THE GENERAL PROCTOR PROGRAM

AN INSTRUCTOR-ORIENTED INFORMATION STORAGE, RETRIEVAL AND PROCESSING SYSTEM

R. A. AVNER

Computer-based Education Research Laboratory

University of Illinois Urbana Illinois
THE GENERAL PROCTOR PROGRAM

An Instructor-oriented Information Storage, Retrieval, and Processing System

R. A. Avner
CONTENTS

Basic Features.................................................. 1
Requirements for use of the Program..................... 2
Alteration of Data Storage.................................. 4
To Start New Students........................................ 6
Operating Procedures........................................ 7
Adaptation of the Prototype to your Course............ 9
Storage Planning Sheet......................................... 13
GENERAL PROCTOR PROGRAM

A prototype PROCTOR Routine now exists. This material can be copied into a new lesson space and modified for use in a particular course with minimal time and effort. Basic features should be available with less than 30 minutes work. Formatting of special displays and information storage would take more time.

Basic Features of the Program

(1) Position of all operating students in the course material can be monitored from a single station during class. The prototype displays student name, station number, identification of course position, and cumulative time spent on the current segment. This information may be modified and/or additional information may be presented subject to storage and display space limitations. Suggested modifications include listing of stations at which students have requested HELP, exceeded a predetermined time limit or number of errors, etc.

(2) Cumulative data are kept for all individuals in the course. These data may be transferred to magnetic tape for printing at any time. The prototype PROCTOR keeps count of total sessions (up to 255) on PLATO, total time (up to 327.67 hours) on PLATO, and current position (up to 262,143 distinct areas) in the course. Twenty-four "flags" are available for "yes-no" type data and 111 bits of space are available for packed cumulative information of special interest to a given course instructor. Suggested additional data might include cumulative test scores, aptitude test scores, counts of numbers of course segments completed, amount of remedial material used, etc.

(3) Cumulative data are kept on each student for certain course segments specified and identified by the instructor. The prototype PROCTOR keeps cumulative time (up to 409.5 minutes) spent on the course segment. Thirty-six bits are available for reporting information such as total questions completed, total errors made, total correct first responses, types of deficiencies encountered, etc. This information may be transferred to magnetic tape at any time for printing or may be interrogated at a PLATO terminal by routines which process the data automatically and present results in graphic and tabular forms. If computer space is available during the class, interrogation of student and lesson data may be done while
being collected.

(4) Minimal modification of current lesson material is required. The PROCTOR lesson is read in independently of the course lessons. Thus, possible conflicts in duplicate unit names, use or non-use of COMMON, etc., are avoided. Amount of space required may be varied by the instructor within certain limits.

Requirements for use of the Program

(1) One student variable must be available for use by the PROCTOR lesson during the entire course. The variable used is specified by the instructor. The limitations are:

(a) the same variable must be used in all lessons and for all students in the course.

(b) this variable must not be overwritten or altered during the course except by PROCTOR routines.

(2) The first Unit of each lesson must contain a single-line DEFN statement. A copy of the statement appears in the prototype PROCTOR and may be copied by use of the "SAVE" feature of AUTHOR MODE.

(3) Each course segment must be specified by the author within each lesson. This is done by inserting the statement

```
CALC AREA(N)
```

where it will be encountered by the student upon beginning the course segment. In actual use, the letter "N" is replaced by a number or TUTOR integer variable having a value in the range 1 to 262,143. The number may have any value in this range. The values need not be in any particular order nor need every value be used. Examples of possible course segment coding schemes include

(a) Linear numbering, e.g. a sequence of 500 "tasks" is numbered from 1 to 500.

(b) Combined lesson and subtask coding, e.g. section 5 of the 14th lesson in a course might have the code "1405".

Course segment codes need not be restricted to specific sections of lesson material. Since a TUTOR variable may be used in the AREA function, a lesson segment could be given a different identification code on each of several uses of the material by the student. Segment codes can even be defined on
the basis of time or prior performance, e.g., certain material might be
given one code if it were encountered before the second session and another
if encountered after specified remedial material had been received.

