Fall, 2016

CS 58000 Program 2

Part 1

I am posting this as a Word doc so you can copy sections of code into your own code file(s). Again, as for all C++ projects in this class, you want to create an empty console project.

One file I suggest you use is utility.h, shown here. It's handy to have this file containing, all in one place, many of the most useful "includes" for standard libraries such as iostream, fstream, cmath, string, etc. This saves you having to figure out which includes you need for any particular program.

#ifndef UTILITY\_H

#define UTILITY\_H

//Gives ANSI version of standard includes

#include <iostream>

#include <limits>

#include <cmath>

#include <cstdlib>

#include <cstddef>

#include <fstream>

#include <cctype>

#include <ctime>

#include <string>

using namespace std;

#endif //UTILITY\_H

You should therefore do an

#include "utility.h"

at the beginning of your files where you would normally put include statements, and of course make the utility.h file part of your Visual Studio project.

An include statement actually inserts the referenced file into your program before it is compiled, so including utility.h - which in turn includes lots of other files, some of which a particular program might not use - is electing simplicity for the programmer over code optimization. Not a bad choice!

The second file, filer.h, appears below.

#ifndef FILER\_H

#define FILER\_H

//header file FILER.H for class filer

//a class object can create a file of a specified

//number of random integers in a specified range

class filer

{

public:

void makefile(int n, int range, string file\_name);

//precondition: n, range, and file\_name have

//been set;

//postcondition: n random integers in the range 0-range,

//inclusive, have been written to a file named file\_name,

//one per line;

private:

int next\_number(int range);

//precondition: range has been set to a nonnegative value

//postcondition: returns a random integer in the range

//0-range, inclusive

};

#endif //FILER\_H

This is a class interface file. Objects of class *filer* have no member variables, but do have available two member functions, *makefile* and *next\_number*. Carefully read the prototype comment for each function to see what service it is supposed to provide.

The *filer* class allows an application program to create a data file of random integers. The application program should get the information from the user about:

how many numbers to put in the file

the upper limit on the size of the numbers (the lower limit is 0)

the name of the file

**You need to write the *filer.cpp* file and also the application file. Then use the application file to create 3 different files each containing 100,000 random integers with maximum numeric value of 32,000, which will guarantee duplicate values, but that doesn't matter. (You can name the 3 files whatever you want as they will be for your use only, but they should be text files, i.e., xxx.txt.)**

Hints:

1. Use a variable of type *string* to store the filename the user supplies. The extraction operator >> has been overloaded for the string class, so you can read in the filename just as easily as any other simple variable. As of C++11, the open function can also take a string variable parameter.

1. Your .cpp files should contain includes for both utility.h and filer.h Be sure to put the include for utility.h first, otherwise the compiler will not recognize such things as the string data type for the file name (in the application file or in the makefile prototype) because it hasn't yet seen the include for the string class.
2. Use the *rand* function, declared in library file cstdlib. This function, called with no arguments, returns a pseudorandom integer between 0 and some system-defined maximum. In order to get integers that are the right size, think in terms of modular arithmetic, i.e., if you want numbers between 0 and 9, think about the remainder when dividing by 10.
3. You want truly random numbers (as opposed to pseudorandom numbers that look random but actually repeat with every program execution). To get truly random numbers, you must "seed" the *rand* function before you use it the first time. You can do this by using the *srand* function, also in cstdlib. The statement is

srand(int(time(NULL)))

The function *time* is found in the ctime library, which you have already included by the use of utility.h, so you don't need to do anything special here.

1. To write to a file, you must instantiate an object of class ofstream. This object should then use the *open* method (available to objects of this class) to open a file, passing as a single argument the string variable holding the file name.

The object should also use the close method to close the file when access to the file is no longer needed. The object is an output stream, just like cout, but directed to the file rather than to the screen.

6. Just to be sure you are getting random numbers as opposed to pseudorandom numbers, glance over your 3 files to be sure they are different.