This programming assignments is based on one developed by Prof. Nathan Sprague of Computer Science at James Madison University for his course, CS 354: Autonomous Robots given in Fall semester 2017.

CSCI 4972/6972: Algorithmic Robotics
Programming Assignment Zero (PA0)
Deadline: 2pm GMT-05 January 24, 2019

Goal

The purpose of this programming assignment is to help you get up to speed with the Python programming language.

Part 1: Warm-up

Get started by completing the online tutorials in `http://www.learnpython.org`. You should complete all of the tutorials in the "Learn the Basics Section" with the exception of "Classes and Objects."

Next, carefully read the tutorial, "Improve Your Python: Python Classes and Object Oriented Programming" through the "Class Methods" section. .

Feel free to skim the tutorials above if you are already comfortable programming in Python. If you are new to Python, I strongly encourage you to work through each step of the tutorials carefully. It will probably take a couple of hours, but the time spent will pay dividends throughout the semester.

Part 2: RoboRally

For this portion of the project you will complete a simplified version of the RoboRally board game. The following excerpt from the RoboRally Wikipedia page describes the game:

> *In RoboRally, players assume control of one of many "Robot Control Computers" in a dangerous widget factory filled with moving, course-altering conveyor belts, metal-melting laser beams, bottomless pits, crushers, and a variety of other obstacles; the concept is that this is what the control computers do for fun after hours. The goal in a game of RoboRally is, apart from survival, to be the first to reach a pre-designated number of checkpoints in a particular order. However, the real difficulty in RoboRally is movement, which is accomplished with the randomly dealt program cards.*
>
> *The program cards specify movement, such as move one space forward, turn left or U-turn. The cards have to be arranged by the player in the specific manner they wish the robot to move. Each player receives up to nine cards each turn. They use five of the cards to specify their robot's movement for the given turn, playing each card face down into one of five available "registers". All robots move simultaneously, each player revealing each register in turn. Robots attempting to move into the same space at the same time are resolved by priority numbers printed on the cards. A robot moving into an occupied space "pushes" the blocking robot (even more than one of them) into the next space.*

For the purposes of this assignment, you don't need to worry about board features (pits, conveyor belts etc.) or checkpoints. You only need to provide the code that handles robot movements and interactions. Specifically, you must complete the unfinished methods in the RoboRally class in the file roborally.py. The file roborally_test.py includes unit tests that will be used to grade your submission.

You may add as many additional instance variables and helper functions as you like, but you must not modify the existing method signatures or public attributes in the provided classes. You do not need to worry about error handling.

Submitting

Unless I get a better submission solution in place, email roborally.py to trinkle@gmail.com before the deadline. Your code should be well organized and clearly documented. Make sure to include an acknowledgment statement describing any resources that you used or assistance that you received.

Grading

Submissions will be graded according to the following rubric:

Passes Unit Tests	85%
Readability and approximate conformance with PEP8	15%

Note that the provided unit tests evaluate both the correctness and efficiency of your solution. Your solution should easily scale to handle thousands, or tens of thousands, of cards and robots. This will require you to consider algorithmic efficiency in developing your solution and to be careful in your selection of Python collection types