoot}{\settotalsbox\runExamFooter}
```

When doing any re-definition of the running footers at the `\@oddfoot` command, be sure to include `\settotalsbox` on the *left side of the running footer*; otherwise, you will not have a totals box when you request one.

- **Exam Strings**

In this section we list a new commands that contain information about the exam.

`\nPagesOnExam`

The command `\nPagesOnExam` expands to the total number of pages in the exam.

`\nQuesInExam[`*exam_name*`]`

`\nQuesInExam` expands to the total number of questions in the exam. The command takes an optional argument, the *exam_name* (this is the name given the exam as the required argument of the `exam` environment). If the argument is not given, the name of the current exam is used (when executed within an `exam` environment). If `\nQuesInExam` appears outside an `exam` environment, the value of the optional argument needs to be specified.

There are several commands are useful for documents that have several `eqexam` environments, these are

```
\theGrandTotal
\totalForPart{exam_name}
\percentForPart{exam_name}
```

The first command sums the point totals for each of the `exam` environments. The latter two, each taking one argument, the name associated with the `exam`, reports the points for that `exam` environment and the percent of the total for that `exam` environment.

Below is a recent example taken from a final exam that I constructed for my class.

```
\begin{eqComments}[Final Exam:] (\theGrandTotal\space points) The
final exam has two parts:
    \textbf{Part I} (\totalForPart{InstrQuestions} points or
       \percentForPart{InstrQuestions} of the total points) consists of
       questions written by the instructor;
    \textbf{Part II} (\totalForPart{GenEd} points or
       \percentForPart{GenEd} of the total points) consists of
       questions provided by the Department of Mathematics.
\end{eqComments}
```

☛ The calculation of `\percentForPart` is done in one of two ways:

1. If the `fp` package is loaded, "floating point arithmetic" is used and results are rounded to the number of decimal points determined by `\nPctDecPts`, the default definition of which is `\newcommand{\nPctDecPts}{1}`. This command may be redefined to another nonnegative integer value.

2. Otherwise—if the `fp` package is *not loaded*—TEX's count registers are used, the percentage rounded to the nearest integer.

The calculations are made when the \maketitle command is expanded. If, for whatever reason, you are not using \maketitle, you can place the command that does the calculations, \EQEcalculateAllTotals, just after \begin{document}, and before the first use of \theGrandTotal and \percentForPart.

```
\firstPageOfExam{exam_name}
\lastPageOfExam{exam_name}
```

These two commands expand the page numbers of the beginning and the ending of the exam environment, respectively, with name *exam_name*.

- **Localization of Strings**

In this section we list various macros that expand to text appearing on an eqexam document. The default text is in English. These commands can be redefined to other English language phrases, or to other languages, and placed in the preamble of your document, or in one of the .cfg files.

- \examNameLabel: On each page of the exam, there is a place for the student to enter her/his name. \examNameLabel can be used to define the name label, the default is

  ```
  \examNameLabel{Name:}
  ```

- \examAnsKeyLabel: When the answerkey option is in effect, the line in which the student enters her/his name (labeled by \examNameLabel) is filled by the value of the text macro \examAnsKeyLabel. The default definition is

  ```
  \examAnsKeyLabel{Answer Key}
  ```

  Thus, when the answerkey option is used, the name field appears as follows:

  Answer Key _____

- \ptLabel and \ptsLabel: Labels for indicating the points of a problem, the first is the singular form of the second. The default is

  ```
  \ptLabel{pt}            % singular form
  \ptsLabel{pts}          % plural form
  ```

- \eachLabel: Label for indicating the common point value of each of several parts of the same problem.

  ```
  \eachLabel{ea.}
  ```

- \pointLabel and \pointsLabel: The word for 'points' used in the instructions environment that lists the number of points in this exam. The default is

  ```
  \pointLabel{point}      % singular form
  \pointsLabel{points}    % plural form
  ```

The \pointsLabel command defines \eq@pointsLabel, which, in turn, is used in the \summaryTotalsTxt, the definition of which follows:

```
\newcommand{\summaryTotalsTxt}
    {($\summaryPointTotal\,\text{\eqpointsLabel}$)}
```

- \defaultInstructions: The instructions environment has a default heading. The command \defaultInstructions allows you to change this heading. The default is

```
\defaultInstructions{Instructions.}
```

See 'The Point and Totals Boxes' on page 56 as well as the section 'Course Info Commands' on page 43 for additional details on these and commands useful for laying out the standard text of an eqexam document.

- **Customization of Color**

When the forcolorpaper option is used, various elements—such as section titles, instruction headers, color for fill-in problems, and so on—have default colors. In this section we list the color controls, along with their default definitions.

```
\proofingsymbolColor{red}
\instructionsColor{blue}
\eqCommentsColor{blue}
\eqCommentsColorBody{black}
\universityColor{blue}
\titleColor{black}
\authorColor{black}
\subjectColor{blue}
\linkcolor{blue}
\nolinkcolor{blue}
\fillinColor{red}
\forceNoColor
```

Description of Color Commands:

1. The color of the proofing symbol, it appear for multiple choice questions with the answerkey option in effect.
2. The color of the header text for the instructions environment. The header is the text that appears in the optional argument.
3. The color for the header text for the eqComments environment. The header is the text that appears in the optional argument.
4. The color for the body of the text for the eqComments environment.
5. The color of the university, visible only when the coverpage option is taken.

6. The color of the title, visible only when the `coverpage` option is taken.
7. The color of the author, visible only when the `coverpage` option is taken.
8. The color of the subject, visible only when the `coverpage` option is taken.
9. The color applied to a link, applies only when `hyperref` is included through one of the PDF options, `links`, `online`, or `email`.
10. The color applied to a link that has been turned off, applies only when `hyperref` is included through one of the PDF options, `links`, `online`, or `email`.
11. The color of the a fill-in (including a True/False question) when one of the options `answerkey`, `vspacewithsolns`, or `solutionsonly` is taken.
12. This convenience command forces all the above colors to black. Useful when you want to use the `showgrayletters`. This produces a black and white document, with gray letters. (If you use the `forpaper` option, the gray letters appear black.)

10.2. Creating Multiple Versions of Exam

Unfortunately, I teach multiple sections of the same course, and am faced with the problem of writing different exams for the same course each administered to a different section.

Typically, I only have a need for two variations on the test; however, further extensions can be made, if needed (See 'New Version Control' on page 53)

- **The Original Version Scheme**

The `eqexam` package defines a boolean switch, `\ifVersionA` for this purpose. The two sections of the same course are "Version A" and "Version B". The default is that you are preparing an exam for "Version A".

The command `\forVersion` sets the version: `\forVersion A` sets version to "Version A", and `\forVersion B` set the version to "Version B". (The argument of the `\forVersion` command is case insensitive, so you also type in `\forVersion b`.)

For small variations in text, there is the `\ifAB` macro,

`\ifAB{⟨Version A text⟩}{⟨Version B text⟩}`

for example, one could say,

```
\begin{problem}[2]
Compute $\frac{d}{dx}\ifAB{x^2}{x^3}$.
\end{problem}
```

For longer variations, the `comments` package is used to create comment environments that are included or excluded. The two environments are `verA` and `verB`.

