# Package 'ProfessR'

May 7, 2026

**Type** Package

**Title** Grades Setting and Exam Maker

**Version** 2.4-3

**Date** 2023-08-09

**Depends** R (>= 2.12)

**Imports** RPMG

**Author** Jonathan M. Lees [aut, cre]

**Maintainer** Jonathan M. Lees <jonathan.lees@unc.edu>

**Description** Programs to determine student grades and create
examinations from Question banks. Programs will create numerous
multiple choice exams, randomly shuffled, for different versions of same question list.

**License** GPL (>= 2)

**NeedsCompilation** no

**Repository** CRAN

**Date/Publication** 2023-08-21 08:42:38 UTC

## Contents

---

`ProfessR-package` *Grades Setting and Exam Maker*

---

## Description

Programs to determine student grades and create examinations from Question banks. Programs will create numerous multiple choice exams, randomly shuffled, for different versions of same question list.

## Author(s)

Jonathan M. Lees

Maintainer: Jonathan M. Lees<jonathan.lees@unc.edu>

## Examples

```
################  making tests:

## Not run:
data(QBANK1)
make.exam(QBANK1, ofile="exam1.tex")

## End(Not run)


#################  setting grades:
g = rnorm(n=130, m=82, sd=10)
g[g>100] = 100
g[g<1] = 1

B = boxplot(g)

divs = c(min(g), B$stats[1:4] + diff(B$stats)/2, max(g) )

D1 = do.grades(g, divs=divs, tit="GEOL 105 Exam 1")
```

---

`autoemail` *AutoEmail*

---

## Description

Automatically email a file to an address using the perl program.

## Usage

```
autoemail(eadd, sfile, hnote = "Exam Results")
```

## Arguments

| | |
|---|---|
| `eadd` | Email address |
| `sfile` | file to be sent |
| `hnote` | subject line |

## Details

This program will work well in Linux and Mac where Perl is installed - I am not sure about Windows. Creates a unix executable file, if perl is present.

## Value

Side Effects.

## Note

Need to change the from designation.

## Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

## See Also

IDandEM

---

`CHECKbank` *Check a set of Question banks*

---

## Description

Sequentially check a set of Question banks. Makes sure there is a QUESTION: and ANSWER for each question.

## Usage

```
CHECKbank(QB)
```

## Arguments

| | |
|---|---|
| `QB` | list of question banks |

## Value

Printed Side Effects

## Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

## See Also

seebank

## Examples

```
 data(QBANK1)
CHECKbank(QBANK1)

############  modify by inserting an error:
QBANK1[[4]]$numANS=NULL

### recheck:
CHECKbank(QBANK1)
```

---

checkgrades *Check Grade Distribution*

---

## Description

View grades sorted and listed with raw score, letter and scaled score, with optional ID and name

## Usage

```
checkgrades(D1, id = NULL, names = NULL)
```

## Arguments

| | |
|---|---|
| D1 | output of do.grades |
| id | character vector, ID for students |
| names | character vector, names of students |

## Value

Side effects

## Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

## See Also

do.grades, DUMPgrades

## Examples

```
g = rnorm(n=130, m=82, sd=10)
g[g>100] = 100
g[g<1] = 1

B = boxplot(g)

divs = c(min(g), B$stats[1:4] + diff(B$stats)/2, max(g) )



###  to run interactively, remove the divs
###  D1 = do.grades(g, tit="GEOL 105 Exam 1")

### otherwise use previously calculated divs:
D1 = do.grades(g, divs=divs, tit="GEOL 105 Exam 1")
checkgrades(D1 )
```

---

COMPbank *Compare Question Banks*

---

## Description

Compare two question banks to find non-duplicated questions

## Usage

```
COMPbank(Qbank1, Qbank2)
```

## Arguments

| | |
|---|---|
| Qbank1 | Question Bank 1 |
| Qbank2 | Question Bank 2 |

## Details

Uses match to find matching questions in the two question banks.

## Value

Vector index of questions in Qbank2 that are not found in Qbank1.

## Note

Only the questions are compared, the answers are ingnored. The return vector will be a set of questions that are not duplicated, i.e. unique to question bank 2.

## Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

## See Also

SELbank

## Examples

```
## Not run:
LF = list.files(path="/home/lees/Class/GEOL_105/TESTBANK/EXAM_1", pattern="txt", full.names=TRUE )

kbank = vector(mode='list')
######   read in the question banks, each in one file
for(i in 1:length(LF))
  {
    h = Get.testbank(LF[i])
    kbank[[i]] = Get.testbank(LF[i])

  }
names(kbank) = LF
Kbank =  vector(mode='list')

for(i in 1:length(kbank))
  {

Kbank = c(Kbank, kbank[[i]])

  }

q2 = COMPbank(Kbank, kbank[[3]] )

############  to extract these:
subq2 = subsetbank(kbank[[3]] , q2)
############  to get the overlapping questions:

olap = 1:length(kbank[[3]])
olap[-q2]



## End(Not run)
```

---

deblank *Remove blanks from strings.*

---

## Description

Remove blanks from strings.

