ECOR Winter 2014 - Assignment #6

Notes:

- **Program comments (including name / student number in the first line) and proper formatting (i.e. indentation) are required for full marks on all programs!**
- **Programs that do not compile and run in the Dev-C++ lab environment (ME4390/ME4377/MC5010) get a maximum of 25%.**

Your best 5 out of 6 assignments count towards your final grade (1% each). To encourage those who have done well on the first five assignments to do this one (as it will help you on the final exam!), it will be counted as 2% of your final grade. In other words, there is a bonus mark. You can get up to 6/5 on the assignments portion of the course.

Question 1

Tic-Tac-Toe is a game where two players alternately place X's and O's onto a 3x3 grid until one of them wins by getting three in a row (line, column, or diagonal) or the game is tied when the grid is full. We are going to represent the grid by an array of 9 integers. The initial state will have the grid containing the numbers 1 through 9, starting with the upper left square of the grid. If an "O" is placed in a square, its value is changed to 10. If an "X" is placed in a square, its value is changed to 11.

After a game is complete, the user is prompted as to whether he/she wants to play again. If so, the player that goes first in the new game is the player that lost the previous game or, in event of a tie game, it should be the player who went second in the game just completed.

You have been provided with a skeleton program ("*a61skel.cpp*"). It contains the function prototypes and the main program. You need only write the functions. Do not change the function prototypes in any way! **When downloading the skeleton, do not cut and paste into a file as the tabs will likely be wrong. Instead, right click the filename and select "save target as" and save the file as a cpp file.**

You have also been provided with a sample executable. As usual, your program should behave just like that one.

Note that the sample executable is "bullet-proof." If any values entered are not integers, it outputs an error message and continues. Your program must behave in the same way. See Chapter 8. Hint: The "bullet-proof"ing will be in the *getIntBP* function.

Call your C++ file "*a61.cpp*". Submit "*a61.cpp*" as Assignment 6; A#6 – Question 1.

Question 2

A data file from the experiment has been provided where the data has been analyzed for day of the year resulting is a list of days, 1 through 366, for the participants' activities. You need to analyze the data to determine which days are the most common for activity. These days of the year have been stored in a file. Each line of this file contains a single day of the year (1 to 366). Here is an example:

```
365
2
15
29
56
100
15
2
4
365
366
4
1
3
366
365
```

You are to write a program that will read in any file of this format and produce an output file containing a table like that shown below:

Day of Year	Events	%Events
365	3	18.8
2	2	12.5
15	2	12.5
4	2	12.5
366	2	12.5
29	1	6.3
56	1	6.3
100	1	6.3
1	1	6.3
3	1	6.3

Note that the table must be sorted, with the most frequent day in the first position and so on. If two or more days have the same number of votes (i.e. there is a tie), then they may be in any order in the output file. For example, in the table above, 2, 15, 4, and 366 could be in any order, and the output would still be correct.

Your program should read in the names of the input and output files. If any problems occur in opening these files, your program simply outputs an error message and terminates.

If the input file contains non-numeric data (e.g. "junk"), your program should report the problem, recover from the error, and keep reading values. Day of year numbers must be in the range of 1 through 366, inclusive. If an invalid number is encountered, your program should report the problem, ignore the invalid number, and continue reading.

You will need two arrays, one to keep track of the days of the year that have been seen, and the other to keep track of the number occurrences of that day. These are "parallel arrays." The first element of one is implicitly associated with the first element of the other, and so on. For example:

Day of Year	Number of Occurrences
366	2
1	1
3	1
15	2
100	1
365	3
29	1

You may assume that there will never be more than 100 different days of the year in an input file.

Don't forget that, when you sort the occurrences array, every change made must also be applied to the day of year array as well. **For full marks, you must use the "swap" function from page Ch6-22 of the notes when sorting your arrays**, i.e.:

```
void swap (int &v1, int &v2) {
   int temp;
   temp = v1; v1 = v2; v2 = temp;
}
```

A sample input file is provided. Note, however, that you should create **many** more test input files to make sure that your code works properly in all circumstances!

As usual, a sample executable is provided. **** Note that to run the sample executable, you must first save it to your computer, put the data file(s) in the same folder, and then run it. ****

Call your C++ file "*a62.cpp*". Submit "*a62.cpp*" as Assignment 6; A#6 – Question 2.