# project[3]: formatted input/output

Due Thursday, 2/18/2016, 12:59:59pm

## Project Goals

The objective for this assignment is to write a robot interface that will command a robot. Your completing this assignment demonstrates that you understand:

- Variables
- User interfaces
- `If-then` and `switch` statements
- Reasoning using logic

**Important Notes**:

1. **Formatting:** Make sure that you follow the precise recommendations for the output content and formatting: do not change the formatting of the menu, or the format of the display message under command (5) below. Your assignment will be auto-graded and any change in formatting will result in a loss in the grade.
2. **Comments:** Header comments are required on all files and recommended for the rest of the program. Points will be deducted if no header comments are included.

## Problem 1

In this assignment we assume that we have a robot living in a grid world of unit-size cells. The robot starts at position with coordinates (x,y) = (0,0), with heading = East, and can perform the following actions: **move forward** (advance one cell in the direction in which it is facing), **move backward** (go back one cell, keeping the same heading), and **turn left/right** by 90 degrees. Your program should run in an infinite loop, at each step prompting the user with a menu of possible actions, as follows:

```
#This program controls the movement of a simulated robot.
(1)   Drive forward
(2)   Drive backward
(3)   Turn left
(4)   Turn right
(5)   Display robot position
(6)   Reset to start position
(7)   Quit program
Please select from the above options:
```

If the user enters an option that is not among the ones above, your program should print

```
Selected option not valid.
```

and then display the above menu again.

For each of the above options, your program should implement the appropriate computations in order to correctly update the current position of the robot. For example:

(1) **Drive forward**: the robot should move one position ahead in the direction it is facing. For example, from the start position (x,y) = (0,0), heading = East, if a forward command is given, the new location of the robot will be (x,y) = (1,0), heading = East. However, if the robot's initial location would have been (x,y) = (0,0), heading = West, after a forward command, the new location of the robot will be (x,y) = (-1,0), heading = West.
(2) **Drive backward**: the robot should move one backward, while maintaining the direction it is facing. For example, from the start position (x,y) = (0,0), heading = East, if a forward command is given, the new location of the robot will be (x,y) = (-1,0), heading = East.
(3) **Turn left**: the robot should turn left 90 degrees. For example, if the robot's heading is currently West, after a turn left command, the robot's new heading will be South. The (x,y) coordinates are not changing during a turn command.
(4) **Turn right**: the robot should turn right 90 degrees. For example, if the robot's heading is currently West, after a turn left command, the robot's new heading will be North. The (x,y) coordinates are not changing during a turn command.
(5) **Display robot position**: this command will display the current location and heading of the robot in the following format:

```
#Robot location is (2, 3); Heading: East
```

(6) **Reset to start position**: this command resets the robot's position to the initial values (x,y) = (0,0), heading = East.
(7) **Quit program**: ends the program.

**Constraints:** your program should use a `switch` statement for handling the user menu commands.

**Challenge:** For the drive forward and drive backward commands, your program should ask the user to enter a number of steps (integer) that the robot should advance forward or backward, as follows:

```
#This program controls the movement of a simulated robot.
   (1)      Drive forward
   (2)      Drive backward
   (3)      Turn left
   (4)      Turn right
   (5)      Display robot position
   (6)      Reset to start position
   (7)      Quit program
Please select from the above options: 1
Please enter the number of steps: 5
```

Your program should take into account the number of steps entered by the user in order to properly update the position of the robot.

Your program should be saved in a file called `robot.c`.

## Grading Rubric

Grading will be done for each problem as follows:

| | |
|---|---|
| **Correctly-named file** | 10% |
| **Header comment** | 4% |
| **Program compiles** | 10% |
| **Correctly-reading data from terminal** | 36% |
| **Correct result printed** | 40% |

## Submission details

**The project needs to be submitted, Tuesday, 2/18/2016, 12:59:59pm (before class).**

To submit your project, you will have to save your project files to an ECC machine using the Linux VM or the nomachine client:

- create a directory called “project3”
- save your *.c files in that directory
- save your description file into that directory
- DO THIS ONCE: Install the submission script ***(don’t type the ‘>’ symbols)***
  > cd ~
  > wget http://www.cse.unr.edu/~newellz2/submit
  > chmod +x ./submit
- TO Submit:
  > cd project3
  > ~/submit

The submission script copies all files in the current directory to our directory. You may submit as many times as you like before the deadline, we only keep the last submission.

# Academic Honesty

Academic dishonesty is against university as well as the system community standards. Academic dishonesty includes, but is not limited to, the following:

Plagiarism: defined as submitting the language, ideas, thoughts or work of another as one's own; or assisting in the act of plagiarism by allowing one's work to be used in this fashion.

Cheating: defined as (1) obtaining or providing unauthorized information during an examination through verbal, visual or unauthorized use of books, notes, text and other materials; (2) obtaining or providing information concerning all or part of an examination prior to that examination; (3) taking an examination for another student, or arranging for another person to take an exam in one's place; (4) altering or changing test answers after submittal for grading, grades after grades have been awarded, or other academic records once these are official.

Cheating, plagiarism or otherwise obtaining grades under false pretenses" constitute academic dishonesty according to the code of this university. Academic dishonesty will not be tolerated and penalties can include canceling a student's enrollment without a grade, giving an F for the course, or for the assignment. For more details, see the University of Nevada, Reno General Catalog.