Problem

Code a program, (BAL), to coordinate lists of bowling analysis data, stored in a Bowling Analysis Management, (BAM) files. [For a description of the BAM file format see the Bowling Analysis Checker, (BAC), specifications.]

Discussion

This program will allow a user to create and perform maintenance upon lists of bowling data, (Alleys, Events, Tools, Balls, Releases, Conditions, Games, and Frames), for a Bowling Analysis Management system. In addition to allowing a user to perform management operations upon existing bowling data lists, BAL will also allow a user to create BAM files.

The interface for this program will be simple one-line text menus. A user chooses an operation by entering the first letter of the menu operation name enclosed in parentheses and hitting the enter/return key. After an operation completes a menu line is then (re)displayed.

The initial menu line will provide a user with the following operations:

BAL: (N)ew, (O)pen, or (Q)uit?

The New operation allows the user to enter the name of a non-existing BAM file to be created interactively. The BAL lists are initialized, (i.e., set to empty, containing no records). After the BAL linked-lists are set up, the BAL section menu line below is displayed. The Open operation allows the user to enter the name of an existing BAM file. The BAM file is checked for existence, input and parsed, an error report is produced and the BAL lists are built, [see the Data Structures section following]. This involves building double-linked lists of all of the bowling data sections and records stored in the BAM file, ordered upon the record indexes. After the BAL linked-lists are built, the second menu line below is displayed. The Quit operation queries the user to be sure they wish to leave the system, (basically to annoy them ☺), and then issues a brief termination message before exiting. The second BAL section menu line below will provide a user with the following operations:

BAL section: *BAM file name*
(A)lleys, (E)vents, (T)ools, (B)alls, (R)eleases, (C)onditions, (G)ames, (F)rames or c(L)ose?

The user selects which list, (section), they wish to operate upon by entering the corresponding letter. A selection of a list leads to the display of the list records and the menu line below. The Frames operation will require a user to first select the Game containing the frames with which they wish to work. The selection of the desired Game will be accomplished through the implementation and use of a numbered menu list as described later in these specifications. The cLose operation first queries the user to be sure they wish to close the current BAL/BAM file, (and to continue to annoy them ☺). It then writes out the BAM file, with a '`.bam`' extension, (if it has been changed), destroys the internal BAL lists, closes all open files and returns control to the previous BAL initial menu line.

BAL Records: *list name*
(A)dd, (D)elete, (E)rase, (M)odify, (L)ist or (T)erminate?

The Add operation allows a user to interactively enter record data to be inserted into the current section, (list). The user is queried for entry of each section field, in order of occurrence in the BAM section record specification, (except for index fields). The user must be given the option of skipping entry of any field. In which case a corresponding default BAM field value is assigned. The user is not queried for section indexes; these are assigned automatically by BAL. Unique index generation should be simple. BAL need only increment the largest existing index in a section by one to create a new unique index for an added record. For Games and Frames section record entry, the index references to records in other sections must be presented as a numbered menu list for selection by the user. For example, when a user is adding a game record, this is the mechanism they will use to associate an alley record with the game to indicate in which bowling alley the game was rolled. A numbered list of the existing alley records will be presented to the user for their selection by the menu list number. The alley record indexes should be used for the menu numbers. Note that optionally not all of the alley record fields need be displayed for this selection. Only the name, city and state need be shown. To prevent numbered list menu entries scrolling off the display, only 10/20 at a time should be displayed allowing the user to select a displayed entry or proceeding to the next 10/20 entries.

The Delete operation allows the user to remove a section record. The section record (L)ist number, [see the (L)ist menu operation description following], is input and the corresponding entry is then removed from the current BAL section list. If the entered section record (L)ist number is not in the current section an appropriate error message must be given. Be aware that this may invalidate records in the Games and Frames sections which contain references to the removed record through their associated indexes. BAL will not be responsible for checking the integrity of the associated index references when a delete is performed.

The Erase operation provides the user the ability to destroy all records in the current section. Due to the severity of this operation, the user is queried to confirm that they do wish delete all the section records. Upon confirmation the section list nodes are destroyed and the section list is set to empty. The user at this point may elect to add records into the list or terminate and move to editing another section.

The Modify operation is a future operation. Selection of the operation should inform the user that it is not currently functioning in the current version of the system. When implemented later, it will allow a user to enter a section record (L)ist number, to be modified. The record entries will then displayed and the user will be allowed to indicate and change any field.

The List operation simply displays a numbered listing of the records of the current section. It should display the section record's field contents using the record indexes presented in a labeled tabular format. Displaying 10 to 20 list elements at a time, allowing the user to hit the return key to list the next 10/20 elements. More than one line may be used to display an entry.
In order to allow for the creation of unique indexes for the frame records the Frames section record format will be changed slightly. An additional Shot field will be added, immediately following the Frames field, shifting all following fields appropriately on the first line. The Shot field will store the number of the ball roll in the game, [1-21]. By multiplying the game index by 100 and adding the shot number, a unique frame index number can be generated. The updated Frames section record format table description follows:

line	cols	label	contents	cols	type	limits	comment
1	1-5	Gdex:	*game index*	7-11	int	> -1	
1	13-18	Lanes:	*lanes: 01-02*	20-24	ints	> 0	
1	26-31	Frame:	*frame number*	33-34	int	> 0 & < 11	
1	**36-40**	**Shot:**	***shot number***	**42-43**	**int**	**> 0 & < 22**	
1	45-49	Bdex:	*ball index*	51-55	int	> -1	
1	57-61	Rdex:	*release index*	63-67	int	> -1	
2	1-9	Approach:	*feet location*	11-14	float	> 0 & ≤ 16	distance from foul line
2	16-20	Feet:	*feet horizontal*	22-25	float	> -41 & < 41	board placement of foot
2	27-31	Mark:	*board target*	33-36	float	> 0 & < 40	aiming point on lane
2	38-46	Accuracy:	*target hit/miss*	48-51	float	> -41 & < 40	
2	53-59	PinHit:	*ball impact point*	61-64	float	≥ 0 & ≤ 40	
2	66-71	Score:	*ball score*	73	char	set:	{ '0' … '9', 'X', '/', '-', 'F'}
2	75-80	Leave:	*spare difficulty*	82	char	set:	{ 'S', 'W', 'D' }

The Terminate operation first queries to user to be sure they wish to finish editing the current section list, (and to continue to really annoy them ☺). It then writes out the BAM file, (if it has been changed), and returns control to the previous BAL section menu.