## Usage

```
deblank(a)
```

## Arguments

| | |
|---|---|
| a | Character string |

## Details

Removes all blanks from strings. The functuon works on vectors of strings, removing blanks on each element.

## Value

Character string with no blanks.

## Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

## Examples

```
j = c(' James', 'Jones ', 'Alpha Dog')

deblank(j)
```

---

# do.grades *Do Grades*

---

## Description

Calculate the grades of a class of students, given raw scores on exam

## Usage

```
do.grades(ggrades, divs = NULL, cut = 0, tit = "Exam Grades",
breaks=length(ggrades)/3, ...)
```

## Arguments

| | |
|---|---|
| ggrades | Raw grades |
| divs | divisions for grades (optional) |
| cut | low end Cut off to remove 0 from statistics |
| tit | Title for Figure |
| breaks | breaks for the histogram, default=length(ggrades)/3 |
| ... | other parameters for hist |

## Details

To remove students who do not take the test a low end cut off is used to excise any grades below that level. Both mean, and standard deviations are shown as well as median and quartiles.

## Value

grades=ggrades, lett=letts, scor=scores, divs=divs, LETS=LETS, SCRS=SCRS, hist=HA LIST:

| | |
|---|---|
| grades | raw scores |
| lett | letter grades |
| scor | scaled grades |
| divs | divisions, estimated by user or provided as input |
| LETS | letter grades assigned |
| SCRS | Scores related to LETS |
| hist | histogram structure |

## Note

Grades are determined linearly within a division

## Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

## See Also

jist, DUMPgrades, getlet

## Examples

```
g = rnorm(n=130, m=82, sd=10)
g[g>100] = 100
g[g<1] = 1

B = boxplot(g)

##########   set divisions automatically:
```

```
divs = c(min(g), B$stats[1:4] + diff(B$stats)/2, max(g) )



###  to run interactively, remove the divs
###  D1 = do.grades(g, tit="GEOL 105 Exam 1")

### otherwise use previously calculated divs:
D1 = do.grades(g, divs=divs, tit="GEOL 105 Exam 1")

## Not run:

####  this is interactive
D1 = do.grades(g, tit="GEOL 105 Exam 1")

#######  list the grades:
cbind(D1$grades,  D1$lett,  D1$scor)

####### if you have names or ID's try:
#######  cbind(IDs, D1$grades,  D1$lett,  D1$scor)

\dontrun{

 DUMPgrades(D1, file="TEST1grades", id=IDS )
}


## End(Not run)
```

---

`droplowest` *Drop lowest grade*

---

## Description

Drop the lowest grade from a matrix of grades. Matrix is assumed to be N by m where m is the number of exams (columns), N the number of students (rows)

## Usage

```
droplowest(z)
```

## Arguments

| | |
|---|---|
| z | Matrix of scores, rows are students, columns are exam scores |

### Details

Best matrix output is sorted, so the grades do not reflect the original order of exam scores. To drop the two lowest scores, apply this program twice, running it a second time on the best outut.

### Value

| | |
|---|---|
| minind | Index of minimum score |
| best | matrix of scores with the lowest dropped |
| midgrade | mean value of best scores |

### Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

### See Also

do.grades

### Examples

```
#########   generate fake exam scores, 10 students, 3 exams
z = matrix(runif(3*10, 50, 100), ncol=3   )
 A  = droplowest(z)
 cbind(A$best,  A$minind, z, A$midgrade)
```

---

## DUMPbank *Dump a Question Bank*

---

### Description

Save an ASCII version of a selected Question Bank

### Usage

```
DUMPbank(ofile, QB, sep = "\n", append=TRUE)
```

### Arguments

| | |
|---|---|
| ofile | character, output file |
| QB | QuestionBank Structure |
| sep | separator between questions |
| append | logical, if FALSE a new file is created |

## Value

Side effects

## Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

## See Also

Get.testbank

## Examples

```
## Not run:
data(QBANK1)
DUMPbank("my.questions", QBANK1, sep = "\n")

QB1=Get.testbank("my.questions")

## End(Not run)
```

---

# DUMPgrades *Dump grades to a file*

---

## Description

Dump grades to a file

## Usage

```
DUMPgrades(D1, file = NULL, id = NULL, names = NULL)
```

## Arguments

| | |
|---|---|
| D1 | list output from do.grades |
| file | file name, a csv will be added as a suffix |
| id | vector of student IDs |
| names | character vector of student names |

## Value

Side effects

## Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

## See Also

do.grades

## Examples

```
g = rnorm(n=130, m=82, sd=10)
g[g>100] = 100
g[g<1] = 1

B = boxplot(g)

divs = c(min(g), B$stats[1:4] + diff(B$stats)/2, max(g) )



###  to run interactively, remove the divs
###  D1 = do.grades(g, tit="GEOL 105 Exam 1")

### otherwise use previously calculated divs:
D1 = do.grades(g, divs=divs, tit="GEOL 105 Exam 1")

## Not run:

DUMPgrades(D1, file="TEST1grades" )


## End(Not run)
```

---

DUPbank *Find Duplicate Questions*

---

## Description

Finds dupliucated questions in a set of Question Banks

## Usage

```
DUPbank(Qbank)
```

## Arguments

| | |
|---|---|
| Qbank | a list of Question Banks |

## Details

The program only checks the questions, not the answers. One could thus have several questions with the same wording, but different answers. I might change this in the future. Given the list of duplicated questions one should edit the original question bank files to remove them.

