CIS 1.5 Fall 2008 Lab I, Part 2

Instructions

- This is the second part of the first homework/lab assignment for CIS 1.5. Read the first part of the assignment for complete instructions, due date and submission details.

1 Before you start

- Follow the "How to use CodeBlocks" instructions to create a new project called **roombaWorld**.
- Get a copy of the roombaWorld C++ code from Professor Parsons. If you are in the lab, the you can get the code directly from him. The code is also on the course website, and is given in Appendix A.
- Copy the program into the **roombaWorld** project.

  Make sure you change the comment to include your name.

  Compile, **Build** and **Run** your program.

(0 points)

2 Limiting the world

Now, we will make the roombaWorld 11 squares in both directions, so that the x and y values must be between 0 and 10. For each of the exercises below, you will have to repeatedly edit the roombaWorld file, then compile, build and run it, until it does as described.

- Add error checking in the **moveNorth**, **moveSouth**, **moveEast**, **moveWest** methods so that roomba does not go out of the world (in other words so that x and y don't go above 10 or below 0). To do this you will need to add a check before you increment or decrement x and y.

  (*1 point*)

3 Wrapping the world around

- Now make it so that if roomba moves north off the north side of the world, it appears at the south side. Write a message to the screen when this happens.

  Make sure that your changes do not alter the error checking you just introduced.

  (*1 point*)

- Extend your answer to the previous question so that when roomba moves south off the south side of the world it appears on the north side, when it moves west off the west side of the world it appears on the east side, and when it moves east off the east side it reappears on the west side.

  Write a message to the screen when each of these things occurs.

  (*1 point*)

4 Leap to the edge

- Add a new behavior **moveNorthToEdge()** that causes roomba to move forward until it gets to the North edge of the world and then stop.
- Add similar behaviors **moveSouthToEdge()**, **moveEastToEdge()**, and **moveWestToEdge()** which move roomba to the edge of the world in the indicated direction.

- Extend the set of user commands so that the user can tell roomba to move forward until it gets to the edge of the world. Use the following letters:

Q	Quit the program
n	move roomba north one step
s	move roomba south one step
e	move roomba east one step
w	move roomba west one step
N	move roomba north to the edge of the world
S	move roomba south to the edge of the world
E	move roomba east to the edge of the world
W	move roomba west to the edge of the world

(*2 points*)

5 Finish up

- Save this last version of the code (the one with the new commands). This version of the project should be submitted as the second part of your first homework/lab assignment.

  This version of the code should contain *all* of the changes you have made during these exercises — you will only get credit for the things that I can see you have done. You can use comments to make sure that I see all these things.

6 Submit Lab I

- Gather up your final version of the roombaWorld program (remember, just the `.cpp` and the programs you wrote for Part 1, and email them to Prof Parsons, as described in the instructions for Part 1.

Appendix A

```
// An interactive world for roomba to play in.
//
// Simon Parsons
// September 7th 2008

//
// Include C++ library definitions

#include <iostream>
using namespace std;

//
// 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?

//
// Declare methods

void displayPosition()
{
  cout << "The robot is at location (";
  cout << x;
  cout << ",";
  cout << y;
  cout << ")\n";
}

void moveNorth()
{
  cout << "moving North...\n";
  y = y + 1;
}

void moveSouth()
{
  cout << "moving South...\n";
  y = y - 1;
}

void moveWest()
{
  cout << "moving West...\n";
  x = x - 1;
}

void moveEast()
{
  cout << "moving East...\n";
  x = x + 1;
}

//
// Define main method
```

```
int main()

{
  x = 0;                // Set variables
  y = 0;
  q = false;

  displayPosition();

  // We keep doing this bit

  while ( q==false ){

   // Get input from user

   cout << "Which way should the robot move (enter N,S,E,W or Q)? ";
   cin  >> c;
   cout << "You entered: ";
   cout << c;
   cout << "\n";

   // Depending on what the user entered, move the right way

   if ( c=='N' ) {
     moveNorth();
     displayPosition();
   }

   if ( c=='S' ) {
     moveSouth();
     displayPosition();
   }

   if ( c=='E' ) {
     moveEast();
     displayPosition();
   }

   if ( c=='W' ) {
     moveWest();
     displayPosition();
   }

   // If the user entered Q, set up so that we finish.

   if ( c=='Q' ) {
     q = true;
   }
  }

  cout << "Time to go!" << endl;
  return 0;
}
```