probsoln.sty v2.04: LaTeX 2$_\varepsilon$ Package to help create problem sheets

Nicola L.C. Talbot

21st August 2007

Contents

1 Introduction

The probsoln package is designed for teachers or lecturers who want to create problem sheets for their students. This package was designed with specifically mathematics problems in mind, but can be used for other subjects as well. The idea is to create a file containing a large number of problems with their solutions which can be read in by LaTeX, and then select a number of problems to typeset. This means that once the database has been set up, each year you can easily create a new problem sheet that is sufficiently different from the previous year, thus preventing the temptation of current students seeking out the previous year's students, and checking out their answers. There is also an option that can be passed to the package to determine whether or not the solutions should be printed. In this way, one file can either produce the student's version or the teacher's version.

2 Package Options

The following options may be passed to this package:

`answers`	Show the answers
`noanswers`	Don't show the answers (default)

3 Creating a New Problem

`\newproblem` `\newproblem[`⟨*nargs*⟩`]{`⟨*label*⟩`}{`⟨*problem*⟩`}{`⟨*solution*⟩`}`

A new problem is defined using the command `\newproblem`. This does not print anything, but merely stores the problem. As from version 2.03, there is also a starred version of this command for problems without a solution (for example, essay style questions):

`\newproblem*` `\newproblem*[`⟨*nargs*⟩`]{`⟨*label*⟩`}{`⟨*problem*⟩`}`

The argument ⟨*label*⟩ is a unique string that is assigned to this problem so that it can be used later. The argument ⟨*problem*⟩ is normal LaTeX code that should be used to typeset the problem. The argument ⟨*solution*⟩ is normal LaTeX code that should be used to typeset the solution, if required. For example, the following defines a problem with the label `quaddiff`:

```
\newproblem{quaddiff}{%
%This is the problem
\begin{displaymath}
f(x) = x^2 + 3x + 4
\end{displaymath}
}{%
%This is the solution
\begin{displaymath}
f'(x) = 2x + 3
\end{displaymath}
}
```

The following is an essay question, so doesn't come with a solution:

```
\newproblem*{inheritance}{Describe what is meant by the term
\emph{inheritance} in object-oriented programming. Use examples.}
```

The optional argument ⟨*nargs*⟩ specifies the number of parameters this problem will take. By default this value is 0, but any value from 1 to 9 may be used. Each parameter is referred to by `#1`, `#2`, ..., `#9`. For example, the following problem, labelled `sindiff`, takes one parameter:

```
\newproblem[1]{sindiff}{%
\(f(x) = \sin(#1x)\)
}{%
\(f'(x) = #1\cos(#1x)\)
}
```

The `quaddiff` problem shown above can be made more generic by using parameters:

```
\newcount\ctr
\newproblem[3]{diff:quad}{%
\(f(x) =
\ifnum#1=0
\else
\ifnum#1=1\else#1\fi x^2
\fi
\ifnum#2=0
\else
\ifnum#2>0 \ifnum#1=0 \else + \fi \fi
\ifnum#2=1\else#2\fi x
\fi
\ifnum#3=0
\else
\ifnum#3>0 \ifnum#2=0 \ifnum#1=0 \else + \fi \else + \fi\fi
#3
\fi\)
}{%
\(f'(x) =
\ifnum#1=0
\else
\ctr=2
\multiply\ctr by #1
\the\ctr x
\fi
\ifnum#2=0
\else
\ifnum#2>0 \ifnum#1=0 \else + \fi \fi
#2
\fi
 % print 0 if both #1 and #2 are 0
\ifnum#1=0 \ifnum#2=0 0 \fi\fi
\)
}
```

The three parameters correspond to the coefficients. Note that they must all be integers since TEX only performs integer arithmetic.

4 Creating a Database

To generate a database, simply create a `.tex` file where all the problems are defined using `\newproblem` or `\newproblem*`, and either `\input` it at the start of your document if you want to use specific problems (see Section 5), or pass it to `\selectrandomly` (see Section 7). This package comes with some sample databases, along with some sample documents that use these databases.

5 Displaying a Problem

`\useproblem` `\useproblem{`⟨*label*⟩`}`

Once a problem has been defined using `\newproblem`, it can be typeset using the

command `\useproblem`. If the problem was defined to take arguments, the arguments to the problem should come after the label. In the case of the `sindiff` example above, the command `\useproblem{sindiff}{2}` would produce the following:

$f(x) = \sin(2x)$

Solution: $f'(x) = 2\cos(2x)$

whereas the command `\useproblem{diff:quad}{3}{0}{-2}` would produce:

$f(x) = 3x^2 - 2$

Solution: $f'(x) = 6x$

Suppose all the above problems are defined in the file `probs.tex`, then the following code will create a problem sheet with four questions in it:

```
\documentclass{article}

\usepackage{probsoln}
\input{probs}

\begin{document}
Differentiate the following functions with respect to $x$:
\begin{enumerate}
\item \useproblem{quaddiff}
\item \useproblem{sindiff}{4}
\item \useproblem{diff:quad}{2}{3}{1}
\item \useproblem{diff:quad}{0}{1}{2}
\end{enumerate}
\end{document}
```

The answer sheet can then be generated by passing the option `answers` to the `probsoln` package.