## Value

| | |
|---|---|
| A | vector of duplicated questions |
| F | vector of duplicated files where the questions were extracted |
| I | vector of duplicated indexes where the questions were extracted |
| N | vector of duplicated indexes where the questions were extracted |

## Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

## Examples

```
data(QBANK1)

###  force some questions to be duplicates:
QBANK1[[51]]=QBANK1[[25]]
QBANK1[[52]]=QBANK1[[12]]
QBANK1[[14]]=QBANK1[[4]]


DQ = DUPbank(QBANK1)

DQ
```

---

# E2grades *Examination grades from Test 2 in 2007*

---

## Description

Real exam raw scores from test in Geology 105, University of North Carolina. Zeros are assigned to students who did not take the test.

## Usage

```
data(E2grades)
```

## Format

numeric vector

## Examples

```
data(E2grades)

g = E2grades

B = boxplot(g[g>1], plot=FALSE)
divs = c(min(g), B$stats[1:4] + diff(B$stats)/2, max(g) )
```

```
###  get(getOption("device"))(width = 12, height = 7)

D1 = do.grades(g, divs=divs, cut = 15, tit="GEOL 105 Exam 1")

jist(D1$hist, D1$grades, D1$lett, col='purple')
```

---

EXAMstats *Exam Statistics*

---

## Description

Statistical Analysis of Examination where the results are either correct or incorrect.

## Usage

```
EXAMstats(j, key)
```

## Arguments

| | |
|---|---|
| j | matrix of student responses |
| key | key of correct answers |

## Details

At this statge no partial credit is given.

## Value

List

| | |
|---|---|
| H | Matrix: question, correct response, student responses, difficulty, Desc, BiSer |
| kr20 | Kruder-Richardson reliability statistic |

## Note

There is a slightly different implementation if partial credit is employed. See

## Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

## References

Kuder, G. F., and Richardson, M. W. (1937). The theory of the estimation of test reliability. Psychometrika, 2(3), 151-160.

Cortina, J. M., (1993). What Is Coefficient Alpha? An Examination of Theory and Applications. Journal of Applied Psychology, 78(1), 98-104.

## See Also

readSCANTRON

## Examples

```
## Not run:
B2 = readSCANTRON(rawfn2)

Estat = EXAMstats(B2$studans, B2$key)

Estat$kr20


## End(Not run)
```

---

`fix.names` *Fix Down Loaded Names*

---

## Description

Fix names to remove problematic alphanumeric characters like spaces, quotes

## Usage

```
fix.names(nam,  upper=FALSE, lower=FALSE)
```

## Arguments

| | |
|---|---|
| `nam` | string |
| `upper` | logical, TRUE= convert to upper case |
| `lower` | logical, TRUE= convert to lower case |

## Details

Currently only space, single and double quotes.

## Value

string, with quaote replaced with underscore

## Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

## Examples

```
####  examples with embedded quotes are not available
####    because they interfere with R documentation

LAM = "SILENCED LAMB"
fix.names(LAM, lower=TRUE)

LAM = "Silence my Lamb"
fix.names(LAM, upper=TRUE)

LAM = "SILeNCED LAMB"
fix.names(LAM)

###   try with single quote
LAM = "O'brian LAMB"
fix.names(LAM)
```

---

`Get.testbank` *Get Test Bank From Ascii Text Files*

---

## Description

Get Test Bank From Ascii Text Files

## Usage

```
Get.testbank(fn)
```

## Arguments

| | |
|---|---|
| `fn` | File Name |

## Details

Structure of input file is strict: see the vignette for an example. Each questions starts with the tag QUESTION: (there is a space following the colon on all tags) followed by answers with the correct answer indicated by the tag ANSWER: . The tag FIG: allows the examiner to include a figure with a latex tag for reference. For example: ‘ `QUESTION: What was the world like during the Late Paleocene Torrid Age? ANSWER: a. Most of the world was wetter and warmer. b. Most of the world was drier and warmer. c. Most of the world was wetter, but a little cooler. d. Most of the world was a desert. e. It is impossible to estimate conditions at that time.` ’

## Value

List: list of Questions

## Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

## Examples

```
## Not run:
fn = "MY.questions"
Qbank = Get.testbank(fn)

##########  use existing database:
data(QBANK1)
#####  dump out question bank in correct format:
DUMPbank("my.questions", QBANK1, sep = "\n")
###  read it in:
QB1=Get.testbank("my.questions")

## End(Not run)
```

---

`getgroup` *Create Groupings of Students*

---

## Description

Create groups of students and plot groups to screen.