```
\begin{problem}[2]
Compute $\frac{d}{dx}\ifAB{x^2}{x^3}$.

\begin{solution}[1in]
We use standard techniques:
\begin{verA}
$$
    \frac{d}{dx} x^2 = 2x
$$
\end{verA}
\begin{verB}
$$
    \frac{d}{dx} x^3 = 3x^2
$$
\end{verB}
\end{solution}
\end{problem}
```

There are several convenience macros for referring to the exams generated by the two variations.

Usually, an exam, test, homework assignment has a number associate with it, e.g. “Exam 1”, “Test 2”, “Assignment #12”, etc. This number should be defined using the `\examNum` macro.

`\examNum{`*num*`}`

where *num* is the number to be associated with the exam (test, assignment) under construction.

This command *must appear before* `\title` in the preamble. The command `\examNum` takes its argument and defines another macro `\nExam`, which has no arguments, but expands to *num*.

The `eqexam` package defines two commands `\Exam` and `\sExam` to automatically enter the test information for the current version. In the preamble, you can say,

```
\title[\sExam]{\Exam}
```

`\Exam` is the long version of the test name, and takes as its argument the exam number. `\sExam` is the short version, having no argument. Both `\Exam` and `\sExam` use the value determined by `\examNum`, described above.

The text of `\Exam` and `\sExam` are generated by the four commands,

1. `\VersionAtext{`*text*`}` This is the text for the long version of the exam name for “Version A”. The default text is

   ```
   \VersionAtext{Exam~\nExam--Version A}
   ```

2. `\VersionBtext{`*text*`}` This is the text for the long version of the exam name for “Version B”. The default text is

```
\VersionAtext{Exam~\nExam--Version B}
```

3. `\shortVersionAtext{`*text*`}` This is the text for the short version of the exam name for "Version A". The default text is

```
\VersionAtext{Exam~\nExam A}
```

4. `\shortVersionBtext{`*text*`}` This is the text for the short version of the exam name for "Version B". The default text is

```
\VersionAtext{Exam~\nExam B}
```

▶ All the above commands, 1–4, need to appear before `\title` in the preamble.

Below is a "typical" example of how to correctly redefine all the version text.

```
\documentclass{article}
\usepackage{amsmath}
\usepackage[forpaper,pointsonleft,noparttotals,nosolutions]{eqexam}

\examNum{1}
\forVersion{B}
\VersionAtext{Test~\nExam--003}
\VersionBtext{Test~\nExam--007}
\shortVersionAtext{T{\nExam}s3}
\shortVersionBtext{T{\nExam}s7}

\title[\sExam]{\Exam}
\author{D. P. Story}
\subject[C2]{Calculus II}
\date{Spring \the\year}
\keywords{Test \nExam, Section \ifAB{003}{007}}
```

- **New Version Control**

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

The steps for creating a multiple version `eqexam` document are as follows.

`\numVersions{`*num*`}`

In the preamble, declare the number of versions for this document using `\numVersions`, e.g., `\numVersions{3}`.

```
\longTitleText
    {⟨Text_1⟩}
    {⟨Text_2⟩}
    ...
    {⟨Text_n⟩}
\endlongTitleText
```

```
\shortTitleText
    {⟨Text_1⟩}
    {⟨Text_2⟩}
    ...
    {⟨Text_n⟩}
\endshortTitleText
```

Note: If there are more titles than what are declared, the rest of the titles are absorbed (gobbled). If there are fewer titles than declared, a LaTeX package error is generated, and "fake" titles are generated.

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

These two commands give values to `\Exam` and `\sExam`. If `\forVersion{a}` is executed, `\Exam` expands to the text `Test~\nExam--Version A` and `\sExam` expands to `T\nExam A`, using the example above. The value of `\nExam` is determined by the `\examNum` command, as described above.

Next is the command that does all the work. It creates alternate text macros for each of the versions declared using `\numVersions`. The syntax is

`\forVersion{`*letter*`}`

For example, assuming `\numVersions{3}`, `\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 Magna Carta.
```

Since `forVersion{a}` was declared, only the `\vA` text is displayed, the others are gobbled up. But wait, time out, 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 $\vA{x}\vB{y}\vC{z}$:
\[
\begin{verA}
       2x + 4 = 7
\end{verA}
\begin{verB}
       5y + 2 = 4
\end{verB}
\begin{verC}
       3z - 2 = 2
\end{verC}
\]
```

Here is a final example of the multiple version scheme, taken from the preamble of one of my Calculus tests.

```
\documentclass{article}
\usepackage{amsmath,graphicx}
\usepackage[forpaper,pointsonleft,nototals,nosolutions]{eqexam}
%\usepackage[forpaper,pointsonleft,nototals,answerkey]{eqexam}

\numVersions{3}
\forVersion{a}
\examNum{1}
\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

\subject[C3]{Calculus III}
\title[\sExam]{\Exam}
\author{Dr.\ D. P. Story}
\university
{%
            THE UNIVERSITY OF AKRON\\
    Department of Theoretical and Applied Mathematics
}
\date{\thisterm\space\the\year} % Fall 2005
\duedate{09/26/05}              % actual date of the test
```

```
% If you convert to pdf using a pdf (links, online, email)
% option, this will appear in the  keywords field of the
% document info dialog.
\keywords{\Exam, administered \theduedate}
```

There is one additional command that can be used to locally control which version that is typeset in the document.

`\selectVersion{`*num*`}{`*total_versions*`}`

You can place the `\selectVersion` command in front of a question or a part of a question that has multiple versions. Through this command you can select which version to typeset, provided the option `obeylocalversions` is set. For example,

```
\selectVersion{3}{4}
\begin{problem}[10]
...
\end{problem}
```

The `\selectVersion` command says there are four variations on the next question and the document author wants to use the third one (that would correspond to C, in the `\forVersion` command). Again, the `obeylocalversions` must be taken for `eqexam` to obey this command.

Recommendation: Each problem should have the command `\selectVersion` in front of it, even for parts. Suppose the document author says `\numVersions{5}`, but some problems don't have five versions, what do you do? If there is a `\selectVersion` in front of a problem with multiple versions, the `\selectVersion` will partially expand to determine if it is needed. It is needed if the version specified by `\forVersion`, is greater than the number of versions for the problem. In this case, `\selectVersion` performs modular arithmetic to compute which version is to be used. For example, if `\forVersion{E}` has been declared in the preamble, but a problem has only three variations, the `eqexam` will use variation B; if `\forVersion{D}` was declared, version A is used, and so on.

10.3. The Point and Totals Boxes

There are two types of points boxes, but only one type of totals box. All the commands listed below can be redefined for language localizations, for example.

- Points that appear in the left margin (the `pointsonleft` or `pointsonboth` options). There are two text macros that are used,

```
\newcommand\leftmarginPtsTxt[1]{(\small$#1^{\text{pts}}$)}
```

when the total points for that problem are shown, and the other

```
\newcommand\leftmarginPtsEaTxt[1]
    {(\small$#1_{\text{ea.}}^{\text{pts}}$)}
```

when the author indicates that each sub-part of a problem is weighted the same, (when the author begins a `problem*` environment with `\begin{problem*}[3ea]`, for example).

- Points that appear in the right margin (options `pointsonright` or `pointsonboth`). These points appear in the bottom half of a box, the text for that box is determined by the following definition.

```
\newcommand\marginpointsboxtext[2]{\small$#1\,\text{pts}$}
```

By the way, the purpose of the upper part of the box is for the instructor to enter the number of points a student received for that problem.

- Points specified by the `\PTs` command. This text is defined by `\itemPTsTxt` as follows. See the paragraphs on '`problem*`' on page 14 for a discussion of the use of `\PTs`.

```
\newcommand\itemPTsTxt[1]{($#1$ pts)}
```

- **The totals box.** When you specify either option `totalsonleft` or `totalsonright`, you get a page totals box appearing in the lower left or right bottom corner.

