GRTEX — To Make Multiple Exam Versions from LATEX

Joel A. Shapiro March 24, 1997

—

Based on Grex, by Richard J. Plano, which used WordMarc input.
Modified for Shapiro's latex input, by Plano ∼1993
Modified by Shapiro ∼1996

Grtex is a set of TEX or LATEX macros, templates and examples, together with a program which can input a conforming (La)TEX file and produce multiple versions suitable for multiple choice exams in large courses. Unlike Grex, the input and output files can not be encrypted at the time of LATEXing or grtexing, and care must be taken to preserve their privacy.

1 Simplified Instructions

This section outlines the steps necessary to produce a normal exam when everything works the first time. See the other sections for more detail and to understand what to do in case of trouble. Suggested file name conventions are indicated.

1. Prepare a "Question" file which contains the header material and the questions, as described below. For illustration I will use the name `227f96db.t` for this file. It can contain more questions than will actually be used, and in my case this was a data base (hence the db) of questions. This file should be debugged in LATEX before you run it into grtex.

2. Prepare a "Want" file in the precise format described below. By convention this has an extension .w, so for the first hour exam this might be called `227f96h1.w`.

3. Grtex will produce a "Key" file and an "Exam" file, which it will ask you to name. For example, `227f96h1.k` and `227f96h1.e`. Currently grtex refuses to overwrite existing files, so make sure these files do not exist in your current directory before running grtex. Now run grtex:

```
GRTeX  30-Dec-94

NOTICE:  GRTeX is a revised version of GREX.
        *****     GRTeX handles TeX.     *****
  For sample format, try DRA1:[MAX]TeXAM.TeX

Current size limitations: Questions:200.  Multiple answers: 100. Exams: 20.



Name of file for Questions :  (or EXIT)* 227f96db.t
Name of file for Want & Par:  (or EXIT)* 227f96h1.w
Name of file for    KEY out:  (or EXIT)* 227f96h1.k
Name of file for   EXAM out:  (or EXIT)* 227f96h1.e
READ QUESTION:     1/    1
READ QUESTION:     2/    2

                                   ⋮

READ QUESTION:   201/  201
 NQUES: 30 QUESTIONS READ/KEPT. 201/ 30
 Average and maximum copy scores:  0.000  0.000
ONUMBER OF QUESTIONS WITH ANSWER A,B,C,D,E, <A:, >E:
    1    2    3    4    5    6    7    8    9   10
    7    9    4    3    7    0    0    0    0    0
```

4. LaTeX the exam file

   ```
   latex 227f96h1.e
   ```

   and then print it with `dvips`, not `texprint`.

2 General Description

Two input files are needed — the "question" file, which is a TeX or LaTeX file containing the text; and a "want" file specifying the questions wanted on the exam, which may be a subset of those in the question file, and various options about right answers, permutations for the versions, etc.

2.1 The "question" file

The question file should be started by copying

```
/physics/doc/physics/grading/grtex/examex.template 227f96db.t
```

replacing the latter name by the one you are choosing. Detailed instructions for how to add and modify that files are given in the documentation/example

`/physics/doc/physics/grading/grtex/examexample.tex`.

If you go to that directory, you can look at the example with

```
ghostview examexample.ps
```

[If it comes out upside-down, pull down from `orientation` to `Swap Landscape` and it will become upright. If it comes out with the top cut off, complain to the system manager of the machine.]

You can also print `examexample.ps` with whatever command you use for postscript printing.

2.2 The "want" file

This file is an ascii file in a definite format. I will be gradually improving grtex to be less demanding of the exact format, but for now, this is it.

The want file specifies all the parameters not only for producing the exam file but also for grading the file. There are many obscure options for this. Some effect the output exam file, while others are passed by the key file to `grad`, the grading program, without affecting the LaTeX exam file.

The main parameters which do affect the exam file are

NQUES: The number of questions which will be extracted.

NEXAMS: The number of versions of the exam you want

MSORT: Specifies the form of permutation to make on the questions and/or answers for the multiple versions of the exam.

Plano's original grtex had three allowed values of MSORT

1: Scramble the order of the questions, but keep each question unchanged.

2: Scramble the order of the answers for each question, but keep the order of the questions unchanged.

3: Scramble the order of the questions and also the order of the answers given to each question.