This program will later be incorporated into a bowling science (BS), bowling analysis management (BAM) system that will be developed in the next version of the system. Note: The standard double linked-list operations, (insert, remove, etc.), from the text/notes may be used, but must be properly modified and referenced. Code from other sources must be approved by the instructor before it can be used. (To use code from other sources without prior approval is an Honor Code violation.)

Execution

The program should initially present the user with a startup screen displaying the name of the program, a brief 3-4 line explanation of the program and the programmer's name, and email address. After the user hits the return key, the startup screen is to be cleared. A brief help screen should appear explaining the program, which is cleared upon the pressing of the return key. (The startup and help screen may be combined if desired.) Following the brief help screen the BAL, (initial), menu line is displayed and execution continues as described above in the discussion section until the user Quits the system.

Structure Charts

The initial structure chart design for BAL must be submitted by **Friday March 5th**, in a zip archive through the automated acceptor. The project number for the acceptor for this project will be "`bal`", typed in lowercase, omitting the quotes. Submission formats are described on the course Web site. The chart must be embedded in a single MS Word document file named "`<schrt1p2>.doc`", on 8.5" X 11" formatted pages. The student's name and email address must be listed at the top of the first page of the file. The size of the file must be < 2MB. Students who insist on hand-drawing the chart and scanning it into Word must first compress the images into GIF formats before insertion into Word to control the file size. The chart may drawn using MS Draw, or other drawing program of choice, but must be viewable without problem in the Word document. The "`pledge.txt`" file as described on the Web site program submission format must also be in the archive. A final structure chart reflecting design changes and corresponding to the code must be produced and submitted with the executable. The evaluated initial chart will be compared against the final chart to determine the overall quality of the design.

Input
BAL is only responsible for being able to read files that follow the BAM file specifications as described in the BAC specifications and as amended above for the frame section records. BAL is also required to be able to read and process BAM files that have been previously saved in BAL.

Output
BAL is required to write 'clean', (replacing discovered field errors with default values), BAM files. No file output beyond the saved BAM files, and error report is required.

Data Structures
The BAL section list(s) must be represented as abstract double linked-list data structures, implemented with a pointer representation. The double linked-list abstract data structures may be implemented based upon a either procedural model or a class (OOP) paradigm. (An object oriented based implementation will be required in the final version of the system.) Optionally, for students who have experience coding C++ templates, an alternative implementation using these features is permitted. Be forewarned, if you have not previously written programs with templates you are advised not to try to learn them on the fly and utilize them. To do so and failing to produce a working program will have severe grade penalties and will not constitute a basis for an argument of your grade!

A design decision has to be made with regards to the storage of the frame records. Either a separate list containing all of the frame records must be implemented or the Game record format must be extended to include storage for an array to hold all of the frame records for a game. Either approach may be taken. This is the only instance where a BAM record section may be merged into another section record and represented as an array.

Assumptions
It may NOT be assumed that a BAM file section will contain any maximum number of entries limit. Error checking must be performed whenever possible, (file existence, illegal menu operation selection, etc.). Note -- don't waste time on a snappy interface; no brownie points will be given for one and no points will be lost for not implementing one ☺.

Grading
Separate compilation, and abstract double linked-list data structures, (pointer implementation), are requirements of this program. Failure to utilize these coding features in the program will result in severe grade penalties. The due date for this program is Wed., **March 31st**. Standard C++ class libraries must be used instead of the non-OOP (Object Oriented Programming) libraries. This requires the `using namespace std,` (standard).

The submitted archive must contain soft copies of the initial evaluated structure chart, final structure chart, (in MS Word file .doc format named "`schrt2p2.doc`".), MS Visual C++ IDE project files, (the project workspace file must be named "`BAL.DSW`"), source code, input/output files, executable image (named "`BAL.EXE`"), pledge statement file, and an ASCII readme file, (named "`BAL.TXT`"): with execution instructions, the name of the project, programmer's name and email address, this course number (CS2574), and lecture section (day/time). Protection or encryption must not exist on the archive or any files. (Note only the project `.dsw`, `.dsp`, `.h` and `.cpp` files in the project folder must be included. The `obj`, `ncb`, `ilk`, `pdb`, `pch`, `idb`, and `exe` files should be omitted to control the size of the submitted archive.)

To receive partial credit for programs that are non-working, or are not fully functional, a one or two paragraph description of each of the problem(s) with the program must be included in the assignment archive in a text file named "`PARTIAL2.TXT`"). The location, routine minimum, of each suspected problem/error must be specified along with possible corrections that need to be made. Non-working/non-implemented menu operations must not bomb the program when selected. They must display a message informing the user that the features are not currently implemented.

Everyone is required to demonstrate their program to the TAs. Instructions for the McBryde 116/118 CS lab demonstration times will be placed on the course Web site. Demonstrations will take place the week of April 1st to April 8th. TAs will have signup sheets available for demo slots.