```
\newcommand\totalsboxtext{\small$\theeqpointsthispage\,\text{pts}$}
```

where `eqpointsthispage` is a counter whose value at the end of each page *should* be the page total. For tests that have multiple `exam` environments, if one exam part ends on a page, and another begins on the same page, this number (`eqpointsthispage`) is the total on the page from the beginning of the new exam part. In this case, at the end of the exam part, there should also appear a remaining total for that part on that page.

The totals boxes are located in the lower right corner (`totalsonright`) or in the lower left corner (`totalsonleft`). Vertical placement of these boxes is controled by the length `\eqevtranstotbox`, its default value is `0pt`. To raise the boxes `6pt`, for example, execute `\setlength{\eqevtranstotbox}{6pt}`; the box may be lowered by using a negative length.

- **Summary Totals.** When you use the `instructions` environment to give initial instructions for an exam, the total points appears automatically in the text, unless you specify the `nosummarytotals` option. This text is defined by `\summaryTotalsTxt`, whose definition follows:

```
\newcommand\summaryTotalsTxt{($\summaryPointTotal\,\text{points}$)}
```

where `\summaryPointTotal` is a macro that expands to the total for this `exam` environment.

10.4. The `eqComments` Environment

In addition to the `instructions` environment, as explained in the section 7.2, entitled '`The exam` Environment' on page 9, should you want to insert additional instructions from within the body or the exam, use the `\eqComments` environment. The `\eqComments` environment has one optional argument, a formatted heading for the comments you want to make. For example,

```
\begin{eqComments}[Proofs.]
Solve each of the problems~5--8 on a separate sheet of paper,
do not write on the back of the paper. Follow the instructions
provided for each problem. Use your little gray cells.
\end{eqComments}
```

Such instructions must go between problems, of course, not within the body of either a `problem` or a `problem*` environment.

The optional argument has a color associated with it, and is visible when you compile the document with the `forcolorpaper` option. `\eqCommentsColor` colors the text that may appear in the optional argument of the environment; `\eqCommentsColorBody` colors the body of the text. Each take a single named color argument.

```
\eqCommentsColor{blue}
\eqCommentsColorBody{black}
```

The above are the default definition.

10.5. The \OnBackOfPage Command

In order to reduce the number of pages needed for an exam, I often cheat by asking the student to work on the back of one of the test pages.

```
\newcommand\bopText{on the back of page~\boPage}
\newcommand\bopCoverPageText{(the cover page)}
\newcommand\OnBackOfPage[1][\bopText]{%
```

For this, I use the `\OnBackOfPage` command

`\OnBackOfPage[`*text*`]`

The optional argument allows you to enter variational text, text that varies from the default text. The default text is contained in `\bopText` macro, its definition is

```
\newcommand\bopText{on the back of page~\boPage}
```

where `\boPage` is the page the student is instructed to do the work. Thus, if you say, "Continue `\OnBackOfPage`." This would expand to "Continue on the back of page 2.", or whatever `\boPage` is determined to be.

To illustrate the use of the optional argument of `\OnBackOfPage`, you might say,

> `\OnBackOfPage[The back of page~\boPage]` can be used to continue work, if necessary.

This expands to "The back of page 2 can be used to continue work, if necessary."

The algorithm used to compute the page, `\boPage`, on which to continue to work is as follows: For all pages, except for the first page of the test, the student works on the back of the previous page. For the first page of the test, the student works on the back of the first page, unless there is a cover page, in which case the student is instructed to work on the back of that page.

In the case of working on the back of the cover page, there is a variation on the instructions, `\OnBackOfPage` expands to "on the back of page 1 (the cover page)". The phrase "(the cover page)" can be redefined using the `\bopCoverPageText` command. The definition of this command is

```
\newcommand\bopCoverPageText{\space(the cover page)}
```

We could change this as follows,

```
\renewcommand\bopCoverPageText{, the cover page}
```

so that it would now read, "on the back of page 1, the cover page". To remove this feature altogether, you could redefine as

```
\renewcommand\bopCoverPageText{}
```

10.6. The `\pushProblem` and `\popProblem` Commands

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 `multicols` environment below, we `\pushProblem`, then close `multicols`, we execute `\popProblem`, and then continue with the multi-parts in single column.

```
\begin{problem*}[\auto]
Do each of the following without error.
\begin{multicols}{2}
    \begin{parts}
        \item\PTs{3} This is a problem.
        \begin{solution}[1in]\end{solution}

        \item\PTs{3} This is a problem.
        \begin{solution}[1in]\end{solution}
\pushProblem
\end{multicols}
\popProblem
        \item\PTs{4} Do this harder problem.
        \begin{solution}[.5in]\end{solution}
    \end{parts}
\end{problem*}
```

In the example, the first two questions appear in two column format, while the third appears in single column format. The same thing can be done in reverse, like so:

```
\begin{problem*}[\auto]
Do each of the following without error.
    \begin{parts}
        \item\PTs{3} This is a problem.
        \begin{solution}[1in]\end{solution}
\pushProblem
```

```
\begin{multicols}{2}
\popProblem
        \item\PTs{4} This is a hard problem.
        \begin{solution}[1in]\end{solution}
        \item\PTs{4} Do this harder problem.
        \begin{solution}[1in]\end{solution}
    \end{parts}
\end{multicols}
\end{problem*}
```

Now, first question is in single column and the next two are in two column format.

☛ In order to get the correct formatting, the `multicols` environment must begin before the `parts` environment.

See `quiz02.pdf` for an example of `\pushProblem` and `\popProblem`.

10.7. \qNewPage, \aNewPage, and \promoteNewPage

The command `\qNewPage` (questions newpage) and `\aNewPage` (answers newpage) are convenience commands for creating new pages. The first one expands to `\newpage` when the `\ifanswerkey` is false and the second one expands to `\newpage` when the `\ifanswerkey` is true; their definitions are

```
\newcommand\qNewPage{\ifanswerkey\else\newpage\fi}
\newcommand\aNewPage{\ifanswerkey\newpage\fi}
```

The `\promoteNewPage` can be used to insert a page break, if needed.

`\promoteNewPage[`*vspace*`]`

If there is more than *vspace* left on the current page, a `\newpage` command is executed (a `\columnbreak` if in a `multicols` environment). The default value for *vspace* is `\@fvsizeskip\textheight`, where `\@fvsizeskip` is a decimal number between 0 and 1 that is set by `\fvsizeskip`. The default is `\fvsizeskip{.3}`. This means that if there is less than `.3\textheight`, and new page is created. This may be too large for most applications, so you can pass a *vspace* through the optional argument. For example, `\promoteNewPage[.25in]`.

10.8. Support for Solution Sets from a Text

I use `eqexam` not only for exams, quizzes and homework assignments, but also for solution sets for problems assigned from the text.

Suppose the assignment was to solve, on a certain page in the text, problems which include **2**, **6** and **12(b)(d)** and it is desired to provide solutions to these problem. For this purpose, `eqexam` provides `\forproblem` and `\foritem`. These commands are used as follows:

```
\begin{exam}{HW\nExam}
\begin{instructions}[Description] (10 points)
Selected solutions from Assignments~24, 25, \S7.1.
```

```
\end{instructions}

\begin{eqComments}[\S7.1]
\textbf{Solving Linear Equations}
\end{eqComments}

\forproblem{2}
\begin{problem}
Statement of problem.
\begin{solution}
Solution to this problem.
\end{solution}
\end{problem}

