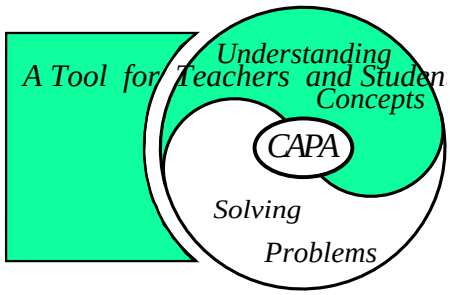


CAPA

Computer-Assisted Personalized Approach
for Assignments, Quizzes and Examinations.

CAPA System User's Manual

Version 5.0

Y. Tsai, E. Kashy, D.J. Morrissey, N.E. Davis, G. Albertelli, and F. Berryman
Michigan State University, East Lansing, MI 48824, USA.

August 3, 2000

1 Installation of *CAPA*

- Expand the *CAPA* file by typing:
  `uncompress` *Your_OS*`.tar.Z`
  at the prompt.
- Untar the *CAPA* directories by typing:
  `tar xf` *Your_OS*`.tar`
  at the prompt.
- This will create a *DIST5* directory which will contain the following items:

```
drwxr-xr-x   2 owner group         8192 May 21 20:05 LinkToBin
-rw-r--r--   1 owner group      1862901 May 21 20:05 Manual46.ps
drwxr-xr-x   4 owner group         8192 May 21 20:05 PutInCAPA5
drwxr-xr-x   3 owner group         8192 May 21 20:05 PutInPublic
drwxr-xr-x   3 owner group         8192 May 21 20:05 PutInPublicWeb
-rw-r--r--   1 owner group        11489 May 21 20:05 README.txt
-rw-r--r--   1 owner group        10163 May 21 20:05 UPGRADE.txt
drwxr-xr-x  22 owner group         8192 May 21 20:05 demolibrary
```

- Print the file *README.txt*. This gives instructions for system installation. If you are upgrading to *CAPA* version 5.0, print the file *UPGRADE.txt* instead and follow those instructions for a system upgrade.
- When checking your permissions with those given in either the *UPGRADE.txt* file or *README.txt* file, be informed that your file sizes may differ.
- **Some Notes:**
  - The LaTeX and dvi programs must be properly installed on the host machine in order for Quizzer to format the problem sets. See your system administrator for help in installing these programs.
  - For a broad range of terminal emulators to be able to access the system[1], it is necessary to modify the `/etc/termcap` file to accommodate the appropriate vt100 terminal attributes, including handling non-standard nomenclature used by terminal emulation programs on Macs and PCs. A notable example is the widely used *tn* program that is part of FTP Software, Inc.'s PC/TCP package. This program negotiates its terminal type with the host computer using the prefix `dec-` for the `vt` family of terminals (*e.g.,* it requests terminal type `dec-vt100` instead of `vt100`). This prefix, if unknown to the Unix host, may cause the negotiation to result in an incorrect terminal type or to fail completely.

[1] by G. Perkins, MSU Physics-Astronomy Dept.

The line in the standard `/etc/termcap` file that reads:

```
d0|vt100|vt100-am|vt100am|dec vt100:\
```

should be edited to read:

```
d0|vt100|vt100-am|vt100am|dec-vt100|dec vt100:\
```

thereby establishing `dec-vt100` as an official synonym for `vt100`. Note that the "`dec-vt100`" entry is different from the "`dec vt100`" entry. We advise adding the new alias rather than changing the existing alias and possibly causing other problems. A similar change can be made on other operating systems which use the `/etc/termcap` method of terminal definition. Unix systems using the *terminfo* program instead will support similar minor changes to the appropriate file or files in the `/usr/lib/terminfo` directory tree (consult your system administrator for details). Be prepared to make similar changes in case other terminal emulation programs at your site use their own non-standard terminal type nomenclature.

- The *CAPA* system should now be completely installed. We recommend taking the guided tour, skimming through the manual, and reading about the *CAPA* functions before coding problems and building sets.

2 *CAPA* Software Components

There are nine application codes in the complete system. They are: Quizzer, Grader, Manager, Capalogin, Capasbin, Capahtml, Qzparse, Capautils and AllCapaID. The Quizzer, Grader, and Manager applications are tools for the instructor to create problem sets, grade exams and hand-scored problems, and manage the class. It was necessary in earlier versions of *CAPA* to use the Qzparse, AllCapaID, and CapaUtils to perform certain tasks. Now however, almost all of their functions can also be found or accessed in either Quizzer, Grader, or Manager. The other three applications are necessary for the student interface. The Capalogin code is executed automatically for each student telnet session. Similar in function are the Capahtml and Capasbin programs executed by the web server when a student uses web browser to answer the problem set. In order for many of these components to work properly, the instructor customizes the necessary *capa.config* file which must reside in each classname folder.

2.1 The *capa.config* File

A file called *capa.config* located in the class directorry contains general configuration information needed for the *CAPA* system. In the usual installation, the class directory contains the homework assignment sets. When generating statistics and reports, Manager requires the user to select a *capa.config* file, and assumes that the user will go the the class directory to select that file. The *capa.config* is divided into sections containing configurable parameters so that instructors can customize the *CAPA* system.

- **Class Structure:** The first section contains information related to the organization of a course. It defines the paths pointing to other possible course components, such as exam_path, quiz_path, supp_path, and others_path. Below is the exact text as it appears in the *capa.config* file. Every comment begins with a "#" symbol. The comments explain how to customize the file.

```
# Comments in this file begins with a # mark in the first column
# when setting a value there MUST be a space before and after the =

# ================ class structure ============
# Defined below are the paths to the various portions of the class.
# (Note: The homework is usually in the class directory,
#           /usr/users/teacher/CAPA51/phy183f0
# The file  /usr/users/teacher/CAPA51/phy183f0/capa_config is used by
# manager to generating stats such as the student course profile.
#
# Possible other "classes" (paths) a user MAY define are exam_path, quiz_path,
# supp_path (supplementary problems), and others_path. Data from these 4 paths
# are used by Manager and included in the stats generated (for example, when
# generating a summary of a student's performance, a Student Course Profile)

quiz_path       = /usr/users/teacher/CAPA51/qph183f0
exam_path       = /usr/users/teacher/CAPA51/xph183f0
supp_path       = /usr/users/teacher/CAPA51/xpc183f0
others_path     = /usr/users/teacher/CAPA51/fci183f0

# A popular feature we have used is to allow students (optionally) to earn
# partial credit on Mid-term exams. Since our exams are mostly or totally
# computer scored, we have allowed students to correct their exams and earn
# partial credit: At the end of the exam, students pick up a different version
# of the  exam as if it were a homework assignment, and enter answers in the
# computer with a short deadline (2-3 days). They must do all the problems, and
# earn partial credit on the improvement in score.
# Example: For 25pct partial credit, a student with 60% on exam 1 (i.e.,
# set1.qz in xph183f0) and  90% on the corrections 1 (set1.qz in xpc183f0),
# gets                  60+0.25*(90-60)=67.5pct
# To use that feature in CAPA, the path where the corrections sets are entered
# must be defined.

correction_path = /usr/users/teacher/CAPA51/xpc183f0

# Note that we used the supp_path for that purpose: set1.qz in xph183f0 has the
# same questions as set1.qz in xpc183f0, but the former is proctored and
# students fill out scantron forms, while for the latter, students enter as
# they would for homework, but in the xpc183f0 (eXamPartialCredit) "class".
```

- **Extrapolate Term Score:** An applet that allows students to extrapolate their term score can be enabled, and will include the Partial Credit by Corrections discussed above. A button on the main menu of the homework web page initiates the applet

```
# ================ Extrapolate Scores Applet ================
# Server where the applet resides:
capa_server = capa2.nscl.msu.edu

# Width and height of the applet:
tscore_width = 500
tscore_height = 450

# yes means show the applet, no inhibts the display of the applet
term_score_applet = no

# The weights used in term score calculation formula:
#    term_score = homework_weight * sum of homework + quiz_weight *
#    sum of quizes + exam_weight * sum of exams + final_weight * final exam
# These values will in most cases add up to 1.
homework_weight   = 0.3
quiz_weight       = 0.05
exam_weight       = 0.3
final_weight      = 0.35

# This specifies the weight of the correction exam:
#    corrected exam score = raw exam score + correction_weight *
#    (correction score - raw exam score)
# If correction score is less than raw exam score, no correction is taken.
correction_weight = 0.25

# final_exam_set_number specifies what set number will be or is the
# final exam in the class (directory) defined by exam_path.
final_exam_set_number = 4

# homework_count is the total number of homeworks that will be given
# to the students
homework_count        = 12

# quiz_count is the total number of quizzes that will be given to the
# students
quiz_count            = 20
```

- **Configuring the "Discuss" forum:** A discussion forum for posting questions and replies associated with problems in assignments can be enabled for each set, using the "set db Header" menu.

```
# =============== capadiscuss configuration ====
# "Admin" sections: Teaching staff listed in Admin sections can delete
# and hide messages (instructor/TA sections). Examples:
# admin_section = 033
# admin_section = 33
# admin_section = 033,12,099,2
admin_section = 099

# You can ban particular users (problem students) from using the "discuss"
# forum page.  Examples:
# banned_users = A12345677
# banned_users = a12345677
# banned_users = A12345677, a12345678,A12345666
banned_users = A12312312
```

- **Manager Configuration:** Data to be included in various reports and graphs:

```
# ================== Manager configuration ====
# Manager will compile statistics with data from set1.db,
# set2.db and so on until the setX.db file which does not
# exist or until it reaches the set number specified below.
# Defining the limits with a number of 99 will make the program
# to include all scores from  all setX.db files up to set 99.
homework_scores_limit_set = 99
exam_scores_limit_set = 99
quiz_scores_limit_set = 99
supp_scores_limit_set = 99
others_scores_limit_set = 99

# Note that when calculating stats, Manager will ask the user to select a
# capa_config file. Manager will assume that the capa_config file selected
# resides in the directory containing the homework sets for which the
# limit "homework_scores_limit_set" is set above.
```

- **Command Specifications**: This section specifies the absolute path to commands that are used by the *CAPA* system, such as `latex`, `dvips`, `mail`, `allcapaid`, and `qzparse`. It should be unnecessary to change these.

```
# ================ Command specifications =====
#
#  When specifying a command, use double quotes.
#
latex_command      =  "latex"
dvips_command      =  "dvips"
mail_command       =  "mailx"
allcapaid_command  =  "allcapaid"
qzparse_command    =  "qzparse"
answers_command    =  "answers"
```

- **Printing Configuration**: The next section lets the instructor identify the printers he/she will use to print assignments, exams, quizzes, etc. The command used to choose one or two sided printing should be specified in this section also. The commands listed below may differ for different operating systems. Check with your system administrator to determine both the printer name and print command syntax.

```
# ================= Printing configuration ====
#
#  -printer list-
#  add available printer name here
#   they will appear in a selection list in the order listed here
#   multiple printer quese can be defined
#
printer_option  = "print_queue1"
printer_option  = "print_queue2"

#  -print command-
#  The predefined variables: $PS_file and $Printer_selected
#  indicate the file to be printed and the printer queue to be
#  printed to.
#
lprOneSided_command = "/usr/bin/lpr -P$Printer_selected"
lprTwoSided_command = "/usr/bin/lpspr -K2 $PS_file | /usr/bin/lpr -P$Printer_selected"
```

- **Capalogin/Web Settings:** This section specifies options for the Capalogin shell and the Web display. It is unnecessary to change these.

```
# ================= capalogin/web settings =========
#
# Specify the capalogin goodbye message delay in seconds.
# This if for the telnet interface only.
capalogin_goodbye_delay  = 4

# Specify the capalogin inactivity delay time in minutes.
# Default value is 60 minutes.  0 means never time out.
# This if for the telnet interface only.
capalogin_inactivity_delay  = 25

# If set to "none", then the summary score will not show.  Any other
# value or not defined will cause the summary score to be displayed.
# This if for the telnet interface only.
#capalogin_show_summary_score = show

# Used to define how many problems to be displayed on the webpage.
# Default is all. If specified must have and integer argument, otherwise
# uses default value.
#num_questions_per_page = 26

# Used to define the length of the status line on the top of the webpage.
# If there are more problems than this, multiple status lines are created.
# By default, one line of the length of the # of problems in the set
# is created.
#web_status_line_length = 26

# If set to "no", then the viewing of term summaries is not an option.
# Default is yes.  This is for both web and telnet interface.
#term_summary_button = yes

# Used to define the cgi path, if unspecified, defaults to capa-bin
#capaweb_cgibin_path = /path/to/cgi
```

- **Quizzer Settings:** The next section is related to an option in Quizzer. Everything between "BeginStandardQuizzerHeader" and "EndStandardQuizzerHeader" is inserted into the *.qz* file when the user clicks on the Std. Header button while using the Quizzer application.

```
# ================ Quizzer settings ==========
#Everything between BeginStandardQuizzerHeader and
#EndStandardQuizzerHeader is inserted into the .qz file when using the
#Std. Header button in Quizzer

BeginStandardQuizzerHeader
//CAPA library problems NOT included in the demolibrary are copyrighted
//by the author, institution, publisher, etc..  By using these materials,
//the user agrees to:
// 1) Protect the problem source code files from unauthorized copying.
// 2) Limit access of the problem source material to teaching staff.
//    This includes installing the CAPA system securely (please see
//    README%security.txt from the CAPA distribution).
// 3) The user is free to mix, cut and paste, modify, adapt, delete,
//    improve, etc. the problems and graphics for her/his own use.
//
//See quizzer "Info" or http://capa.msu.edu/license for more information
//about terms and conditions.
//
/IMP "/demolibrary/Tools/StdMacros"
/IMP "/demolibrary/Tools/StdUnits"
/IMP "/demolibrary/Tools/StdConst"
/IMP "/demolibrary/Tools/GreekWeb2"
/IMP "HWTop"
EndStandardQuizzerHeader
```

- **Unit Section:** This is divided into three subsections.

  1. **Base Units:** The capa.config file defines the base units first. It begins with the special marker `<< Base Unit >>`. Each line contains a definition of a base unit which consist of (1) a full name, (2) a symbol representing that base unit, (3) and a comment beginning with a "#" character. All three fields are required and are space delimited (no tabs). There is a 47 character limit to the unit name, 15 character limit for the unit symbol and a total of 32 base units can be defined.

```
# ======================= Unit section =============
# Base Unit
# name      symbol     comment

<< Base Unit >>

meter           m      # length
kilogram        kg     # mass
second          s      # time
ampere          A      # electric current
kelvin          K      # thermodynamic temperature
mole            mol    # amount of substance
candela         cd     # luminous intensity
```

2. **Prefixes:** The next subsection of units includes prefix symbol definitions. It consists of three fields on each line, separated by white spaces or tab characters. This section needs to begin with the marker `<< Prefix >>`.

```
#   Prefix     symbol     factor
<< Prefix >>

yotta            Y         10^{24}
zetta            Z         10^{21}
exa              E         10^{18}
peta             P         10^{15}
tera             T         10^{12}
giga             G         10^9
mega             M         10^6
kilo             k         10^3
hecto            h         10^2

deci             d         10^-1
centi            c         10^-2
milli            m         10^-3
micro            u         10^-6
nano             n         10^-9
pico             p         10^{-12}
femto            f         10^{-15}
atto             a         10^{-18}
zepto            z         10^{-21}
yocto            y         10^{-24}
```

3. **Derived Units:** The final part of the unit subsection is the definition of derived units. It begins with the special marker `<< Derived Unit >>` and consists of FOUR required fields on each line. The fields are as follows:

   Field (1) is the unit name; maximum of 47 characters with no spaces.
   Field (2) is the symbol representing the unit; maximum of 15 characters with no spaces.

   Fields (1) and (2) are separated by spaces. Field (3) is the way the unit is calculated from base units or other previously defined derived units. Calculations of derived units use the standard operands for multiplication (*), division (/), and power (^). Addition and subtraction are not allowed. Numbers can be in floating point or exponential format. The comment, field (4), is **required** and begins with a hashmark (#).

```
# Derived Unit
# name       symbol     unit          comment
# name must be in oneword
<<Derived Unit>>

gram            g      0.001 kg      # mass
hour            hr     3600. s       # time
minute          min    60 s          # time
day             day    24.0 hr       # time
year            yr     365.24 day    # time
pound           lb     0.45359237 kg # mass
ounce           oz     1.77185E-3 kg # mass
inch            in     2.54 cm       # length
foot            ft     12 in         # length
mile            mi     5280 ft       # length
yard            yd     0.9144 m      # length
nautical_mile   n_mi   6080 ft       # length, nautical mile (UK)
rood            rood   1210 yd^2     # area, rood
acre            acre   4840 yd^2     # area, acre
hertz           Hz     1/s           # frequency
litre           L      10^3/cm^3     # volume
newton          N      m*kg/s^2      # force
pound_force     lbf    4.44822 N     # force
dyne            dyn    1E-5 N        # force
pascal          Pa     N/m^2           # pressure, stress
bar             bar    1E5 Pa          # pressure
mmHg            mmHg   1.33322E2 Pa    # pressure, millimeter of mercury
torr            torr   1 mmHg          # pressure
atm             atm    760 torr        # standard atmosphere
joule           J      N*m             # energy, work, heat
electronvolt    eV     1.6021892E-19 J # energy
calorie         cal    4.1868 J        # energy
Btu             Btu    1.05506E3 J     # energy
watt            W      J/s             # power, radiant flux
coulomb         C      A*s             # electric charge
volt            V      J/C           # electric potential, electromotive force
ohm             ohm    V/A           # electric resistance, use this in /ANS
ohm             ohms   V/A           # electric resistance
ohm             Ohm    V/A           # electric resistance
ohm             Ohms   V/A           # electric resistance
siemens         S      1/Ohm         # electric conductance
farad           F      C/V           # electric capacitance
tesla           T      V*s/m^2       # magnetic flux density
weber           Wb     V*s           # magnetic flux
henry           H      V*s/A         # inductance
radian          rad    m/m            # plane angle
degree          deg    1.745329E-2 rad # plane angle (Pi rad=180 deg)
steradian       sr     m^2 /m^2        # solid angle
lumen           lm     cd*sr         # luminous flux
lux             lx     cd*sr/m^2     # illuminance
becquerel       Bq     1/s           # activity (radioactive)
gray            Gy     J/kg          # absorbed dose (of radiation)
sievert         Sv     J/kg          # dose equivalent (dose equivalent index)
astroUnit       AU     1.49598E11 m  # mean distance earth to sun
```

- **Note:** The coded units are interpreted in the order of basic unit, derived unit, then prefix. For example, "T" will be matched against "tesla" instead of considered the prefix "T". Another example is that "min" will match "minutes" instead of treated as a combination of the prefix "m" and units "in"

2.2 Quizzer

- **Location:** The Quizzer application can reside in any directory and can be linked to a directory within the user's path. For example, linking Quizzer to */usr/local/bin* as was directed in the installation instructions makes the application available to all users on the machine.
- **Function:** The major functions of the Quizzer application are:
  - Create and edit problem set files (for homework, quizzes, and exams), named *set1.qz*, *set2.qz*, etc. which contain ASCII text only.
  - Edit existing question files in ASCII text.
  - Calculate and preview the answers generated by the problem set code.
  - Preview the resulting problem sets in the following three modes:
    1. Enscript Mode: See how the problem set appears to students who log into *CAPA* through a terminal.
    2. Tex: Preview the LaTeX typeset.
    3. Web: Preview the resulting problem set showing the coded HTML flags.
  - Create a *setx.dvi* and view an image of the printe student problem set.
  - Generate the database file (*setx.db*) which contains the time constraints on the availability of the problem sets to students for a given assignment. Each problem set must have an opening-date, a due-date, and a date when answers are available.
  - Print the problem set(s) for a student, for one or more sections, or for the entire class. (We recommend printing the *CAPA* ID on student papers. The *HWTop* file included in the distribution does this already.)
  - Analyze set: This gives the low end, high end, and distribution of answers for problems in a set. This can be done for all students in a class or for a random sample of students.
- **Mandatory Files**: *set1.qz*, *TeXheader*, *TeXfooter*, and *classl*.
  - **set*x*.qz**: Problem set files are the files containing the coded problems created by the instructor for each given assignment. They are labeled *set1.qz*, *set2.qz*, etc. There is a limit to 99 problem sets for each class.
  - *TeXheader* and *TeXfooter:* These files specify the information needed by the LaTeX formatter to create the document for printing.
  - *classl*: The class list file for each course using the system is the *classl* file. The maximum number of students in one class is 4096. The format of the entries in the class list file is strict and must correspond exactly to that in the following example.

```
12345678901234567890123456789012345678901234567890123456789012345678901234567890123456789012345...
nsc 121   001 A12345678 Albertelli, Guy II                  albertel@pilot.msu.edu
nsc 121   001 A23592320 Berryman, Felicia V.                berryma5@pilot.msu.edu
nsc 121   001 A73336318 Kashy, Edwin N.                     kashy@nscl.msu.edu
nsc 121   001 A87654321 Student, Jaimie .
```

Note: The first row of digits is **not** part of the file but is given here to locate the information in the file.

