<u>Problem</u>

Provide a graphical user interface (GUI) with editing features for opscan, (multiple-choice), test results files produced by a National Computer Systems™ test scanner from Va Tech Measurement and Research Services.

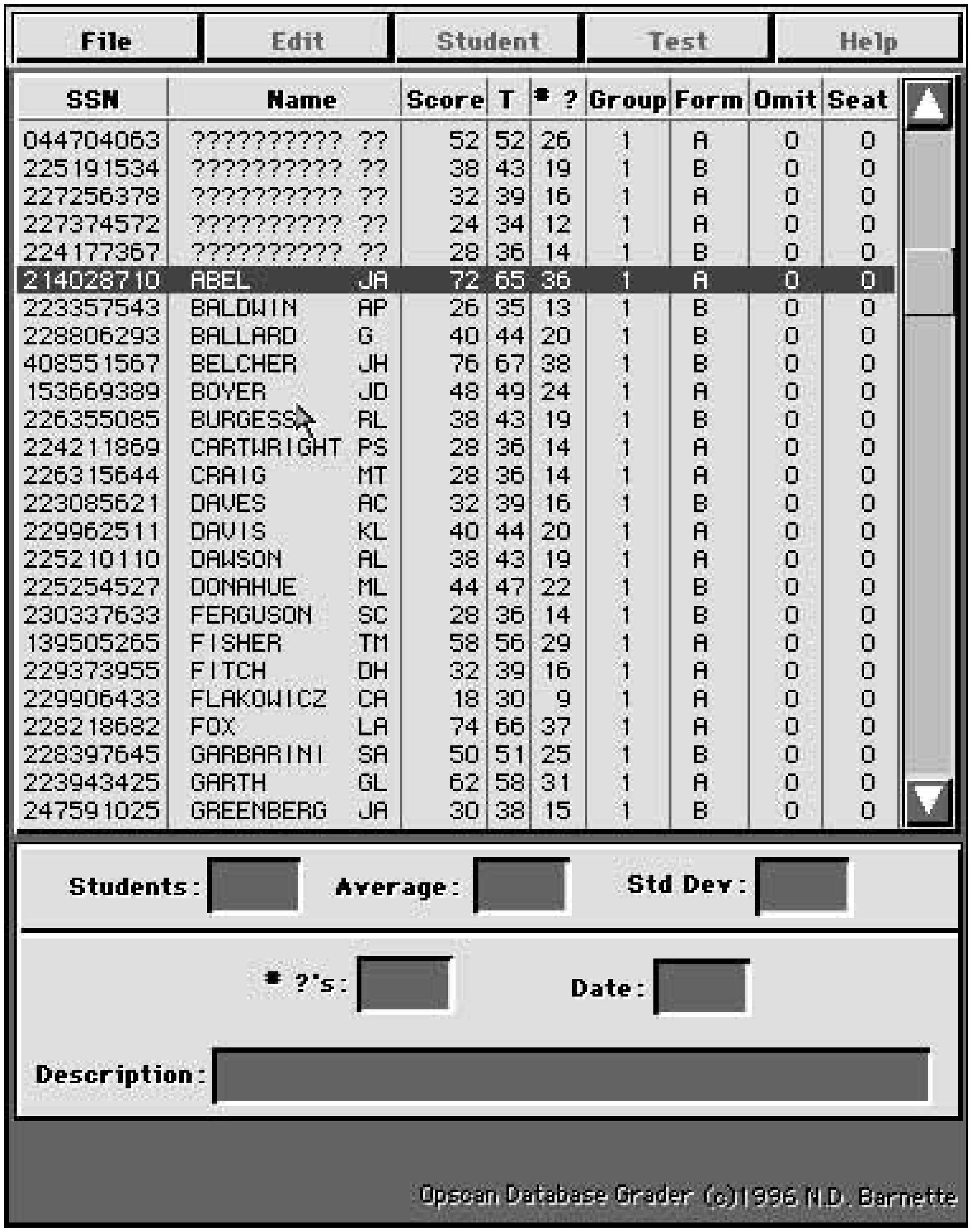
<u>Interface</u>

The GUI must resemble the interface depicted in Figure 1 almost identically. Minor variations, (colors, etc.) are permissible. Major changes to the interface must be approved. The programmer's name should appear at the lower right corner of the screen, following the copyright. All access to the programs operations, (except for the scrolling provided by a picklist), will be provided through the system menus as discussed below.