\forproblem{6}
\begin{problem}
Statement of problem.
\begin{solution}
Solution to this problem.
\end{solution}
\end{problem}

\forproblem{12}
\begin{problem*}
Statement of problem.
\begin{parts}
    \foritem{b} Statement for item (b)
    \begin{solution}
    Solution to this problem.
    \end{solution}

    \foritem{d} Statement for item (d)
    \begin{solution}
    Solution to this problem.
    \end{solution}
\end{parts}
\end{problem*}
\end{exam}
```

10.9. Referencing Multiple Choice Questions

When the `showgrayletters` option is used, each alternatives in a multiple choice question will have a gray capital letter `A`, `B`, `C`, etc. underneath it. This letter can then be referred to in the text or the solution.

The use of this option is global and is controlled by the switch, `\ifaebshowgrayletters`. The gray letter feature can be turned on and off locally: To turn on this feature, insert the command `\grayletterson` at some appropriately chosen point in the document; to turn off the gray letter feature insert `\graylettersOff`.

In support of the `showgrayletters` option is a new command `\REF`. `\REF` acts like the

LaTeX command `\ref` with the hyperref modifications, but it converts the reference to uppercase. When `\ref` would typeset the letter 'a', for example, `\REF` would typeset the letter 'A'. `\REF`, like `\ref`, typesets a hyperref link. Hyperref defines a * version of `\ref`; `\ref*` typesets the reference, but does not create a link; `\REF*` does the same. When `\aebshowgraylettersfalse` is in effect, `\REF` does not capitalize the reference.

- Below is an example of this.

```
\begin{problem}[5]
Answer this if you can!
\begin{answers}{2}
\bChoices
    \Ans0\label{testsqFirst} This is a possible answer.\eAns
    \Ans1\label{testsqSecond} Try this one (the correct one).\eAns
    \Ans0 This is an answer.\eAns
    \Ans0 Another alternative.\eAns
\eChoices
\end{answers}
\begin{solution}
We reference alternatives (\REF*{testsqFirst}), an incorrect answer,
and (\REF{testsqSecond}), the correct answer.
\end{solution}
\end{problem}
```

Notice that the gray letters was not turned to off until after the usage of `\REF`.

Important The gray letters are typeset into the document. Do not use a background color for the checkboxes as this color will cover up the gray letters. The default background color checkboxes is transparent, keep it that way.

When typesetting an exam for paper (using the `forpaper` option), the gray letters appear as black letters. If you want actual gray letters, you have to use the `forcolorpaper` option. In this case, you'll see the blue color appearing in various places. Change these blue colors to black using the following commands in the preamble:

```
\forceNoColor
```

10.10. Displaying Points between two Markers

Some instructors might like a subdivide the exam into segments (or parts) and to have a total for that segment of problems. The `eqexam` package offers three commands for that purpose.

```
\placeMarkerHere{name}
\calcFromMarkers[formatting]{name2}{name3}
\markerTotalFmt{formatting}
```

Place `\placeMarkerHere` outside of a `problem`/`problem*` environment, giving each a unique name; for example `\placeMarkerHere{bQForm}`. Place `\calcFromMarkers` wherever you wish a calculation to be displayed, for example,

```
\placeMarkerHere{bQForm}
\begin{eqComments}[Quadratic Formula\calcFromMarkers{eQForm}{bForm}.]
Solve each of the following equations using the quadratic formula.
\end{eqComments}
..
\begin{problem}[5]
...
\end{problem}
...
% Finished with problems that use the quadratic formula,
% now create another marker for the next set of questions.
\placeMarkerHere{eQForm}
...
```

After you LaTeX three times (and the totals are all brought up to date), the header of the **eqComments** should read **Quadratic Formula (12 points)**, where the **12 points** are the total of all points assigned between the `bQForm` marker and the `eQForm` marker.

The formatting for the total points between markers is determined by the optional first parameter of `\calcFromMarkers`, and if there is no optional first parameter, by a global command, `\markerTotalFmt`, which sets the default formatting. The default definition of `\markerTotalFmt` is

```
\markerTotalFmt{ (\themarkerCnt\space points)}
```

The command `\themarkerCnt` references the counter `markerCnt` in which the calculations are made. Any redefinition of `\markerTotalFmt` should use `\themarkerCnt` to reference to value.

You use the optional first parameter the same way as the definition of `\markerTotalFmt`. You can say, for example, you can type

```
\calcFromMarkers[ $\themarkerCnt^{\text{pts}}$]{eQForm}{bForm}
```

to get a formatted total `$12^{\text{pts}}$`, which typesets to '12^{pts}'.

You might have noticed that I've inserted a space character at the beginning of the definition `\markerTotalFmt{ (\themarkerCnt\space points)}`, and place `\calcFromMarkers` up against the previous word, as in

```
Quadratic Formula\calcFromMarkers{eQForm}{bForm}.
```

This is so that when the required totals are not defined—early in the LaTeXing process—there is no space between `Formula` and the period (.); this is nothing but a cosmetic trivial point. After you LaTeX enough times, the full expansion appears as,

```
Quadratic Formula (12 points).
```

10.11. Extending the `\fillin` Command

When the document author uses the `usexkv` option, and the `xkeyval` package is found on the document author's system, the `\fillin` command is redefined to use key-value pairs in the optional first argument. The syntax for `\fillin` now is,

```
\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⟩,
    enclosesoln=⟨true|false⟩,fitwidth=⟨true|false⟩
]{width}{answer}
```

Parameter Description. `\fillin` takes three parameters.

1. The first optional parameter uses a key-value system, these allow the document author to set the appearance and behavior of the field. The keys are described below.

2. The second parameter *width* is the amount of horizontal space to leave for the student to write in the response. For example,

   ```
   \fillin[boxed,boxsize=Large]{1.5in}{$2x - 3$}
   ```

   Yields the following results, depending on the option.

`nosolutions` option	`answerkey` option
	$2x - 3$

   If this parameter is left empty, the `eqexam` uses the *answer* parameter to determine the width of the field. The code

   ```
   \fillin[boxed,boxsize=Large]{}{$2x - 3$}
   ```

   gives a box the same width as the answer, $2x - 3$, which is not very wide in this case. On the test, (compiled with the `nosolutions` option) the student would see this box . Though we can typeset the answer into this box, it is too narrow for the student to write the answer into; a better strategy is as follows:

   ```
   \fillin[boxed,boxsize=Large]{\ifNoSolutions{1.5in}{}}{$2x - 3$}
   ```

   Now, when the file is compiled under different options, we obtain

`nosolutions` option	`answerkey` option
	$2x - 3$

   For the answer key version, the answer is enclosed is tightly enclosed, a cosmetic difference. See also the `fitwidth` option.

   The command `\ifNoSolutions{#1}{#2}` executes the first parameter if the compile option is `nosolutions`, otherwise it executes the second parameter.

3. The third argument, *answer*, is the correct answer; this correct answer appears when the document is compiled with the `answerkey` option.

The description of the key-value pairs for `\fillin`:

`underline`: A Boolean switch, which if `true`, the fill-in region is underlined. The default is `false`, the region is not underlined.

`u,b`: Legacy options. If `u` is chosen, the region is underlined, if `b` is chosen, the region is not underlined. Use of the `underline` key is recommended.

`boxed`: A Boolean switch, which if `true`, the fill-in region is boxed in using the `\boxed` command of `amsmath` package. The default is `false`, the region is not boxed.

`boxpretext`: A key that takes *`text`* as its value. This value will be placed in front of the third argument, labeled *`answer`* above. The *`text`* appears in the box even when the `answerkey` is not in effect. This key is ignored if the `boxed` option is not taken.

This option allows you to create an expression like

$y =$ and with `answerkey` $y = 2x^2 + 1$

`boxsize`: This is a choice key, the choices being `tiny`, `scriptsize`, `footnotesize`, `small`, `normalsize`, `large`, `Large`, `LARGE`, `huge` and `Huge`. the smaller sizes probably are not useful, I give them to you for free. This key allows you to adjust the height of the box. This key is ignored if the `boxed` key does not appear in the option list. For example,

`boxsize=Large` `boxsize=Huge`

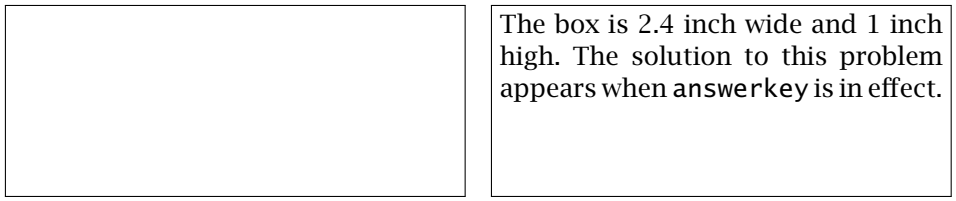
Choice of size depends on the height of the anticipated response of the student. The default is `normalsize`.

`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.

A simple example would be