* The fields used by this system are:
  1. Section number which appears in columns 11 to 13
  2. Student Number in columns 15 to 23
  3. Student Name in columns 25 to 35
  4. Student Email in columns 61 to 101

  The maximum number of characters in the Student Name field is 30. The course acronym and number (in columns 1 to 10) are not used by the system, but are convenient for recordkeeping. The format of the file must be exact. The section number is used for printing.
* Each line, including the last line, must end with a carriage return.
* The period (.) added for students with no middle names has facilitated importing *CAPA* summaries into some spread sheet programs (i.e. Lotus) when preparing final grades for a course.

- *records* directory: This folder must be present as a sub-directory of the class directory. Quizzer writes the date information from *setx.qz* in *records/datex.db*. Capalogin saves student input from the telnet session for *setx* in *records/logx.db*. Capasbin saves student input for the web session for *setx* in *records/weblogx.db*. The summary information from both *logx.db* and *weblogx.db* is written to *setx.db* . Both Capalogin and Capasbin create a *setx* folder which contains other folders called *problemy* if the set has a `/SUBJECTIVE()` question. This folder contains files of the `/SUBJECTIVE()` answer essays with student numbers as their names. The duration of student telnet sessions is saved in the file *records/duration.db*. As students login via telnet, a file named *active.log* is created. It is used to limit the number of concurrent telnet sessions per student. Also created are two files, *submissionsx.db* and *websubmissionsx.db* which records all student entries. For a 500 student class having 11 assignments you can expect this directory to grow to approximately 35Mb in size.

  The above files are also used by Grader and Manager for generating statistics and summaries.

- Qzparse is an application separate from from Quizzer. It's functions though can now be done using Quizzer in *CAPA* 5.0.

  – **Functions**: Qzparse can be used to generate the .tex files needed to prepare problem sets in a batch mode without using **quizzer**. It can create one output file for an entire section or a multiple number of sets for a given student. The output file can then be processed by LaTeX and Dvips to produce a postscript file for printing. The options of Qzparse are displayed by typing `qzparse -h`.

```
capa2.nscl.msu.edu> qzparse -h
USAGE: qzparse [ -[T|H|A][a|b] ] [-Sec [n|n:m] | -Stu sn [-o filename] ]
          [ -Set [n|n:m] ] [-c path_to_class] [-d outputdirectory]
 Example 1: qzparse -Tb -sec 2:3 -set 2:5
            will generate tex files with both questions and answers
            for sections 2 to 3, sets 2 to 5
 Example 2: qzparse -Ha -stu A12345678 -set 3
            will generate html files with answer only
            for student A12345678 set 3
          -T     = tex   mode
          -H     = html  mode
          -A     = ascii mode
                 = default question only
            a    = answer   only
            b    = both question and answer
          -Sec 3 = for section 3
          -Sec 3:7    = from section 3 to section 7
          -Stu A12345678 = for a specified student
          -Set 1 = output set 1
          -Set 3:4 = output from set 3 to set 4
          -c class_path
          -o output_filename_with_absolute_path (only for a student)
          -d directory_to_create_files_in (default is class_path/TeX)
-------This is version 5.0.3 @ 11:23-Apr-07-1999
------------------------------------------------------
```

- Examples:
  1. For preparing TeX files for set 4 papers for all students in sections 1 thru 45:
     `qzparse -T -sec 1:45 -set 4`
  2. For preparing papers which only contain the answers of a particular set for all students in section 3:
     `qzparse -Ta -sec 3 -set 4`
  3. For preparing set 1 thru 5 for a student with student number A87654321:
     `qzparse -T -stu A87654321 -set 1:5`
  4. For preparing set 1 for a student with student number A87654321, with the output file *A87654321.tex* placed in the */usr/users/teacher* directory.
     `qzparse -T -stu A87654321 -set 1 -d /usr/user3/teacher`
- **Files**: Qzparse uses the same files as Quizzer. In addition, running Qzparse will create a *TeX* sub-directory in the class directory. Qzparse will write output files such as *section1-set1.tex* or *a87654321.tex* to this directory. The *.tex* files must be passed through LaTeX and the dvips codes to make postscript files for printing. (See guided tour of Qzparse for explanations on how this is accomplished.)

2.3 Capalogin, Capahtml, and Capasbin

- Capalogin
  - **Function**: The main function of the Capalogin code is to handle remote sessions of the students who access CAPA with VT100 terminals. The code is run instead of a UNIX shell in order to allow large numbers of students to easily login while controlling their access to the data files. Capalogin queries active.log if the student is already logged-on in this class. If

the student is already logged in, then a warning message is sent (see below) and the student is allowed in to the system. If the student has opened 2 sessions without exiting properly, the student is not allowed into the system and is sent a different warning message.

– **Files:** The instructor can send general information to the entire class by typing messages into certain files. The login-specific files, *welcome.msg* and *goodbye.msg*, (and if needed the *help.msg*, *second-login.msg*, and *third-login.msg*), can be edited to send information to students, reminding them of deadlines or of exam dates, telling them to disregard a particular problem (when a serious error has been made in coding), etc. Note that the instructor can code hints and explanations for individual problems directly in the set$x$.qz files. Files and their functions are listed below:

welcome.msg	⇒ The message displayed to the student upon login.
help.msg	⇒ The message displayed when the student selects the menu item `Help`.
goodbye.msg	⇒ The message displayed after a student selects menu item `eXit`.
second-login.msg	⇒ The message displayed when a student already is logged on and tries to begin a second session.
third-login.msg	⇒ The message displayed when a student already is logged on twice and tries to begin a third session.
capa.config	⇒ Controls the functions as described in the *capa.config* section.

- Capahtml
  - **Functions**: The major functions of Capahtml are to:
    * Authenticate the student number and *CAPA* ID entered by the student and generate the main menu page after verifying them.
    * Produce the page that contains the corresponding problem set when a student selects "Try current set" button from the main menu page.
    * Display a summary of student grades when "Term summary" button is selected.
- Capasbin
  - **Functions**: The major functions of Capasbin are to:
    * Check the correctness of answers submitted by the student.
    * Give the appropriate response to a student's entry and display the correct answer when the entry is satisfactory.

2.4 Grader

- **Functions**: The major functions of Grader are to:
  - Display a summary of student grades and *CAPA* ID numbers for any problem set.
  - Generate the reports found in the *records* directory.
  - Grade subjective answers, such as essays entered in through the *CAPA* system.
  - View a summary of a students login file (the problems correct vs. incorrect) with their *CAPA* ID for that set.
  - Grade hand-graded problems for a student while viewing the correct answers.
  - Generate grade reports for a student, a section, or for the entire class.

– Excuse a problem for the class.

- **Files**: Grader reads the *classl* file and the files in the *records* directory generated by Quizzer and Capalogin and/or Capasbin. The reports created by Grader are put into the class directory. The reports for the entire class are labeled *ClassSet1.rpt* and reports for a section are labeled *Sec1Set1.rpt*. They should be renamed if permanent records are required, i.e., a subsequent *ClassSet1.rpt* file will overwrite any previous *ClassSet1.rpt*. Grader also writes a file called *gradingstatus* to the *records/setx* directory.

2.5 Manager

- **Functions**: The following are functions of Manager:

1. **Manage Classl**. A convenient way to edit the classl file (add student, delete, sort, find, change section, merge, etc.). Don't forget to "save" after making a change.
2. **Scoring** Included in Manager is a utility called Scorer which the instructor uses to import a score report from a machine scored scantron individualized exam and grade that *CAPA* generated exam where the correct pattern differs for each student. Up to 50 questions can be included in a quiz or exam. The information on the score report file is shown below:

```
Sample format for one student, 3 capa questions
Character numbers begin with zero!
0         1         2         3         4         5         6         7         8         9
01234567890123456789012345678901234567890123456789012345678901234567890123456789012345678901234567890123456789
515000012001090999003   5327 #0001      BERRYMAN  F     A23592320002EDEIAI   1 1 1    1 111     1

              start  end    Example:
 SerialNumber  {  5  8 }    Exam above was the 12th exam to be scored
 LastName      { 40 49 }    BERRYMAN   as last name.
 FirstInitial  { 50 50 }    F as first initial
 StudentNumber { 56 64 }    student number is A23592320
 Section       { 65 67 }    entered section 002 on scantron sheet
 CapaID        { 68 73 }    CapaID+  This is a six letter code which is printed on each
                            student's exam. There is a space to enter the  code on the
                            scantron sheets (available from MSU). Here, the code was EDEIAI
 Question 1    { 76  85 }   (starting at character 76, the student's response is recorded.
 Question 2    { 86  95 }   The CAPA scantron form is has 10 bubbles with choices labeled
 Question 3    { 96 105 }   both numerically and alphatically 1 through 10 and A through J.
  .....
 Question 50   {566 575 }   Each question is 10 columns and there are 50 possible questions.
                            Each column for a question must have either a space or a 1
                            1 means it was marked, space means it wasn't.
```

   Scorer puts the results of the scoring into a setX.sb file which the user may view to acertain that the scores are reasonable. The user can have the code also write the file to setX.db, at which time the students may see their grades.

3. Print one or more assignments for a student without having to open the *setx.qz* file in Quizzer. You can choose the student by either name or student number.
4. Randomize a user made seating file for assigned seating in exams.
5. Use Manager to access CapaUtils 1.1 which performs the following functions:
   (a) View summarized log files.

(b) Run statistics on a set.

(c) Generate a student course profile which summarizes all data from the class, exam, quiz, supplementary, and others paths.

(d) Get CAPA IDs for one student or the entire class.

(e) Analyze the problems in the set and see how each problem discriminates between the upper and lower percents of the class, the correlation between different problems, and what degree of difficulty each problem had for your class.

(f) View the exact submissions a student enters into a telnet and Web session.

(g) Analyze a class report generated in grader and view the distribution of correct verses incorrect.

(h) Analyze and generate the output generated by scorer.

- **Files**: Manager reads the classl file, instructor generated seating files, and files generated from machine scoring. CapaUtils reads report files in the class name directory and files found in the *records* directory.

- An older version of CapaUtils can be accessed through a terminal.

```
capa2.nscl.msu.edu> capautils.pl
USAGE: capatools.pl -c Full_path_to_class
Please enter the CLASS absolute path:
/usr/users/teacher/CAPA5/nsc121s9

       +-----------------------------------------------+
       |        Welcome to CAPA Utilities Ver 1.0      |
       +-----------------------------------------------+

       +------/usr/users/teacher/CAPA5/nsc121s9--------+
       |                                               |
       |  1: Change class path                         |
       |  2: Run capastat                              |
       |  3: Log analysis on Y, N, S, U, and u         |
       |  4: Student course profile                    |
       |  5: CAPA IDs for one student                  |
       |  6: All CAPA IDs                              |
       |  7: Item analysis                             |
       |  8: Item correlation                          |
       |  9: Print assignment(s) for a student         |
       | 10: View submissions for a student            |
       | 11: Quit                                      |
       |                                               |
       |SELECT:                                        |
       +-----------------------------------------------+
```

- CAPA IDs can also be generated through the terminal using the function AllCapaID.

```
capa2.nscl.msu.edu> allcapaid -h
USAGE: allcapaid [-s start-set] [-e end-set] [-stu student-number] [-c class-dir
ectory] [-d output-directory] [-h] [-i] [-sec [n|n:m]]
       start-set : default 1
       end-set   : default 10
       student-number  : no default
       class-directory : no default
       output-directory: class-directory/capaID
       -Sec 3          : for section 3
       -Sec 3:7        : from section 3 to section 7
       -i              : don't create files, print to the screen
       -h              : prints this message
       CAPA version 5.0.3, 11:23-Apr-07-1999
```

3 Running *CAPA* via X-Server software

Your computer system can be used to access the applications and edit your files on a remote *CAPA* host. This section describes examples for using a PC or a Macintosh computer to edit files and run the Quizzer application.

- **Running *CAPA* via X-WIN32**[2]

  NOTE: These instructions are for using X-WIN32 (v4.1.2) in connection with a DEC Alpha host running OSF1 and may differ depending upon your host machine and operating system.

  Procedure

  - To set up X-WIN32 to automatically open upon start-up of your PC:
    1. Install the X-WIN32 software on your machine.
    2. Once the software has completed its installation, a small window appears which contains two shortcut icons.
    3. Open the path *C:/Winnt/Profiles/Username/Start Menu/Programs/Startup* from the "My Computer" icon.
    4. Drag the shortcuts into the Startup folder.
    5. When you reboot your machine, X-WIN32 will execute upon startup.
  - From the toolbar, choose X-Win (blue), choose Sessions, then choose XDMCP-broadcast. This will generate a listing of all machines able to run XDMCP mode.
  - Open the X-Win Utilities (green). An "X-Win32 Utility" window appears. Choose Sessions, then choose New Session.
  - Enter the name you wish to use for your session. Example: `capa3`
  - Check the button in front of "XDMCP".
  - Make sure Query is checked. Enter the hostname. Example: `capa3.nscl.msu.edu`
  - Select Save
  - From the same "X-Win32 Utility" window, click on Options then Window Settings and make sure that "Multiple" is selected. For operating systems in which the background is necessary (Linux Debian for example), check the option "switch to single window" under "window settings".
  - You can now run your X-Server sessions by clicking on X-Win (blue), choosing Sessions, and choosing the session name you have just created. Your screen should then appear exactly as if you were sitting at the monitor of your host machine with your Windows-NT desktop in the background. If you chose the option "switch to single window" in the step above, a large window should appear which looks exactly as if you were sitting at the monitor of your host machine.
  - Open up an xterm and enter `quizzer` at the prompt.

[2] *X-WIN32 ©StarNet Communications*

– The Quizzer main menu will appear in the upper left-hand corner of your screen. You can edit and preview your files through the Quizzer application. Be certain to quit the Quizzer application when you are finished editing.

- **Running *CAPA* via eXceed**[3]

These instructions are for using eXceed Version 4.0 from a PC running Windows-NT[4] in connection with a DEC Alpha host running OSF1 and may differ depending upon your host machine and operating system. **Note:** The following instructions for eXceed have not been updated since the CAPA 4.6 Manual.

Procedure

– To set up eXceed to automatically open upon start-up of your PC:
  1. Install the software following the directions from the producer.
  2. Open the path *C:/Winnt/Profiles/Username/Start Menu/Programs/Startup* from 'My Computer'.
  3. Open the path *C:/Winnt/Profiles/Username/Start Menu/Programs*
  4. Highlight the eXceed.exe by single clicking on it. From the menu bar choose File, then Create Shortcut.
  5. Drag the Shortcut you have just created into the Startup folder you opened previously.
  6. When you reboot your machine, eXceed will execute upon startup.

– To set up your *CAPA* server as an xhost: Two methods:
  1. The simplest way is to leave the host.txt file in eXceed as deactivated. This allows connection with any host as X-server.
  2. If you wish, you can specify each host by creating the host.txt file in eXceed.
     (a) From the Windows-NT main menu bar, choose Start, choose Programs, choose Exceed, then choose X-Config.
     (b) Double click on the Security icon.
     (c) A pop-up menu will appear where you can specify using the host.txt file by clicking on the associated radio button.
     (d) You can choose to edit this file and add your hostname. Remember to save the host.txt file after you have finished editing it.
     (e) This allows your machine to connect with only the hosts listed in the host.txt file.

– To run a telnet session through eXceed:
  1. From the Windows-NT main menu bar, choose Start, choose Programs, choose Exceed, choose htelnet. From the menu bar, choose File, then Create Shortcut. Drag the Shortcut you have just created to the Desktop.
  2. Double click on the shortcut you have just created to open a telnet session into your host machine.
  3. The Connect pop-up screen appears. In the field "host", enter your host name. Example: `capa3.nscl.msu.edu`

[3] *eXceed ©Hummingbird Communications, Ltd.*
[4] *Windows-NT ©Microsoft Corporation*

4. Select OK.
5. You will then see your Telnet window with the `login:` prompt displayed.
6. Login to your host as you normally would for a telnet session.
7. You must then command your host to display to your local machine by entering the command
   `setenv DISPLAY 35.8.33.131:0.0`
   Note: Enter the IP address of the machine you are using. For this example, 35.8.33.131 is used. You can, if you wish, create an alias for this command by editing your .cshrc file in your login directory.
8. You can now run the Quizzer application by entering `quizzer` at the telnet prompt.
9. The Quizzer main menu will appear in the upper left-hand corner of your screen. You can edit and preview your files through the Quizzer application. Be certain to quit the Quizzer application when you are finished editing.

– To map the shortcut keys for Quizzer:

1. From the Windows-NT main menu bar, choose Start, choose Programs, choose Exceed, then choose X-Config.
2. Double click on the Input icon.
3. There will be displayed a pull-down menu to set the choices for the Alt key.
4. You can choose to have either the left or right Alt key function for your shortcut key in Quizzer, Grader, and/or Manager.

- **Running *CAPA* via MacX**[5]

NOTE: These instructions are for using MacX Version 1.5 in connection with a DEC Alpha host running OSF1 and may differ depending upon your host machine and operating system.

Procedure

1. Double click on MacX1.5 icon.
2. Got to Edit and select XDMCP Preferences
3. An "XDMCP Preferences" window pops up. In the "Type of Query" box, select Broadcast. Then select the button that says "User selects From a List of Responding Hosts".
4. Click Execute Now.
5. A list of all the machines you are networked to will appear.
6. Click File then save to desktop to make an icon.
7. In the list, click on the machine where /capa/ is located and then click select.
8. Login to your /capa server. Open up a terminal and type `quizzer` to begin the Quizzer application. You can start the other items by entering the appropriate command at the prompt (i.e. `grader`, `manager`, etc.).
9. The next time you need to access *CAPA*, double click on your broadcast icon and select your machine.

[5] *MacX ©Apple Computer, Inc. All rights reserved.*

4 *CAPA* Functions

All of the examples and descriptions used below are for student submissions of answers through a Web or telnet session. There are some changes and additions to the *setx.qz* code when using *CAPA* for machine scored exams. Please see the next section for those examples.

4.1 Question source text

CAPA assumes that any characters in the *setx.qz* without a special command at the beginning of a line will be displayed as text for the student. The following are those special commands:
`/LET`, `/BEG`, `/DIS`, `/IMP`, `//`, `/HIN`, `/EXP`, `/MAP`, `/RMAP`, `/IF`, `/ELSE`, `/ENDIF`, `/WHILE`, `/ENDWHILE`, `/ANS`, `/AND`, `/OR`, `/SUBJECTIVE`, `/VERB`, `/ENDVERB`, `/START`, and `/END`.
Text (excluding white space) preceding the forward slash will disable the all of the commands except `//` and `/DIS`. Variables must be defined before use.