(4) When data are to be kept, the AREA function should be preceded
by "IJOIN ZSTART." At completion of the course segment the lesson must
contain "IJOIN ZSTORE" to store accumulated data. If no data are to be
kept, neither IJOIN is necessary (though they will do no harm). Also if
the prior course segment was terminated by "IJOIN ZSTORE" it is not
necessary to use IJOIN ZSTART." In general an AREA function and an "IJOIN
ZSTORE" should not appear in the same Unit. Note that "IJOIN" is illegal
if it appears after an "ARROW" command.

Units ZSTART and ZSTORE require a total of 16 words of space and must
be present in at least one of the course lessons read into the computer
together at each class session. Copies of these Units appear in BLOCK
ZSTART of the prototype PROCTOR lesson and may be copied into course lessons
by use of the BLOCK Copy option of AUTHOR MODE.

A general rule for delimiting course segments (which will work even if
data are not to be kept for a given segment) is to place an "IJOIN ZSTORE"
in the last Unit of each course segment and a "CALC AREA(N)" in the first
Unit of each course segment. If a course segment has only one Unit, the
instructor should seek special help from the author of this note.

(5) Data for individual course segments are kept only if space is
reserved for such storage. Data are stored in COMMON storage kept in the
lesson space of the PROCTOR program. The amount of COMMON space required is
a function of the total number of students in the course (a maximum of 63
students per course is allowed in the prototype) and the total number of
course segments for which data are to be stored at one time.

\[
\text{Common Space (in words)} = 10 + 5 \times SS + ST + SS \times ST
\]
where SS = total number of students in the course
ST = total number of course segments for which
data are kept

In addition to Common storage the Prototype PROCTOR lesson requires about
700 words of space to perform monitoring and data keeping functions. If
monitoring is not required, this requirement may be halved. Modifications
for specific applications might cause the space requirement to decrease by
a small amount (50 words or so) or increase by several hundred words.
Space requirements for functions not ordinarily necessary during class operation (such as interrogating data) are on the order of 3000 words. Partial READIN features permit use of only essential portions of the program if space requirements for course material are critical.

Bookkeeping features of the prototype allow reallocation of storage spaces for course segment data. Thus, if data are to be kept for 100 course segments but it is known that no more than 20 segments will be in use by students at any one time, it would be possible to set aside space for 20 rather than 100 segments. This situation would normally occur where students proceed through a course in a roughly linear fashion. As all students complete a given segment the cumulative data for that segment can be printed out in permanent form and the storage space reallocated for storage of data from a different segment.

(6) Students terminate each session by using -TERM- "EXIT" rather than -TERM- "FINISHED." If the class is monitored, a fail-safe function insures that data will be properly updated even if students halt in some other manner.

Alteration of Data Storage

The prototype PROCTOR contains several features designed to simplify modification of data storage if such modification should be necessary. Situations requiring modification of COMMON storage include

(a) initializing the COMMON for a new class
(b) deleting records for an individual student
(c) assigning data storage space to a given course segment
(d) examining and/or altering information in portions of storage space not available in standard displays

All of the above operations are made available by reading the prototype PROCTOR into the computer by partial READIN using part BOOKING. No lesson title is provided for such a readin (as a security measure). Hence access to the lesson must be by means of student records set to Unit "BK1". The program permits the user to proceed to a Unit which allows modifications of the types listed above. The displays are self-explanatory and operations are heavily protected against recognizable errors. Checks are automatically made to insure that, for example, the same course segment is not assigned
several different storage spaces even if a user might try to make such an incorrect assignment.

Special notes for use of the first 3 options include

(a) Initializing storage - this option **must** be selected before every new class (including the very first one) is started. Note that the user is required to follow several sets of instructions before the operation is complete and the message "COMMON IS READY FOR A NEW CLASS" is displayed. Follow the directions carefully. Remember, once this operation is completed all data stored from prior class sessions will be destroyed. The complicated procedure required is solely to insure that the operation is not performed unintentionally.

(b) Deleting records for a single student - this option deletes all data stored for the specified student. It also makes the space used available for reassignment to a new student (use the procedure described in the section entitled "To Start New Students").