```
\fillin[boxed,enclosesoln,parbox={[t][1in]}]{2.4in}{%
    The box is 2.4 inch wide and 1 inch high. The solution to
    this problem appears when \texttt{answerkey} is in effect.
}
```

Below shows how this command appears under different options.



The box on the left is appears when the `nosolutions` or `vspacewithsolns` is in effect. The one on the right appears when the `answerkey` option is taken. Note the size of the two boxes are the same.

Note: The explanation of the `parbox` option used in this example appears below.

`fitwidth` This key is an implementation of the `\ifNoSolutions` strategy discussed earlier. If `fitwidth` is specified and the `nosolutions` option (an `eqexam` option) is in effect, the width of the fill-in region is taken from the second parameter; however, if the `answerkey` option is specified, the natural with of the answer (the third parameter) is used.

`align`: A key that takes one of three values, `l`, `r`, and `c`. This key aligns the text within the fill-in field (when the `answerkey` option is taken): `l` (left-aligned), `c` (center, the default), `r` (right-aligned). This parameter affects the position of the *`answer`*, and does not affect the position of the *`text`*, which is aligned left, of the `boxpretext` key. The alignment becomes visible when the `answerkey` option is in effect.

`hiddenbox`: When the `boxed` option is used, `eqexam` uses an `\fbox` to enclose the content. The `hiddenbox` is used with the `boxed` option and it resets the lengths of `\fboxrule` and `\fboxsep` to `0pt`. In this way, some of the features (`enclosesoln` and `boxpretext`, for example) that apply to boxed content can be used without the surrounding boundary rule. See `parbox` for an example.

`parbox`: When this option is taken, `eqexam` encloses the third argument (the answer) in a `\parbox`. The value of the `parbox` option is the first three parameters of a `\parbox`; for example `parbox={[t][.5in][c]}` causes `\fillin` to enclose the content in a `\parbox` with a height of `.5in`. The width of the `\parbox` is taken to be the second parameter of `\fillin` macro. The third parameter of `\fillin` becomes `\parbox[t][.5in][c]{`*`width`*`}{`*`answer`*`}`. The value of the `parbox` key needs to be enclosed in braces, as just illustrated.

Here is an example of the use of this key-value pair:

```
\fillin[parbox={[t][.5in]},boxed,hiddenbox,
    boxpretext={\textbf{\textcolor{black}{Conclusion: }}},
    format=\normalfont,enclosesoln]{\linewidth}{%
    There is sufficient evidence at the $\alpha=0.05$ level to
    conclude that the mean microwave repair cost is less than
    \$100.}
```

If you say `\fillin[parbox,boxed,`*`options`*`]{`*`width`*`}{`*`answer`*`}` (`parbox` with the empty argument), translates into `\parbox{`*`text`*`}`, the height of the box is the natural height of the content.

Note the presence of the `hiddenbox` key, in this case, no rule or additional spacing appears around the box.

This typesets, when `answerkey` option is used as

Conclusion: There is sufficient evidence at the $\alpha = 0.05$ level to conclude that the mean microwave repair cost is less than \$100.

When `answerkey` is not used, the following is seen (by the student at test time):

Conclusion:

The same vertical space is used in both cases because of the `enclosesoln` option taken in this example.

If the `hiddenbox` key is removed, a proper boxed region appears,

Conclusion: There is sufficient evidence at the $\alpha = 0.05$ level to conclude that the mean microwave repair cost is less than \$100.

Similarly, when `answerkey` is not used, the following is seen,

Conclusion:

`color`: The value of this key is a named color. The *`answer`* appears in this color, when `answerkey` option is in effect. The default is red.

`format`: A general purpose formatting key. The value of the key can be most anything: `\bfseries` (to change font series), `\ttfamily` (to change font family), `\Large` (to change size of the *`answer`* and *`text`* (the value of `boxpretext`). Several formatting commands can appear as the value; thus, `format={\bfseries\Large}` makes the answer, when `answerkey` is in effect, appear in large bold font. The default is `format={\fillInFormatDefault}`, where the definition of the command `\fillInFormatDefault` is

```
\newcommand{\fillInFormatDefault}{\bfseries}
```

The next three keys apply when a `boxed` option is not taken, that is, they apply to the case of `u` (`underlined`) or `b` (blank). The keys were designed for the `underline` option, however.

`lift`: The syntax is `lift=length` This key lowers the underline without changing the baseline reference point of the content. For example, `lift=10pt` lowers the un-

derlining by `10pt`.

$\frac{1}{2}+\frac{1}{3}=$ $\frac{3}{4}$ $\frac{1}{2}+\frac{1}{3}=$ $\frac{3}{4}$

The expression on the right corresponds to

```
$...\fillin[lift=10pt]{1in}{\frac{3}{4}}...$
```

The underline is lowered to include the baseline of the denominator of the fraction.

`autolift`: The `autolift` key takes the guesswork out of the `lift`. When `autolift` is used, the depth of the content of the answer field. For example, with the `autolift` key, `\fillin[autolift]{1in}{\frac{3}{4}}` yields $\frac{3}{4}$. Now the fraction is exactly underlined.

`addtoautolift`: The `addtoautolift` key is a companion to `autolift`. If we assign `addtoautolift=3pt`, `3pt` additional lift is added to the amount of lift determined by `autolift`. For example,

`\fillin[autolift,addtoautolift=3pt]{.5in}{\frac{3}{4}}` yields $\frac{3}{4}$

the number three-fourths is underlined plus `3pt` more.

The keys are processed by an `\edef`, this allows you to define a command with your favorite options; for example, you can define

```
\newcommand{\myBoxOpts}{boxed,boxsize=Large,align=l}
```

then in the exam, type

```
$\frac{\sqrt{-18}}{\sqrt{6}}=\fillin[\myBoxOpts]{1.5in}{3\imath}$
```

`\fillin` boxed. When the `boxed` key is used, the `\boxed` command of `amsmath` is used; This command needs to be in math mode. If the `boxed` key is used and the `\fillin` command is in math mode, then `\fillin` puts the *`answer`* and `boxpretext` in math mode too; if `\fillin` command *not in math mode* (i.e., it is in text mode), `\fillin` typesets the *`answer`* and `boxpretext` in text mode. For example,