4.2 Variable Definitions and Expressions (`/LET` and `/BEG`)

- Definitions are specified in the form:
  `/LET VariableName=expression`
- Variable definitions and expressions begin on a single line with a `/LET` command that is terminated by a carriage return. Long entries will be line-wrapped by the editor and may appear to extend past one line on the display. The line continuation symbol (\ followed by a carriage return) can be used to break a long line into several lines and still be considered as a single long line.
- `/BEG` is simply an alias for `/LET`, and is used to indicate the beginning of each problem in the following manner:
  `/BEG prob_val=3`.
  This sets the variable prob_val which can then be used to define the weight of the problem in the `/ANS` specification the signals the end of the problem.
- The *CAPA* system does reserve variable names for certain functions used in the system. They are listed in the **Intrinsic Functions Table**, and must not be used as variable names.
- **List of Available Expressions**:

Integer	◇ An integer.
Real number	◇ A real number. It could be of the form 123.4, 1.234E+2, 1.234E+2, 1.234E+02, 1.234e+02.
"string"	◇ A string. It is specified in the form `"A block of text"`.
Variable Name	◇ A variable previously defined.
(expression)	◇ Precedence, evaluate *expression* first.
- expression	◇ Negative of *expression*.
function([expression, ···])	◇ Call a function with arguments. (see below)
*expression * expression*	◇ Multiply expressions.
expression / expression	◇ Divide expresssions.

expression + *expression*	◇ Add expressions, concatenate strings
expression − *expression*	◇ Subtract expressions.
expression == *expression*	◇ Logical; expressions equal? Returns 0 if false, 1 if true
expression != *expression*	◇ Logical; expressions different? Returns 0 or 1
expression >= *expression*	◇ Logical; greater than or equal? Returns 0 or 1
expression <= *expression*	◇ Logical; less than or equal? Returns 0 or 1
expression > *expression*	◇ Logical; greater than? Returns 0 or 1
expression < *expression*	◇ Logical; smaller than? Returns 0 or 1

- There are three types of variables: integer, real and string. The type is not explicitly specified but rather is assigned by context at the time the variable is defined.
- **Note:** If you define a variable with an equation consisting of only integer numbers, then the resulting answer will be an integer. For example, we have the following code:
  `/LET variable1=1/2`
  `/LET variable2=35/12`
  Then *variable1* will be assigned the value of "0" and *variable2* will be assigned the value of "2". On the other hand,
  `/LET variable1=1.0/2.0`
  will now be assigned the value of 0.5 because at least one real number was used in the equation.
- The variable names must begin with a letter but may contain letters, numbers, and underline characters.
- Variable names (and function names) **are case sensitive**, and there is no limit on the length of a variable name.
- Variables must be defined before they are used in any other expression.
- **Note:** Quotation marks indicate the beginning and end of strings. You must use a backslash to display the quotation marks within a string. For example:
  `/LET string= ''I'll be back.''`
  will produce the ouput "I'll be back."
- An expression may be be broken down into several lines using the line continuation character \ followed immediately by a carriage return. This can help improve the legibility of the code. Do not use the line continuation character \ within a command, variable, or function name.

```
/LET a_long_line = "This is a very, very, very long statement. It is convenient to \
divide it into two lines."
```

- Once defined, the same variable can be used at any subsequent point in the entire problem set or alternatively it could be be redefined.

4.3 Display of Variables (/DIS)

- The *value* of a previously defined variable can be displayed in the displayed text by placing the variable name in the `/DIS()` command, e.g. `/DIS(`*variable*`)`.

- The format of the display of a numerical value can be controlled using a colon and a format specification. For example, `/DIS(LENGTH:3f)` means display the variable *LENGTH* as a floating point number with three places after the decimal (*x.xxx*). `/DIS(LENGTH:2E)` will display the variable in scientific notation with two decimal places (*x.xx*E*x*). The colon is a delimiter indicating that a format specification follows.
- Quizzer shows a default line length of 80 characters. In the x-windows version, a greater line length can be obtained by resizing the window, accomplished by dragging the lower right corner.
- `/DIS(stdline)` will display a short line that is used to seperate problems. When you build sets with Quizzer using import, you have the option of displaying a "standard line" or a `webonlyline` (a line that appears on the web only) after the problem. When printing two sided, you have the option of diplaying a `stdlineOvr` which displays the word "Over" so students realize that the set is two sided. All of these variables are defined in the */demolibrary/Tools/StdMacros* file.
- **Note:** A total of 37 lines are available for telnet display for each question on the VT100 terminal. These lines are separated onto two pages with 20 lines available on the first page and 17 available on the second. Three lines are repeated for reading consistency. Hints and explanations are displayed on separate screens, each of which can have up to 20 lines. There are no line limitations on the printed or Web versions of the text.

4.4 The Import Function (`/IMP`)

- The `/IMP` function calls and uses the contents of the specified file, however the file's content is not displayed in the *setx.qz* file.
- The /IMP command requires a string input, either the string filename or a variable which provides the string filename.

  1. For example:
     `/IMP "HWTop"`
     will import the *HWTop* file from the local directory.
  2. For files in other directories, use either relative or absolute paths as part of the filename, such as:
     `/IMP "/demolibrary/Tools/StdMacros"`
     or:
     `/IMP "../mystuff"`.
     Note that if relative paths are used, the `/IMP` command will not work unless it is the appropriate directory and can cause errors if the file is moved.
  3. The file may be selected dynamically from a set, for example,

```
/LET integer=random(1,10,1)
/LET filename="File"+integer
/IMP filename
```

     which will import a randomly selected *Filex* from *File1, File2, ...File10.*

Note: Adding an integer to a string results in a concatenated newstring. This can be used to provide even greater variety among sets for students, since different problems or graphics on a given topic can be selected. This feature may be especially useful for producing standardized tests.

With a large enough problem base, students can be given a randomly selected set to prepare for the test.

4.5 Comment Lines (//)

- Comments are any character strings after a double forward slash, (//). For example:

```
//A line of comments
//You can also put comments after code.
/LET number=random(1,4,1) // chooses a random number
```

- The comments are only displayed by the Quizzer module and are for the benefit of the persons writing and reviewing the actual problem code.
- Instructors are strongly encouraged to include comment lines in order to indicate authorship and to describe the structure of the problems for future use.

4.6 Hints (/HIN)

- Hints are optional. There can only be one hint per problem. The hint becomes available after a wrong answer is entered by the student. The answer function has reserved syntax for changing the number of entries before displaying the hint. The notation `/ANS(variable:2f, hint=6)` would require that a student enter 6 incorrect answers before being presented with the hint.
- The text for hints contain character strings (no expressions) and can display string variables defined before the hint to match a problems content, and is specified as:

```
/LET index=random(1,4)
/LET pronoun=choose(index,"he","she","she","he")
/LET ppronoun=choose(index,"his","her","her","his")
/LET person=choose(index,"son","daughter","niece","nephew")
/HIN While her /DIS(person) was being picked up, /DIS(pronoun)
/HIN accidently let /DIS(ppronoun) balloon escape.
```

- This hint has two lines and two carriage returns. The contents of both lines are displayed simultaneously; it is not two separate hints.
- Content of all lines typed with a /HIN at the beginning are displayed as the hint. The relative ordering of the hint lines is preserved, and the hint is displayed as a separate page on the VT100 screen and below the answer box on the web version.
- Only one /HIN is needed if the text is very long and uses the continuation character.

```
/HIN A hint is often welcomed by students. It sometimes requires \
a large amount of text  and might require several lines.
```

Or (no carriage return at all, line wraps automatically):

```
/HIN A hint is often welcomed by students. It sometimes requires a large amount of text and
might require several lines.
```

- Hints can also be viewed after a problem set's due date.

4.7 Explanations (/EXP)

- Lines of explanations begin with the **/EXP** characters and are displayed on the login-terminal only when requested after a problem set is closed. The VT100 display offers the option to view a separate page containing the explanation. The web version automatically displays the coded explanation when the student views a closed problem set.
- The syntax for coding explanations is very similar to hints. It is useful to include a detailed explanation for some problems. Students often review old problem sets at exam time and may not exactly remember how to solve a specific problem.
- The following are explanation examples:

```
/EXP An explanation is often welcomed by students.
/EXP It sometimes requires a large amount of text
/EXP and might require several lines.
```

Or:

```
/EXP An explanation is often welcomed by students. It sometimes requires \
a large amount of text  and might require several lines.
```

Or (no carriage return at all, line wrap automatic):

```
/EXP An explanation is often welcomed by students. It sometimes requires a large amount of
text  and might require several lines.
```

4.8 The Mapping Function (/MAP and /RMAP)

- The **/MAP** function is used to map and permute the assigned values from a set of variables onto another set of variables according to a given seed value.
- For example:

```
/LET seed=random(1,3000,1)
/MAP(seed;M1,M2,M3;m,n,o)
```

Assigns to the variables $M1$, $M2$, and $M3$ the values of the variables m, n, and o. The correspondence (i.e. which of m, n, or o is assigned to $M1$, etc....) is determined by the value of the variable 'seed', which in this example is selected from 1 to 300.

- The arguments of the `/MAP` function are divided into three portions by a semicolon symbol (the seed, the variable set, and the defined variable set). The value of the seed is used to setup a random number generator which is needed to select one value from the **second** set of variables and assign it to one of the variables in the `first` set. In the above example, two random numbers will be generated. The first random number is then divided by three and the remainder is used to determine the first value to be selected from the set of three values and gets assigned to the first variable in the list. The second random number is then divided by two and the remainder is used to determine the second value from the remaining two values and is mapped to the second variable. The very last value left is then assigned to the last variable. Therefore, the number of variables in both sets must be equal. If any of the variables that appear in the second variable set were not previously assigned a value, a warning will be issued. Variables used in the first variable set may contain no data at first, but all variables in the first variable set will receive some value after the `/MAP` function is called.
- The `/MAP` function is used in all the auxiliary files of the multiple choice templates prepared to facilitate coding of qualitative or conceptual questions.
- The `/RMAP` function does the reverse action of the `/MAP` function. The value of the seed is used to setup a random number generator which is needed to select one variable from the **first** set of variables and map it with one of the values from the `first` set. For example, we have the following code:

```
/LET seed=random(1,300,1)
/LET w=1
/LET x=2
/LET y=3
/LET z=4
/MAP(seed;a,b,c,d;w,x,y,z)
/DIS(a), /DIS(b), /DIS(c), /DIS(d)

output:  3, 1, 2, 4
```

The values of the the second set of variables were randomly assigned (dependent on the seed value) to the first set variables. See how the **value** of w (the first of the four values in its value set) is mapped to the **variable** b (the second of the variables in its set). If we used `/RMAP` instead with the same seed value used above in the code below, the resulting mapping is reversed. Notice how the **variable** a (the first of the four variables in its set) has the **value** of x (the second of the four values) mapped to it.

```
/LET w=1
/LET x=2
/LET y=3
/LET z=4
/RMAP(seed;a,b,c,d;w,x,y,z)
/DIS(a), /DIS(b), /DIS(c), /DIS(d)

output:  2, 3, 1, 4
```

Therefore, `/RMAP` can also be used to unscramble `/MAP`:

```
/LET seed=random(1,300,1)
/LET w=1
/LET x=2
/LET y=3
/LET z=4
/MAP(seed;a,b,c,d;w,x,y,z)  //same seed as above - not redefined
/RMAP(seed;p,q,r,s;a,b,c,d)
/DIS(p), /DIS(q), /DIS(r), /DIS(s)

output:  1, 2, 3, 4
```

- To view an example of both the `/MAP` and `/RMAP` see the second problem in *set19.qz* from the *nsc121s9* class directory(*/demolibrary/type-other/OR-MAP-RMAP*).

4.9 Conditional Statements (/IF, /ELSE, and /ENDIF)

- `/IF`, `/ELSE`, and `/ENDIF` are used to add conditions within the *CAPA* coding.
- For example:

```
//NOTE: Each /IF needs an /ENDIF
/LET choice=random(1,4,1)
/IF (choice==1)
If the statement within the parentheses is true, this text
will be printed for the student to read.
/ELSE
 /IF (choice==2)
This line will be printed instead.
 /ELSE
  /IF (choice==3)
This line will be printed.
  /ELSE
If all the conditions above are false, then this text
will be printed.
  /ENDIF  //ends the last /IF condition
 /ENDIF  //ends the second /IF condition
/ENDIF  //ends the first /IF condition
```

- Other commands can be entered in the `/IF` statement. For example, you may have the following:

```
/IF (choice!=1)
/ANS(5)
/ELSE
/ANS(0)
/ENDIF
```

Therefore, if the variable choice happens to **not** equal the interger 1 then the the answer is 5, otherwise the answer is 0.

4.10 Loops (/WHILE and /ENDWHILE)

- Below is an example of the /WHILE and /ENDWHILE commands:

```
Written below are the multiples of 5 up to 12:
/LET integer=0
/WHILE (integer <= 12)
/LET answer = 5*integer
/DIS(integer) * 5 = /DIS(answer) /DIS(web("","//",""))
//see the functions table to see how web() works
/LET integer=integer+1
/ENDWHILE

The output will display:

Written below are the multiples of 5 up to 12:
0 * 5 = 0
1 * 5 = 5
2 * 5 = 10
3 * 5 = 15
4 * 5 = 20
5 * 5 = 25
6 * 5 = 30
7 * 5 = 35
8 * 5 = 40
9 * 5 = 45
10 * 5 = 50
11 * 5 = 55
12 * 5 = 60
```

  As long as the condition in the parentheses is true, the code between the /WHILE and /ENDWHILE will be processed. When the condition (in this case, the value of the variable *integer* is less than or equal to 12) is false, control passes to the next line below the /ENDWHILE.

- Every /WHILE was must have a /ENDWHILE to mark the end of the /WHILE loop.

4.11 Answers (/ANS)

- Each individual problem must be ended by an expression specifying the answer. This answer expression consists of the keyword /ANS() starting on a new line.

- The answers are indicated specifically by the /ANS() command, with the parentheses containing the answer and the attributes of the answer. The answers to problems can be previously defined variables. There must be at least a variable, integer, real number, of string defining the answer within the parentheses, (/ANS(variable), /ANS(5), etc.).

- Several options are available to format the answer. The attributes of the /ANS() function are **case insensitive**. The available options for defining /ANS() attributes are:

- **List of Available Expressions:**

wgt=	◇ problem weight
tries= *or* try=	◇ number of allowed tries to input the answer
tol=	◇ answer tolerance

`calc=fmt/unfmt`	◇ sets tolerance to be calculated from the formatted or unformatted answer.
`sig=`	◇ fixed number of significant digits.
`sig=`x `minus` y `plus` z	◇ a range of significant digits from x minus y to x plus z
`unit=` *or* `units=`	◇ units of answer (dimensions)
`hint=`	◇ number of wrong student entered answers before the hint appears.
`str=mc/cs/ci/fml`	◇ different ways of accepting a string answer with fml for math formula as answer.
`hgr=on`	◇ hand graded answer
`pc=on`	◇ partial credit
`ansbox=on/off`	◇ Web generated answer box
`br=on/off`	◇ Web line break

- **More detailed explanations are found below:**

1. **Problem Weight:** The problem point value (or weight) is set by:
   `/ANS(variable, wgt=2)`
   The problem value can be an integer value between 0 and 9. A variable can be can be defined at the beginning of a problem by either of the following commands:
   `/BEG prob_val=3`
   or
   `/LET prob_val=3`
   Then, this value can be used in the `/ANS()` function:
   `/ANS(variable, wgt=prob_val, tries=20)`.
   If the problem weight is not defined, the default value of 1 point is used.
2. **Tries:** The number of attempts to enter an incorrect answer can be limited to an integer value less than or equal to 99.
   - The format for setting the number of tries is:
     `/ANS(variable, tries=35)`
     or
     `/ANS(variable, tries=try_val)`
   - In the second example shown above, *try_val* is assumed to be a previously defined variable. The student receives a message near the answer field stating the number of attempts and the total number of tries available for each problem in which the `tries=`x has been specified. A warning message is displayed when only one try remains. Errors in either significant figures or units will **not** decrease the number of remaining tries for the student. Both the significant figure and unit must be correct before a "Correct" or "Incorrect" response to the numerical portion of the answer is given to the student.
   - **Note:** `try=` is equivalent to `tries=`
3. **Tolerance:** The tolerance can be specified as (1) a numerical value, (2) as a percentage of the correct answer or as (3) a pre-defined variable.
   (a) The format for a set numerical value would be given by:
   `/ANS(variable:3f,tol=0.5)`

(b) The format of a % tolerance would be given by:
`/ANS(variable:3f, tol=1.1%)`.
This is specified if there is a possibility that the answer will be calculated as equal to zero. The tolerance is thus given a value rather than a percentage.

(c) The format for a defined tolerance variable would be:
`/ANS(variable:3f, tol=TolVar)`.

(d) The format for a % tolerance with a defined variable would be:
`/ANS(variable:3f, tol=TolVar%`

Note that the display of the answer can be formatted according to the same commands used in /DIS.

- **Real answers:** A tolerance should be specified for 'real' answers. It can be absolute or relative. An absolute tolerance can be a variable. Answers with real values can be **formatted** with specifications `:1E, :3E, :2f,` etc., where `1E` means exponential notation with one decimal place, `:3E` means exponential notation with three decimal places, and `:2f` means two floating decimals.
  (a) `/ANS(variable:2E,tol=1.2%,wgt=prob_val,tries=try_val)`
  When an answer submitted by a student is within the relative tolerance (1.2%) of the correct answer, the student receives 'correct'. **Note:** The relative tolerances are based on the **unformatted** answer.
  (b) `/ANS(variable:2E,tol=5.0,wgt=prob_val,tries=try_val)`
  An answer submitted by a student within the absolute tolerance of 5.0, would be graded 'correct'. That is the answer must lie within the range *variable*-5 to *variable*+5.
  (c) `/ANS(variable:3E,tol=tol_val,wgt=prob_val, tries=try_val)`
  Here the tolerance is a previously defined quantity, `tol_val`.
  (d) `calc=fmt/unfmt`: Selects the application of the tolerance range for the formatted or unformatted numerical answer value. For example, say we have `variable=2/3` for the following answer line:
  `/ANS(variable:3f,tol=.01,wgt=prob_val, tries=try_val)`
  the error would be between two-thirds plus or minus .01. If we used this instead though:
  `/ANS(variable:3f,tol=.01,calc=fmt,wgt=prob_val,tries=try_val)`
  the error would be between 0.667 plus or minus 0.01. The default value is unformatted.
- **Integer answers:** You may not want a tolerance for an integer answer.
  `/LET variable=34`
  (a) `/ANS(variable, tol=0, wgt=prob_val,tries=try_val, hint=hint_val)`
  Only the number "34" will be accepted as correct.
  (b) `/ANS(variable, wgt=prob_val, tries=try_val, hint=hint_val)`
  Same as above. Only the number "34" will be accepted as correct.
  (c) You can also use a percentage tolerance or a non-integer real number as a tolerance, but then students can get a "correct" answer by entering in an appropriate non-integer number. For example:
  `/ANS(variable, tol=4, wgt=prob_val, tries=try_val, hint=hint_val)`

Any answer within plus or minus 4 units is accepted as correct. For example, the real number, 36.5 will be accepted as "correct".

- **Note:** If you define your answer variable with an equation consisting of only integer numbers, then the resulting answer will be an integer. For example, we have the following code:
  `/LET variable1=1/2`
  `/LET variable2=35/12`
  Then *variable1* will be assigned the value of "0" and *variable2* will be assigned the value of "2". On the other hand,
  `/LET variable1=1.0/2.0`
  will now be assigned the value of 0.5 because at least one real number was used in the equation. [6]

4. **Significant Figures:** If unspecified, significant figures are not checked and any number of significant figures between 1 and 15 are accepted provided the answer entered falls within the specified numerical tolerance range. The number of acceptable significant figures can be specified by the following format:
   (a) `/ANS(variable, sig=4 plus 1 minus 1, tries=try_val, wgt=prob_val, hint=hint_val)`
   This will accept 3, 4, or 5 significant figures in the answer.
   (b) `/ANS(variable, sig=3 plus 2, tries=try_val, wgt=prob_val, hint=hint_val)`
   This will accept 3, 4, or 5 significant figures in the answer.
   (c) `/ANS(variable, sig=4 , tries=try_val, wgt=prob_val, hint=hint_val)`
   This will only accept 4 significant figures in the answer.

