Managing the paper trail of student projects: `datatool` and more

B. Tomas Johansson

Abstract

It is described how the paper trail for final year undergraduate projects can be handled using the package `datatool` and a master file of student data. In particular, it is shown how to generate a list of students and supervisors, a randomised mark sheet and timetable for project presentations, and how to sum the marks. To achieve this, the `ifthen` and `lcg` packages are used as well.

1 Introduction

At our institution, being responsible (or coordinator) for a final year undergraduate report writing module typically involves keeping track of around a hundred students, their project titles, supervisors and examiners. Moreover, various documents have to be produced, for example, a list of students with accompanying project titles and supervisors, a mark sheet as well as a timetable for the oral presentations, and a document having the final marks, to mention only a few. Naively setting up those documents in LaTeX by copying in student data, one finds that as the term goes by, students drop out or are added, supervisors turn up informing you that a project title has changed, or it is suddenly agreed that the timetable for the oral presentations should not be in name order but randomised. Starting feverishly to make changes to each document, most of us soon lose track of which data one has changed for what student and in what document.

A remedy is given by the package `datatool` [4]. Then only a master file of student data is needed, and documents are generated by reading from that file. Focus here is on basic constructions relevant for managing undergraduate projects; examples of handling marks of students and printing them out can be found in [4].

A csv-file with student data first needs to be constructed. It is a text file with comma-separated items on each line (other separators are also allowed). We'll give examples using this sample `students.csv`:

```
FirstName,Surname,Title,Supervisor,Examiner
Joe,Allen,Prime Numbers,Gert Li,Abbie Wan
Deeksha,Bhai,Inverse Problems,Alice Ince,Ma Ren
Gina,Gil,Algebraic Notes,Gert Li,Bob Al Turner
```

The first row consists of key words to be used when constructing documents, and the remaining rows are filled with student names and their data. For simplicity, only the students' names are split into first name and surname. One can easily add more keywords to the file such as `ReportMark` and `VivaMark`, and fill in the corresponding marks.

To generate a list of students, their supervisors and the titles of the projects, one can proceed as follows. In the preamble of a `.tex` file put

```
\usepackage{datatool}
\DTLloaddb{students}{students.csv}
\newcounter{nrstudents}
```

The command starting with `\DTLloaddb` has the effect of storing the database from the csv-file in a variable named `students`. The counter `nrstudents` will be used to number the students.

The lines below generate a table with student names and their corresponding supervisors, also numbering each student:

```
\begin{tabular}{rll}
 & Student & Supervisor\\
\DTLforeach{students}
  {\firstname=FirstName,\surname=Surname,
   \supervisor=Supervisor}
  {\stepcounter{nrstudents} \thenrstudents.
   & \firstname \space \surname
   & \supervisor\\}
\end{tabular}
```

Here, the command `\DTLforeach` goes through each line in `students` and stores the surname, first name and the supervisor's name of that line in the variables `firstname`, `surname` and `supervisor`, respectively. The keywords to pick out the data are given in the first line of the file `students.csv`. The number $1, 2, \ldots$, is in the first column thanks to the counter `studentnr`. A basic table is the result, as below; the design of such tables is a separate issue not dealt with here.

Note that if, for example, the name of the examiner should also be present in the table, then simply adjust to include `\examiner=Examiner` in the `\DTLforeach` command, and put `\examiner` in a separate column.

	Student	Supervisor
1.	Joe Allen	Gert Li
2.	Deeksha Bhai	Alice Ince
3.	Gina Gil	Gert Li

To get a corresponding list of titles, use

```
\setcounter{nrstudents}{0}
\noindent\\
\DTLforeach{students}
  {\title=Title}
  {\stepcounter{nrstudents}
   \thenrstudents. \title \\}
```

where again `\DTLforeach` is the key part. The result:

1. Prime Numbers
2. Inverse Problems
3. Algebraic Notes

If changes are made to the data in `students.csv`, it is only necessary to run the file having the above

commands to obtain an updated list of students, supervisors and project titles. This is a clear advantage compared to copying in the data in the file itself, especially as the number of files grow.