```
Let $ x = 4 $ and $ y = -3 $ and let $ z = 2x - 4y $ then
    $\fillin[boxed,align=l,boxpretext={z=}]{1in}{20}$
```

When `answerkey` is used, the above question appears as

Let $x = 4$ and $y = -3$ and let $z = 2x - 4y$ then $\boxed{z = 20}$

Here is an example in text mode

```
The first president of the US is
    \fillin[boxed,align=c]{1.5in}{Washington}
```

When `answerkey` is used, the above question appears as

The first president of the US is [Washington]

Here is an example of mixed mode

```
The width of a rectangle is 20 feet and length is 15 feet.
Find the area: $\fillin[boxed,align=l,
    boxpretext={\text{Area: }}]{1in}{300\,\text{feet}}$
```

The answer is typeset in math mode, but the value of `boxpretext` needs to be typeset in text mode. When `answerkey` is used, the above question appears as

The width of a rectangle is 20 feet and length is 15 feet. Find the area: [Area: 300 feet]

`\fillin` not boxed. The `\fillin` command will typeset the answer in math mode if `\fillin` is in math mode, and in text mode, otherwise. For example, each of the following typesets the same.

```
The area of a circle of radius $2$ is \fillin[u]{.5in}{$4\pi$}
The area of a circle of radius $2$ is $\fillin[u]{.5in}{4\pi}$
```

- **`\fillineol`: Fill-in to end-of-line**

The `\fillineol` is a variant of `\fillin`, its main feature is that the width parameter of the `\fillin` command is automatically calculated to the end of the line.

`\fillineol*{⟨phrase⟩}[⟨fillin-opts⟩]{⟨answer⟩}`

Calculus originator, Isaac [ ]

```
\fillineol{Calculus originator, Isaac }[boxed,align=l,boxsize=Large]{Newton}
```

The star-option allows ⟨*phrase*⟩ to contain verbatim text; in this case, the collectbox package is required.

The `\fillineol` command takes the ⟨*phrase*⟩, measures its width, and uses the result as the value of ⟨*width*⟩ in the `\fillin` command. Usually, the ⟨*phrase*⟩ occurs at the beginning of the sentence. However, that is not always convenient.

For this next example, the fill-in occurs at the end of the sentence, which wraps around to the next line. Enter the name of your professor for the course you are currently taking, his name is _____

The demo file for this feature is named `fillineol.pdf`, found on the AcroTEX Blog website.

10.12. Vertical Space Fill Types

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.[12]

The demo file for this feature is named `hw03.pdf`

There are two sets of commands.

```
\useFillerLines
\useFillerDefault
```

The first command sets the code to draw rule lines in the vertical white space, the second command reverts back to the default, plain vertical white space.

The `\setFillLinesFmt` command is used to set the design parameters of filler lines.

```
\setFillLinesFmt{KV-pairs}
```

There is an extensive list of key-value pairs.

Key-value pairs of `\setFillLinesFmt`. These key-values document-wide scope *if not declared within a group.*

`fltype=⟨line|dash|dots|blank|grid⟩` When the value of `fltype` is different from `grid`, the key determines the type of horizontal line drawn (including a "blank" line); when `fltype=grid`, the type of line drawn is determined by `gridtype`, described below. The default is `line`.

When `fltype=grid` and neither `bgonly` nor `outlineonly` are specified, a grid is drawn consisting of horizontal and vertical lines. The thickness of the lines drawn is determined by the length. The default is `\setlength{\flfboxrule}{.4pt}`. Any dotted line is unaffected by the value of `\flfboxrule`.

The rest of the keys are listed in alphabetical order. The `fltype` key is separated out due to its importance.

`align=⟨default|left⟩` This key sets the horizontal alignment of the filler lines; a key-value of `align=default` is the natural way of aligning the filler lines, while `align=left` shifts the lines to the left margin. The default is `default`, `align` is equivalent to `align=default`.

`bgcolor=⟨named_color⟩` The color of the background, when `bgonly` is in effect.

fltype=grid required

`bgonly*=⟨true|false⟩` When the key is set to true, `eqexam` colors the targeted region with the color determined by the `bgcolor` key. This key is *ignored* unless `fltype=grid`.

[12] The filler line feature has been rewritten and extended from previous versions of `eqexam`.

If `bgonly[=true]` is declared, `bgonly=false` must be declared later to turn off this effect, as `eqexam` cannot determine the intention. If the star-form is used (`bgonly*[=true]`), `bgonly` is automatically set to false after the next region is drawn. Use the `bgonly` key to create backgrounds for several consecutive regions. Use the `bgonly*` form of the key for a single region.

`equalcells=⟨true|false⟩` This key is only effective when `fltype=grid`, but the keys `bgonly` and `outlineonly` are both false. When the key `equalcells` (or the key-value `equalcells=true`) is specified, grid cells of equal width are drawn; the total width of a row may not be equal to the `\linewidth`, however. The key is initially set to `false`.

The command versions of this key may also be expanded explicitly; these are `equalCellSizesOn` and `equalCellSizesOff`.

`color=⟨named_color⟩` The value of the `color` key sets the color to be used when drawing filler lines (both horizontal and vertical).

`gridtype=⟨line|dash|dots⟩` The value of the key `gridtype` determines the line type when `fltype=grid`, otherwise, the value of this key is ignored. Values of `line` and `dash` obey the value of `\flfboxrule`.

the gap is \wlVspace

`linegap=⟨dimen⟩` The "gap" between two horizontal or vertical lines; the `linegap` key sets the length register `wlVspace` to ⟨*dimen*⟩. This key executes the command `eqWLSpacing{⟨dimen⟩}`. The default is 14pt.

`numbers=⟨none|left|right⟩` The `numbers` key controls whether the lines are numbered and their relative position to the line; the default is not to number the lines (`numbers=none`).

`numbersep=⟨dimen⟩` The value of `numbersep` sets the separation between the line and the number, the default is 2pt.

`numfmt=⟨num-fmt⟩` This key-value pair formats the line numbers. Within the value ⟨*num-fmt*⟩, the macro `\flnum` refers to the number itself; for example,

```
numfmt=\textcolor{red}{\flnum}
```

produces red line numbers. The initial declaration for this key is `numfmt=\flnum`, which creates a black line number.

font size of the numbers

The font size of the line numbers is `\tiny`, but this can be overwritten by incorporating a size into the `numfmt` (`numfmt=\small\flnum`).

fltype=grid required

`outlineonly*=⟨true|false⟩` If this key is true, an outline of the region is drawn using line type determined by `gridtype`. This key is *ignored* unless `fltype=grid`.

If `outlineonly[=true]` is declared, `outlineonly=false` must be declared later to turn off this effect, as `eqexam` cannot determine the intention. If the star-form is used (`outlineonly*`), `outlineonly` is automatically set to false after the next region is drawn. Use the `outlineonly` key to create backgrounds for several consecutive regions. Use the `outlineonly*` form of the key for a single region.

`topline=⟨true|false⟩` If true, `eqexam` draws an extra horizontal line above the topmost line. Any vertical lines are extended to meet this additional line.

The following is an example of filler lines under the `nosolutions` assumption.

1. Expound on all you know on the subject.

When the `flextended` option is not in force, these three lines are blank. However, in this example, `flextended` is active, and I can write to these lines. Refer to 'On the `flextended` option' below.

2. Expound on all you know on the subject.

The verbatim listing of this example is presented and discussed in the next section.

- **On the `flextended` option**

nosolutions
vspacewithsolns

The `flextended` option brings in code that extends filler lines. It defines one new environment `priorworkarea`, and several commands. One significant enhancement is the ability to write to the lined regions in the `nosolutions`, `vspacewithsolns`, and `answerkey` options are in effect.

The contents of the `priorworkarea` environment will be superimposed on the filler lines when the `nosolution` or `vspacewithsolns` option is in effect. It is placed prior to the `solution` environment.