<u>Discussion</u>

In order to provide more efficient scrolling, insertion and deletion operations, the underlying physical data structure for this assignment must be changed to double linked-list, (see Figure 2). All of the standard linked-list operations, (insert, delete, etc.), must be implemented as an Abstract Data Type (ADT). Additional non-standard list operations will be required to support the GIL GUI routines and must be incorporated into the list ADT accordingly, adhering to the principles of information-hiding and encapsulation. C++ object-oriented code (classes) may NOT be used in this assignment. Separate compilation implemented via a make file is required.

This assignment will comprise approximately 20%-30% of the final term project. Additions and modifications to the interface should be anticipated and planned for accordingly. An addendum specifying minor changes and alterations to the interface and clarifying it's behavior for this assignment will be produced if required. Screen shots of the pull-down (popup) menus will be made available on the course web site.



SSN	Name	Score	T	* ?	Group	Form	Omit	Seat
044704063	?????????? ??	52	52	26	1	A	0	0
225191534	?????????? ??	38	43	19	1	B	0	0
227256378	?????????? ??	32	39	16	1	A	0	0
227374572	?????????? ??	24	34	12	1	A	0	0
224177367	?????????? ??	28	36	14	1	B	0	0
214028710	ABEL JA	72	65	36	1	A	0	0
223357543	BALDWIN AP	26	35	13	1	B	0	0
228806293	BALLARD G	40	44	20	1	B	0	0
408551567	BELCHER JH	76	67	38	1	B	0	0
153669389	BOYER JD	48	49	24	1	A	0	0
226355085	BURGESS RL	38	43	19	1	B	0	0
224211869	CARTWRIGHT PS	28	36	14	1	A	0	0
226315644	CRAIG MT	28	36	14	1	A	0	0
223085621	DAVES AC	32	39	16	1	B	0	0
229962511	DAVIS KL	40	44	20	1	A	0	0
225210110	DAWSON AL	38	43	19	1	A	0	0
225254527	DONAHUE ML	44	47	22	1	B	0	0
230337633	FERGUSON SC	28	36	14	1	B	0	0
139505265	FISHER TM	58	56	29	1	A	0	0
229373955	FITCH DH	32	39	16	1	A	0	0
229906433	FLAKOWICZ CA	18	30	9	1	A	0	0
228218682	FOX LA	74	66	37	1	A	0	0
228397645	GARBARINI SA	50	51	25	1	B	0	0
223943425	GARTH GL	62	58	31	1	A	0	0
247591025	GREENBERG JA	30	38	15	1	B	0	0

Figure 1. ODB GUI

<u>Execution</u>

The program should initially present the user with a startup window displaying the name of the program, a short description of the program and the programmer's name. (This window should be the same window that appears when the user selects the “About” option from the Help menu.) After a few seconds has elapsed, the startup window is to be cleared and the main ODB GUI screen drawn (see Figure 1).

Illegal operations should be disallowed by the appropriate disabling of menu items. Non-functioning operations, during demonstration, (see grading section), must not bomb the program, but display a ‘warning’ notice informing the user of this fact.

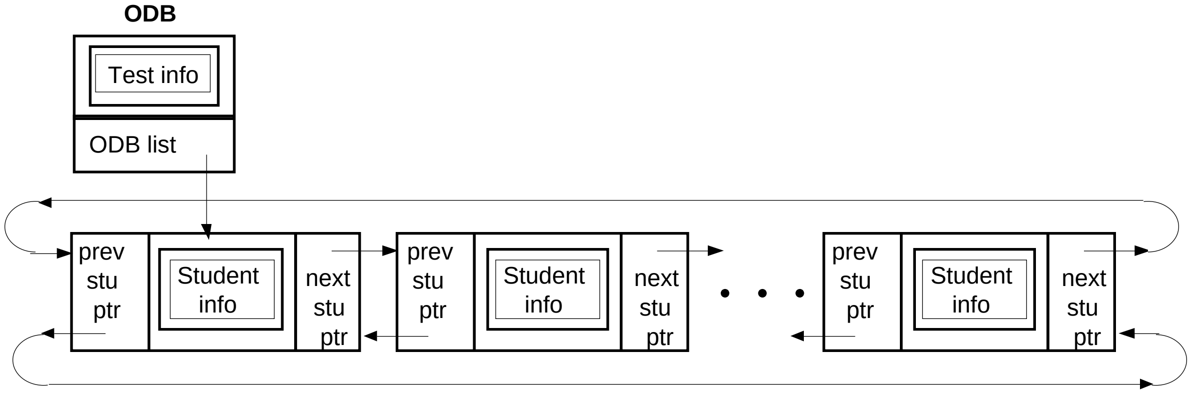
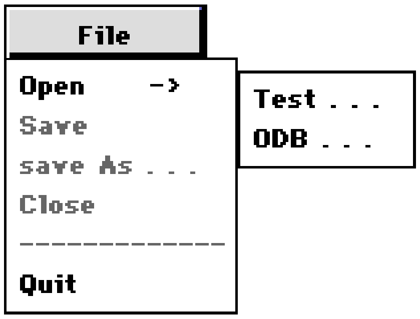
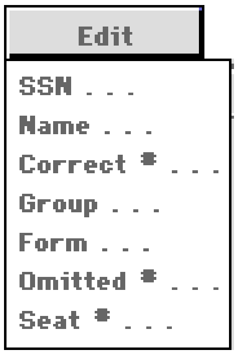