   Care must be exercised when limiting the significant figures when a percentage tolerance is used. It is possible to create a situation where the student cannot enter a correct answer for a problem. For example, the server might calculate the unformatted *variable* to be equal to 0.409. The following answer format will create such a situation:
   `/ANS(variable:0f, sig=1, tol=1%, wgt=prob_val)`
   The student will calculate this answer to be 4E-1 which is not within 1% of the answer.

5. **Units:** A file in the class directory named *capa.config* lists all SI units and other user defined acceptable units with their relationships. That file can be edited by the instructor. Units are entered as strings.
   - A typical format for answers requiring a unit is:
     `/ANS(perimeter:3f,unit="cm",sig=4)`
     or
     `/ANS(area:2f,unit="cm^2")`
   - For a specially defined unit, one can use a variable, e.g.

```
 /LET StringVar="dollar"
/ANS(cost:2f,unit=StringVar,tol=.005,wgt=2,tries=6,sig=4 plus 3)
```

   - */demolibrary/type03/msu-prob22.txt* and */demolibrary/type03/msu-prob23.txt* are two problems included in the distribution package which aid in teaching students the

[6] You may have noticed this text from the /LET section, but it is worth repeating because it can be a major source of frustration to the instructor and students if real numbers and integer numbers are not dealt with correctly.

rules for entering answers with units in a *CAPA* session. These are useful for students in an initial *CAPA* set.

- **Note:** `units=` is equivalent to `unit=`

6. **Show Hints:** The hint can be configured so that it is available or displayed only after a certain number of tries. This is achieved by setting `hint=AnInteger` (e.g. `hint=3`) in the answer format. For example:
   `/ANS(variable, hint=3, tries=10, wgt=1)`
   or
   `/ANS(variable, hint=hint_val, tries=10, wgt=1)`
   will both (1) show the hint on a Web browser after 3 tries and (2) allow it to be viewed on telnet after 3 or more tries by entering :H which shows up on the menu lines.
7. **String answers** can be exact comparisons, or have the order and/or case disregarded.
   (a) `/ANS(letters,str=ci,wgt=prob_val,tries=20)`
   The command `str=ci` is the default value, the order is checked but case insensitive.
   (b) `/ANS("KCl",wgt=prob_val,tries=20)`
   The default is case insensitive. Therefore, answers such as "kcl" and "KcL" are accepted as correct.
   (c) `/ANS(letters,wgt=prob_val,tries=20)`
   letters is a string variable previously defined. This works like the example above. (The order is essential, but not case sensitive.)
   (d) `/ANS(letters,str=mc,wgt=prob_val,tries=20)`
   The command `str=mc` accepts any order of the correct responses. Hence, "abc" and "acb" are equivalent.
   (e) `/ANS(letters,str=cs,wgt=prob_val,tries=20)`
   This is an exact string comparison. Both the order and the case are required.
   (f) **Formula as string answer:**

```
/ANS(fm1,str=fml,eval=<"x" @ pt1:pt2#20>,TOL=1E-3, tries=try_val,
wgt=prob_val,hint=hint_val)
```

   In this one-variable case, *CAPA* will sample the student's input formula and the formula fm1, the string formula representing the correct answer. Comparison will be at 20 values of x between x = pt1 and x = pt2). All comparisons must fall with the tolerance, here 1E-3, for a correct answer.
   For examples, see the templates:
   `/demolibrary/MCTools/equationA` *and* `/demolibrary/MCTools/equationXY`
8. **Hand Grading:** A problem can be designated to be hand graded. For this case students are not allowed to answer via telnet or web sessions and the question is graded by the instructor using the Grader application. The format for this is:
   `/ANS(variable,hgr=on, tries=1, wgt=1)`
   (See /SUBJECTIVE for telnet or Web submitted essay questions for hand grading.)
9. **Partial Credit:** A multiple point problem can be specified to have partial credit. This attribute works like `hgr=on` in that students cannot login to answer the question. The format for this is:
   `/ANS(variable, wgt=9, pcr=on, tries=1)`

This allows an instructor to manually enter an integer point value from 0 to 9 through the Grader application.

10. **Miscellaneous Answer Attributes**
    (a) `br=on/off`: This starts/stops the generation of
 (a line break) on the Web display. The default value is on.
    (b) `ansbox=on/off`: Controls the generation and display of the answer box for the Web. The default value is on.
    (c) `Yes/no` is equivalent to `on/off`.

4.12 Multiple Answers (/AND and /OR)

- `/AND` or `/OR` after an answer specification requires another `/ANS` statement be present. In the case of `/AND`, two successive answers will be required from the student. In the case of `/OR`, the student will receive credit if either answer is supplied.
- Below is an example for `/AND`:

```
/ANS(variable_1, wgt=prob_val, tries=try_value)
/AND
/ANS(variable_2, wgt=prob_val, tries=try_value)
```

The above example requires two answers from the student. During a telnet session the student will be prompted to enter the first answer and then the second **in the order that they are coded**. During a web session, there will be multiple answer boxes labeled "Answer 1" and so on. Again, they must be entered in order by the student. It is recommended that the problem text give instructions to the student to answer the questions in order. Note that the problem is worth the number of points of the last defined `wgt` specification, i.e. if in the first answer `wgt=3` and in the second answer `wgt=1`, the student will only receive one point for the total question.

- Below is an example for `/OR`:

```
/ANS(variable_1, wgt=prob_val, tries=try_value)
/OR
/ANS(variable_2, wgt=prob_val, tries=try_value)
```

The `/OR` is suitable for questions such as asking the student to choose one of the correct answers from a list.

- Note that the variable *try_val* was used in the examples above to simplify the script. If the number of tries is different within the answer specifications, the last defined number of tries is given to the student for their amount of tries in that problem.

4.13 Essay Questions (/SUBJECTIVE)

`/SUBJECTIVE` is used instead of `/ANS` when the instructor wants to allow students to submit a short essay. The student can type their essay in either the telnet or web session. The instructor grades this essay with the Grader application and the score is recorded in the database. `hgr=on` must be specified when using `/SUBJECTIVE`.

```
/SUBJECTIVE(wgt=prob_val, hgr=on, tries=try_val)
```

You may choose to set a number of tries also. The student will be allowed to rewrite their essay according to the number of tries that the instructor allows.

4.14 Displaying text verbatim (/VERB and /ENDVERB)

Used to display text verbatim:

```
Enter the answer as in a calculator:
/VERB
Example: 4.567 + 2.431 * x^4 - 0.310 * x^2
/ENDVERB
```

4.15 Displaying Information Before the Problem Set (/START)

All the material coded before /START in a *setx.qz* file is both printed out and displayed on the Web for students to see. See *set19.qz* in the *nsc121s9* class directory for an example (note the placement of the imported *HWTop* file).

4.16 Ending the Problem Set (/END)

- /END() or /END(stdendline) causes the parser to stop and ignore all input characters beyond that command. String variables can be defined and constructed and additional text (such as instructions for students) may be displayed between the last /ANS() and /END.
- All sets should include /END(stdendline), so that both on paper and on the Web, the department is identified and the copyright nature of the *CAPA* software is indicated. The string for stdendline is created in the *HWTop* file in the class directory and can be edited by users, such as changing the Dept. ID for your particular class.
- The *CAPA* system counts the number of problems in a problem set by counting the number of **proper** answer lines (lines containing /ANS or /SUBJECTIVE). If there is an error in coding and the answer cannot be evaluated, that problem is not counted by Quizzer.

Table of Intrinsic Functions: Names Are Case Sensitive.	
Functions	Description, Sample quizzer input and output for printing

`sin(x), cos(x), tan(x)`	Trigonometric functions. x is in radians.
	
	`/LET angle=60.0`
	`/LET var1=sin(angle*3.141592654/180)`
	`Sine of /DIS(angle:1f) degrees is /DIS(var1:3f).`
	
	Output: `Sine of 60.0 degrees is 0.866.`

`asin(x), acos(x), atan(x), atan2(y,x)`	Inverse trigonometric functions. Returns radians. asin() computes the principal value of the arc sine of x, in the interval [-pi/2,pi/2] radians. The value of x must be in the domain [-1,1]. acos() computes the principal value of the arc cosine of x, in the interval [0,pi] radians. The value of x must be in the domain [-1,1]. atan2() computes the principal value of the arc tangent of y/x, in the interval [-pi,pi] radians. The sign of atan2() is determined by the sign of y.
	
	`/IMP "demolibrary/Tools/StdConst"`
	`//pi_c is a variable set to equal pi and is found in the imported file above.`
	`/LET var1=-1.65`
	`/LET var2r=atan(var1)`
	`/LET var3d=atan(var1)*180.0/PI`
	`The angle whose tangent is /DIS(var1:3f) can be expressed`
	`as /DIS(var2r:3E) radians or as /DIS(var3d:3E) degrees.`
	
	Output for telnet session: `The angle whose tangent is -1.650 can be expressed as -1.026E+00 radians or as -5.878E+01 degrees.`

`log(x), log10(x)`	Natural logarithm and base-10 logarithm. Note that the the variables `natlog` and `tenlog` were defined to simplify the coding of this problem by allowing complicated text (with subscripts and superscripts) to be printed by displaying a variable.
	
	`/LET natlog=tex("$log_e$","log_e")`
	`/LET tenlog=tex("$log_{10}$","log_10")`
	`/LET x=2546.7`
	`/LET var1=log(x)`
	`/LET var2=log10(x)`
	`/LET ratio=var1/var2`
	`/LET var3=log(10.0)`
	`The /DIS(natlog) of /DIS(x:1f) is /DIS(var1:3E) while its /DIS(tenlog)`
	`is /DIS(var2:3E). Note that the ratio /DIS(natlog)//DIS(tenlog)`
	`= /DIS(ratio:3f), which is just /DIS(natlog)(10), i.e. /DIS(var3:3E).`
	
	Output for print out or Web: The log_e of 2546.7 is 7.843 while its log_{10} is 3.406. Note that the ratio $log_e/log_{10} = 2.303$, which is just $log_e(10)$, i.e. 2.303.

`exp(x), pow(x,y), sqrt(x)`	Exponential, power, and square root. Compute e^x, x^y, and $\sqrt{x}$ respectively.
	
	`/LET varx=1.526`
	`/LET vary=0.5`
	`/LET var1=exp(varx)`
	`/LET var2=pow(varx,0.5)`
	`/LET var3=pow(varx,vary)`
	`/LET var4=sqrt(varx)`
	`var1=/DIS(var1:2f); var2=/DIS(var2:2f); var3=/DIS(var3:2f);`
	`var4=/DIS(var4:2f).`
	
	Output: `var1 = 4.60; var2 = 1.24; var3 = 1.24; var4 = 1.24.`

Functions	Description, sample quizzer code and output

`abs(x), sgn(x)`	`abs(x)` returns the absolute value of x. `sgn(x)` returns 1, 0 or -1 depending on value of x
	
	`/LET xx=-2.5`
	`/LET var1=abs(-4.5)`
	`/LET var2=sgn(xx)`
	`/LET var3=sgn(-4.5*xx)`
	`var1=/DIS(var1:2f) var2=/DIS(var2) var2=/DIS(var2:2f) var3=/DIS(var3)`
	
	Output: `var1=4.50 var2=-1 var2=-1.00 var3=1`

`erf(x), erfc(x)`	Error functions. $erf(x) = \frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2} dt$ and $erfc(x) = 1.0 - erf(x)$.
	
	`/LET varx=0.51`
	`Evaluate the Normal Probability Integral from -/DIS(varx)`
	`to /DIS(varx). /DIS(newline)`
	`/LET prob = erf(varx/sqrt(2.0))`
	`Probability = /DIS(prob:4f)`
	
	Output: `Evaluate the Normal Probability Integral from -0.51 to 0.51.`
	`Probability = 0.3899`

`ceil(x),`
`floor(x)`
	Ceiling and floor functions. $ceil(x) = \lceil x \rceil$ and $floor(x) = \lfloor x \rfloor$. The ceiling function returns an integer rounding x toward positive infinity. The floor function rounds toward negative infinity.
	
	`/LET varx=ceil(3.65)`
	`/LET vary=floor(3.65)`
	`/LET varz=ceil(-5.73)`
	`/LET varw=floor(-5.73)`
	`varx=/DIS(varx) vary=/DIS(vary) varz=/DIS(varz) varw=/DIS(varw)`
	
	Output: `varx=4 vary=3 varz=-5 varw=-6`

`min(...),`
`max(...)`
	Minimum and maximum functions, with indefinite number of arguments. Arguments must be of either all integer or real type. String comparison are such that $A < a$. In other words, lower case letters have a higher value than upper case letters.
	
	`/LET a=min(23,45,12,7,9)`
	`/LET b=max(23.1,45.3,12.6,7.1,9.0)`
	`/LET c=min("a","ae","aeg","Aeg")`
	`/LET d=max("a","ae","aeg","Aeg")`
	`a=/DIS(a:2f) b=/DIS(b:2f) c=/DIS(c) d=/DIS(d)`
	
	Output: `a=7 b=45.30 c=Aeg d=aeg`

Functions	Description, sample quizzer code and output

`factorial(n)`	Arguments must be an integer. Returns an integer if n is or smaller, else or real value.
	
	`/LET number=factorial(6)`
	`/DIS("6!=")/DIS(number)/DIS(newline)`
	`/DIS("0!=")/DIS(factorial(0))/DIS(newline)`
	`/LET number2=factorial(21)`
	`/DIS("21!=")/DIS(number2:3E)/DIS(newline)`
	
	Output: `6!=720`
	`0!=1`
	`21!=`5.109×10^{19}

`N% M`	N and M are integers, and the remainder of the integer ratio is returned.
	
	`/LET ratio = 98/5`
	`/DIS("ratio=")/DIS(ratio)/DIS(newline)`
	`/LET leftover=98%5`
	`/DIS("remainder=")/DIS(leftover)`
	
	Output: `ratio=19`
	`remainder=3`

`sinh(x),`
`cosh(x),`
`tanh(x)`
	Hyperbolic functions.

`asinh(x),`
`acosh(x),`
`atanh(x)`
	Inverse hyperbolic functions.

`roundto(var1,n)`	Rounds the value of real variable 'var1' to n places (fixed point notation).
	
	`/LET value=100.0/3.0`
	`Initial value = /DIS(value:5f)/DIS(newline)`
	`/LET value=roundto(value,2)`
	`The new value is now /DIS(value:5f) and can be used as`
	`/DIS(value:2f) with no rounding error.`
	
	Output: `Initial value = 33.33333 The new value is now 33.33000 and can be used as 33.33 with no rounding error.`

`web("a","b","c")`
`or web(a,b,c)`
	ASCII (a), tex(b) and html(c) strings or variables displayed respectively.
	
	`Example: \verb+{/DIS(web("M2","M$_2$","M<sub>2</sub>")) }+`
	
	`Output for telnet session: M2`
	`Output for printout or Web: M$_2$`

`html(a) or` `html("a")`	variable or string 'a' parsed for Web browser only. `Example:` `/DIS(html(" <a href=/nsc121s9/Links/webPulley.html> Motion of Masses \` `on a Pulley</a><p>"))` Displays link "Motion of Masses on a Pulley"

Functions	Description, sample quizzer code and output

`jn(0,x)`, `jn(1,x)`, `jn(n,x)`	Bessel functions of the first kind, with orders 0, 1 and n respectively. Note that the first argument is integer, the second real.
	
	`/LET aa=jn(0,3.0)`
	`/LET bb=jn(1,3.0)`
	`/LET cc=jn(2,3.0)`
	`aa=/DIS(aa:4f) bb=/DIS(bb:4f) cc=/DIS(cc:4f)`
	
	Output: `aa=-0.2601 bb=0.3391 cc=0.4861`

`yn(0,x)`, `yn(1,x)`, `yn(n,x)`	Bessel functions of the second kind, with orders 0, 1 and n respectively. Note that the first argument is integer, the second real.
	
	`/LET dd=yn(1,3.0)`
	`dd=/DIS (dd:4f)`
	
	Output: `dd=0.3247`

`random(l,u,d)`	Returns a uniformly distributed random number between l and u with steps of d. Note that all arguments must be the same type, integer or real. [When real number are used, avoid having the step size such that the highest value lands on "u" as different machines may have different representations and accuracy.]
	
	`/LET index=random(2,5,2) //returns integers 2 or 4`
	`/LET value=random(2.3,5.15 ,0.2 ) //returns 2.3, 2.5, 2.7,...., 5.1`
	`index=/DIS(index) value=/DIS(value:2f)`
	
	Output: `index=4 value=3.70`

`choose(i,...)`	Choose the ith item in the argument list. Integer i must be greater than zero and it's maximum possible value must not exceed the number of arguments following it.
	
	`/LET indx=3`
	`/LET realvar=choose(indx, 23.0,45.4,67.3) // will select 67.3`
	`/LET intvar=choose(indx, 23,45,67) // will select 67`
	`/LET stringvar=choose(indx,"No","Yes","Maybe") // will select "Maybe"`
	`realvar=/DIS(realvar:2f) intvar=/DIS(intvar) stringvar=/DIS(stringvar)`
	
	Output: `realvar=67.30 intvar=67 stringvar=Maybe`

`tex(a,b)`, `tex("a","b")`	When in TeX mode, return the first argument a, and return the second argument b in enscript mode (ASCII).
	
	`/DIS(tex("This, in the .tex file.","This, in the ASCII version."))`
	`/LET A=33`
	`/LET B=66`
	`/DIS(tex(A,B))`
	
	Tex Output: This, in the .tex file. 33

`var_in_tex(a)`	Equivalent to `tex("a","")`

Functions	Description, sample quizzer code and output

`to_string(x)`, `to_string(x,y)`	If variable x is an integer, to_string(x) returns a string. If x is real the format is given by y as follows:
	
	`/LET name1=to_string(34.56789)`
	`/LET name2=to_string(34.56789, ".3E")`
	`/LET name3=to_string(34.56789, ".3f")`
	
	`No format is /DIS(name1), with ".3E" format is /DIS(name2),`
	`and with ".3f" format is /DIS(name3)`
	
	Output: No format is 34.56789, with ".3E" format is 3.457E+01, and with ".3f" format is 34.568

`capa_id()`, `class()`, `section()`, `set()`, `problem()`	The CAPA ID number, class name, section number, set number and problem number respectively. Variables can be assigned to their current values or they can be displayed directly as shown below.
	
	`/LET capaIDval=capa_id()`
	`Your CAPA ID is /DIS(capaIDval). /DIS(newline)`
	`The class name entered at login is /DIS(class()). /DIS(newline)`
	`Your section number is /DIS(section()). /DIS(newline)`
	`This is part of problem set /DIS(set()). /DIS(newline)`
	`The current problem number is /DIS(problem()). /DIS(newline)`
	
	`Output:`
	`Your CAPA ID is 8659.`
	`The class name entered at login is nsc121s9.`
	`Your section number is 1.`
	`This is part of problem set 2.`
	`The current problem number is 2.`

`name()`, `student_number()`	student name and student number. The latter can be printed on quizzes when the student_number is used for identification to resolve ambiguities. `/DIS(name()) /DIS(student_number())` Output: Student, Jamie . A12345678

`open_date()`, `due_date()`, `answer_date()`	Problem set open date, due date, and answer date.
	