## Usage

```
getgroup(g.first, n = 2)
```

## Arguments

| | |
|---|---|
| `g.first` | Character vector of student names. |
| `n` | number per group |

## Details

Class roster will be divided into n groups and displayed on the the screen.

## Value

List of groups with names.

## Note

The class is currently randomized in this version.

### Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

### See Also

GetStudentNames

### Examples

```
g.last =c('Joyce', 'Einstein', 'Hertz', 'Bailey',
'Compton', 'Jones', 'Wilson', 'Smith', 'Anderson' )

getgroup(g.last, n = 3)
```

---

## getKEY *Read Key output*

---

### Description

Read Key output

### Usage

```
getKEY(fn)
```

### Arguments

| | |
|---|---|
| fn | character string file name |

### Details

Reads in the file output of ProfessR and returns a vector of answers

### Value

vector of correct answers

### Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

### See Also

version.exam, prep.solution

getlet *Get Letter Grades*

## Description

Get letter grades from list of numeric scores

## Usage

```
getlet(ggrades, divs)
```

## Arguments

| | |
|---|---|
| ggrades | vector of grades |
| divs | numerical vector of divisions |

## Details

Returns letter grades scaled linearly between divisions.

## Value

LIST:

| | |
|---|---|
| ggrades | Input grades |
| lett | letter values |
| scor | scores after scaling |
| divs | divisions used in setting scores |
| LETS | Letters for grades |
| SCRS | numeric divisions used for LETS |
| olett | letter values, older version |
| oscor | scores after scaling, older version binned |

## Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

## See Also

do.grades

## Examples

```
g = rnorm(130, m=82, sd=10)

g[g>100] = 100
g[g<1] = 1



B = boxplot(g)


divs = c(min(g), B$stats[1:4] + diff(B$stats)/2, max(g) )

G = getlet(g, divs)

cbind(G$LETS, G$SCRS)



data.frame(G$grades, G$lett, G$scor)
```

---

GetStudentNames *Extract Student Names from Roster.*

---

## Description

Given a roster of students, with (lastname, first name) format, extract a unique set of first names, with no blanks.

## Usage

```
GetStudentNames(c1, dup.lets=1)
```

## Arguments

| | |
|---|---|
| c1 | Character vector |
| dup.lets | NUmber of letters to add from last name in the event that first names are duplicated. |

## Details

The function assumes the names are comma separated with lastname, firstname order. The code separates the names, removes blanks from the first name, and finds a unique set of names. If first names are not unique, the function extracts the first letters of the last names and the duplicated names and appends with a period.

## Value

Character vector of unique first names

## Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

## Examples

```
g.first =c("Jason","Skyler","Adrian","Berkley","Jack",'David',
'David', 'Jim', 'Jim')
g.last =c('Joyce', 'Einstein', 'Hertz', 'Bailey', 'Compton',
'Jones', 'Wilson', 'Smith', 'Anderson' )

c2 = paste(g.last, g.first, sep=', ')

K = GetStudentNames(c2)
```

---

# gradeSCAN *Grade a SCANTRON*

---

## Description

Grade each row of a matrix which is a record of the scanned anwers from a test.

## Usage

```
gradeSCAN(j, key)
```

## Arguments

| | |
|---|---|
| j | matrix, scanned answers from the grading center |
| key | vector, key for grading |

## Details

Program sums correct answers and returns the score for each row.

## Value

vector of scores

## Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

IDandEM *Match ID and Email file*

## Description

Match ID and Email file

## Usage

```
IDandEM(scrfn, sisroster, sel = 1:2, hnote = "Exam Results", SEND = TRUE)
```

## Arguments

| | |
|---|---|
| scrfn | list(ID=number, nam="name on scantron") |
| sisroster | list(ID=number, lastname='last name of student', fullname='full name of student') |
| sel | numeric, index= specify for a specific student |
| hnote | text, subject line on E-mail |
| SEND | logical, if FALSE, do not send |

## Details

A set of files has been separated and stored. Each file is to sent to a different student with the exam results.

## Value

Side Effects

## Note

The IDs of the reference data base (the roster) must match the IDs in the list of files. If not, use repair.id to fix the scantron IDs

## Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

## See Also

repair.id

## Examples

```
## Not run:
##  read in the names of the files
zfile = scan(file="ALLIDS", list(name="", ID=0, tfile=""), sep="," )
##  read in a roster.  The roster has
##   email addresses that are attached tot he files
##  by matching the ID in the zfile with the IDs in the data base
load(file="/home/lees/Class/GEOL_105/Grades_2008/EXAM1/BB1.RDATA")

jroster = BB1

IDandEM(zfile, jroster, sel=1:10, hnote="GEOL105 EXAM3 Results", SEND=FALSE )
IDandEM(zfile, jroster,  hnote="GEOL105 EXAM3 Results", SEND=FALSE )

######## actual sending
IDandEM(zfile, jroster, hnote="GEOL105 EXAM3 Results", SEND=TRUE )




## End(Not run)
```

---

`jist` *Add letter grades to histogram*

---

## Description

Given a vector of grades, add the letters to an existing histogram.

