CIS 1.5 Fall 2009 Lab IV.3

1. Connect to the Scribbler

   Follow the instructions on the handout "How to connect to the Scribbler".

2. Remote control.

   Now run the second program in the folder `labIV.3`

   As with `test.cpp` you have to compile it and then run it. Type:

   ```
   make demo
   ./demo
   ```

   in the Terminal window.

   This program allows you to operate the robot by remote control. You can make the robot go forwards or backwards, you can make it turn, stop and beep.

   Drive the robot around a while.

3. Taking control

   Now you are ready to start programming the robot. You will write your code in the file `test.cpp`.

   First you need to edit the program. YOU CANNOT EDIT THIS PROGRAM IN CODE::BLOCKS (more correctly, using Code::Blocks in the same way as you have in the past, you won't be able to compile the program). Type:

   ```
   nano test.cpp
   ```

   in the Terminal window.

   This opens up an editor that allows you to modify the code in the file `test.cpp`. Hopefully this looks a lot like other C++ programs you have written in the past.

   Edit the program so that it makes the robot go forwards for a short time, then stops, then backwards for a short time.

   To do this you will have to tinker with the commands in `test.cpp` trying to get the robot to do what you want it to do.

   When you have something to try, save the program (instructions are in the bottom of the window), exit `nano`, and then type:

   ```
   make test
   ./test
   ```

   in the Terminal window.

   Don't be surprised if it takes several attempts to get the program to work correctly.

   HINT: you might want to use `nano` *to look at the code for* `demo.cpp` *to give you some ideas about what to do.*

   HINT: if you find that the robot is ignoring some of the commands that you send it, try adding in some additional "wait" commands to space the other commands out..

4. Back and forth

   Modify the program so that the robot goes forwards and backwards 3 times.

   Then modify the program to go forwards and backwards until the user enters `q`.

5. All square

   Now, modify your program so that the robot drives in a square.

   First, get the robot to drive straight and turn through 90 degrees.

   Then use a loop to make the robot repeat this four times.

   How far does the robot end up from where it started?

6. Functions

   Now, alter your square-driving program so that it includes two functions.

   One should make your robot drive straight. The other should make it turn through 90 degrees.

   Use these functions and a `for` loop to make the robot drive in a square.

   Modify the function that makes the robot turn so that it uses a parameter which controls how long the robot turns for.

7. Reusing functions.

   Modify your program so that it makes the robot drive in a square and then in a triangle.

   You should be able to make it drive in a triangle using just the functions that you already wrote.

   Now extend the program so that it offers the user a choice of either driving the robot in a square, or driving the robot in a triangle.