Introduction to Programming

CMPT 120

Project 2 – User Interaction – 100 points

Deadline

Final push no later than 8AM Monday on 16 October 2023

Goals

To continue development of your "robot" simulator term project by adding user interaction to control the robot and move it around an empty room. To reinforce your growing Git and Python skills.

Preparation

For this assignment, you must have already completed the initial version of your project hosted in the *{YourLastname}-Work* repository of our class organization on GitHub.com. Open your Python script in the VSCode IDE to continue editing your code.

Instructions

Create the next version of your project by implementing the required features listed below.

★ *Commit #1 – Draw the robot in the empty room.* **15pts**
- After prompting the user to begin, draw the room again but this time with the letter "R" representing the robot printed at the correct position within the room.

★ *Commit #2 – Display the robot's operating instructions to the user.* **10pts**
- Print a list of valid commands that the user can enter to control the robot.
  - We haven't implemented these yet, but valid commands for the robot will be: `forward`, `reverse`, `turn left`, `turn right`, and `quit`.
- Also update your README file to list the valid robot commands.

★ *Commit #3 – Read and distinguish commands until the user quits.* **15pts**
- Use a loop to repeatedly read commands from the user via keyboard input.
- If the user enters the `quit` command, then break out of the loop. You can now remove the code from version one that prompts the user to press <Enter> to quit.
- If the user enters any other command, print a message for that command (e.g., "Robot moves forward.") and then draw the room again.
- If the user does not enter a valid command, then simply tell them so.

★ *Commit #4 – Implement turning left and right.* **20pts**
- Add a variable to keep track of which direction the robot is facing.
  - You may start the robot in any direction you like.
- Update the robot's direction in response to commands from the user.
- Each time you print the room, also print a sentence stating which direction the robot is facing (e.g., "The robot is facing North").

★ *Commit #5 – Implement moving forward and backward.* **20pts**
- Use variables to keep track of the robot's x and y position if you're not already.
- Update the robot's position in response to commands from the user.
  - If the user is facing east/west, then you must change its x-position.
  - If the user is facing north/south, then you must change its y-position.

★ *Commit #6 – Prevent the robot from leaving the room.* **20pts**
- Finally, check the robot's position before moving it.
- If honoring the user's command would place the robot on a wall or outside the room, then do not move the robot. Instead, report the collision.

Advice

Follow coding best practices with mnemonic naming, consistent indentation and spacing, frequent comments., and appropriate use of user-defined functions.

Be sure to stage your changes a little at a time and commit frequently. Push any local work to your remote GitHub repository regularly. Don't forget to:

- Write short but meaningful messages for every commit. (Tip: Consider using the instructions themselves as your commit messages.)
- Look at the differences between successive versions of your code. You can do this on the GitHub website as well as in GitHub Desktop app itself.

Test, test, test… and test again. Then test some more. When you think you've tested enough, go back and test yet again. Then get someone else to test for you while you test theirs. Etc.

Submitting

You must push your changes to GitHub before the due date.

Note: Pushing regularly will reduce the risk of losing your work, so do not wait until after you have made all changes and commits before pushing.