## Usage

```
jist(h, Z=1, L=1, col=2)
```

## Arguments

| | |
|---|---|
| `h` | histogram list |
| `Z` | grades from original data |
| `L` | letters associated with grades |
| `col` | color for plotting letters |

## Details

This will add information on an existing histogram plot. If h is the output of do.grades() then Z and L are ignored.

## Value

Graphical Side effects

## Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

## See Also

do.grades

## Examples

```
g = rnorm(130, m=82, sd=10)
g[g>100] = 100
g[g<1] = 1

B = boxplot(g)


divs = c(min(g), B$stats[1:4] + diff(B$stats)/2, max(g) )

####G1  = do.grades(g, cut=20, tit="GEOL 105 Exam 1")

##########  replot with existing divisions:
D1 = do.grades(g, divs=divs, tit="GEOL 105 Exam 1")

jist(D1$hist, D1$grades, D1$lett)

##########  or simply:

D1 = do.grades(g, divs=divs, tit="GEOL 105 Exam 1")

jist(D1)
```

---

# LETGRADE *Letter Grade*

---

## Description

given a numeric grade return a letter grade

## Usage

```
LETGRADE(g)
```

### Arguments

| | |
|---|---|
| g | numeric grade between 1-100 |

### Details

returns a grade based on a 4 point spread

### Value

character vector of grades

### Note

Failing grade is "E" by default. There is no "A+" in this program (UNC policy)

### Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

### Examples

```
g = rnorm(25, m=82, sd=10)
g[g>100] = 100
g[g<1] = 1


L = LETGRADE(g)

cbind(g, L)
```

---

## make.exam *Make Exam*

---

### Description

Given a question bank, create a test.

### Usage

```
make.exam(Qbank, ofile = "examq.tex", ncol=2)
```

### Arguments

| | |
|---|---|
| Qbank | Question bank list |
| ofile | Output file |
| ncol | number of columns on page, default=2 |

### Details

Dumps out a tex file with the questions

### Value

Side Effects - output to a TEX file.

### Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

### See Also

prep.exam

### Examples

```
data(QBANK1)

## Not run:
make.exam(QBANK1, ofile="exam1.tex")

## End(Not run)
```

---

## make.solution *Create Solution File*

---

### Description

Create Solution File in Latex

### Usage

```
make.solution(Qbank, ofile = "answers.tex")
```

### Arguments

| | |
|---|---|
| Qbank | Question Bank |
| ofile | Output File |

### Details

Creates a latex file suitable for printing solution to the exam.

## Value

Side Effects

## Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

## Examples

```
data(QBANK1)

## Not run:
 make.solution(QBANK1, ofile= "solutions.tex")

## End(Not run)
```

---

# phist *Plot Histogram with Grades labeled*

---

## Description

Plot Histogram with Grades labeled

## Usage

```
phist(G, Z = 1, L = 1, col = 2, add = FALSE, tit = "GEOL 105 Exam 1")
```

## Arguments

| | |
|---|---|
| G | Histogram list from do.grades |
| Z | numerical grades |
| L | text, vector, Letter Grades |
| col | color for text |
| add | logical, add=TRUE, add to existing plot |
| tit | title for plot |

## Value

List:

| | |
|---|---|
| x | x location on plot |
| y | y location on plot |
| L | Label printed |

### Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

### See Also

do.grades

### Examples

```
## Not run:
newID3 =  repair.id(DBB, raw3)
raw3$id=newID3
raw3$ID=newID3



## End(Not run)
```

---

`prep.exam` *Prepare Exam for Latex (simple style)*

---

### Description

Prepare Exam for Latex - use simple styles

### Usage

```
prep.exam(OF, incfile, instructor="", examdate="",
 course="",  examname="", instructions="", ncol=2)
```

### Arguments

| | |
|---|---|
| `OF` | Character string output files |
| `incfile` | Character, include file name for questions |
| `instructor` | name of instructor |
| `examdate` | Date of the examination |
| `course` | Name of the course, character |
| `examname` | title of exam |
| `instructions` | character vector of instructions |
| `ncol` | number of columns on page, default=2 |

### Value

Side Effects

## Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

## See Also

version.exam

## Examples

```
## Not run:
######  since the program produces a file on the local
######   system, do not run this example


examdate="THURS Sep 20 2007"


seqnum="1"
exnumber="Exam 1"
V = "exam1A"
outtex = paste(sep=".",V, "tex" )
outMAST  = paste(sep="", V, "MAST" )


MASTtex  = paste(sep=".", outMAST , "tex" )

outsolut  = paste(sep="", V, "solutions.tex" )
Me = "Jonathan M. Lees"

course="GEOL 105"

examname=paste(sep=" ", exnumber, "Seq", seqnum)



instructions=c("There are 50 questions.",
"Answer all questions.", "Mark clearly.")
\dontrun{
prep.exam(outMAST, outtex , instructor=Me, examdate=examdate,
course=course, examname=examname, instructions=instructions)
}


## End(Not run)
```