```
\begin{priorworkarea}
  ⟨content⟩
\end{priorworkarea}
\begin{solution}[vspace]
  ⟨solution⟩
\end{solution}
```

For the ⟨*content*⟩ to be superimposed on the vertical space specified by the optional parameter *vspace* of the `solution` environment, `\useFillerLines` must be in force and `\turnflnosolnsOn` must be expanded prior to the statement of the problem. (Turn off the writing to the vertical space with `\turnflnosolnsOff`). The verbatim listing of the example above, which does use the `flextended` option as well as the `priorworkarea` environment is presented below.

```
\begin{exam}{fl1}\chngToNoSolns

\begin{problem}
Expound on all you know on the subject.
\begin{priorworkarea}\baselineskip\wlVspace
When the \texttt{flextended} option is not in force, these three
lines are blank. However, in this example, \textsf{flextended} is
```

```
active, and I can write to these lines. Refer to ...
\end{priorworkarea}

\begin{solution}[nLines=3]
Your guess is as good as mine. ...
\end{solution}
\end{problem}

\turnflnosolnsOff
\setFillLinesFmt{fltype=grid,gridtype=line,
  numbers=right,topline,color=lightgray}
\begin{problem}
Expound on all you know on the subject.
\begin{priorworkarea}
Essay area.
\end{priorworkarea}

\begin{solution}[nLines=3]
Your guess is as good as mine.
\end{solution}
\end{problem}
\end{exam}
```

answerkey

For the `answerkey` option, content of the `solution` environment is written to the vertical space allotted by is optional parameter, *vspace*. As with `priorworkarea` there is a gatekeeper command `\turnflanskeyOn`. For the content to be written to the vertical space, `\userFillerLines` and `\turnflanskeyOn` must be expanded. To turn off writing to the vertical space, expand `\turnflanskeyOff`.

The last example is repeated, but under the assumption of the `answerkey` option.

1. Expound on all you know on the subject.

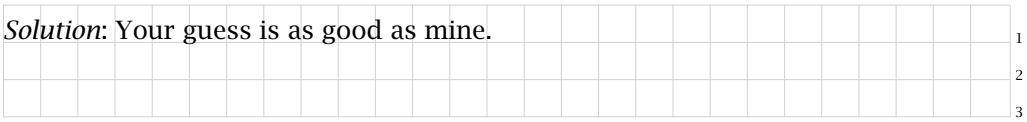
Solution: Your guess is as good as mine. The content has the capability of breaking across pages.

I've added more lines because this problem has a chance breaking across a page boundary. Let's prattle on until we go to the next page.

We'll jump down a couple of lines cause I don't have much to say. Gotta keep going to get to the next page. If all works as it should, I'll see you one the other side! Perfect! As mentioned in the 'Breaking across pages' below. There are limitations to this wonder.

2. Expound on all you know on the subject.

Solution: Your guess is as good as mine.



All in all, this is very cool.

- **Breaking across pages**

When the vertical space of the `solution` environment is filled with filler lines (assuming `\useFillerLines` is in force), *the lines and the content can break across pages*, provided the content is not within a box. It has limited capabilities of breaking across columns (for `fltype=line|dash|dots`), but when the `flextended` option is in effect, may fail dramatically at superimposing the content over the filler lines.

When `\turnContAnnotOn` is expanded prior to the problem, should the fill lines break across a page boundary, the annotation string (`annotContStr`) appears at the top of the next page. The topic of annotating solutions is taken up again in the next section.

- **Annotating a continuing problem with `\useFillerLines`**

When the command `useFillerLines` is expanded, any vertical space declared with the optional argument of `solution` is created one line at a time. Just before each of the lines is drawn, two commands are expanded, the first just prior to the a page break, if one occurs, and the second just after the page break, if one occurs. The second command is the internal version of `insertContAnnot`, the first command is accessed through the command `\priorPageBreakMsg`.

```
\priorPageBreakMsg{msg}
\flPageBreakMsg{msg}
```

The `\priorPageBreakMsg` command allows you to insert *msg* at the beginning of each line, but does not determine whether a page break is near. The `\flPageBreakMsg` command, which uses both `\priorPageBreakMsg` and `emitMessageNearBottom`, is the appropriate vehicle for writing *msg* just prior to a page break. Its definition is found next.

```
\newcommand{\flPageBreakMsg}[1]{%
    \priorPageBreakMsg{\emitMessageNearBottom[\iacvspace]
    {\eqfititin{{\Large\strut}#1}}}}
```

The command may be redefined for your needs. Notice that the star-option of the command `\emitMessageNearBottom` *is not taken*, so the command does not create a new page if near the bottom, but it does create a message (`#1`). The optional argument of `\emitMessageNearBottom` is set to `\iacvspace`, this is the interpretation of 'near the bottom' of the page. A recommended value for `\iavspace` is `2\wlVspace` (`\acvspace{2\wlVspace}`). A typical declaration might be,

```
% Set \iacvspace for \insertContAnnot and \emitMessageNearBottom
\acvspace{2\wlVspace} %<-- the recommended value
\flPageBreakMsg{\textbf{Problem~{\eqeCurrProb}(\thepartno)\space
    continues on next page}}
```

At the top of the next page the `\annotContStr` string appears as well.

10.13. Keep vertical space with `answerkey`

One thing that has bothered me in my efforts to create the "perfect" exam package, is the differences in vertical spacing between the exam the student sees (which is compiled with the `nosolutions` option), and the solutions document the instructor sees (as compiled with the `answerkey` option). The differences in vertical space of the two documents makes it harder for the eye to move from the student's exam to the answer key document and back again. In the past, I've used `\aNewPage` or `\qNewPage` to force page breaks so that the page breaks of the two document match, this makes it easier to get one page at a time.

Below is a simple example to illustrate the above points. In the example below, we simulate

```
\begin{solution}[1in]
$ x+1 = 4 \implies x = 3 $
\end{solution}
```

nosolutions

answerkey

Solution: $x + 1 = 4 \implies x = 3$

We have put an `\fbox` around the vertical space so you can see the bottom of the vertical space, and better appreciate the point I am trying to make. When `nosolutions` is in effect, a vertical space of 1 inch is created; when the `answerkey` is used, the vertical spacing is ignored, and the solution is typeset, as shown above. Notice the difference in the vertical spacing between the two.

The `eqexam` package now attempts to build the solution environment so the vertical spacing is (roughly) the same.

```
\vspacewithkeyOn
\vspacewithkeyOff
```

The first command turns on this new feature—the feature of trying to place the vertical spacing in the `answerkey` mode as requested by the optional argument of the `solution` environment. The second one, `\vspacewithkeyOff` (the default) turns off this new feature, and reverts to the old behavior of `eqexam`.

Here is a representation of the effects of the command `\vspacewithkeyOn`.