These modes **do not produce** random orderings. They do not permute version 1 at all. While random orders are produced internally for the other versions, favorite orderings are extracted to minimize the scores of those copying off perfect papers of the wrong version. I found the results unacceptible, so I have produced several new allowed values for MSORT:

0: Does no scrambling at all, and also doesn't check correct answers. This can be used only with NEXAMS = 1. It is useful to extract the desired questions from a database for consideration by the instructors, not for producing the exams themselves. If your database contains questions with correct answers not yet determined, the answer can be given as {0}, although this must be fixed before running the real exam, of course.

5: Like 1, it scrambles question order only, on versions [2,NEXAMS], keeping the answers unpermuted within the question. But this MSORT really does give a random ordering.

6: Like 2, scrambles answers on versions [2,NEXAMS] but not the question order, but really giving a random order for the answers.

7: Like 3, scrambles both question and answer orders on [2,NEXAMS], but truly randomly.

9: Like 5, but version 1 is scrambled as well as [2, NEXAMS]

10: Like 6, but version 1 is scrambled as well as [2, NEXAMS]

11: Like 7, but version 1 is scrambled as well as [2, NEXAMS]

The other parameters you need to specify are

KOURSE the course number, *e.g.* 228

KDATE Date of the exam, in format YYMMDD, say 960227

NMULTS The number of questions with multiple correct answers. Usually this is 0 until you discover, after the exam has been given, that you made a mistake. See below.

KPSCOR The total number of points for a perfect exam.

KWRONG The number of points to **subtract** for a wrong answer. Most people use 0, I believe.

KVALUE If 0, each correct answer is awarded KPSCOR/NQUES points. If > 0, the number of points must be individually specified for each question.

MEXTRA "reserved"

M32 If 0, only one answer per question may be marked. If 1, each question will have various combinations of marks as correct. See below.

2.2.1 The format of the want file is

1. One line of 80 or fewer characters which will appear on the student answer sheets. What is there does not affect anything else, but there must be a line for it.

2. One line which is ignored but usually gives the names of the variables required on the following line, in order, to help in entering them correctly. So it should read

```
KOURSE KDATE NQUES NEXAMS NMULTS KPSCOR KWRONG KVALUE MSORT MEXTRA M32
```

3. On the next line, one integer for each of the above variables, free format.

4. If NMULTS $\neq 0$, there are $\lceil$NMULTS/5$\rceil$ lines in the (fortran) `FORMAT (1X,5(I2,I3,1X,4A1))`, each of which specify, for five questions, the number of points to give for one of the extra answers (I2), the question number we are talking about (I3), and the other answers which are considered (partially) correct, or 0 (4A1). Thus `15␣␣5␣2300` will give 15 points to answers 2 or 3 on question 5, while `␣␣␣␣5␣2300` will give answers 2 and 3 the same credit as the "correct" answer.

5. If KVALUE > 0, the credit assigned for each correct answer and the penalty for each wrong answer is given in the next two lines in `FORMAT (20F4.1)` format. (??? in what order? Suppose NQUES> 20?).

6. Finally, the numbers in the input file of the questions to be output (on unscrambled versions) in the order specified. This must be in `FORMAT (10I6)` format, though clearly this needs to be changed.

Here is an example of a "want" file which will make question 138 of the input file question 4 of version 1, and will make 4 versions with 19 questions, all weighted equally:

```
COMMON HOUR 2 EXAM -- PHYSICS 123 -- NOVEMBER 13, 1989
COURSE DATE  NQUES NEXAMS NMULTS KPSCOR KWRONG KVALUE MSORT MEXTRA M32
 123, 841201,   19,     4,     0,   100,     0,     0,    1,     0,  0
    2     3     4   138     1     7     9    10    11    12
   13    14    15    16    17     5    19    20    21
```

3 More Detail

The grading program has a very long history, and many complex options, which have not had their documentation updated. Before grtex there was a program called grex which performed the same function, but with a word processor called WordMark rather than latex. The file `/physics/doc/physics/grading/grex.asc` describes this program, including an appendix on "Preparation and Administration of Exams". In the same directory there is also `gred.asc`, which describes the programs `GREAD`, which reads the mark sense forms, `GRED`, which lets one edit the forms read to correct for errors, and `GRAD`, which grades the exam. This document contains a lot of details about the options to GRAD.