6 Selecting All Problems

`\selectallproblems`

To select all problems defined in a database, in the order in which they were defined, use the command: `\selectallproblems{`⟨*filename*⟩`}` For example, suppose the problems are defined in the file `easy.tex`, then the following will create a problem sheet which uses all these problems:

```
\documentclass[a4paper]{article}

\usepackage{probsoln}

\begin{document}
Differentiate the following functions:
\begin{enumerate}
\selectallproblems{easy}
\end{enumerate}
\end{document}
```

7 Selecting Problems at Random

\selectrandomly `\selectrandomly{`⟨*filename*⟩`}{`⟨n⟩`}`

The command `\selectrandomly` will select ⟨n⟩ problems that are defined in the file ⟨*filename*⟩. Each problem is proceeded by `\PSitem` which is defined to be `\item`, so the command `\selectrandomly` should occur within one of the list-like environments, such as `enumerate`. For example:

\PSNitem

```
\begin{enumerate}
\selectrandomly{easy.tex}{4}
\end{enumerate}
```

will result in four numbered problems, selected at random from the file `easy.tex`. (The `.tex` extension may be omitted.) Each problem is followed by the command `\endPSNitem` which by default does nothing.

Multiple `\selectrandomly` commands may be used, however a different file must be used each time. For example:

```
\begin{enumerate}
\item Differentiate the following functions with respect to $x$:

\begin{enumerate}
\selectrandomly{samples/easy.tex}{3}
\selectrandomly{samples/args.tex}{1}
\end{enumerate}

\selectrandomly{samples/implicit.tex}{1}
\selectrandomly{samples/1stprncpl.tex}{1}

\end{enumerate}
```

This will result in a total of 6 problems, numbered 1(a), 1(b), 1(c), 1(d), 2 and 3.

If a randomly selected problem requires arguments, a message similar to the following will be displayed:

```
Problem diff:quad requires 3 argument(s), please specify (e.g. {5}{3}):
```

Enter the required arguments, where each argument is enclosed in braces ({ }).

8 Other Commands

\PSNrandseed The command `\PSNrandseed{`⟨n⟩`}` specifies the seed for the random number generator. For example, if you are using `\selectrandomly`, `\PSNrandseed{\year}` will produce a different set of problems each year, whereas `\PSNrandseed{\time}` will produce a different set of problems each time you LaTeX the problem sheet (as long as you leave at least a minute between runs.)

\random The command `\random{`⟨*counter*⟩`}{`⟨a⟩`}{`⟨b⟩`}` generates a random number from ⟨a⟩ to ⟨b⟩ and stores it in the LaTeX counter ⟨*counter*⟩. For example, to select 2, 3 or 4 problems from the file `implicit.tex`:

```
\newcounter{numproblems}
\begin{enumerate}
\random{numproblems}{2}{4}
```

```
\selectrandomly{implicit.tex}{\value{numproblems}}
\end{enumerate}
```

(Note the use of `\value`.)

This command can also be used to generate random values for problems that take arguments. Consider the problem `diff:quad` defined earlier. Three counters can be defined to represent the three coefficients:

```
\newcounter{A}
\newcounter{B}
\newcounter{C}
```

Random values can now be assigned to these counters:

```
\random{A}{-5}{5}
\random{B}{-5}{5}
\random{C}{-5}{5}
```

Finally, the problem can be used (note the use of `\arabic`):

```
\useproblem{diff:quad}{\arabic{A}}{\arabic{B}}{\arabic{C}}
```

`\doforrandN` The command `\doforrandN{`$\langle n \rangle$`}{`$\langle cmd \rangle$`}{`$\langle list \rangle$`}{`$\langle text \rangle$`}` will apply $\langle text \rangle$ for a random selection of $\langle n \rangle$ items in the comma separated $\langle list \rangle$. In each iteration the list item is denoted by $\langle cmd \rangle$. For example, suppose you have three files called `file1.tex`, `file2.tex` and `file3.tex`, and you want to select 1 problem from two of the three files, then you can do:

```
\doforrandN{2}{\file}{file1,file2,file3}{\selectrandomly{\file}{1}}
```

Note that it is also possible to do

```
\newcounter{numproblems}
\random{numproblems}{1}{3}
\selectrandomly{file\arabic{numproblems}}{1}
\random{numproblems}{1}{3}
\selectrandomly{file\arabic{numproblems}}{1}
```

however there is a possibility that the same file may be selected twice which will cause an error.