```
\begin{solution}[1in]
$ x+1 = 4 \implies x = 3 $
\end{solution}
```

nosolutions

answerkey

Solution: $x + 1 = 4 \implies x = 3$

As you can see, the vertical spacing with solutions is the same as without, the page breaks should be the same, and the positioning of the problems should be (roughly) the same throughout the test.

The commands `\vspacewithkeyOn` and `\vspacewithkeyOff` may be used anywhere (between problems or parts), but normally, one would put `\vspacewithkeyOn` in the preamble.

10.14. Annotating a Continuing Problem with Parts

The demonstration file for this feature is

Annotating a Continuing Problem with Parts, and Page totals

found at the **AcroTEX Blog**

A situation often encountered is when a problem with parts crosses a page boundary; the problem continues on the next page without any annotation. This feature works to insert a text string at the top of the next page, a string that gives the reader (student) the context of the part.

(Page 2)
4. Solve each of the following.

(a) Solve this

(b) Solve this

(Page 3)
Problem 4 continued.

(c) Solve this

(d) Solve this

Above is an illustration of how this feature works.

```
\turnContAnnotOff
\turnContAnnotOn
```

The feature may be turned off and on using the above two commands. These commands are usually in the preamble, but they can appear between problems or between `exam` environments.

The annotatation text that appears is determined by the definition of `\annotContStr`.

```
\newcommand{\annotContStr}{%
    \textbf{Problem~\eqeCurrProb\space continued.}}
```

where \eqeCurrProb is the current problem number. \annotContStr may be redefined, but be sure to include the problem number \eqeCurrProb.

The insertion of the continuation annotation uses the \promoteNewPage command, see Section 10.7 on page 60 for a description of this command.

```
\acvspace{vspace}
\resetacvspace
\newcommand{\ic@vspacedefault}{1in}
```

The above three lines describes some commands for controlling the generation of a new page through the use of \promoteNewPage. The first one \acvspace defines the value of the length \iacvspace, which is used to set the option parameter of \promoteNewPage. The default value for promoting a new page is 1in, as defined by \ic@vspacedefault, which can be redefined. The command \resetacvspace resets the optional argument of \promoteNewPage back to its default. All definitions are local, so the revert to their original values when a group is exited.

\iacvspace

for items in a parts env.

Automatic annotation. Just prior to an \item within a parts environment, the command promoteNewPage is expanded with optional argument of \iacvspace in this case. This command calculates the amount of space remaining on the page, if it is less than \iacvspace, a \newpage is emitted and \annotContStr is typeset at the top of the next page. As a result of this algorithm, if there is more space than \iacvspace, but a page break occurs by the TEX page breaking algorithm, the continuation string does not appear.[13] If you don't get the continuation annotation at the page break, change \iacvspace (possibly to a smaller value) through its \acvspace interface command.

When the useFillerLines command is used, there are additional automatic automation commands, refer to the paragraph **Annotating a continuing problem with \useFillerLines** on page 74.

Manual annotation. There are two commands for manually inserting annotations into the document.

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

For both commands, the default value for the optional argument *vspace* is \iacvspace. Both commands start with a \par, meaning they must be in vertical mode to work, so they should not be used within a paragraph. Each command determines the amount of space left (*spaceleft*) on the page.

For \insertContAnnot, if *spaceleft* < *vspace*, a \newpage command is expanded and the continuation string \annotContStr is inserted at the top of the next page; otherwise, it does nothing.

[13] Remember, after each \item a solution environment may request vertical space for the student to do his work.

The action of \emitMessageNearBottom is similar but it *does not insert* start a new page (\newpage) unless the star-option (*) is specified. If *spaceleft* < *vspace*, then *msg* is typeset (at the bottom of the page). If the star-option is specified, a \newpage is invoked and \annotContStr is typeset at the top of the next page.

10.15. The Exam Record

The demonstration file for this feature is

The Exam Record on the Cover Page

found at the **AcroTEX Blog**

When choose the coverpage option and the coverpagesumry option is set to either byparts or bypages, you get an **Exam Record** appearing on the cover page. See Figure 1, page 33.

There are several commands that can be used to customize the layout of the **Exam Record**

```
\eqeSumryHoriz
\eqeSumryVert
```

These two commands arrange the **Exam Record** relative to the student/instructor information. The former is a horizontal arrangement, the latter is a vertical arrangement, \eqeSumryVert is the default.

Below are several text commands for customizing the text in the **Exam Record** box:

```
\newcommand{\cpSumryHeader}{\textbf{Exam Record}}
\newcommand{\cpSumryPts}{\,\text{pts}}
\newcommand{\cpSumryPage}{Page}
\newcommand{\cpSumryTotal}{Total:}
\newcommand{\cpSumryGrade}{Grade:}
\cpSetSumryWidth{.5\linewidth}
```

The meanings of numbers (1)–(5) are apparent from Figure 1, page 33, where these strings are on display. Number (6) determines the width of the **Exam Record**, its default is half the \linewidth.

The **Exam Record** by default is enclosed in an \fbox, use the command \cpNofbox in the preamble to remove this enclosing box.

Naming conventions for byparts. When coverpagesumry=byparts, there are several naming options available. The default name of each part is the exam name (the required argument of the exam environment. By specifying \useUIPartNames in the preamble, the user friendly name is used (the one that appears as the optional first argument of the exam environment. The user friendly names are used when the solutions are listed in the back of the document (for example, when the option vspacewithsolns is specified). These user-friendly names may be too wide to put in the **Exam Record**, in this case, you can executed \useCustomPartNames in the preamble. This allows you to define your own names that are to appear in the **Exam Record**. To define your custom names, use the \customNaming command:

```
\customNaming{name}{text}
```

where, *name* is the name of the exam, and *text* is the text to appear in the **Exam Record**. For example, the following definitions might be made for a two part final exam, the first part the instructor is free to pose questions, the second part are Department supplied questions.

```
\customNaming{Part1}{Instr Qs:}
\customNaming{Part2}{Dept Qs:}
```

10.16. Calculate problem range between two markers

I have defined the command In response to a user who wanted more information about problem ranges between two marks, the command `\calcQsBtwnMarkers` is the proposed solution. The syntax is…

```
\calcQsBtwnMarkers[Mrk2]{Mrk1}
```

The required parameter (*Mrk1*) is called the *primary mark*. The command defines a number of other commands based on the primary mark name. `\calcQsBtwnMarkers` calculates the first and last problem numbers of the questions between the command `\calcQsBtwnMarkers` that has *Mrk1* as its *primary marker* and another command `\calcQsBtwnMarkers` that has *Mrk2* as its *primary marker*. The names of the commands produced are all based in the primary marker name *Mrk1*.

As mentioned above, there are several other commands `\calcQsBtwnMarkers` defines. `\calcQsBtwnMarkers[Mrk2]{Mrk1}` defines the following commands:

- `\Mrk1Start` is the first question number that follows the placement of the command `\calcQsBtwnMarkers` with *Mrk1* as its primary marker.
- `\Mrk1End` is the last question number between two `\calcQsBtwnMarkers` commands having *Mrk1* and *Mrk2* as their primary marks.
- `\Mrk1nQs` is the number of questions appearing between `\calcQsBtwnMarkers` commands having primary marks of *Mrk1* and *Mrk2*.

Three other commands are defined for easy user access.

- `\markStartFor{Mrk1}` expands to `\Mrk1Start`.
- `\markEndFor{Mrk1}` expands to `\Mrk1End`.
- `\markNumQsFor{Mrk1}` expands to `\Mrk1nQs`.

Use the command `\calcQsBtwnMarkers` without the optional first argument as the last mark in your `exam` environment.

The demo file for this feature is named `markqs.pdf`

References

[1] D. P. Story, *AcroTEX eDucation System Tools: LaTeX for interactive PDF documents*, in preparation. See page 4.