`prep.solution` *Prepare Solution Files*

## Description

Prepare Latex Solution Files

## Usage

```
prep.solution(ofile)
```

## Arguments

| | |
|---|---|
| `ofile` | output file name |

## Details

Prepares the Latex header for the solution files

## Value

Side Effects

## Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

## See Also

prep.exam

## Examples

```
## Not run:
prep.solution("solfile")

## End(Not run)
```

printSCANTRON *Print Scantron*

## Description

Print results from scantron center

## Usage

```
printSCANTRON(B1)
```

## Arguments

| | |
|---|---|
| B1 | list, output of readSCANTRON: must have elements studans, Nams, ids |

## Value

side effects

## Note

Prints the matrix returned from the scantron center.

## Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

## See Also

readSCANTRON

## Examples

```
## Not run:

datadir =  "./DATA"
rawfn1 = paste(datadir,'t6200a.raw.csv', sep="/")

B1 = readSCANTRON(rawfn1)
printSCANTRON(B1)


## End(Not run)
```

QBANK1 *Example Question Bank*

## Description

Example Question Bank, 50 question, multiple Choice

## Usage

```
data(QBANK1)
```

## Format

List:

**Q** Question in latex format (character string)

**A** Possible Answers in latex format (vector of character strings)

**a** Correct Answer in latex format (character string)

**numANS** index number corresponding to correct answer

**FIG** character: full path to figure, tag for figure

## Details

An example input question in ascii format is constructed using three tag identifiers: "QUESTION:", "ANSWER:" and (optionally) "FIG:". The format is shown here:

## Examples

```
data(QBANK1)
## maybe str(QBANK1) ; plot(QBANK1) ...
print(QBANK1[[1]])
```

ran.exam *Random order of Exam*

## Description

Randomly re-order the questions in a Question Bank

## Usage

```
ran.exam(Qbank)
```

### Arguments

| | |
|---|---|
| Qbank | Question Bank List |

### Details

randomly re-order the questions in a Question Bank

### Value

Question bank

### Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

### See Also

Get.testbank

### Examples

```
data(QBANK1)
NEWQB = ran.exam(QBANK1)
```

---

## readSCANTRON *Read Scantron*

---

### Description

Read UNC scantron

### Usage

```
readSCANTRON(fn = "t9543b.raw.csv", nq = 50, istart = 6)
```

### Arguments

| | |
|---|---|
| fn | character, name of digital file with raw scores |
| nq | integer, Number of questions to read |
| istart | integer, start of column for first question |

### Details

The data is scanned by machine. If a student marks on the exam past the correct number of questions, the machine assumes there are legitimate repsonses beyond the key.

## Value

list:

| | |
|---|---|
| Nstudents | number of students |
| Nquestions | number of questions |
| Nams | names of students |
| ids | Ids of students |
| studans | matrix, student answers |
| key | key for grading |

## Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

## Examples

```
## Not run:

datadir =  "./DATA"
rawfn1 = paste(datadir,'t6200a.raw.csv', sep="/")

B1 = readSCANTRON(rawfn1)

## End(Not run)
```

---

# rename.answers *Rename Answers*

---

## Description

Rename the answers on a Question Bank

## Usage

```
rename.answers(Qbank, newnames = letters[1:26], sep = ") ")
```

## Arguments

| | |
|---|---|
| Qbank | Question Bank |
| newnames | vector of new names |
| sep | separator between name of Answer and Answer String |

## Details

Takes the given list of questions, and returns same list with answers rpefaces by a different set of itemizers

## Value

Question Bank List

## Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

## See Also

Get.testbank

## Examples

```
data(QBANK1)


newnames=letters[1:26]
NEWQB = rename.answers(QBANK1, newnames=newnames )
NEWQB[[35]]

newnames=1:26
NEWQB = rename.answers(QBANK1, newnames=newnames )
NEWQB[[35]]

newnames=LETTERS[1:26]
NEWQB = rename.answers(QBANK1, newnames=newnames )
NEWQB[[35]]
```

---

# repair.id *Repair Poorly Bubbled Student ID*

---

## Description

Repair Poorly Bubbled Student IDs by matching to a reliable data base of names and IDs. Routine offers a set of possible matches if several may be appropriate.

## Usage

```
repair.id(sisroster, scrfn)
```

## Arguments

| | |
|---|---|
| sisroster | Reference Data set |
| scrfn | Scantron Output |

## Details

Program searchers for missing ID's by attempting to match up names.

