Clayton Kimber
CS 202
Assignment 1b

When designing part 1b from 1a, the entirety of the code was scrapped, save for the pseudo-code wrapper of the previous assignment. I did this due to the fact that my previous assignment did not work as intended, and had several obvious issues with the functionality that it was intended to have. By re-coding the entire project, I was able to learn the items that were lacking in part 1a.

Starting with the main menu, the case statement issues textual prompts and requires user input to navigate the menu system. This system has six main objectives, as outlined in the design parameters, which will be discussed below.

Reading in from the default file provided was taken care of by the *readFile* function that simply iterates through the file that is opened called "symbols", and places the character string in each array space. This action was also used in the second menu item, but in the *userFileRead* function, the user is prompted to name the file that is being read. In future versions, I would like to compress the functions together and prompt in the case statement to reduce the function count.

Creating the slot machine reels array was done by *createSlot*, which used the random number generator to load a integer variable from 0 through 5 in order to get an index of the symbols array. These random symbols are loaded into the array passed to *createSlot* within a for loop set by hard-coded values. Future versions will remove the brute force *strCopy* function that is inside of *createSlot* and put it into its own function. This was an oversight of the actual use of the function, and will be corrected.

Printing the reel array out the the screen is handled by *bonusToScreen* and *reelsToScreen* fuctions, each printing their descriptors part of the display. Each function is very similar to each other, and in future versions will be compressed into a single overloaded function that will take either a two or three column array as the parameters instead of having two separate functions.

Once either selection 1 or 2 have been made, and an array is loaded into the reels array, selection 4 becomes available. It is technically always selectable, however will through a warning that the arrays aren't loaded and fall back to the main menu. Printing out the selection of reel and stop is

handled by the *selectionToScreen*, *bonusFetch*, and *strComp* functions, each calling the following function in order to accomplish the desired output. To answer the design question:

> How will the correct bonus value be found when printing out the symbol and bonus?

The *strComp* function can be used iteratively through the reel and stop reference to find the exact match in the symbols array. From there, you pass the index of the matched character string in the symbols array to the value column of the symbols array. In my solution, I used *atoi* to convert the character array into an integer, which was only needed for the 10 bonus value, and that is passed back to the *selectionToScreen* function for printing.

Printing to file is handled by *writeFile*, and is the same process as the read file functions that were used to read in the symbols file and reels files, except that the data is being written, not read. Simple prompts allow the user to input the file name as per design requirement.