	`/DIS(open_date()) /DIS(newline)`
	`/DIS(due_date()) /DIS(newline)`
	`/DIS(answer_date()) /DIS(newline)`
	`/DIS(due_day()) /DIS(newline)`
	
	`Output:`
	`Mon, May 3, 1999 at 08:00. \newline`
	`Thr, May 6, 1999 at 08:00. \newline`
	`Fri, May 7, 1999 at 08:00. \newline`
	`Thr, May 6, 1999.`

Functions	Description, sample quizzer code and output

`get_seed()`, `set_seed()`	get_seed() returns the value of the seed for the random number generator. set_seed() allows resetting the value of the random number generated seed to an earlier value of the seed.
	
	`/LET seed1=get_seed()`
	`/LET number=random(1,1000,1)`
	`The current seed is: /DIS(seed1)`
	`The random number was: /DIS(number)`
	
	`/LET seed2=get_seed()`
	`/LET number=random(1,1000,1)`
	`The current seed is: /DIS(seed2)`
	`The random number was: /DIS(number)`
	
	`/LET any_variable=set_seed(seed1)`
	`/LET seed3=get_seed()`
	`/LET number=random(1,1000,1)`
	`The current seed is: /DIS(seed3)`
	`The random number was: /DIS(number)`
	
	`Output:`
	`The current seed is: 1516840802 1582785437`
	`The random number was: 727`
	`The current seed is: 539910159 1530382995`
	`The random number was: 671`
	`The current seed is: 1516840802 1582785437`
	`The random number was: 727`
	
	Note: get_seed() works like name(), i.e. /DIS(get_seed()) displays the current seed. On the other hand, set_seed() behaves like most other functions, i.e. seed=set_seed(previous_seed_variable) which is similar to y=sin(x).

`sub_string(a,b,c)`	Allows instructor to retrieve part of a previously defined string (a=initial string, b=place of starting character, c=length of substring).
	
	`/LET string1="Have a nice day."`
	`/LET string2=sub_string(string1,8,4)`
	`/DIS(string2)`
	
	`Output: nice`

Functions	Description, sample quizzer code and output
array[xx]	"xx" can be a variable or a calculation. For example, for the array with name "NaMe": /LET index=1 /WHILE (index<10) /LET NaMe[index]=index*10.5 /LET index=index+1 /ENDWHILE The value of NaMe[7] is /DIS(NaMe[7]:1f) Output: The value of NaMe[7] is 73.5
array_max(Name), array_min(Name)	For example, for the array NaMe[xx] defined above: maximum value of array element = /DIS(array_max(NaMe)) minimum value of array element = /DIS(array_min(NaMe)) Output: maximum value of array element = 94.5 minimum value of array element = 10.5
array_moments(B,A)	The moments of array A[] are put into array B[i] with i=0 to 4. B[0]=number of elements, B[1]=mean, B[2]=variance, B[3]= skewness, and B[4]=kurtosis. Note (standard deviation =square root of variance. Example (using NaMe[] array above] /LET elements=array_moments(MMT,NaMe) Displaying the moments: number of elements=/DIS(MMT[0]:2f) mean=/DIS(MMT[1]:2f) variance=/DIS(MMT[2]:2f) skewness=/DIS(MMT[3]:2f) kurtosis=/DIS(MMT[4]:2f) Output: Displaying the moments: number of elements=9 mean=52.50 variance=826.87 skewness=0.00 kurtosis=-1.60
init_array(Name)	Important to initialize when the same array name is used in more than 1 problem in a set. For example, for the array NaMe[xx] above: Current value of NaMe[5] = /DIS(NaMe[5]:1f) /LET deletd=init_array(NaMe) New value of NaMe[5] = /DIS(NaMe[5]:1f) Number of array elements deleted = /DIS(deletd) Output: Current value of NaMe[5] = 52.5 New value of NaMe[5] = VAR "NaMe[5]" NOT DEFINED! Number of array elements deleted = 9

Functions	Description, sample quizzer code and output

`random_normal(,,,,)`
`random_beta(,,,,)`
`random_gamma(,,,,)`
`random_exponential(,,,)`
`random_poisson(,,,)`
`random_chi(,,,)`
`random_noncentral_chi(,,,,)`
	`random_normal(return_array,item_cnt,seed,av,std_dev)`
	`random_beta(return_array,item_cnt,seed,aa,bb)`
	`random_gamma(return_array,item_cnt,seed,a,r)`
	`random_exponential(return_array,item_cnt,seed,av)`
	`random_poisson(return_array,item_cnt,seed,mu)`
	`random_chi(return_array,item_cnt,seed,df)`
	`random_noncentral_chi(return_array,item_cnt,seed,df,xnonc)`
	
	`Generate item_cnt of random numbers according to the particular distribution with the specified parameters and seed and store them in the array named return_array.`

Sample Code	Sample Code for `random_poisson(,,,)` and `random_normal(,,,,)` is shown on the next page.

`random_poisson(,,,)`

FORM: random_poisson(return_array,item_cnt,seed,av)

Array name and 3 parameters needed. Below, the return_array is called "RET":

```
/LET dleted=init_array(RET) // initialize array "RET"
/LET myseed=777  // could be = random(1,10000)
/LET count=10
/LET average=2.2
/LET samples=random_poisson(RET,count,myseed,average)
samples=/DIS(samples); Note 'samples' it is the same as 'count'
/LET j=0
/WHILE (j <10)
   RET[/DIS(j)] = /DIS(RET[j])
  /LET j=j+1
/ENDWHILE
OUTPUT: samples=15; Note 'samples' it is the same as 'count'
   RET[0] = 3      RET[1] = 0
   RET[2] = 3      RET[3] = 1
   RET[4] = 2      RET[5] = 4
   RET[6] = 2      RET[7] = 2
   RET[8] = 1      RET[9] = 2
```

`random_normal(,,,,)`

FORM: random_normal(return_array,item_cnt,seed,av,std_dev)

Array name and 4 parameters needed. (return_array called "RET")

```
/LET dleted=init_array(RET) // initialize array "RET"
/LET myseed=555  // could be = random(1,10000)
/LET count=10
/LET average=6.2
/LET stddev=2
/LET samples=random_normal(RET,count,myseed,average,stddev)
samples=/DIS(samples); Note 'samples' it is the same as 'count'
/LET j=0
/WHILE (j <10)
  /LET RETB[j] = roundto(RET[j],2) // This array is rounded to 2 places
   RET[/DIS(j)] = /DIS(RET[j]:4f) and RETB[/DIS(j)] = /DIS(RETB[j]:2f)
  /LET j=j+1
/ENDWHILE
OUTPUT: samples=15; Note 'samples' it is the same as 'count'
   RET[0] = 2.9195 and RETB[0] = 2.92
   RET[1] = 6.4934 and RETB[1] = 6.49
   RET[2] = 7.2427 and RETB[2] = 7.24
   RET[3] = 7.0054 and RETB[3] = 7.01
   RET[4] = 7.1111 and RETB[4] = 7.11
   RET[5] = 8.8491 and RETB[5] = 8.85
   RET[6] = 6.9555 and RETB[6] = 6.96
   RET[7] = 9.7349 and RETB[7] = 9.73
   RET[8] = 8.1034 and RETB[8] = 8.10
   RET[9] = 6.8346 and RETB[9] = 6.83
```

5 Coding Templates and Auxiliary Files

Template files that facilitate problem coding and improve the legibility of the set source file are available. They have proven to be very useful for multiple choice conceptual problem coding, coding problems with tables, labeling graphics in problems, etc. and can be expanded and modified to suit a user's particular needs. For more detailed information about these templates, view the output of sets 3-18 in the *nsc121s9* class folder (print out or display on a web browser).

1. The */demolibrary/Tools* directory contains the following files:

```
GreekWeb          Numprob-withfig   StdMacros          lets_chem
GreekWeb2         Problem#          StdUnits           parProblem#
Numprob           StdConst          Stylefile-include
```

   The files include some variable definitions that can be used to simplify problem-coding.

2. */demolibrary/MCTools* directory contains files that were written to simplify the coding of qualitative conceptual questions, labeled graphics, changing variable answer questions into multiple choice questions for exams, etc. Answers types include T/F, matching, ranking, exam style, labeling, etc.. These are used to build new problems by posting the text of a template into a new problem. Note that the code in *exam1of8* can be pasted into a problem and edited to transform the single numerical answer format into a multiple choice format with 8 choices. Listed on the next page are the templates included within the MCTools directory.

```
0-ReadMe_All      SM4aux          i4p5auxw        i9p4auxw
0-ReadMe_iXpY     SM5             i4p5w           i9p4w
90.120grids       SM5aux          i5p3auxw        i9p5auxw
90.90grids        SM6             i5p3w           i9p5w
M3T7              SM6aux          i5p4auxw        i9p6auxw
M3T7aux           SM7             i5p4w           i9p6w
M4T7              SM7aux          i5p5auxw        i9p7auxw
M4T7aux           SM8             i5p5w           i9p7w
M5T7              SM8aux          i5p6auxw        i9p8auxw
M5T7aux           SM9             i5p6w           i9p8w
M6T7              SM9aux          i6p3auxw        i9p9auxw
M6T7aux           exam1of8        i6p3w           i9p9w
M7T7              exam2E1W_aux    i6p4auxw        label-10auxw
M7T7aux           exam2E2W_aux    i6p4w           label-10w
M8T7              exam2E_aux      i6p5auxw        label-11auxw
M8T7aux           exam2Epm_aux    i6p5w           label-11w
M9T7              exam3of6        i6p6auxw        label-12auxw
M9T7aux           exam3of6ABaux   i6p6w           label-12w
Nof3              exam3of6CDaux   i6p7auxw        label-13auxw
Nof3-Ex-Hw        exam3of6EFaux   i6p7w           label-13w
Nof3P1            exam3of9        i7p3auxw        label-14auxw
Nof3P1aux         exam3of9ABCaux  i7p3w           label-14w
Nof3aux           exam3of9DEFaux  i7p4auxw        label-15auxw
Nof4              exam3of9GHIaux  i7p4w           label-15w
Nof4P1            fig10to10auxw   i7p5auxw        label-16auxw
Nof4P1aux         fig10to10w      i7p5w           label-16w
Nof4aux           fig11to11auxw   i7p6auxw        label-17auxw
Nof5              fig11to11w      i7p6w           label-17w
Nof5aux           fig12to12auxw   i7p7auxw        label-18auxw
Nof6              fig12to12w      i7p7w           label-18w
Nof6aux           fig13to13auxw   i7p8auxw        label-2auxw
Nof7              fig13to13w      i7p8w           label-2w
Nof7aux           fig4to4auxw     i8p3auxw        label-3auxw
Nof8              fig4to4w        i8p3w           label-3w
Nof8aux           fig5to5auxw     i8p4auxw        label-4auxw
Nof9              fig5to5w        i8p4w           label-4w
Nof9aux           fig6to6auxw     i8p5auxw        label-5auxw
Rank3             fig6to6w        i8p5w           label-5w
Rank3aux          fig7to7auxw     i8p6auxw        label-6auxw
Rank4             fig7to7w        i8p6w           label-6w
Rank4aux          fig8to8auxw     i8p7auxw        label-7auxw
Rank5             fig8to8w        i8p7w           label-7w
Rank5aux          fig9to9auxw     i8p8auxw        label-8auxw
Rank6             fig9to9w        i8p8w           label-8w
Rank6aux          i4p3auxw        i8p9auxw        label-9auxw
SM3               i4p3w           i8p9w           label-9w
SM3aux            i4p4auxw        i9p3auxw        labelingweb.aux
SM4               i4p4w           i9p3w
```

3. The */demolibrary/Tables5* directory includes the following templates to make tables that will both print and display on the web.

```
Tables5.aux           row1column4          row5column2
Tables5.aux.noborder  row1column5          row5column3
Tables5pg1.aux        row1column6          row5column4
Tables5pg2.aux        row1column7          row5column5
readme                row2column2          row5column6
row10column2          row2column3          row5column7
row10column3          row2column4          row6column2
row10column4          row2column5          row6column3
row11column2          row2column6          row6column4
row11column3          row2column7          row7column2
row11column4          row3column2          row7column3
row12column2          row3column3          row7column4
row12column3          row3column4          row8column2
row12column4          row3column5          row8column3
row13column2          row3column6          row8column4
row13column3          row3column7          row9column2
row13column4          row4column2          row9column3
row14column2          row4column3          row9column4
row14column3          row4column4          rowRcolumnC.1
row14column4          row4column5          rowRcolumnC.2
row1column2           row4column6
row1column3           row4column7
```

4. Note: The file *teacher/CAPA5/nsc121s9/HWTop* contains two lines that should be edited. That part of the file is shown below.

```
.....
//
//************* Edit the 2 statements below to suit ****************
//
/LET coursename="Sample \capa Questions"
/LET DeptID="College of Natural Science, Michigan State University"
//
//****************************************************************
//
```

This file contains the header for assignments or exams, as well as defining the variable *stdendline* (standard end line). The file should be edited to reflect the name of the course, department, and University. Note that it is coded so that its information is not part of the text sent to vt100 terminals as this would waste several lines on the screen each time the first problem is displayed. Thus the *HWTop* file is imported in each set **after** the *StdMacros* file.

6 Prototype Set with Three Problems

6.1 General Description.

The source code for a class's problems are contained in files named *setx.qz* where *x* is the problem set number. These files usually incorporate information from other files (problem libraries) using the import command, `/IMP`. The example[7] below was assembled with Quizzer by (1) selecting File in the main menu then (2) choosing New. You will then be prompted to choose the *capa.config* file for the class you are working in, for example */usr/users/teacher/CAPA5/nsc121s9/capa.config*. In the new window, (3) click Std. Header then (4) Import three times, each time selecting a problem to be imported from the */demolibrary /type00/sample-probxx*. Each time you import a problem you will be asked to set the weight of the problem and the number of tries allowed. Finally, (5) click Endline. Note: each time you click on one of the buttons, you will be prompted if your cursor is in the correct position. Make sure the cursor is at the beginning of a blank line and not within header and question text.

[7] Boxed items denote Clicking on a Button selection

```
//CAPA system software is copyrighted by Michigan State University.
//By using these materials, the User agrees to:
//1) Protect the source code files  from unauthorized copying.
//2) Limit  access  of the source material to teaching staff.
//3) The User is free to mix, cut and paste, modify, adapt, delete,
//   improve, etc. the problems and graphics for his/her own use.
//
/IMP "/demolibrary/Tools/StdMacros"
/IMP "/demolibrary/Tools/StdUnits"
/IMP "/demolibrary/Tools/StdConst"
/IMP "/demolibrary/Tools/GreekWeb2"
/IMP "HWTop"
//
//
/BEG prob_val=1
/LET try_val=99
/LET hint_val=1
/DIS("/demolibrary/type00/sample-prob01")
/IMP "/demolibrary/type00/sample-prob01"
/DIS(stdline)
//
/BEG prob_val=1
/LET try_val=99
/LET hint_val=1
/DIS("/demolibrary/type00/sample-prob02")
/IMP "/demolibrary/type00/sample-prob02"
/DIS(stdline)
//
/BEG prob_val=1
/LET try_val=99
/LET hint_val=1
/DIS("/demolibrary/type00/sample-prob03")
/IMP "/demolibrary/type00/sample-prob03"
/DIS(stdline)
/END(stdendline)
```

CAPA processes the above set in a number of ways. By choosing Create .dvi and selecting a student by entering `A87654321`, one obtains the result displayed on the following page. This set is similar to *nsc121s9/set2.qz* in the distribution.

Student, Jaimie . **Section 1**

Sample CAPA Questions *Set 2*

nsc121s9 Due Fri, May 28, 1999 at 10:00 CAPA_ID 8239

/demolibrary/type00/sample-prob01

1. [1pt] Find the area of a rectangle with a length of 8.0 cm and width of 4.0 cm.

<== The first 3-lines on the left are generated by the HWTop file. This file uses definition in the Std-Macros file so that the order of /IMP in the set$x$.qz is important.

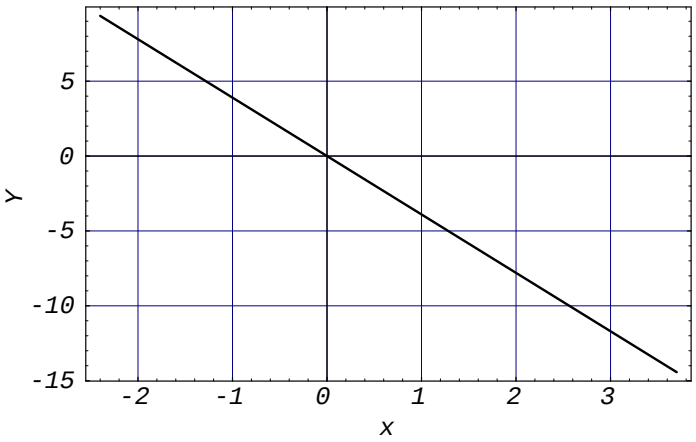
<== This line shows the path of the imported problem. It is essential for classifying the content of a library. It is also useful while debugging a set. Before printing for students, just change the `/DIS...` to a comment, i.e., to `//DIS...` in the *set$x$.qz* file.

/demolibrary/type00/sample-prob02

2. [1pt] The graph below shows the function

$$Y = ax$$

Make a careful determination of the value of a.



<== The paths for *.ps* or *.gif* version of the picture are specified in the problem file. This is one of 14 pictures randomly selected for students as will be seen below from the problem code.

/demolibrary/type00/sample-prob03

3. [1pt] A mass M = 0.13 kg is on a smooth horizontal surface with negligible friction. It is attached to a spring with spring constant k = 15.3 N/m. The mass is displaced from its equilibrium position by a distance of 0.14 m and then released from rest. (Give all of the correct answers, i.e., A or BC or ABC...)

A) The amplitude doesn't vary with time

B) The motion of M is simple harmonic motion.

C) The frequency of oscillation is independent of k.

<== The point value here is set by the `/BEG prob_val=2` at the start of Problem 3. [Note: `/BEG` is just an alias for `/LET`

<== This type of problem should have at least 4 statements unless it is in an exam where students have just one try. That is, at least one of the three statements should have two choices for *CAPA* to randomize from. Therefore, there may be several versions of each of the statements (see problem code in "Problem 3" section.)

6.2 Imported Standard Files.

All the files below should be read as examples of CAPA coding and may be edited and adapted by users.

- The */demolibrary/Tools/StdMacros* contains a set of useful macros such as "stdline" to facilitate coding.
- The */demolibrary/Tools/StdUnits* has a large number of unit combinations pre-formatted for presentation on the printed page and Web browsers.
- The */demolibrary/Tools/StdConst* defines variables for a number of often used fundamental constants.
- The */demolibrary/Tools/GreekWeb2* file is a list of variables for displaying greek letters, symbols, etc. You can view the symbols in the output of *set13.qz* in the distribution class.
- The *HWTop* file resides in the classname folder. Two lines which construct the stdendline require editing. They are visually separated from other items in the file by rows of asterisks. Please edit these lines to reflect the correct course and department information.

```
//************* Edit the 2 statements below to suit ****************
/LET coursename="Sample \capa Questions"
/LET DeptID="College of Natural Science, Michigan State University"
//*****************************************************************
//
/IF (tex("Y","N")=="Y")
/DIS("\noindent{\large\bf "+name()+"\hfill Section {\Large "+section()+"}}\newline")
/DIS("\vskip -.1in")
/DIS("\noindent {\large \it "+coursename+" \hfill Set "+set()+"}\newline")
/DIS("\vskip -.16in")
/DIS("\noindent{\bf "+class()+"} Due "+due_date()+"\hfill CAPA\_ID "+capa_id())
/DIS("\vskip .05in \pagestyle{empty}")
/ENDIF
/DIS(html("<b>"+name()+"</b>"))
/LET stdendline=stdendlineA+DeptID+stdendlineB  //A & B are in  Tools/StdMacros
```

- *HWTopSN* is similar except for an extra line of code which displays the student number. This is not recommended because the combination of the student number and CAPA ID is all that is needed to access the system for this and preceeding problem sets.

