Honors Computer Science C++
Mr. Clausen
Program 8A

Special Note: Every program from Chapter 4 to the end of the year needs to have functions!

PROBLEM 8A: I'm_Sort_of_Searching (50 points)

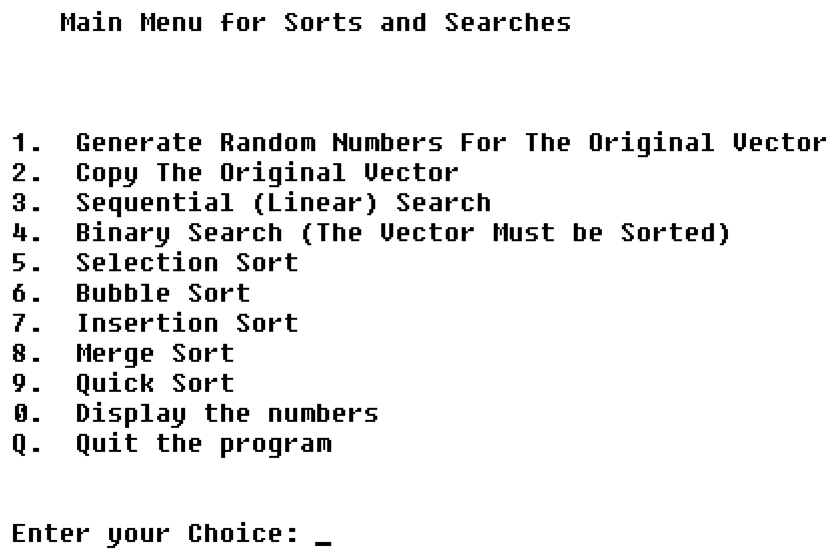
The purpose of this program is to practice vectors and the sorting and searching algorithms. Create a vector **(apvector)** to hold as many integers as the **user** would like. Ask the user how many elements they would like to use for this vector (NOTE: I will test your program with at least 1000 elements). Fill this apvector with random numbers from 0 to **list.length( ) - 1**. Make a duplicate copy of this apvector. A good name for the original vector is list, and list_copy is a good name for the copied vector. Test the program with 20 or 30 numbers, and then when your program works, increase the size of the apvector (you don't have to "resize" the vector while the program is running, just enter a larger number when you run the program).

Your program should have a main menu to sort the list_copy apvector of random numbers by bubble sort, selection sort, and insertion sort (leave room for merge sort and quick sort) and duplicate the original apvector of random numbers. **After each sort, copy the duplicate apvector back into the original apvector, so that we will be sorting the same numbers each time for a better comparison of each sort.**

Also in your menu should be choices for linear search or binary search (remember that a binary search assumes that the list of numbers is previously sorted). Ask the user to enter a number to search for from 0 to list.length() - 1. Search through the apvector for that number, keeping track of how long it takes with each search. Make sure your program takes care of the case where the number is not found in the list. Also make sure to consider the case where the number occurs more than once, list every index where the number occurs and count how many times the number occurs.

When displaying the lists of numbers use rows and columns so that we can see more numbers on the monitor screen.

For now we will only fill in the details for Bubble Sort, Selection Sort, Insertion Sort, and the Sequential Search. We will add the other sorts and searches at a later date. **Below is a sample of the menu choices you should use…**



```
   Main Menu for Sorts and Searches


1.  Generate Random Numbers For The Original Vector
2.  Copy The Original Vector
3.  Sequential (Linear) Search
4.  Binary Search (The Vector Must be Sorted)
5.  Selection Sort
6.  Bubble Sort
7.  Insertion Sort
8.  Merge Sort
9.  Quick Sort
0.  Display the numbers
Q.  Quit the program


Enter your Choice: _
```

Save the program as LastNameFirstNameP8A.cpp in your "S:" directory. To see a model for the menu setup for this program look at the source code for the program "MenuSample.cpp" in the network directory titled: HonorsCompSciCFiles. Look in the folder, Text Book Programs and Ch8.
As you type all your programs this year, be sure not to type past the 80-column line in Borland C++ 5.02 for Windows. If you have any statements longer than 80 columns, press the return key to "wrap" the statement around to the next line.

1. Type comments at the beginning of the program to display your name and other information just like those used for program 2A. **Make sure to change the program name and program description in these comments, so that the program number, name, and description say what is listed above.**

2. Include <iostream.h> so we can use the **cout** and **cin** commands. Also include <conio.h> so you can use **getch**() to leave your output displayed on the screen until the user presses any key to continue. Include <iomanip.h> so we can use the **setw** command. Also add the following preprocessor directives to your program: #include <math.h>, #include <stdlib.h>, #include <time.h>, and #include "apvector.h" (we will use apvector.h, not apvector.cpp).

3. Declare a constant of int LOW = 0;.

4. Don't forget to include comments for your functions, right before each of your function declarations.

5. Declare the function Display_My_Info ( ) in the function declaration section of the program. This should be a void function that doesn't take any parameters and doesn't return any values. In addition, declare these functions:

char Menu_Choice(); **void** Control_Menu_Execution(**char** menu_choice, apvector<int> &list, apvector<int> &list_copy, **int** logical_size); **void** Random_Array(apvector<int> &list, **int** logical_size); **void** Duplicate_Array(apvector<int> &list, apvector<int> & list_copy); **void** Sequential_Search(apvector<int> & list_copy, **int** logical_size); **void** Binary_Search(); **void** Selection_Sort(apvector<int> & list_copy, **int** logical_size); **void** Swap_Data (**int** &number1, **int** &number2); **int** Find_Minimum (apvector<int> & list_copy, **int** first, **int** length); **void** Bubble_Sort();**void** Insertion_Sort();**void** Merge_Sort();**void** Quick_Sort();**void** Display_Numbers(apvector<int> &list2, **int** logical_size); and **void** Log_Off_Message();

6. Use a comment line with **equal signs** to separate all of the above from the int main function. For example:
   //===

7. Inside the **int** main() function, declare all of your variables right away except for the two vectors. Ask the user what the upper limit should be for the vectors and then declare the two apvectors. After that call the functions Display_My_Info ( ), and the functions Menu_Choice() and Control_Menu_Execution should be called in a do …while loop. All of the other functions will be called from the Control_Menu_Execution function. **Don't forget that value-returning functions are "called" in assignment statements, while void functions can be called just by writing the function name including any of the "actual" parameters.** Then tell the user to press any key to continue, have your getch() command, and your return 0 command.

8. After the **int** main() function, have another comment line of **subtraction signs** to separate the above from your function implementation lines below. For example:
   //---

9. Implement all of the functions, separating each one from the other ones using comment lines of **subtraction signs**.

10. When you implement the function: Display_Numbers, don't forget to display the numbers as a table using 10 numbers across each row. To format the numbers nicely use the setw command to set a width for each of the columns. You also want to right justify these numbers. This will require that you add #include <iomanip.h> to your preprocessor directives and use the following commands: cout << setiosflags (ios::right) ; and cout<< setw(5);

11. After the last function implementation of every program, end your program with two comment lines of equal signs. This signifies the end of your source code. For Example:

```
//==========================================================
//==========================================================
```

When you are finished with your program, have tested it thoroughly to make sure that your calculations are correct, and are sure that you don't need to make any changes, then save your program in the "W" network mapping, and the Program 8A folder.