## Value

| | |
|---|---|
| newid | New vector of IDs that correspond to the scantron input |

## Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

---

# ridNA *Replace NA with somehting else*

---

## Description

Replace NA with somehting else

## Usage

```
ridNA(z, temp)
```

## Arguments

| | |
|---|---|
| z | vector |
| temp | replacement |

## Value

vector with NA's replaces

## Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

## Examples

```
z = 1:10
z[z>8] = NA

ridNA(z, 0)
```

`scramble.answers` *Scramble Answers*

## Description

Randomly rearrange answers within a question of a test bank

## Usage

```
scramble.answers(Qbank)
```

## Arguments

| | |
|---|---|
| `Qbank` | Question Bank (list of Questions) |

## Details

Takes the given list of questions, and returns same list with answers scrambled.

## Value

Question Bank List

## Note

Since some question require that the answers be ordered in a certain way, these are not Randomized in this scrambling process. These include:

`c("all of the above", "none of the above", "None of these are correct", "all of the choices are correct", "All of the choices are correct", "Both choices are correct", "None of the choices are correct", "Both of the choices are correct", "All of these are correct", 'Neither of these are correct')`

## Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

## See Also

Get.testbank

## Examples

```
data(QBANK1)


QBANK1[[35]]

 NEWQB = scramble.answers(QBANK1)
NEWQB[[35]]
```

SEARCHbank *Search Question Bank for Keyword*

## Description

Search a question bank for key words.

## Usage

```
SEARCHbank(gw, y = "humidity")
```

## Arguments

| | |
|---|---|
| gw | Question Bank |
| y | key word |

## Details

Dumps to the screen the questions that match the key.

## Value

Side effects - dumps to the screen. returns a vector of questions that match.

## Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

## See Also

seebank,Get.testbank,SELbank,COMPbank

## Examples

```
## Not run:
####  seebank program is interactive -
data(QBANK1)
SEARCHbank(QBANK1, "humidity" )

## End(Not run)
```

## seebank *Print out a bank of questions*

### Description

Prints out a bank of questions, one at a time

### Usage

```
seebank(QB)
```

### Arguments

| | |
|---|---|
| QB | QuestionBank Structure |

### Value

Side effects

### Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

### Examples

```
## Not run:
####  seebank program is interactive -
data(QBANK1)
seebank(QBANK1)

## End(Not run)
```

## seequestions *See Questions Sequentially*

### Description

Print questions to the screen

### Usage

```
seequestions(QB)
```

## Arguments

| | |
|---|---|
| QB | Question Bank |

## Details

Prints just the questions to the screen.

## Value

Prints to screen

## Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

## See Also

seebank

## Examples

```
## Not run:
LF = list.files(path="/home/lees/Class/GEOL_105/TESTBANK/EXAM_1", pattern="txt", full.names=TRUE )

kbank = vector(mode='list')
######   read in the question banks, each in one file
for(i in 1:length(LF))
  {
    h = Get.testbank(LF[i])
    kbank[[i]] = Get.testbank(LF[i])

  }
names(kbank) = LF

cbind( seequestions(kbank[[1]]) )




## End(Not run)
```

SELbank *Select Questions from a bank*

## Description

Select, random set of questions from a test bank.

## Usage

```
SELbank(QB, N, xclude=NULL)
```

## Arguments

| | |
|---|---|
| QB | Question bank |
| N | integer, number of questions to select |
| xclude | integer vector, index of questions to exclude, default=NULL |

## Details

Progam uses sample to get a random perturbation, and then pulls out the first N questions

## Value

Question bank

## Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

## See Also

Get.testbank

## Examples

```
## Not run:
LF = list.files(path="/home/lees/Class/GEOL_105/TESTBANK/EXAM_1", pattern="txt", full.names=TRUE )

kbank = vector(mode='list')
######   read in the question banks, each in one file
for(i in 1:length(LF))
  {
    h = Get.testbank(LF[i])
    kbank[[i]] = Get.testbank(LF[i])

  }
names(kbank) = LF
Kbank =  vector(mode='list')
```

```
for(i in 1:length(kbank))
  {

Kbank = c(Kbank, kbank[[i]])

  }

##########   get 50 sample questions
NEWQB = SELbank(Kbank, 50)


## End(Not run)
```

---

show.dist *Show Distribution of Grades*

---

## Description

Show Distribution of Grades

## Usage

```
show.dist(W)
```

## Arguments

| | |
|---|---|
| W | list output of do.grades |

## Details

Print out the distribution of letter grades

## Value

Side Effects

## Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

## See Also

do.grades

## Examples

```
g = rnorm(n=130, m=82, sd=10)
g[g>100] = 100
g[g<1] = 1

B = boxplot(g)

divs = c(min(g), B$stats[1:4] + diff(B$stats)/2, max(g) )
D1 = do.grades(g, divs=divs, tit="GEOL 105 Exam 1")

show.dist(D1)
```

---

# subsetbank *Subset a Question Bank*

---

## Description

Extract a subset from a question bank

## Usage

```
subsetbank(QBANK, sel)
```

## Arguments

| | |
|---|---|
| QBANK | Question Bank List |
| sel | integer vector of index to specific questions |

