CSCI 445 Programming Assignment 1 — Python

Prof. Ayanian, University of Southern California Spring 2017

Python is a great programming language because it is easy-to-use, powerful, and flexible. Furthermore, it is very popular and widely used in practice[1]. Nevertheless, not all of you have experience with the language, yet. In this programming assignment you will learn (or refresh) the basics of Python. During the later labs, you will apply this knowledge to robotics problems.

1 Read About Python

A good starting point is the official python tutorial: https://docs.python.org/3.6/tutorial/. As far as our lab is concerned, you only need to know the following sections: 2,3,4.1-4.6,4.8,5,6,9.1-9.4. Of course, there are many other resources available, including books, online classes, and video tutorials. You can find a list here: https://wiki.python.org/moin/BeginnersGuide.

2 Functions and Control Flow

Create a new file named `task1.py` and add the following content:

1. Write a function `mysum1` that takes a list of integers and returns the sum of integers. Your function should use a for-loop internally. Test your function with the following inputs: `[1,5,7]`, `[]`, and `[-5,3]` and print the results to the screen for each of those examples.
2. Write a function `myfib1` which computes the nth Fibonacci number:

$$F(n) = \begin{cases} F(n-1) + F(n-2) & n > 1 \\ n & n \leq 1 \end{cases} \quad \text{for } n \geq 0$$

   Your function can be iterative or recursive. Test your function with the inputs 3, 5, and 7 and print the results to the screen for each of those inputs.

3 Classes and Modules

Classes and modules help you to write more modular code. If used during the lab, this will allow you to re-use a lot of your code in your final project.

In the first lab, you demonstrated how to use motion primitives on the robot to let it move in various ways. In this exercise, you will make your code more generic.

1. Modify your solution of Lab1 to use more generic functions. In particular, you can create the following member functions of the `Run` class: `forward(distance, speed)`, `backward(distance, speed)`, `turn_left(duration, speed)`, `turn_right(duration, speed)`, `stop()`. The speed should be in meters per second, the distance in meters, and the duration in seconds. You can test your code in simulation (similar to Lab1).
2. We now move the code to its own class. This way you can just re-use the class, rather than copying the function if you need motion primitives in a future lab.
   (a) Create a new file `my_robot.py`.
   (b) Add a new class `MyRobot` and move the motion primitive functions to that class.
   (c) Change `lab1.py` to use your new class rather than the locally defined functions.

[1] http://spectrum.ieee.org/computing/software/the-2016-top-programming-languages

3. One issue with your current solution is that you still need to specify the speed for each motion primitive. It would be nicer if you could specify the `base_speed` once and then all function use that `base_speed` by default. Change your code accordingly.

4 SciPy

In engineering, Python is frequently used together with packages from `SciPy`[2], which provide basic routines for scientific computation. It includes `numpy` for basic linear algebra and random numbers, as well as `matplotlib` for data visualization. Your Python distribution, Anaconda, includes SciPy already; hence, you do not need to install any additional software.

1. Create a new file `task3.py` and import numpy using `import numpy as np`.
2. Write a function `mysum2` that takes a list of integers and returns the sum of the integers. Use a `numpy`-function rather than a for-loop and test with the same inputs as you did for `mysum1`.
3. Write a function `plotcircle1` that creates two arrays of data points for x and y coordinates of a circle (Hint: You can use trigonometric functions to produce such points). Plot those arrays using an xy-plot.
4. Write a function `plotnorm1` that randomly generates 10000 numbers from a Normal distribution and plots the data as a histogram with 20 bins. You can use https://docs.scipy.org/doc/numpy/reference/generated/numpy.random.normal.html for the generation and the `hist`-function of `matplotlib` (http://matplotlib.org/api/pyplot_api.html).

5 Submission

Please create a zip-file named `firstname_lastname.zip` containing `task1.py`, `my_robot.py`, `lab1.py`, and `task3.py` and submit the archive using BlackBoard. All files should be in a working condition, i.e. we should be able to execute your code on our side using Python3 during grading.

[2] https://www.scipy.org/