6.3 Problem 1.

The text of */demolibrary/type00/sample-prob01* is:

```
//************************************
//BEG prob_val=1
//LET try_val=20
/IMP "/demolibrary/Tools/Problem#"
// By  kashy@nscl.msu.edu, No commercial use.
/HIN The area of a rectangle is the product of the two sides.
/EXP Multiply the length by the width and enter the result.
//
/LET long=random(7.0,9.1,1.0)
/LET wide=random(2.0,5.1,2.0)
//
Find the area of a rectangle with a length of  /DIS(long:1f)\
/DIS(cm_u) and width of /DIS(wide:1f)/DIS(cm_u).
//
/LET area=long*wide
/ANS(area:1f,tol=0.2,wgt=prob_val,tries=10,hint=hint\_val,unit="cm^2")
//************************************
```

<== The `//BEG prob_val=2` and `//LET try_val=2` are comments. The variables `prob_val` and `try_val` are defined in the *setx.qz*.

<== Hints and Explanations are optional. Additional hints can be inserted directly in the *setx.qz* file for each problem.

<== This questions was built when the *CAPA* sets were built on a different computer than the ones that the students logged into. There were slight parsing differences with respect to real numbers between the computers. Therefore, adding a little to the high end avoided this problem (i.e. using 9.1 instead of 9.0 in the definition of `long`). Even better though would be to parse only random integers and avoid the problem entirely.

<== Note the unit for the problem is defined as a string.

6.4 Problem 2.

/demolibrary/type00/sample-prob02:

```
//************************************
//BEG prob_val=1
//LET try_val=20
/IMP "/demolibrary/Tools/Problem#"
//By E. Kashy, kashy@nscl.msu.edu, No commercial  use
//
/LET k=random(1,14,1)
/LET file=choose(k,"35","45","56","62","77","86","93","m26",\
                  "m39","m47","m51","m66","m74","m84")
//
//
//
//
//
/LET slope=choose(k,3.5, 4.5, 5.6, 6.2, 7.7, 8.6, 9.3, -2.6,\
                  -3.9, -4.7, -5.1, -6.6, -7.4, -8.4)
//
//
//
//
/LET TexGraph="\epsfxsize=3.6in \epsffile{/demolibrary"+ \
"/Graphics/Gtype00/Y"+label+"x.ps}"
/LET WebGraph="<IMG SRC=/demolibrary/Graphics/Gtype00/Y"+ \
label+"x.gif"
//
//
/LET constStr=web("a","$a$","<i>a</i>")
/LET functStr=web("Y=ax","\\ \centerline{$Y=ax$}","<i>Y=ax</i>")
//
//
The graph shows the function /DIS(functStr)
Make a careful determination of the value of /DIS(constStr).
//
/DIS(web("", TexGraph, WebGraph))
//
//
/ANS(slope:2E,tol=5%,wgt=prob_val,hint=hint\_val,tries=try_val)
//************************************
```

<== The `/IMP` command imports a file to format the problem number.

<== The computer code picks a random number, k. For k=4, the choose function selects the string variable, 62 which is then concatenated into the name of the .eps or .gif file to be displayed. NOTE: The back slash is a line continuation character which hides the carriage return so that the `/LET` statement has no carriage return.

<== The answer (slope) is also specified with a `choose()` function using the same selection index `k`. Thus, the data in the first `choose()` function corresponds to the answers in the second `choose()` function.

<== Commands and paths for the TeX and Web figures. The continuation character is outside the "string".

<== This formats the equation for Web and TeX

<== Text of problem.

<== Displays the figure in TeX and Web.

6.5 Problem 3.

/demolibrary/type00/sample-prob03:

```
//************************************
//BEG prob_val=1
/IMP "/demolibrary/Tools/Problem#"
//
//By E. Kashy, kashy@nscl.msu.edu, No Commercial Use
// /demolibrary/CAPA46/MCTools/Nof3  Select N correct
//of 3 Statements
//
A mass M = 0.13/DIS(kg_u) is on a smooth horizontal surface \
with negligible friction. It is attached to a spring with \
spring constant k = 15.3/DIS(NPm_u). The mass is displaced \
from its equilibrium position by a distance of 0.14 m and \
then released from rest. (Give all of the correct answers, \
i.e., A or BC or ABC...)
//---------------------------------------
/LET s1a="The motion of M is simple harmonic motion."
/LET s1b="Statement 1 variation b"
/LET s1c="Statement 1 variation c"
/LET s1d="Statement 1 variation d"
/LET mix1=random(1,1,1)
/LET a1a=1
/LET a1b=26
/LET a1c=26
/LET a1d=26
//
/LET s2a="The amplitude varies with time."
/LET s2b="The amplitude doesn't vary with time"
/LET s2c="Statement 2 variation c"
/LET s2d="Statement 2 variation d"
/LET mix2=random(1,2,1)
/LET a2a=2
/LET a2b=1
/LET a2c=26
/LET a2d=26
//
/LET s3a="The frequency of oscillation is independent of k."
/LET s3b="The Period of oscillation is independent of k."
/LET s3c="The frequency of oscillation depends of k."
/LET s3d="Statement 3 variation d"
/LET mix3=random(1,3,1)
/LET a3a=2
/LET a3b=2
/LET a3c=1
/LET a3d=26
//
//
/IMP "/demolibrary/MCTools/Nof3aux"
//
//
/ANS(Nof3right,wgt=prob_val,str=mc,hint=hint_val,tries=try_val)
//************************************
```

<== This problem starts as a template *Nof3* which is located in */demolibrary/MCTools*.

<== Certain variables (`kg_u, NPm_u`) are not defined within the problem. Because they are used often, they were defined in a files imported once at the start of the set (`/IMP "/demolibrary/Tools/StdUnits"`)

<== The `mix1` variable will allow only the one variation of this statement and all students will have it. It is identified as correct by `a1a=1`.

<== The `mix2` variable will select either of 2 variations of this statement for a particular student. The first leaf is identified as incorrect by `a2a=2`, while the second is correct, with `a2b=1`. This templates allows up to 4 variations for each statement.

<== The `mix3` variable will select one of the 3 variations of this 3rd statement to be given to a particular student.

<== The *.aux* file randomizes the order and formats presentation. Note the names of the statement and answer variables s3a, s3b, a3a, a3b, etc. must correspond to those used in the *Nof3aux* file.

<== The three letters in the answer can be in any order because `str=mc` is selected.

7 Sample Problem Set.

- A set of sample questions is included in the distribution called *set1.qz* in *nsc121s9* class directory. The next page displays the printed version for one student. The *CAPA* source code is given on the following pages. The set is assembled from problems in the */demolibrary* included with *CAPA*.
- The source code for imported problems can be seen by opening */demolibrary/typexx/msu-probyy.txt*. as a reference file in Quizzer. You may also check out the source code by opening */your/path/teacher/CAPA5/nsc1* in Quizzer and double clicking on the highlighted links to the individual problems.
- Note that two-sided (duplex) laser printing is useful when making assignments or exams that require more than one page.
- In physics and chemistry, several users have agreed to share and exchange problems for use in their classes. In physics, this is evolving into the exchange of local physics libraries (i.e. */msuphyslib*) among users. Authors retain commercial rights. A comment line within each problem identifies the author. Also, an e-mail address is useful if some question about the problem or its coded solution arises. Users are also to edit and adapt the problem for their classes.

Student, Jaimie . **Section 1**

Sample CAPA Questions *Set 1*

nsc121s9 Due Sat, Jan 1, 2000 at 16:00 CAPA_ID 2318

Student Number **a87654321**

1. [1pt] The following are possible ways to express the quantity 0.513 (Give ALL correct answers, i.e., B, AC, BCD...) Note: 3.45E-8 is a way you can enter the number 3.45×10^{-8} in most computers.

A) 5.13E-1

B) 0.0513E+1

C) 0.000513E+4

D) 51.3×10^{-3}

E) 0.513E-2

F) 0.00513×10^{2}

2. [1pt] Calculate the perimeter of a rectangle with a length of 19.5 cm and a width of 40 mm. Enter units.

3. [1pt] Calculate, in cm, the perimeter of a rectangle with a length of 13.5 cm and a width of 37 mm.

4. [1pt] Calculate the perimeter of a rectangle with a length of 22.50 cm and a width of 48.0 mm.

5. [1pt] Test your calculator skills by checking that $(4.50)^2$ equals 20.25 and that $(4.50)^3$ equals 91.13. Now, find the value of $(4.50)^{2.27}$.

6. [1pt] Reduce the fraction **90/60** to its lowest terms. [If the result is 4/3, enter 4 (Answer1), and 3 (Answer2)]

7. [1pt] Calculate the volume of a spherical balloon which has a surface area of 0.0373 m^2.

8. [1pt] List the following in order of increasing lengths from shortest to longest. (If B is shortest, then A, then C, and D is longest, enter BACD)

A) 40 mm

B) 1.00 $in.$

C) 0.060 m

D) 0.10 ft

9. [2pt] Match each person with the most appropriate description. (If the first corresponds to B, and the next 6 to C, enter BCCCCCC)

1) Leonardo DaVinci	A. Philosopher
2) Alfred P. Sloan	B. Politician
3) Socrates	C. Philanthropist
4) Percy Shelly	D. Poet
5) Richard M. Nixon	E. Painter
6) Edgar Allen Poe	
7) Emanuel Kant	

10. [1pt] The sequences for each of three fragments of DNA are shown below:
fragment 1 = TTTAAGGAAGAGGCC
fragment 2 = GAGGCCAATTCG
fragment 3 = CTGGCCACTTTTAAG
Assume that the target DNA sequence has 30 bases and reconstruct that DNA sequence.

11. [1pt] The nine elements of a 3x3 matrix are shown below.

3	2	-2
3	-3	3
-7	-1	3

Evaluate the determinant of that matrix.

12. [2pt] In order to practice for a hammering-speed contest, a person purchases the following items:

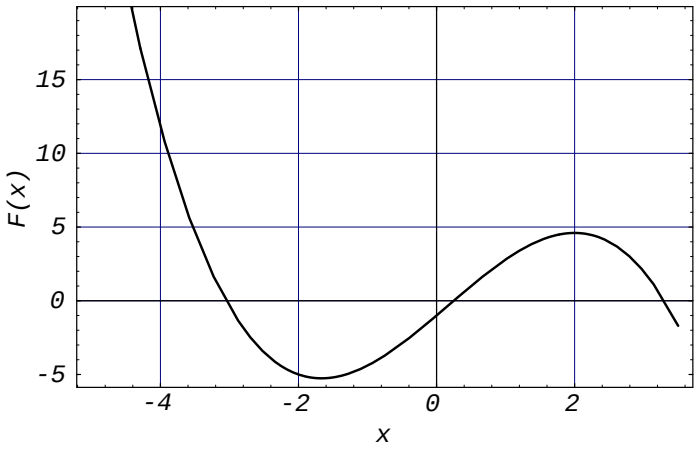
– Item –	Number	$ ea.
Hammer	1	16.77
Nails	72	0.08
Wood Blocks	5	4.25
Band Aids	36	0.14

Calculate the total cost, including a sales-tax of 6.40%. (in $)

13. [1pt] A block is at rest on an inclined plane whose elevation can be varied. The angle of elevation θ is increased slowly from the horizontal, and when it reaches 36.7 degrees, the block begins to slide. Calculate the coefficient of static friction.

14. [1pt] The block reaches a speed of 3.19 m/s in a time of 1.23 s. Calculate the coefficient of kinetic friction.

15. [1pt] Using graphical methods, determine at x = -3.6 the value of the derivative of function $F(x)$ plotted in the graph below.



16. [3pt] A fisherman and his young daughter are in a boat on a small pond. Both are wearing life jackets. The daughter is holding a large floating helium filled balloon by a string. Consider each action below independently, and indicate whether the level of the water in the pond R-Rises, F-Falls, S-Stays the Same, C-Can't tell. (If in the first the level Rises, and in the second it Falls, and for the rest one Cannot tell, enter RFC-CCC)

A) The fisherman lowers himself in the water and floats.

B) The fisherman fills a glass with water from the pond and drinks it.

C) The fisherman lowers the anchor and it hangs vertically, one foot above the bottom of the pond.

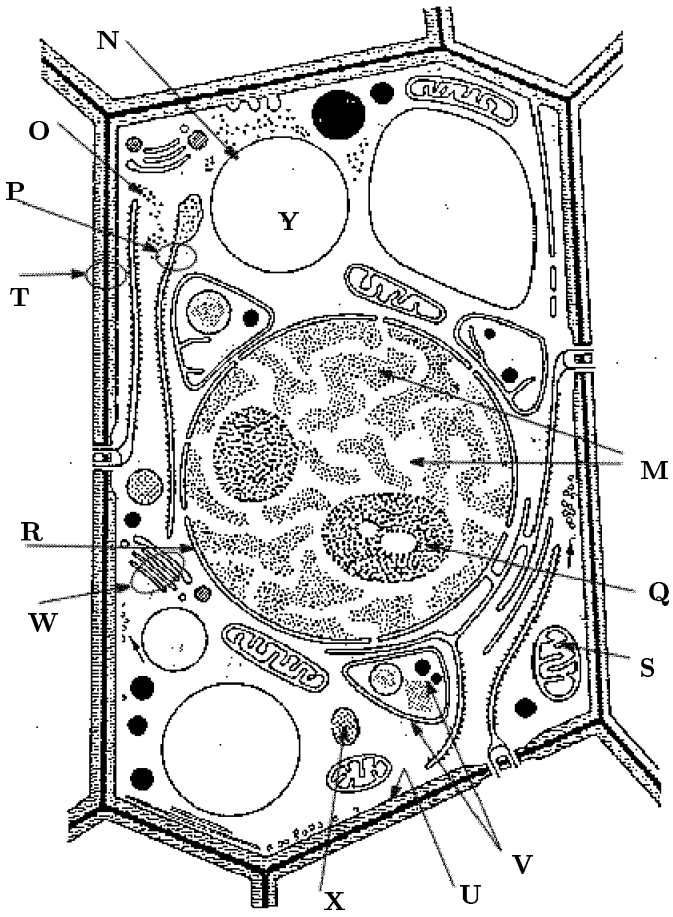
D) The fisherman knocks the tackle box overboard and it sinks to the bottom.

E) The daughter pops the balloon.

F) The daughter gets in the water, looses her grip on the string, letting the balloon escape upwards.

(Over)

17. [3pt] Match the appropriate letter on the diagram with each organelle in the sequence in which they are listed. (Example: If the first organelle corresponds to D on the diagram and the next to C, begin your answer with DC...)



1) Nucleus
2) Ribosomes
3) Nuclear Envelope
4) Cell Wall
5) Plastid (Chloroplast, proplastid, ...
6) Nucleolus
7) Endoplasmic Reticulum
8) Golgi Body
9) Vacuole
10) Cell or Plasma Membrane
11) Peroxisome
12) Tonoplast or Vacuolar Membrane
13) Mitochondrion

---

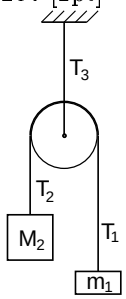
18. [1pt] Consider the reaction of nitrous oxide N_2O with oxygen O_2 to form nitrogen dioxide NO_2:

$$2N_2O_{(g)} + 3O_{2(g)} \rightarrow 4NO_{2(g)}$$

A 5.0 liter vessel contains N_2O gas at 5.0 atm pressure. A second vessel, 2.0 L in volume contains oxygen at 5.0 atm pressure. Now suppose that the two vessels are connected by a pipe of negligible volume and the two gases mix and react to produce as much nitrogen dioxide as possible. Assume that the temperature remains constant. What is the pressure in the apparatus at the end of the reaction?

---

19. [2pt]

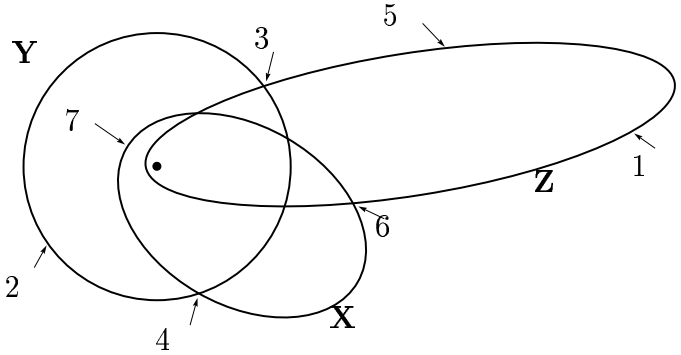


A frictionless, massless pulley is attached to the ceiling, in a gravity field of $9.81\,m/s^2$. Mass M_2 is greater than mass m_1. The quantities T_n and g are magnitudes. (For each statement select T-True, F-False, G-Greater than, L-Less than, or E-Equal to.)

A) m_1g + M_2g is .... T_3
B) T_2 is .... T_1
C) m_1g is ... T_1
D) The center-of-mass does not accelerate.
E) The magnitude of the acceleration of M_2 is ... the magnitude of the acceleration of m_1.
F) T_3 is .... T_1 +T_2

---

20. [2pt] Asteroids **X**, **Y**, **Z** have equal mass (6 kg each). They orbit around a planet with M=7 $\times$ 10^{24} kg. The orbits are in the plane of the paper and are drawn to scale.



TE, KE and PE represent Total, Kinetic and Potential energies. Select G-Greater than, L-Less than, or E-Equal to. (If the first is G and the rest L, enter GLLLLLL).

A) The KE of **Y** at 3 is .... its value at 2.
B) The speed of **Z** at 6 is .... it is at 5.
C) The PE of **X** at 4 is .... the PE of **Y** at 4.
D) The PE of **X** at 4 is .... the PE of **Y** at 3.
E) The TE of **Z** is .... the TE of **X**.
F) The TE of **Y** is .... the TE of **X**.
G) The PE of **X** at 7 is .... its value at 6.

---

21. [5pt] Write a short essay (200 words or less) about the experiment shown in PHY183 class on Tuesday, November 24, and its interpretation. The text of the video can be found in http://capa2.nscl.msu.edu/demolibary/Links/ALNf7/freefall.html. Submit your answer via CAPA (Not by e-mail). Teams of up to five students may collaborate on a single submission.

---

22. [2pt] Vector $\mathbf{C} = \mathbf{A}\times\mathbf{B}$, where

$$\mathbf{A} = 15\mathbf{i} - 17\mathbf{j} - 20\mathbf{k}$$
$$\mathbf{B} = 15\mathbf{i} + 25\mathbf{j} - 20\mathbf{k}$$

The unit vectors in the x, y and z directions are **i**, **j** and **k** respectively. **A** is a displacement and has units of m, and **B** is a force and has units of N. Calculate vector **C** and enter the numerical values of its x, y, and z components.