`\showanswers` The command `\showanswers` will show the solutions from that point on. May be localised by placing within a group.

`\hideanswers` The command `\hideanswers` will hide the solutions from that point on. May be localised by placing within a group.

`solution` The solution is placed inside the solution environment. By default this environment simply does `\par\noindent\textbf{\solutionname:}` at the start, `\solutionname` where `\solutionname` has the value: Solution. See the file `sample3.tex`, which comes with this package, for an illustration of how to customise the way in which the randomly selected problems are displayed. This sample file randomly selects multiple choice problems stored in the file `tabmchoice.tex`, and displays them in a longtable environment. As from version 2.03, the package will not define this environment if another package or class file has already defined it.

`showanswers` The boolean variable `showanswers` is defined to be true if the answers are shown and false otherwise. You can therefore do something like:

```
\ifthenelse{\boolean{showanswers}}{\textbf{Solution Sheet}}{}
```

and **Solution Sheet** will be printed only if the answers are displayed. (For more information on `\ifthenelse` and `\boolean` see the documentation for the `ifthen` package by David Carlisle.)

The `showanswers` switch can also be used within the definition of a problem, if you want the question to appear differently if the solution is displayed. For example:

```
\newproblem{mc:prod}{%
Which of the following is the derivative of $x\sin(x)$?
(Circle the correct answer.)
\ifthenelse{\boolean{showanswers}}{}%
{\begin{description}
\item[A] $\sin(x)$
\item[B] $x\cos(x)$
\item[C] $\sin(x) + x\cos(x)$
\end{description}
}}{%
\begin{description}
\item[A] $\sin(x)$
\item[B] $x\cos(x)$
\item[\textcircled{C}] $\sin(x) + x\cos(x)$ (product rule).
\end{description}
}
```

If the solutions are not displayed, the question will appear as

Which of the following is the derivative of $x\sin(x)$? (Circle the correct answer.)

A $\sin(x)$

B $x\cos(x)$

C $\sin(x) + x\cos(x)$

otherwise it will appear as:

Which of the following is the derivative of $x\sin(x)$? (Circle the correct answer.)

A $\sin(x)$

B $x\cos(x)$

Ⓒ $\sin(x) + x\cos(x)$ (product rule)

9 Troubleshooting

9.1 Unexpected Output

- I have lots of blank space before the first problem when using `\selectrandomly`.

  This is probably because you have lots of extraneous white space in your database. `\selectrandomly` will input the entire file, so any extra space will be included. Try commenting out the extra space using %.

9.2 Error Messages

- `! Package probsoln Error: Label ... already used.`

  Each label identifier used in \newproblem must be unique. Check to make sure you haven't used the same label more than once. Also check to make sure you haven't \inputed or randomly selected from the same file more than once. (Or \inputed and randomly selected from the same file.)

- `! Package probsoln Error: Label ... undefined.`

  You need to define a problem before you can use it. Check to make sure you haven't mis-spelt it, and check to make sure you have \inputed the file in which it is defined.

- `! Package probsoln Error: Requested number too large.`

  You have asked for more problems than are defined within the specified file. All problems in that file will be selected.

- `! Package probsoln Error: Can't randomly select ⟨n⟩ item(s)`

  You have asked to randomly select $\langle n \rangle$ items from a list that has less than $\langle n \rangle$ elements. For example, the following will generate this error:

  ```
  \doforrandN{10}{\file}{file1,file2,file3}{%
  \selectrandomly{\file}{1}}
  ```

  In this case the list, `file1,file2,file3` has only 3 elements, but the user has asked for 10 elements. If you type 'h' at the LaTeX prompt it will tell you how many items it thinks there are in the list. Remember that each item must be separated by a comma.

- `! LaTeX Error: Lonely \item--perhaps missing list environment.`

  Each problem selected using \selectrandomly is proceeded by \PSNitem which by default is defined as \item, and should therefore be placed in one of the list environments, such as `enumerate`. Alternatively, redefine \PSNitem.

- I get an error when I put a command definition in my database when using \selectrandomly.

  \selectrandomly inputs the database twice, so any command definition will be read twice, causing an error. You can use the switch \iffirstpass to prevent the error. For example:

  ```
  \iffirstpass
  \newcommand{\mycmd}{}
  \fi
  ```

  It is generally not a good idea to put anything other than \newproblem commands within the database.

10 Contact Details

Dr Nicola Talbot
School of Computing Sciences
University of East Anglia
Norwich. NR4 7TJ
http://theoval.cmp.uea.ac.uk/~nlct/

Index