CIS 1.5 Fall 2009 Lab II.2

1. Putting things together

   - On the next page is a longer program than others we have seen so far this term. It is called **roomba.cpp**. It simulates the action of a robot moving around in a room. We will work on understanding and modifying this code.
   - Begin by entering the code into a new program called **roomba.cpp**. Prof Parsons has a copy of the program on a memory stick which you can copy onto the laptop you are using.
   - Compile and run it, to make sure that everything is working correctly.

2. Handling other input

   - Notice that the user can enter either Q or q to quit the program.
   - However, if the user enters f instead of F, the program does not recognize the lower case letter.
   - Modify the code so that the user can enter either upper or lower case of all the possible input letters (F or f, B or b, L or l, R or r and Q or q).

3. World's end

   - Like most rooms, the room the robot is wandering around in has walls and therefore has fixed dimensions. This means that the x and y values which indicate the robot's location have limits. They cannot be negative, and they cannot be greater than the size of the room.
   - Assume that the minimum possible value for x is 0 and the maximum possible value is 10.
   - Assume that the minimum possible value for y is 0 and the maximum possible value is 10.
   - Modify the code so that the program prints out a message when the robot gets to the edge of the room and another message when the robot is in a corner.
   - Now modify the code to make sure that the x and y values do not exceed their limits.
     Hint: If the robot's x or y value reaches its minimum, then do not subtract anything from it.
     Another Hint: If the robot's x or y value reaches its maximum, then do not add anything to it.

4. Wrapping up

   - Unlike physical rooms, in the virtual world, it is not uncommon for a bot to be able to "wrap around".
   - This means that if the bot wanders to the righthand edge of the room and keeps going in the same direction, it will leave the screen and re-appear again at the leftmost edge.
   - Similarly, if the bot wanders to the bottom edge of the room and keeps going in the same direction, it will leave the screen and re-appear again at the top edge.
   - Modify the code so that the robot will wrap around.
     Hint: If the robot's x or y value reaches its minimum, then set it to its maximum value.
     Another Hint: If the robot's x or y value reaches its maximum, then set it to its minimum value.

```
//--------------------------------------------------------------------
//
// roomba.cpp
//
// This program simulates a robot wandering around a room.
//// Written by:  Elizabeth Sklar
// Modified by: Simon Parsons
//
// Last modified: 15th September

#include <iostream>
using namespace std;

int main()
{
  // Declare variables

  int x;   // robot's x position
  int y;   // robot's y position
  char c;  // user's input
  bool q;  // does user want to quit?

  // Initialize variables

  x = 0;
  y = 0;
  q = false;

  // Loop until user enters Q to quit

  while ( q==false ) {

    cout << "the roomba is at location (" << x << "," << y << ")" << endl;
    cout << "which way should roomba move (enter F,B,L,R or Q)? ";
    cin  >> c;
    cout << "you entered: " << c << "\n";

    if ( c=='F' ) {
      y = y + 1;
    }
    else if ( c=='B' ) {
      y = y - 1;
    }
    else if ( c=='L' ) {
      x = x - 1;
    }
    else if ( c=='R' ) {
      x = x + 1;
    }
    else if (( c=='Q' ) || ( c == 'q' )) {
      q = true;
    }
    else {
      cout << "Oops! you entered something invalid. please try again :-)" << endl;
```

```
    }

  } // end while q==false
} // end of main()
```