Source Code for *nsc121s9/set1.qz*

```
//CAPA system software is copyrighted by Michigan State University.
//By using these materials, the User agrees to:
//1) Protect the source code files  from unauthorized copying.
//2) Limit  access  of the source material to teaching staff.
//3) The User is free to mix, cut and paste, modify, adapt, delete,
//   improve, etc. the problems and graphics for his/her own use.
//
/IMP "/demolibrary/Tools/StdMacros"
/IMP "/demolibrary/Tools/StdUnits"
/IMP "/demolibrary/Tools/StdConst"
/IMP "/demolibrary/Tools/GreekWeb2"
/IMP "HWTopSN"
/START
//************************************
/IMP "/demolibrary/Instructor_Notes.txt"
/DIS(stdline)
//
/LET hint_val=1
//
/BEG prob_val=1
/LET try_val=12
//DIS("/demolibrary/type01/msu-prob10.txt")
/IMP "/demolibrary/type01/msu-prob10.txt"
/DIS(stdline)
//
/BEG prob_val=1
/LET try_val=12
//DIS("/demolibrary/type01/msu-prob13.txt")
/IMP "/demolibrary/type01/msu-prob13.txt"
/DIS(stdline)
//
/BEG prob_val=1
/LET try_val=12
/LET hint_val=1
//DIS("/demolibrary/type01/msu-prob12.txt")
/IMP "/demolibrary/type01/msu-prob12.txt"
/DIS(stdline)
//
/BEG prob_val=1
/LET try_val=12
//DIS("/demolibrary/type01/msu-prob15.txt")
/IMP "/demolibrary/type01/msu-prob15.txt"
/DIS(stdline)
//
/BEG prob_val=1
/LET try_val=12
//DIS("/demolibrary/type01/msu-prob11.txt")
/IMP "/demolibrary/type01/msu-prob11.txt"
/DIS(stdline)
//
/BEG prob_val=1
/LET try_val=12
/LET hint_val=2
//DIS("/demolibrary/type-math/fraction-LCD-235")
/IMP "/demolibrary/type-math/fraction-rlt-235"
/DIS(stdline)
//
```

```
/BEG prob_val=1
/LET try_val=12
//DIS("/demolibrary/type01/msu-prob07.txt")
/IMP "/demolibrary/type01/msu-prob07.txt"
/DIS(stdline)
//
/BEG prob_val=1
/LET try_val=12
//DIS("/demolibrary/type03/msu-prob21.txt")
/IMP "/demolibrary/type03/msu-prob21.txt"
/DIS(stdline)
//
/BEG prob_val=2
/LET try_val=12
//DIS("/demolibrary/type-other/match.txt")
/IMP "/demolibrary/type-other/match.txt"
/DIS(stdline)
//
/BEG prob_val=1
/LET try_val=12
/LET hint_val=2
//DIS("/demolibrary/type-biology/dna-3fragmentsB")
/IMP "/demolibrary/type-biology/dna-3fragmentsB"
/DIS(stdline)/DIS(newpage)
//
/BEG prob_val=1
/LET try_val=12
/LET hint_val=2
//DIS("/demolibrary/type-math/det33a.txt")
/IMP "/demolibrary/type-math/det33a.txt"
/DIS(stdline)
//
/BEG prob_val=2
/LET try_val=12
/LET hint_val=1
//DIS("/demolibrary/type-math/percentStoryA.txt")
/IMP "/demolibrary/type-math/percentStoryA.txt"
/DIS(stdline)
//
/BEG prob_val=1
/LET try_val=12
//DIS("/demolibrary/type10/msu-prob29a.txt")
/IMP "/demolibrary/type10/msu-prob29a.txt"
/DIS(webonlyline)
//
/BEG prob_val=1
/LET try_val=12
//DIS("/demolibrary/type10/msu-prob29b.txt")
/IMP "/demolibrary/type10/msu-prob29b.txt"
/DIS(stdline)
//
//
/BEG prob_val=1
/LET try_val=12
//DIS("/demolibrary/type02/msu-prob09.txt")
/IMP "/demolibrary/type02/msu-prob09.txt"
/DIS(stdline)
//
/BEG prob_val=3
```

```
/LET try_val=12
//DIS("/demolibrary/type32/msu-prob12v2.txt")
/IMP "/demolibrary/type32/msu-prob12v2.txt"
/DIS(stdline0vr)/DIS(newpage)
//
/BEG prob_val=3
/LET try_val=12
//DIS("/demolibrary/type-botany/01-organelles.txt")
/IMP "/demolibrary/type-botany/01-organelles.txt"
/DIS(stdline)
//
/BEG prob_val=1
/LET try_val=12
/LET hint_val=1
//DIS("/demolibrary/type-chemistry/limiting-reagent.txt")
/IMP "/demolibrary/type-chemistry/limiting-reagent.txt"
/DIS(stdline)
//
/BEG prob_val=2
/LET try_val=20
//DIS("/demolibrary/type09/msu-prob89.txt")
/IMP "/demolibrary/type09/msu-prob89.txt"
/DIS(stdline)
//
/BEG prob_val=2
/LET try_val=20
//DIS("/demolibrary/type30/msu-prob13a.txt")
/IMP "/demolibrary/type30/msu-prob13a.txt"
/DIS(webonlyline)
//
//BEG prob_val=2
//LET try_val=20
//DIS("/demolibrary/type30/msu-prob13b.txt")
//IMP "/demolibrary/type30/msu-prob13b.txt"
/DIS(stdline)
//
/BEG prob_val=5
/LET try_val=3
//DIS("/demolibrary/type-other/subjective")
/IMP "/demolibrary/type-other/subjective"
/DIS(stdline)
//
/BEG prob_val=2
/LET try_val=12
//DIS("/demolibrary/type-other/vectorProduct.txt")
/IMP "/demolibrary/type-other/vectorProduct.txt"
/END(stdendline)
```

8 Guided Tour of the *CAPA* System

This section presents a guided tour through the *CAPA* software system by relying on the files that are part of the distribution package. This tour assumes that the user is logged in as "teacher" and owns the *nsc121s9* class directory. If you do not have permission to login as "teacher", you will be unable to complete sections of the tour which involve writing files to the "teacher" owned directories (i.e. creating *.db* headers and *.dvi* files, printing, generating class records, etc.). [8]

1. **Guided Tour of** Quizzer

   - Instructions for running the Quizzer application:
     - Prior to opening the Quizzer application, set up your X-server software to interact with your server. See the section on using *CAPA* via eXceed, X-win32 or MacX.
     - Through your X-server software, open an rlogin (or telnet) session to your server.
     - Start the application by entering the command Quizzer at the prompt.
   - Select the Source of the problem set.
     - The Quizzer main menu will be placed in the upper left corner of your screen.
     - Select File, then Open.
     - A pop-up screen displaying your current directory will allow you to browse. The file filter, which for quizzer will read **.qz*, selects only *setx.qz* files for editing with the Quizzer application.
     - In this pop-up screen, open the CAPA5 directory.
     - Find and select the sample class file nsc121s9.
     - Open the set1.qz file. (*set1.qz* is identical to the Set of Sample Problems in this manual).
     - The contents of this file will show up in Quizzer's editing window. Browse through the contents of this question file. Be careful not to make any changes at this time, or at least **not** to Save any changes. If you think you did accidently make changes, choose Quit from the main menu and click No if you are asked to save.
     - Click on a highlighted imported file. A reference file pops up in a new window. Close this window by choosing File then Close. A dialog box will appear to confirm your selection.
     - You may wish to use the Save As... under File in the menu option and save the set as *set99.qz*, then close it and open *set1.qz* again.
     - If you make an inadvertent change or if you are not sure of a change, then just Quit the Quizzer application without saving and restart it by entering the command Quizzer.
   - Preview the example question file in enscript mode.
     - Choose Enscript mode button from the options at the top of the Quizzer edit window.

[8] Boxed items denote Clicking on a Menu selection

  - Click the Preview button. A dialog window will appear to ask you to select a specific student from the class list. The choice may be made by randomly selecting a student from a specified section or by entering the student number of a particular student in the class.
  - Before previewing, click File in the menu and select Open Reference File... . Find and open the classl file in the *nsc121s9* folder. You will see all the students in the class, including "Student Jamie ." Select Close under the File menu in the *classl* file window.
  - Select the random-selection button and click Preview.
  - Note: To select a specific student, first, click the Preview button, select Specify the student by, and then enter the Student Number `A87654321` into the text entry panel *followed by a carriage return.* If the Student number is not in the class list file (*teacher/CAPA5/nsc121s9/classl*) an error message is returned.
  - A preview window will appear. This window displays the text from the translated question file as it would be presented to the students during a terminal session. The instructor can use this display to determine if the coding is correct and if the presentation on the remote terminal will be adequate.
  - If any changes have been made to the *set1.qz* file in the editor, Quizzer will ask you to save the file. **Don't** save the file at this point unless you are absolutely sure that the file has not been damaged, instead select Cancel, Quit Quizzer, answer No if asked to save, and start again.
- Close the preview window by clicking on the Dismiss button at the top of the window.
- Preview the example question file in TEX mode.
  - Change the **Mode** of the preview output by clicking on the TeX mode button on the upper part of edit window and then click the Preview button as before.
  - Select a particular student, this time type the student number `A87654321` into the text panel, then <return>, select the Specify the student by button, and click Preview.
  - The preview window will display the contents of the *nsc121s9/quiztemp.tex* file that will be used later by LATEX to typeset the problem set for the selected student, in this case "Jamie Student".
  - Click on Dismiss to close this window.
- Preview the set of sample problems in Web mode similar to how enscript and TEX modes were viewed.
  - This mode will generate a preview window displaying the web specific version of your source file. (This is not the complete *.html* file.)
- Specify the date limits in the database header.
  - The database header (DB header) for *set1.qz* is present from the distribution. However, you can bring up a window to modify that information by clicking on the DB Header button at the top of the main edit window. **Note**: You must preview the set in enscript prior to setting or changing the DB Header information. (We have already done this.) Three dates and times must be set:
    (a) **Open date/time**: when the students can begin to enter answers to this problem set.
    (b) **Due date/time**: the time by which students must finish entering answers and properly log out of CAPA, i.e., when all answer-recording stops.

(c) **Answer date/time**: when the answers to the problems will be made available to the students. This should be *after* the Due date/time.

- Use [Load] to view the current values for this problem set. If the values were not present in the *set1.db* file, you would have recieved a warning.
- Setting the DB Header is necessary for each new problem set before students log in. Without the proper date information, students will not be allowed to enter their answers.
  (a) The database header may be set as the number of problems change, but it *must* be set *after* the number of problems in the set is finalized, otherwise serious errors in the grade records will occur. If you have found it necessary to make major changes in your source file, such as changing the problem value or the grading option to hand graded (`hgr=on` inside the `/ANS()` command) be certain to reset the DB Header prior to printing the final version for distribution to your students.
  (b) Once the problem sets have been distributed to students and students begin logging in, do *not* change the number of problems in the set and reset the database header.
- The dates and times must be entered with the numerical format of `YYYY/MM/DD` and `HH:MM` on the appropriate lines. The hours of the day are based on a 24 hour clock.
- You can specify different dates for different sections. Click on [Add] and add different due dates for section 1. You can also choose to get the default dates for section 1.
- In the "adddate" window, click on [More Options]. You can then add a duration time in hours and minutes. This lets the student choose their own due date. Once the students log in, they only have the set duration of time to work one the set (such as on a take home quiz). You also have the choice to give your students no feedback by inhibiting correct/incorrect responces. Click [Cancel] to not change any of the options.
- The "Creating DB Header" window also has a cancel button to exit the window without changing the dates already present in the records folder. Close the "db Header" window by selecting [Cancel].

- Viewing the file as it will be printed.
  - Select [Create .dvi] from the main Quizzer menu. (The creation of a **.dvi file* is a necessary step before printing any TEX document.) There are no Quizzer errors in the *set1.qz* file, and Quizzer will generate a file called *quiztemp.tex*
  - You will again be prompted to select either a random student or a specific student determined by student number. Make your choice, then click [Preview].
  - Quizzer invokes the LATEX program to typeset the *quiztemp.tex* file.
  - Quizzer will create the *quiztemp.dvi* file for viewing the appearance of the printed document. You should now have two new windows open. The *quiztemp.dvi* file is automatically opened and displayed once it is created. This process takes only a few seconds to complete depending on the computer. All of the *quiztemp.** files are in the *nsc121s9/* directory. The quiztemp files are rewritten every time a new *.dvi file* is created.
  - After you are done viewing both windows, dismiss the LATEX output window and the quit the quiztemp window.
  - Note that new problem sets may have coding errors in their CAPA grammar, LATEX grammar, or HTML grammar. These are independent languages. Therefore, previewing in enscript, creating a *.dvi* file, and viewing the problem set in a web browser are necessary to detect and locate each kind of error.

- Printing the set of sample problems through Quizzer.

- From the main Quizzer menu, choose Print...
- The pop-up screen will indicate your options. Choose one of the following:
  (a) Print current .dvi: This option will print the most recent .dvi file you have generated using the "Create .dvi" menu option, i.e. the *quiztemp.dvi* file listed in your directory.
  (b) Randomly select one student from section: This will generate a random assignment from the section you specify to be printed.
  (c) Specify the... Student Number: This will print out a set for the specified student. Once the student number is entered into the field, be certain to press <return> to call the student name from the *classl* file.
  (d) Print section: This will print the entire specified section's papers.
  (e) Print multiple sections: This will open a menu for choosing the sections to be printed.
  (f) Print whole class: This option will print the assignments for the entire class.
- **Note:** Once you have entered a command for printing, another pop-up screen will appear allowing a choice of printers. The printer options are defined in the *capa.config* file. (See discussion of *capa.config file* in the Software Components section.)
- Click Select and then click Print in the "Print Command" window. You may also have a choice of printers and printing options to choose from in this window.
- After the printing has finished, note the *CAPA* ID number at the top of the printed page.

- Viewing the set on the Web
  - Open a browser and point it to the approriate class.html URL
  - Select the class *nsc121s9* by clicking on "class" box and selecting *nsc121s9* from the drop-down menu. This will allow you to view "Jaime Student's" problem set 1.
  - Enter `A87654321` for the student number and `2318` for the CAPA ID
  - Click on the button to try the problem set. Close the browser when you are done browsing.
- Other Quizzer main menu items:
  - All Quizzer submenus can be "torn off". The submenus have a dashed line at the top. If you click on the dashed line, the submenu becomes a separate, static window which can positioned anywhere on your screen.
  - File: This was discussed above, but also includes "Open Reference File" a useful option allowing copying and pasting from any file into the quiz set.
  - Edit: This submenu contains the standard editing commands (cut, copy, paste, select all, undo, and find). The Find option allows you to find and/or replace using different criteria.
  - Prefs: This feature can be used to choose what style of output is displayed, such as just problems, problems and answers, or answers only. This also determines what is printed on the paper through the print command, so be certain to set it to "problems only" prior to printing the entire classes papers. You can also edit the HTML and TEX header and footer files through the preferences menu by clicking on the menu that displays HTMLheader. However, for most applications of *CAPA* you should not need to edit these files.

  - Windows: This submenu displays all the currently open windows in the Quizzer application. If ever a smaller window is hidden by a larger one, you can shuffle the smaller window to the top by selecting it in this menu.
  - Analyze Set: This allows you to view the range of anwers a certain problem will generate. While *set1.qz* is open, select Analyze Set. From the new window, select Run Random, enter the number 10, then click continue. When the set is finished being analyzed, you can select different problem numbers and view the range of answers generated for the *CAPA* problems. Dismiss this window and try it again, but this time select Run Class.
  - Remap: You have the option of remapping either the backspace or the delete key to suit your particular preference. Xdvi Options: This allows you to resize the display of the *.dvi* file on your monitor. These changes will not affect printing.
- Select Quit from the menu to leave the Quizzer program.
- A "MakeSure Prompt" window will appear. You have the option Cancel to continue editing or Yes to quit Quizzer. If you made changes, you will be asked if you want to save them. Answer No at this time.
- The Quizzer main menu then disappears.

2. **Guided Tour of Capalogin**

After the instructor has created a question set (*set1.qz* in this example) and the *records/set1.db* file has the relevant dates and times, students can login through the network. The students can use equipment or a communication program that emulates a vt100 terminal to enter their answers and view the hints. Note that the emulation is very specifically vt100 and not vt102, vt200, etc.

At Michigan State University, the students log into the system from a large number of remote locations on and off campus. They have used a variety of computers, IBM, Mac, etc., that emulate a vt100 terminal. For the guided tour, you will open a terminal window on your host computer.

- Login Instructions: An example of a login instruction handout which has been used at MSU is included in the next section of this manual.
- A student logs in by entering the class account user name, such as `nsc121s9`. This account does not have a password. Therefore, anyone can get in, but they are 'captured' by the code Capalogin and have limited access to the machine (see details in the software component section).
- The student is then asked to enter his or her student number and the *CAPA* ID number of any assignment.
  - The student number has 9 characters. At MSU the last 8 are digits, but the 9 characters can be a mixture of digits and letters. The *CAPA* ID number is unique for each problem set for a given student in a given course. It has 4 digits and will be printed on a student's individual assignment sheet.
  - Each student has a different *CAPA* ID number for each problem set. The Capalogin code will select the *setx.qz* file that corresponds to the CAPA ID. Previous sets can be reviewed anytime.
  - If the current date and time are within the allowed time range specified in the DB Header file, students can enter their answers. The answers are recorded as they are submitted.

  - Problems arise when students open multiple sessions. If two sessions for one student are active at the same time, the computer records the distribution of correct and incorrect answers of the last session to log-out, not necessarily the session with the most credit earned. This is one the reasons for limiting the number of active sessions by one student.

- **Sample login:[A]**
  - Exit your server and then open a new telnet (or **rlogin**) session into the server. The method for this varies depending upon the type of X-server software being used. (See Section 3 on configuration of X-server software.)
  - At the prompt `login:`, enter the classname. Example: `nsc121s9`
  - The capalogin screen will be displayed.
  - Enter: `A87654321` for the student number and then hit <return>. Note that the cursor should be positioned after the colon following the words `Student Number` near the middle of the screen. If it is not, particularly if it is at the bottom of the screen, the terminal emulation is not vt100 and needs to be reset in the terminal emulator.
  - Enter `2318`, the CAPA ID printed on the homework page for student A87654321. The CAPA ID number must correspond to the one printed on the problem set for this student, or the login will not be allowed.
  - The student's name is displayed at the top of the screen and the main *CAPA* menu near the center.
  - Enter: `S` (or `s`, as the menu is not case sensitive) to view the student's summary for the course. No credit should have yet been earned.
  - Enter: `M` to return to the main menu.
  - Enter: `T` to try the problem set.
  - Enter: `1` to try problem 1.
  - Problem 1 of set 1 will appear on the screen.
  - Follow the instructions, noting that if you put in a wrong answer, a `:H` to receive a hint may become an option. Note that the colon in this screen differentiates a command from a problem answer.
  - After answering a few questions, enter `:m` to return to the main menu.
  - Enter: `s` You can now see the updated student summary.
  - Enter: `x` to exit. This ends the session, records the student responses, and closes the connection to the server. The *goodbye.msg* file is then displayed .

- **Sample login:[B]**
  - Open a web browser session into the server.
  - Select the sample class *nsc121s9* from the pop-up menu.
  - Enter: `A87654321` for the student number. Note that you must click in the window.
  - Enter the *CAPA* ID `2318` (as was printed on the homework page for student number A87654321). The *CAPA* ID number must correspond to the one printed on the problem set for this student, or the login will not be allowed.
  - Click on the " Click here to work on *CAPA* " button.
  - The next document you will see will be the main menu where you can select to try the current problem set.
  - Set 1 will appear on the screen.

- Note that if you put in one or more wrong answers, a hint will be displayed if it has been included in the code.
- Note that the problem numbers are listed across the page at the top and bottom of the set. You can click on the problem number and go directly to that problem.
- Once you're finished previewing this set, you can choose to exit the system.