A typical question is how many projects a supervisor has, to check that there is a fair work load. To count and print out the number of projects assigned to a supervisor, do

```
\def\sumpr{0}
\DTLforeach[\DTLiseq{\supervisor}{Gert Li}]
  {students}{\supervisor=Supervisor}
  {\DTLadd{\sumpr}{\sumpr}{1}}
\sumpr
```

Here, a sum variable `\sumpr` is defined and declared to be zero. The data in `students` is sifted through `\DTLforeach`, where `\supervisor` is defined equal to Gert Li via `\DTLiseq`. If the conditional statement `\supervisor=Supervisor` is true, add one to the variable `\sumpr` (done via the command `\DTLadd`). The number of projects for the given supervisor is then printed with `\sumpr`.

Rather than counting the number of projects for each supervisor, it can be helpful to sort the data with respect to the name of the supervisors (it would have been more realistic to split supervisor names into first name and surname but to keep it simple a list is obtained here sorted with respect to the first name of the supervisors). Sorting is done with

```
\DTLsort{Supervisor}{students}
```

The data in `students` is now sorted and using the commands above, a table can be generated with the rows in alphabetical order of the supervisors; it is left for the reader to try.

For the oral presentations, staff need a mark sheet to score each student. This mark sheet should contain the name of the student, project title and time of the presentation, and have a table for scoring: the level of the content, mastery of the subject, quality of the slides, and presentation skills. Moreover, an overall mark should be given. A complication is that the order of the presentations shall be randomised.

The package `lcg` [2] generates random numbers. Put in the preamble of a `.tex` file for the mark sheet,

```
\usepackage{lcg}
\usepackage{ifthen}
\usepackage{datatool}
\DTLloaddb{students}{students.csv}
\newcounter{hour}\setcounter{hour}{1}
\newcounter{minutes}
```

The counters keep track of the time of a presentation. Assume that each presentation is 10 minutes, with a 10 minute break after 50 minutes. The hour is set to one to have an afternoon session starting at 1 pm.

To generate the mark sheet, first a column `Random` of random numbers is appended (on the fly) to the database. The command `\rand` from the package `lcg` [2] is invoked to generate random numbers. The data is then sorted with respect to this new column,

```
\DTLforeach{students}{}
  {\rand \DTLappendtorow{Random}{\arabic{rand}}}
\DTLsort{Random}{students}
```

It does not matter much to us if some random numbers are equal, the mixing in the database is still sufficiently far from name order. Then continue with

```
\DTLforeach{students}
  {\firstname=FirstName,\surname=Surname,\title=Title}
  {\noindent\\
   Student name: \firstname \space \surname \\
   Project Title: \title\\
   Time: \thehour.\theminutes 0 pm \\
   \addtocounter{minutes}{1}
```

If a presentation is to happen for example at 1.20 pm then `\thehour=1` and `\theminutes=2`. Since the presentations are 10 minutes each, the counter `\minutes` has to be increased by 1 (adding 0 to it manually).

The next part of the mark sheet is a table for scoring. This does not contain any real complication and is only included here for completeness

```
\begin{center}
\begin{tabular}{|l|c|c|c|c|}
\hline
& Poor & Fair & Good & Excellent \\ \hline
Level of content & & & &          \\ \hline
Mastery of subject & & & &       \\ \hline
Slides & & & &                   \\ \hline
Presentation skills & & & &     \\ \hline
\end{tabular}
\end{center}
Please give overall mark between 0 (lowest) to
100 (highest):
\begin{center}\thicklines
\framebox[.98\textwidth][c]{
  \parbox{.95\textwidth}
    {Additional Comments:\noindent\\[0.3cm]}}
\end{center}
```

A conditional statement is written at the end of the file using the `ifthen` package [1], checking whether five talks in a row have been given (that is, if `\theminutes=5`). If five talks have been presented, the counter for the hour is increased by one and the counter for the minutes is reset to zero. Otherwise nothing additional is done. This is coded as

```
  \ifthenelse{\equal{\theminutes}{5}}
    {\addtocounter{hour}{1}\setcounter{minutes}{0}}
    {}
}% end group from \noindent above
```

and this ends the construction of an elementary mark sheet for scoring oral presentations. A part of the file is shown at the top of the next page.