(c) Assignment of Data Storage - use course-segment identification code numbers selected according to the scheme you have decided on for your course. Unless data storage is assigned, data for a course segment will not be kept. A list of currently assigned spaces may be seen by pressing key LAB.

Contents of individual data storage locations may be examined in a Unit accessed by pressing the DATA key. This Unit will generally be used only by someone who knows the exact form in which the data are stored. Upon typing a Common variable number (e.g. "26" if Common variable number 26 is desired) and pressing NEXT, the user will be shown the desired Common variable and the 3 following variables in octal, integer and alpha formats. Addresses of names of specified students, cumulative storage for specified storage areas, and storage for given students on given task areas may be directly accessed by use of a set of functions listed on the display for this Unit. A second ARROW in the Unit accesses a calculator mode which allows conversion of packed octal information to more convenient forms or alteration of Common or Student variables.
To Start New Students

1. Initialize data storage (see section "Alteration of Data Storage")
2. Set up Student Records as you normally do for a class. Then, if all students are to begin in the same Unit of a lesson, temporarily insert the command "JOIN ZSET" as the first statement in this Unit. Following Units may be temporarily "STOPed" out.
3. Read your lesson or lessons in along with the General Proctor lesson using PARTS "SET" for the latter.
4. Read the General Proctor lesson in separately using PARTS "BOOKING".
5. Sign in using each student's records in turn.
6. If "JOIN ZSET" is used, you will immediately go to a SETUP display. Otherwise (if students are scattered through a lesson, for example) go to the SETUP display by pressing TERM, typing "ZST," and pressing NEXT.
7. The SETUP display will automatically assign an unused student number and display the first 8 letters of the student's name. Numbers are assigned in reverse order (highest first) so if alphabetized printed copies of records from the General Proctor are desired you must enter student records in reverse alphabetic order. Thus you would sign in as student "Walker" before signing in as student "Adams."
8. If you wish to alter the name used by the Proctor program for a student or to enter certain desired data (e.g. grade point averages, sex, or pretest scores) type the additions or changes as specified by the display. Otherwise press NEXT alone to be sent back to WELCOME TO PLATO for the next student.
9. After all students have been signed in once, verify that all entries were correctly made by signing in under each name again. The display should indicate that each student is an "old student" and reflect changes or additions made by you during the original sign in.
10. Delete the lessons from memory and remove the "JOIN ZSET" and any STOP statements temporarily inserted in your lesson(s).
Operating Procedures

1. To operate a class:
   A. Read in the Proctor lesson by itself using PARTS "CLASS" if you wish to monitor students as they work or PARTS "BLIND" if monitoring is not to be done. The basic "CLASS" section requires about 600 words plus COMMON space. The basic "BLIND" section requires about 300 words plus COMMON space. Warning: "BLIND" does not have the fail-safe data updating features of "CLASS." Time and session data will be invalid if students fail to encounter an "IJOIN ZSTORE" statement during the session or fail to sign out using -TERM- "EXIT".
   B. Read in all other course lessons.
   C. Upon completion of the class, make sure that all students are signed out. Then delete lessons. The Proctor lesson should always be the first to be read in and the last to be deleted. If other courses operating at the same time as yours also use a Proctor lesson, contact the author of this memo to have TERM and PRESS statements altered in one or both of the Proctor lessons. Conflicts of this nature will otherwise produce error messages on the screens of students in one of the courses (more specifically, for students in the course that read its Proctor lesson in second).

2. To alter or initiate storage
   A. Read in the Proctor lesson using PARTS "BOOKING". The basic BOOKING section requires about 750 words plus COMMON space.
   B. Sign in under student records sending you to Unit "BK1." Do not alter the position of the TITLE statement in Unit ZMON to allow other methods of entry since BOOKING allows total destruction of your COMMON as well as the other feature described in the section "Alteration of Data Storage."