Figure 2. ODB Double List Structure

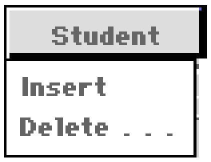
Menus

The five menus listed in Figure 1., their options and operation will now be briefly discussed in order. The menus will be shown in their initial disabled state. The File menu will provide access to the external File I/O operations. The Open option causes another popup menu to appear, (in a cascading hierarchical behavior), allowing the user determine whether they wish to input a previously saved ODB file or translate and save *raw* opscan test results files into ODB files. The Save option writes the currently open ODB file to disk. It should be disabled for the *raw* test results files that have not been previously saved. The save As option prompts the user, (in a dialog box), for a new file name in which to write the current open ODB file. When the Close option is chosen, the user is prompted to confirm that they wish to close the current open ODB file and when confirmed, the ODB list structure is destroyed, initialized to empty and the appropriate menu options disabled. The Quit operation will prompt the user for confirmation, and issue a termination message.

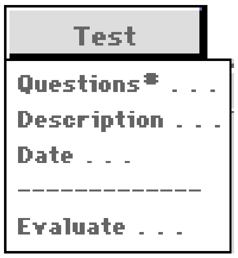




The Edit menu allows a user to alter the contents of the currently selected student record returned by a GIL picklist. Each operation will simply open a dialog box window, displaying the current value of the field from the selected record and providing an input field for entry of the new field value. “Cancel” and “Ok” buttons must be provided, with appropriate action taking place upon their selection.



The Student menu allows a user to add and remove students from the test. In the case of insertion a user must be prompted for each of the seven fields listed in the Edit menu. (This might be accomplished by cycling through the edit dialogs.) The Delete option will query to user to confirm the removal in a dialog box, (the current student SSN and name should be displayed in the dialog box), with “Cancel” and “Ok” buttons provided for the confirmation.



The Test menu provides editing of the test parameters. Each of the first three options operate analogously to the edit menu options. A dialog box window, with the current value of the test parameter field and providing an input field for entry of the new parameter value is displayed. “Cancel” and “Ok” buttons are also to be provided, with appropriate actions. Note that the two lines of descriptive test text has been reduced to one. The Evaluate option will recompute each students score, (score = number correct / number of test questions * 100), T-score (T-score = 10 * ((score - class average)/(standard deviation)) + 50), and test statistics (see Figure 3). The user should be informed that the operation may take a while to perform. Progress bars for showing the user an estimate of the time remaining to completion is not required.

The last menu, the Help menu, provides a user with minimal program information. The About ODB option displays the program startup, (splash), screen . The ODB option brings up a brief help screen explaining the major system features to the user.

$$\sigma = \sqrt{\frac{\sum_{i=1}^{N} (X_i - \mu)^2}{N}}$$

where:

σ	<- standard deviation
Σ	<- sum of the differences
μ	<- mean (avg of the scores)
X_i	<- student score
N	<- number of students

Figure 3. Standard Deviation Formula

<u>Code Reuse/Modification</u>

This assignment, like the first, will be reused in the final course project, thus if it is not fully operational at the time of the due date, it will still need to be working at the end of the final project. Since the code will be reused, it is advisable to make the code as general as possible.

<u>Design</u>

An initial structure chart design is not required for this program, but is highly recommended. The final structure chart for the program is expected to differ greatly from the chart for the previous program, due to the addition of the GUI.

<u>Grading</u>

Turn in hard copies of the final structure chart, source code, input/output files, and a diskette, (system labeled), with files containing: ASCII source code, executable image, I/O files, and an ASCII *readme* file with execution instructions. The main source code file must be named <u>ODB.cc</u> or <u>ODB.cpp</u> and the executable image <u>ODB</u>. The disk should contain no subdirectories. Protection should be set to allow anyone access. All materials to be graded must be placed in a sealed envelope, neatly labeled. In addition, your GTA will require you to <u>demonstrate</u> your program during lab time in the McB 116/118 lab. To receive partial credit for programs that are non-working, or are not fully functional, a brief one or two paragraph description of the problem(s) must be included in the assignment folder. The location, routine minimum, must also be specified along with possible corrections that need to be made.