A timetable for the presentations also needs to be generated. Put the same commands in the preamble as for the mark sheet, and keep the randomisation.

Student name: Gina Gil
Project Title: Algebraic Notes
Time: 1.40 pm

	Poor	Fair	Good	Excellent
Level of content				
Mastery of subject				
Slides				
Presentation skills				

Please give overall mark between 0 (lowest) to 100 (highest):

Additional Comments:

Figure 1: Simple mark sheet example output.

Students are traversed writing out the data in a table and increasing the time of the presentation:

```
\begin{tabular}{lr}
\DTLforeach{students}
  {\firstname=FirstName,\surname=Surname}
  {\firstname \space \surname
   & \thehour.\theminutes 0 pm --
   \addtocounter{minutes}{1}
   \thehour.\theminutes 0 pm\\
```

Following this, if five consecutive talks have been given, the text "Break" and the time of the break are stated, otherwise a new line is started,

```
\ifthenelse{\equal{\theminutes}{5}}
  {\\ \setcounter{minutes}{0}
   Break \thehour.50 pm --
   \addtocounter{hour}{1} \thehour.00 pm
   & \\ \\}
  {\\[-0.4cm]}
}% ends \firstname... group above
\end{tabular}
```

This finishes a timetable for oral presentations, and part of such a file is this:

Gina Gil	1.40 pm – 1.50 pm
Break 1.50 pm – 2.00 pm	
Joe Allen	2.00 pm – 2.10 pm
Deeksha Bhai	2.10 pm – 2.20 pm

The automatic process for the randomised mark sheet and timetable takes away the painstaking job of manually assigning students and their timeslots. Having to make possible changes in the files makes the manual approach even less amusing. Instead only the csv-file needs to be updated (data for that file can often be generated from the student administration).

Let us mention one further construction. Expand the above csv-file with two columns named `ReportMark` and `VivaMark`, and fill in the corresponding marks. That is, a mark for the written report and a separate mark for the oral presentation. The total mark is the weighted sum of these two marks, with the written report counting for, say, 90% and the oral presentation for 10%. A file reporting the student names, marks and the total mark is to be constructed. In the preamble, put

```
\usepackage{datatool}
\DTLloaddb{students}{students.csv}
```

The remaining lines are

```
\begin{tabular}{rccc}
Student & Report Mark & Viva Mark & Final Mark\\
\DTLforeach{students}
  {\firstname=FirstName,\surname=Surname,
  \reportmark=ReportMark,\vivamark=VivaMark}
  {  \firstname \space \surname
   & \reportmark & \vivamark
   & \FPeval{\result}
           {round(0.9*\reportmark+0.1*\vivamark,0)}
     \result \\}
\end{tabular}
```

Here `\FPeval` (a wrapper in `datatool` of a command from the package `fp` [3]) stores in `\result` the value, first rounded to zero decimals, of the weighted sum `0.9*\reportmark+0.1*\vivamark` (more advanced calculations of marks and how to write them directly into the csv-file is given in [4, Sections 6.6–9]). An example of the outcome is

Student	Report Mark	Viva Mark	Final Mark
Joe Allen	68	82	69
Deeksha Bhai	60	65	61
Gina Gil	28	38	29

Other documents can be generated similarly with `datatool`, for example, a certificate for each student with their individual name and final mark, and an attendance sheet with the name of a student and supervisor filled in (to be used to record meetings with the supervisor). Personalized e-mail messages can also be produced.

Limitations of `datatool`, such as being slow on sorting, are stated in the documentation [4]. For our moderately sized documents, no problems have been experienced.

To conclude, involving the package `datatool`, managing the paper trail of student projects becomes manageable.

References

[1] David Carlisle, The `ifthen` package, `ctan.org/pkg/ifthen`
[2] Erich Janka, The `lcg` package, `ctan.org/pkg/lcg`
[3] Michael Mehlich, The `fp` package, `ctan.org/pkg/fp`
[4] Nicola L. C. Talbot, The `datatool` package, `ctan.org/pkg/datatool`

◇ B. Tomas Johansson
School of Mathematics, Aston University
Birmingham, B4 7ET, UK
`b.t.johansson (at) fastem dot com`
ORCID 0000-0001-9066-7922