3. To interrogate data
   A. Read in the Proctor lesson using PARTS "DATA". The basic DATA section requires about 1350 words plus COMMON space. If data processing is desired during class operation, both sections may be read in together. Provision for moving between the monitoring and data sections should be inserted in Units "ZMON" and "ZDATA" if such
dual operation is anticipated (DATA statements in each would provide such a capability).

B. Options are largely self-explanatory. "Student Data" generally refers to data specific to an individual student while "Lesson Data" refers to data for all students. Student data may be requested either by typing the student's name or his code number. Both are cross-listed in the "CLASS ROLL" option in "STUDENT DATA."

Lesson data may be displayed as a histogram (single variable) or a scatter-plot (two variables at a time). When a variable involves data from a specific course segment, the code letter for the type of variable and the code number for the course segment must both be typed (in either order). Requests for nonexistent data will be erased and ignored. The "R" indicated on scatter-plot displays is the Pearson product-moment correlation coefficient. A value of zero indicates that no linear relationship between the two sets of data is present while a positive or negative value of 1 indicates that a perfect linear relationship exists between the two.

Data print-outs require that you be signed in to the Proctor lesson under a real set of student records (i.e. not simply as "STUDENT") and that a data tape be mounted and ready to receive the data.
Adaptation of the Prototype to your Course

1. Determine what information is to be kept and decide where it is to be stored. This step is most easily done by arranging for a conference with a member of the CERL Evaluation and Measurement staff. You should decide also on a plan for identifying course segments, the number of students in the course and the maximum number of course segments for which data are to be kept at any one time.

2. Request a new lesson space containing lesson WATCH (the prototype Proctor lesson).

3. Modify the prototype. Modification may be done in two steps. The first step consists of modifications required to provide the basic features. The second step consists of modifications required to provide special features for your own course.

(a) Basic Modifications - The prototype uses a COMMON named "SAVE." BLOCK "SAVE" should be deleted and enough new BLOCKs started to provide space for your requirements. You will need

\[(10 + 5 \times SS + ST + SS \times ST)\] words of storage.

Each BLOCK permits up to 75 words of DATA. Remember that the first 5 letters of COMMON BLOCK names must all be the same.

Example: Suppose that your class will consist of 20 students and data for no more than 16 course segments need be stored at any one time.

\[SS = 20\]
\[ST = 16\]

Then \[10 + 5 \times SS + ST + SS \times ST = 10 + 5 \times 20 + 16 + 16 \times 20 = 446\] words.

The smallest number of COMMON BLOCKs which will hold 446 words is 6 \((6 \times 75 = 450\) words). For this example we might then start 6 BLOCKs named STORE1, STORE2, STORE3, STORE4, STORE5, and STORE6 immediately after BLOCK ZSTART.

The only remaining step in the basic modification is to alter the lesson title and 5 statements in BLOCK "ZMON." The 5 statements begin with "CON WATCH, SAVE, 37"

(1) Replace the COM statement with one listing your own lesson name, the first 5 letters of your COMMON BLOCK names, and the number of words in your COMMON.
Replace "DEFN SS=3" with DEFN SS=(your class size)

Replace "DEFN ST=3" with DEFN ST=(number of segments to be stored at any one time)

Replace "DEFN SC=01234 4321 0000 0000" with a code of your own choosing. This code is used to insure that only students in your course will be monitored and allowed to store data. The last 8 digits must be 0. You should avoid zero (00) and blank (20) codes since these occur frequently in standard records.

Replace "DEFN LC=30" with DEFN LC=(the number of the student variable which you have chosen for use by the PROCTOR lesson). This variable will contain the secret code specified above and the current student lesson location at all times. Caution: the number must be an integer constant. Do not use a variable name.

Note that the next DEFN statement ("DEFN AREA(X)=...") is the one referred to earlier that must be copied into each of your lessons. At this point your lessons need only be modified by insertion of "CALC AREA(n)," "IJOIN ZSTART," and "IJOIN ZSTORE" statements to delimit course segments, copying Units ZSTART and ZSTORE into your lessons, and copying the "DEFN AREA(X)=..." statement into the first Unit of each of your lessons. After initializing your new COMMON by use of the BOOKING feature you will have a functioning PROCTOR system.

