CMPT 135: Lab Work Week 6

File Input Output

1. Create a text file named **ClassList.txt** ***manually*** using notepad and edit it as follows

John Walter	**20**	**19**	**45**
Sara Gill	**16**	**15**	**35**
Mark Black	**23**	**24**	**50**
Jess Paul	**10**	**20**	**25**
Joe Nash	**14**	**18**	**44**
⋮			

Think of these as students' full names and their assessment marks for exercises, projects and final exam. List as many students' details as you wish; it does not matter how many students are listed in the file.

Write a C++ program that reads this file (ClassList.txt) and creates a new file named **Report.txt** with the same content together with the letter grades of the students on the same line for each student. For the letter grades, use the settings [90, 100] = A, [75, 90) = B, [65, 75) = C, [50, 65) = D and [0, 50) = F. When you write your program remember that you don't know how many students are listed in the file. For this reason, you must use eof() member function to keep on reading input until the end of file marker is reached.

This means when our program finishes execution, then the **Report.txt** file should contain the following data:

John Walter	**20**	**19**	**45**	**B**
Sara Gill	**16**	**15**	**35**	**C**
Mark Black	**23**	**24**	**50**	**A**
Jess Paul	**10**	**20**	**25**	**D**
Joe Nash	**14**	**18**	**44**	**B**
⋮				

2. **[Challenge]** Do Question #1 above again but this time making sure to read from an input text file named **ClassList.txt** and write the output to the same file **ClassList.txt**. In this question assume that there are exactly **five** students listed in the input file.

3. Write a C++ program that generates a random integer **n** in the range [50, 100] and that writes **n** random integers in the range [0, 999] to an output text file named **randomNumbers.txt**.

4. Write a C++ program that reads the file **randomNumbers.txt** you created in Q3 above and prints the minimum and maximum integers in the file. Your output should be printed to the console output screen. Remark: You are not allowed to declare any array. Moreover remember that you don't know how many numbers are on the input file which means you must read until the end of file is reached.

5. Write a C++ program that reads the **randomNumbers.txt** file you created in Q3 above and that prints the data in the file to the console output screen in reverse order; that is the first integer read will be printed last and the last integer read will be printed first. Remark: You may use an array (or better a **SmartArray** object) when you answer this question. Moreover remember that you don't know how many numbers are on the input file which means you must read until the end of file is reached.

6. Write a C++ program that reads the **randomNumbers.txt** file you created in Q3 above and that prints the data in the file to the console output screen in reverse order; that is the first integer read will be printed last and the last integer read will be printed first. Remark: You are not allowed to use any array when you answer this question. Moreover remember that you don't know how many numbers are on the input file which means you must read until the end of file is reached.

   Hint:- You may write a recursive function that takes an input file streaming object argument and that prints the data read by the input streaming object in reverse order.

7. Write a C++ program that reads a positive integer **n** from the keyboard and then creates a new file named **multiplicationTable.txt** that contains the following information

1 2 3 .... n
2 4 6 .... 2n
3 6 9 .... 3n
.
.
.
n 2n 3n ... n^2

   Note that you are not allowed to declare any array here.

8. Consider a file similar to **ClassList.txt** you ***manually*** created in Q1. Write a C++ program that reads the file and prints the top student (i.e. maximum total marks). Your program must print the full name, the marks, and the letter grade of the top student only. Again, you are not allowed to use any array to answer this question.

9. Write a C++ program that does the following:

   a. Create an output file stream and connect it to a file **"RandomFloats.txt"**
   b. Generates a random integer **n** in the range [900, 1000]
   c. Print in the file **n** random floats in the range [0.0, 100.0)
   d. Close the output stream
   e. Create an input stream and connect it to your file **"RandomFloats.txt"**
   f. Read all the **n** floats in the file into an array
   g. Close the input file stream
   h. Sort your array with a bubble or selection or insertion sort algorithm
   i. Create an output stream and connect it to a file **"SortedRandomFloats.txt"**
   j. Write the sorted array of floats into the output file stream
   k. Close the output file stream
   l. Check your file **" SortedRandomFloats.txt"** in Notepad to see indeed it is sorted.

Exception Handling

10. Discuss the pre and post conditions for the following functions

- A function that takes an integer argument n and returns the sum 1+2+3+...+n
- A function that takes an integer argument n and returns true if n is a composite number.
- A function that takes an array of float data type and its size as arguments and returns the sum of the elements of the array.
- A function that takes an array of double data types and its size as arguments and returns the minimum element of the array.
- A function that takes two arrays of string data type and their sizes as arguments and returns true if all the elements of the first array are found in the second array.
- A function that takes a vector of integers and an integer value as arguments and deletes the elements of the vector that are equal to the integer argument.

11. Implement the functions described in Question #2 above with appropriate exception handling techniques. You may use abort or assert functions or try-catch blocks as you see fit. If you think there is no a runtime error that you can handle with exception handling then you may write your functions without any exception handling techniques.

Switch-Case Statements

12. Consider the following program that makes use if if, else-if, and else statements. Convert it to an equivalent program that makes use of only switch-case statements but not any if, else-if, else, or ternary (conditional operator) statements.

```
int main()
{
    srand(time(0));
    int x = rand() % 100;
    int y = rand() % 100;
    cout << "The value of x is " << x << " and that of y is " << y << endl;
    if (x % 2 == 0 && y % 2 == 0)
           cout << "Both even." << endl;
    else if (x % 2 == 0 && y % 2 != 0)
           cout << "x is even but y is odd." << endl;
    else if (x % 2 != 0 && y % 2 == 0)
           cout << "x is odd but y is even." << endl;
    else
           cout << "Both odd." << endl;

    system("Pause");
    return 0;
}
```