3. Guided Tour of Grader

- Prior to opening the Grader application, set up your X-server software to interact with your server. See Section 3 on using *CAPA* via eXceed, X-win32 or MacX.
- Through your X-server software, open an rlogin (or telnet) session to your server.
- If necessary, change directory to the directory in which the Grader application is located.
- Start the application by entering the command Grader.
- The Grader main menu will appear in the upper left corner of your screen.
- Choose File then Specify Class
- A pop-up window appears with "Please select a capa.config file" on the title bar. Search the directory to find the class (in this case, *nsc121s9*). You may need to go back one directory (by clicking the folder button in the upper right hand corner of the window). Double click the nsc121s9 directory, then double click on the *capa.config* file. A new window will appear with *nsc121s9* as the class.
- Specify a section number and problem set number:
  - Enter 1 for the section number in the text entry panel.
  - Enter 1 for the problem set number.
  - Now, click Load Scores button. Grader will scan the record files and the (short) list of students in section 1 and their current grades will appear. Note, for example, the grade of student A87654321. For classes with large student enrollments and long problem sets, the display of the section grades may take a few seconds.
- Select a student to grade:
  - All the students in the selected section are contained in the displayed list. Click and highlight Student Jamie . from the list.
  - Click the Grading button. A new grading window will appear on the screen. The answers to the problem set for that student are displayed along with rows of buttons. You can choose to view questions and answers or answers only.
  - You can change the recorded grade for any problem except those already answered correctly remotely by clicking on the corresponding radio button on the left hand side.
  - The buttons refer to whether the answer was remotely entered as correct **Y**-es: as incorrect **N**-o: or excused **E**-xcused. A dash indicates that the problem was never attempted. Lower case **y** or **n** are written in the *setx.db* file to indicate when a problem was graded by teaching staff using Grader.
  - Problems whose answers cannot be entered directly by students are hand-graded problems. Essay questions or derivations are examples. (As seen in the "*CAPA*" Functions" section, they are identified as such in the answer format of the problem as `hgr=on`.) To assign a grade, click on the box that is presented, assign the number of points received by the student in the panel that opens, and click to enter the grade.
  - `/SUBJECTIVE` questions (essay questions that were directly entered in by students) are graded seperately by selecting File in the Grader main menu and then selecting Grade Subjective. *This is explained in greater detail later.*

- Save results: After changing a student's grade, click on the Save button (upper part of the grading window) to record the result. Confirmation of any grade change is required in a separate panel.
- Grade `/SUBJECTIVE()` questions:
  - You can only grade `/SUBJECTIVE()` past the due date.
  - As you get more comfortable with the Quizzer, Grader, and CapaLogin components, you should complete the steps below.
  - Problem number 22 is a `/SUBJECTIVE` question in *set1.qz*. To view this problem, you should first enter in some essay answers for some of the students in the class list. You should have at least two students in a team with only one submission. Make sure the student not submitting the answer has her or his student number entered in the answer text. Next, change the due date (change the DB header with Quizzer).
  - To grade `/SUBJECTIVE()`, click on File in the Grader main menu and then select Grade Subjective.
  - You will be asked to select your class by choosing the *capa.config* file from your class directory.
  - After selecting the correct file, you will be asked for the set number and then the problem number.
  - Two windows should appear. One window is titled "gradesubjective" and displays the essay and the name of the student(s) who submitted the answer. The other window is titled "scoreandcom" and has options for grading the students.
  - The buttons on the "scoreandcom" window do the following functions:
    (a) The buttons next to the numbers at the top are the points you can assign for the problem.
    (b) Grade: By clicking this button, the points selected will be assigned to the student(s) and the box next to "graded" will be checked.
    (c) Grade&Next: Does the same as Grade except that it also displays the next answer to be graded.
    (d) Next: Displays the next answer in order of ID number.
    (e) Prev: Diplays the previous answer in order of ID number.
    (f) Find ID: This is not commonly used. If one of the student numbers are typed wrong in the essay text, then that can be corrected and the correct student can be found that matches the new student ID by clicking on the Find ID button.
    (g) Find Name: Similar to Find ID except that the name is searced for. Because it searches through the entire text, it may generate a long list. You should highlight the name in the essay text before using the Find Name button.
    (h) Add ID: Allows you to enter another student to the list in case a particular student was not listed in the essay text. You can select the student by either name or ID.
    (i) GoTo: Allows you to go to the answer from a particular student number, name, or an ungraded student.
    (j) Exit: Exits both windows.
    (k) wrap: Wraps long lines to make questions more readable.
    (l) pict: Displays the picture if the appropriate picture files are available.

(m) Print Response: Prints the essay or derivation text.

(n) At the bottom of the "scoreandcom" window is the number of questions graded and the number that need to be graded.

- Generate reports:
  - To generate reports, select File from the main menu, then Create Class Summary and follow the instructions.
  - Two types of reports can be generated by the Grader program, a class report and a section report. Both types are saved in a text file with an *.rpt* extension in the course sub-directory, *nsc121s9*. These files will have a prefix ClassSet$x$ or Sec$x$Set$x$.
- Select Quit on the Grader menu.

4. **Guided Tour of Manager**

- Prior to opening the Manager application, set up your X-server software to interact with your server. See Section 3 on using *CAPA* via eXceed, X-win32 or MacX.
- Through your X-server software, open an rlogin (or telnet) session to your server.
- If necessary, change directory to the directory in which the Manager application is located.
- Start the application by entering the command Manager.
- The Manager main menu will appear in the upper left corner of your screen.
- Choose Actions. You will be given three choices:
  (a) Scoring: This is used to score Scantron *CAPA* exams. [9]
  (b) Generate Stats: This gives different ways to generate statistics on scores, answer submissions, etc. There are also options to look up *CAPA* IDs. This choice is the same to the capautils application.
  (c) Randomize Seating File: This will randomize a user created seat file for exams.
- Choose Generate Stats
- A pop-up window appears with "Select the capa.config in the class directory" in the title bar. Double click on the appropriate *capa.config* file.
- Another pop-up window appears which allows you to choose the following CapaUtils functions:
  - Change Class Path allows you to switch back and forth between classes.
  - Run Capastat gives on-line feedback to instructors on the level of difficulty students have with an on-line assignment.
  - Summarize Log Files gives information when students entered in correct answers (Y), wrong answers (N), errors in significant digits digits (S), incorrect units (U), and no units entered or units entered when units were unnecessary (u).
  - Student Course Profile searches the various *CAPA* records and gives a summary of a particular student's performance in all areas involving *CAPA* such as homework, quizzes, examinations, or supplementary problems.
  - CAPA IDs for one student finds the *CAPA* ID for a particular student for one or more sets.
  - All CAPA IDs generates all the *CAPA* IDs for a class for one or more sets.
  - Item Analysis gives the level of difficulty and discrimination for each of the questions.
  - Item Correlation gives the correlation between problems in a set. (If the display is hard to read, uncheck the wrap button.)
  - View Submissions gives all the answers a student submits for a particular set.
  - Analyze Class Report gives grade distributions and a graph of grade distributions for a previously generated class report.
  - Analyze Responces analyzes output generated by scorer.

[9] For more information about scantron *CAPA* tests and Scorer, please contact capa@capa.msu.edu.

  - Graph a Responces Analysis

- Close the CapaUtils window by selecting Quit.
- Another feature on the Manager window is Print. This print option is for printing assignment(s) for a specified student.
- Quit Manager.

5. **Guided Tour of Qzparse**

- Qzparse is a feature that gives the instructor flexible printing options. **Note:** All of these printing options can be done using Quizzer, while Manager allows you to print for one student by searcing for either a name or a student number. Open an rlogin (or telnet) session as "teacher" and follow the script below. The computer used in this example has a prompt **capa2.nscl.msu.edu>**
- Before you begin, you must be in the appropriate class directory.
  capa2.nscl.msu.edu> `cd /absolute/path/teacher/CAPA5/nsc121s9`
- Entering in `qzparse -h` at the prompt will display all of the options available to this function.
  capa2.nscl.msu.edu> `qzparse -h`

```
USAGE: qzparse [ -[T|H|A][a|b] ] [-Sec [n|n:m] | -Stu sn [-o filename] ]
          [ -Set [n|n:m] ] [-c path_to_class] [-d outputdirectory]
 Example 1: qzparse -Tb -sec 2:3 -set 2:5
            will generate tex files with both questions and answers
            for sections 2 to 3, sets 2 to 5
 Example 2: qzparse -Ha -stu A12345678 -set 3
            will generate html files with answer only
            for student A12345678 set 3
          -T     = tex   mode
          -H     = html  mode
          -A     = ascii mode
                 = default question only
            a    = answer   only
            b    = both question and answer
          -Sec 3 = for section 3
          -Sec 3:7    = from section 3 to section 7
          -Stu A12345678 = for a specified student
          -Set 1 = output set 1
          -Set 3:4 = output from set 3 to set 4
          -c class_path
          -o output_filename_with_absolute_path (only for a student)
          -d directory_to_create_files_in (default is class_path/TeX)
-------This is version 5.0.3 @ 11:23-Apr-07-1999
------------------------------------------------------
```

- **capa2.nscl.msu.edu>** `qzparse -T -sec 1:2 -set 1`
  qzparse running in TeX mode, question only, from section 1 to 2, set 1
- **capa2.nscl.msu.edu>** `Enter the ABSOLUTE path of class`
  `/usr/users/teacher/CAPA5/nsc121s9`

```
Section  1: 4 students
    Student: Albertelli, Guy II           A12345678   set 1.....................
    Student: Berryman, Felicia V.         A23592320   set 1.....................
```

```
  Student: Kashy, Edwin N.              A73336318   set 1.....................
  Student: Student, Jamie .             A87654321   set 1.....................
DONE set 1

DONE section 1    ALL DONE
```

The above message shows that the Qzparse application has generated the appropriate *.tex* files from the quiz set files. The *.tex* files will contain questions for *.qz* set 1 for section 1 and section 2.

- **capa2.nscl.msu.edu>** `cd TeX`
  capa2.nscl.msu.edu> `latex section1-set1`

  ```
  .............
  Output written on section1-set1.dvi (8 pages, 45412 bytes).
  Transcript written on section1-set1.log.
  ```

  The above message shows that the *.dvi* file has been created from the *.tex* file for section 1.

- **capa2.nscl.msu.edu>** `dvips section1-set1.dvi -o section1-set1.ps`
  ....
  This command generates the .ps file which is ready for printing.

- **capa2.nscl.msu.edu>** `ls`

  ```
  section1-set1.aux  section1-set1.log  section1-set1.tex
  section1-set1.dvi  section1-set1.ps
  ```

  The `ls` command lists the files in the current working directory. As seen above, there are five files created. The *.ps* file contains problem sets for all students in section 1. The command to print is written below. To specify two sided printing or any other print option, you need to find the command for your particular printer.

- **capa2.nscl.msu.edu>** `lpr -PLocal_Printer section1-set1.ps`
- Note: Do not type a space between `-P` and the name of your local printer.

Note: It is suggested that the *TeX* classname subdirectory be cleaned at regular intervals as *.ps* file can get very large.

6. **Guided Tour of Capautils 1.0**

   - Through your X-server software, open a telnet session to your server.
   - Start the application by entering the command `capautils.pl`.
   - Enter the absolute path requested, for example:
     `/usr/users/teacher/CAPA5/nsc121s9`
   - The terminal will display a menu with several items. These items are similar to the items from capautils 1.1 which have been explained in the guided tour of Manager.
   - All items are menu driven and some of the information which this application uses (such as printing, classnames, etc.) is in the *capa.config* file.

7. **Guided Tour of AllCapaID**

   - Besides getting *CAPA* IDs from CapaUtils (either from a terminal or a window in Manager), you can also retrieve *CAPA* IDs from the AllCapaID application.

- Typing `allcapaid -h` at the prompt of a telnet session will show all the options available for this function.
  capa2.nscl.msu.edu> `allcapaid -h`

```
USAGE: allcapaid [-s start-set] [-e end-set] [-stu student-number] [-c class-directory] [-d output
       start-set : default 1
       end-set   : default 10
       student-number  : no default
       class-directory : no default
       output-directory: class-directory/capaID
       -Sec 3          : for section 3
       -Sec 3:7        : from section 3 to section 7
       -i              : don't create files, print to the screen
       -h              : prints this message
       CAPA version 5.0.3, 11:23-Apr-07-1999
```

- An example with "Jaime Student" is shown below:

capa2.nscl.msu.edu> `allcapaid -s 1 -e 1 -stu A87654321`

```
Enter the ABSOLUTE path of class from root (/)
/usr/users/teacher/CAPA5/nsc121s9

   1
2318
```

9 Sample Login Instructions

An example of login instructions for students is given on the following page.

It is also useful to have a handout for students listing the Base Units, Prefixes, and Derived units used in a class. These are in the *capa.config* file which resides in each classname directory.

College of Natural Science M. S. U.
nsc121s9

Instructions for Using *CAPA*[1]

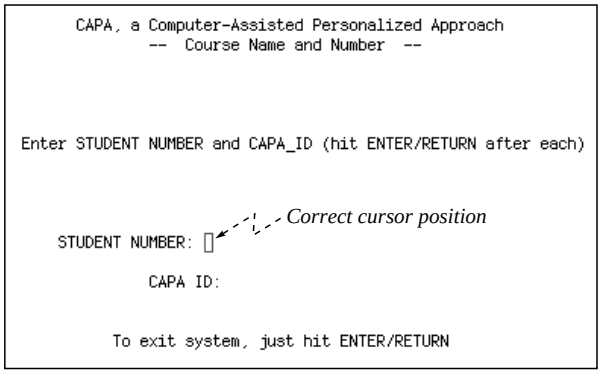
CAPA implements a **C**omputer-**A**ssisted **P**ersonalized **A**pproach for assignments, quizzes and examination. Using *CAPA* in this class is **OPTIONAL**. Your **Name** and a specific **CAPA ID** is at the top on each printed worksheet. Discussion of concepts and methods of solving problems with fellow students is strongly encouraged.

I. Access via telnet.

1. Starting a telnet session:

   C. From wherever you do a **telnet**, enter
   telnet capa2.nscl.msu.edu
   You should next see the prompt **login:**

2. At the '**login:**' prompt, enter **nsc121s9** (note the lower case letters in nsc121s9). Your terminal screen should then look like:



   The cursor on your screen must be in the position shown above. If it is not, try resetting your terminal emulator to **vt100**.

3. Enter your **Student Number**, then the **CAPA ID** (both are printed at the top of the sample set for this demonstrations). Then follow the instructions.

II. Access via www browser.

1. Point your browser to:
   http://capa2.nscl.msu.edu/class.html
2. Select **nsc121s9** from the **Class** pop-up menu.
3. Enter your **Student Number**, then the **CAPA ID** (both are printed at the top of the sample set for this demonstrations).
4. Click on the appropriate button to try your set.

Notes:

1. You may repeat the problems you missed and get full credit. Your instructor sets the number of tries allowed. After an 'incorrect' response, a HINT may be available. It is viewed by selecting :H in telnet, automatically on www. You may login/logout many times. Once a problem is **correct**, credit cannot be lost.
2. Do **not** open multiple sessions or browsers. Be sure to eXit properly each time to terminate a session.
3. If needed when using the www, you can use the **Reload** button at the top of your assignment.
4. Clicking the browser's "**Reload/Refresh**" button right after submitting an incorrect answer re-submits that incorrect answer and uses up a Try.

10 Technical Support, Acknowledgments

The *CAPA* homepage URL is **http://www.pa.msu.edu/educ/CAPA/** It outlines hardware specifications for *CAPA* licensing information, and has the sample set displayed.

Technical Support

Felicia Berryman
Telephone:
(517)333-6396
E-mail:
capa@capa.msu.edu
Other e-mail resources are:
albertelli@nscl.msu.edu
kashy@nscl.msu.edu
morrissey@nscl.msu.edu

Acknowledgments

Many instructors, graduate students, and undergraduate students at MSU and other institutions have participated in the *CAPA* project. Their work and suggestions have helped to make *CAPA* a better tool.

The support for *CAPA* development or operation from the following is gratefully acknowledged: MSU College of Natural Science, MSU Office for Computing and Technology, MSU Department of Physics and Astronomy, MSU Department of Chemistry, MSU Provost, The National Superconducting Cyclotron Laboratory(NSCL) and also from the Alfred P. Sloan foundation and the US Department of Agriculture.

The advice of R. Fox, K. Berhooz, and B. Pollack (of the NSCL) and G. Perkins (of the Department of Physics and Astronomy) has been of considerable help in computer and network operation.

Thanks to M. Albertelli for checking some sections of this manual.

11 Publications

The *CAPA* homepage URL is **http://www.pa.msu.edu/educ/CAPA/** It outlines hardware specifications for *CAPA*, licensing information, and has the sample set displayed.

CAPA Publications

- *CAPA*, An Integrated Computer Assisted Personalized Assignment System Kashy, E., Sherrill, B.M., Tsai, Y., Thaler, D., Weinshank, D., Engelmann, M., Morrissey, D.J., , *American Journal of Physics*, Vol. **61**, No. 12, (1993), pp. 1124-1130.
- Conceptual Question in Computer-Assisted Assignments Kashy, E., Gaff, S.J., Pawley, N.H., Stretch, W.L., Wolfe, S.L., Morrissey, D.J., Tsai, Y., *American Journal of Physics*, Vol. **63**, No. 11, (1995), pp. 1000-1005.
- Using Computer-Assisted Personalized Assignments in Freshman Chemistry, D.J. Morrissey, E. Kashy, and Y. Tsai, *J. of Chem. Edu.***72**,(2)1995, 141-146.
- Computer-Assisted Assignments in a Large Physics Class Thoennessen, M., Harrison, M. , *Computers and Education*, Vol. **27**, No.2, (1996), pp. 141-147.
- Individualized, Computer-assisted, Personalized Assignments, E. Kashy and D.J. Morrissey, in **The Hidden Curriculum**, by Sheila Tobias and Jacqueline Raphael Plenum Press 1997, 165-167
- Using Networked Tools to Promote Student Success in Large Classes, Kashy, E., Thoennessen, M., Tsai, Y., Davis, N.E., Wolfe, S.L., *Journal of Engineering Education* 1998.

A few other papers

- Abbott, H., "Physics Homework Management: Using HyperCard to Generate Individualized Problem Sets", Computers in Physics, Vol. 8, No. 2, (1991) pp. 166-169.
- Borg, W.R., Gall, M.D., "Critical Evaluation of Research, in Educational Research", David Mckay Company (1974), p. 106.
- Brown, D.J., "Mallard, Asynchronous Learning on the Web", http://www.cen.uiuc.edu/Mallard/
- Chiu, C.B., Moore, C.F., "Centralized Computer System for Engineering Physics", Manual, U of Texas at Austin.
- Hestenes, D., "Toward a Modeling Theory of Physics Instruction", American Journal of Physics, Vol. 55, No. 5, (1986), pp. 440-454.
- Hubler, A.W., Assad, A.M., "CyberProf, An Intelligent Human-Computer Interface for Asynchronous Wide Area Training and Teaching", Fourth International World Wide Web Conference Proceedings, WWW Journal Dec 11-14, (1995), pp. 231-238.

- Ilgen, D.R., Fisher, C.D., Taylor, M.S., "Consequences of Individual Feedback on Behavior in Organizations", Journal of Applied Psychology, Vol. 64, No. 4, (1979), pp. 349-371.
- Lewis, R.A., Harper, B.M., Wilson, M., "Computer Assignments and Problems Classes for Physics Students", Computers Educ., Vol. 16, No. 4, (1991), pp. 349-362.
- Mellema, S. Niederriter, C.F., Thompson, H.B., "A Com puter-based Homework System with Individual Problem Solving and Instructor Diagnostics", Bull. Am. Phys. Soc., Vol. 38, (1993), p. 1004.
- Resnick, L.B. "Education and Learning to Think", Washington, DC: National Academy of Sciences, 1987.
- Romer, D., "Do Students Go to Class? Should They?", Journal of Economic Perspectives, Vol. 7, No. 3, (1993), pp. 167-174.
- Sansone, C., "A Question of Competence: The Effect of Competence and Task Feedback on Intrinsic Interest", J. of Personality and Social Psychology, Vol. 51, No. 5, (1986),pp. 918-931.
- Sassenrath, J.M., Garevich, C.M. "Effect of Differential Feedback from Examinations on Retention and Transfer", reprinted from J. Edu. Psychology 1965, in Current Research on Instruction, R.C. Anderson et al., Prentice Hall (1969), pp. 211-215.
- Sokoloff, D.R., "Enhancing Physics Learning in Lecture with Interactive, Microcomputer-Based Demonstrations", Bull. Am. Phys. Soc., Vol. 40, No. 2, (1995), p. 917.
- Treisman, U., "Studying Students Studying Calculus", The College Mathematics Journal, Vol. 23, (1992) p. 362.
- VanHeuvelen, A., "Overview, Case Study Physics", American Journal of Physics, Vol. 59, No. 10, (1991), pp. 898-907.