## Details

for selecting specific questions

## Value

Question Bank with selections

## Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

## See Also

SELbank, COMPbank

## Examples

```
## Not run:
LF = list.files(path="/home/lees/Class/GEOL_105/TESTBANK/EXAM_1", pattern="txt", full.names=TRUE )

kbank = vector(mode='list')
######   read in the question banks, each in one file
for(i in 1:length(LF))
  {
    h = Get.testbank(LF[i])
    kbank[[i]] = Get.testbank(LF[i])

  }
names(kbank) = LF
Kbank =  vector(mode='list')

for(i in 1:length(kbank))
  {

Kbank = c(Kbank, kbank[[i]])

  }

##########   get 50 odd numbered sample questions

isel = seq(from=1, to=100, by=2)

oddset1 = subsetbank(Kbank, isel)


## End(Not run)
```

---

UNCkeytron *Create a KEY for the scantron*

---

## Description

Create a KEY for the scantron

## Usage

```
UNCkeytron(g, fout, LAB = "KEY")
```

## Arguments

| | |
|---|---|
| g | vector of correct answers |
| fout | output file name |
| LAB | Label to print on key |

## Details

Given a vector of correct answers the program will create a postscript file with a facsimile of the scantron used for examinations at UNC Chapel Hill. The Bubbles will be filled and can be used to prepare a number 2 pencil version.

## Value

Side effects

## Note

Currently only eps outputs - future versions may be different. At this time, the code creates postscript code, which can be converted to png, pdf or other formats with software outside of R. In linux I use a perlscript,

```
/home/lees/Progs/Perl/ps2png.prl files.eps
```

which, in turn, calls, epstopdf and

```
gs -dBATCH -sDEVICE=png16m -dNOPAUSE -r200 -sOutputFile=$outpf $inpf
```

## Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

## See Also

getKEY

## Examples

```
## Not run:

fkeyA = "/Users/lees/SCANTRON/A.FINAL.key"
fkeyB = "/Users/lees/SCANTRON/B.FINAL.key"
 FKEY1 = getKEY(fkeyA)

    FKEY2 = getKEY(fkeyB)

UNCkeytron(FKEY1, "AKEYfinal.eps", "A KEY final")
UNCkeytron(FKEY2, "BKEYfinal.eps", "B KEY final")



## End(Not run)
```

version.exam *Create 1 instance of a specific Exam*

## Description

Create 1 instance of a specific Exam

## Usage

```
version.exam(Qbank, V, exnumber = "Exam 1", seqnum = "2", examdate ='',
instructor="",  course="", instructions="", SAMP=TRUE, ncol=2)
```

## Arguments

| | |
|---|---|
| Qbank | question bank |
| V | Character string output files |
| exnumber | Exam number |
| seqnum | Version Number |
| examdate | Date of the examination |
| instructor | character, name of teacher |
| course | character, nqme of course |
| instructions | vector of character strings |
| SAMP | logical, if TRUE a random ordering to the questions is produced |
| ncol | number of columns on page, default=2 |

## Value

Side Effects

## Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

## See Also

ran.exam, make.exam, prep.exam

## Examples

```
##  the example creates files on the local system - thus not run
## Not run:
data(QBANK1)

examdate="THURS Sep 20 2007"

version.exam(QBANK1, "exam1A" , exnumber="Exam 1", seqnum="1", examdate=examdate)
```

```
##################
examdate=date()

seqnum="1"
exnumber="Exam 1"
V = "exam1A"
outtex = paste(sep=".",V, "tex" )
outMAST  = paste(sep="", V, "MAST" )


MASTtex  = paste(sep=".", outMAST , "tex" )

outsolut  = paste(sep="", V, "solutions.tex" )
Me = "Jonathan M. Lees"

course="GEOL 105"

examname=paste(sep=" ", exnumber, "Seq", seqnum)

K = length(QBANK1)

instructions=c(
paste(sep=" ", "There are",K," number of questions."),
"Answer all questions.","Use number 2 pencil",
"Mark each box clearly.")

version.exam(QBANK1, "exam1B" , exnumber="Exam 1", seqnum="B",
examdate=examdate, instructor=Me, course=course , instructions=instructions)


## End(Not run)
```

---

wrist *Write Histogram*

---

## Description

Write grades on Histogram

## Usage

```
wrist(DB)
```

## Arguments

| | |
|---|---|
| DB | Output of do.grades |

## Details

Used internally in plotting programs

## Value

Side Effects

## Author(s)

Jonathan M. Lees<jonathan.lees@unc.edu>

## See Also

do.grades

## Examples

```
g = rnorm(n=130, m=82, sd=10)
g[g>100] = 100
g[g<1] = 1

B = boxplot(g)

divs = c(min(g), B$stats[1:4] + diff(B$stats)/2, max(g) )
D1 = do.grades(g, divs=divs, tit="GEOL 105 Exam 1")

hist(g)
wrist(D1)
```

# Index