(b) Special Modifications - Special modifications are concerned with specifications, storage, and processing of data of unique importance to your course. These modifications may be done in consultation with a member of the CERL Evaluation and Measurement staff or directly by instructors who have experience with octal CALC operations and the MVBIT and MOVE commands. Use of the planning forms enclosed in this report will simplify much of the special modification bookkeeping. The list of Unit names given here is intended as a checklist and guide to "what is done where."

ZMON and ZMODA - Monitor display units

ZIP-UP (in BLOCK "ZOFF") - insert commands for storing special data here. Variables with
useful information include IC3= student number
IC4= course segment
IC6= address of cumulative storage for the student
IC9= address of special storage for the student
when positive (if negative, no space was set aside for the current segment).

ZSETUP - insert provision for storing special historical data here (used during initialization of new student records)

ZLES - display unit for on-line processing of stored data. Variables are identified by letters on this display. A MATCH command in Unit ZLES1 should be altered as new variables are added. Unit ZL must be referenced in Unit ZLES1 (by a GOTO statement) for every variable which involves specific course segments.

ZLES3 (in BLOCK ZLES1) - routes requests to the proper data-unpacking routine. The 1st and 2nd JOIN statements provide appropriate lists biased upward by one (variable "A" goes to branch 1, "B" to branch 2, etc.). Join Unit "ZLT" for variables that involve specific course segments and "ZLS" for variables that involve cumulative student data.

ZLPAK (in BLOCK ZLES1) - The JOIN in this Unit picks up the specific data unpacking routine for each variable. The data unpacking routines are contained in Units named "ZL0" through "ZL(n)" where "ZL0" has the routine for variable "A", "ZL1" for variable "B," etc. When a new variable is specified, add a new "ZL(n)" Unit name to the JOIN list in ZLPAK and write the new Unit. The new Unit should move information from the COMMON location addressed by I129 into variable I137.

ZLABEL (in BLOCK ZL1V) - contains a WRITC statement with
labels for the variables used.

ZSTUD1 - Display unit for cumulative data for an individual student. Unit ZSTUD4 (in BLOCK "ZSTUD2") contains the statements which retrieve the data. Variable IS has the address of the cumulative data.

ZSTU - Display unit for data from course segments currently stored (for a specified student). Unit ZPRNTSK (in BLOCK "ZPRIN-T") unpacks these data both for this display and for printed output.

ZPRIN-T - Contains format for printed output of data for specific course segments.

ZPRIN-S - Contains format for printed output of cumulative student data. Unit ZPRNS (in BLOCK "ZPRIN-S") accesses Unit ZSTUD4 and specifies the format for each line of data.
### STORAGE PLANNING SHEET - GENERAL PROCTOR

<table>
<thead>
<tr>
<th>Bits</th>
<th>Characters</th>
<th>Largest Integer</th>
<th>Bits</th>
<th>Characters</th>
<th>Largest Integer</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>13</td>
<td>13</td>
<td>8,191</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>14</td>
<td>14</td>
<td>16,383</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>7</td>
<td>15</td>
<td>15</td>
<td>32,767</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>15</td>
<td>16</td>
<td>16</td>
<td>65,535</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>31</td>
<td>17</td>
<td>17</td>
<td>131,071</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>63</td>
<td>18</td>
<td>262,143</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>127</td>
<td>19</td>
<td>19</td>
<td>524,287</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>255</td>
<td>20</td>
<td>20</td>
<td>1,048,575</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>511</td>
<td>21</td>
<td>21</td>
<td>2,097,151</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>1,023</td>
<td>22</td>
<td>22</td>
<td>4,194,303</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>2,047</td>
<td>23</td>
<td>23</td>
<td>8,388,607</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>2</td>
<td>4,095</td>
<td>24</td>
<td>16,777,215</td>
<td></td>
</tr>
</tbody>
</table>

Student Cumulative Storage - Beginning address is 7 + SS + 4 x (student number)

<table>
<thead>
<tr>
<th>character</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>bit</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Lesson Data -

<table>
<thead>
<tr>
<th>character</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>bit</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>