Homework for Python Basic

2024 Fall

Overview

Due October 8, 22:00, UTC+08:00

This homework includes some small programs of Python basic usage. You are asked to complete the code in the source files. In the following, we will provide instructions for each question. After finishing all the questions, screenshot the output of `python autograder.py`. Please **zip the whole folder incuding the screenshot** (make sure you have completed the missing code) and name it as `${student¯id}.zip`, e.g., `2023012345.zip`. You should submit thezip file to Web Learning before the deadline.

You should edit or run the following files:

- `addition.py`: source file for question 1
- `buyLotsOfFruit.py`: source file for question 2
- `shop.py`: source file for question 3
- `shopSmart.py`: source file for question 3
- `autograder.py`: autograding script

Other files are related to `autograding` system. You can ignore them.

1 Python installation guide (with Micromamba)

1.1 Install Micromamba

The following content is excerpted from the official guide.

For Linux, macOS, or Git Bash on Windows install with:

```
"${SHELL}" <(curl -L micro.mamba.pm/install.sh)
```

1.2 Install Python 3.11

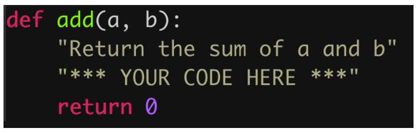
Create an environment with Python 3.11 by:

```
micromamba create -y -n aihw -c conda-forge python=3.11
```

Then activate the environment by:

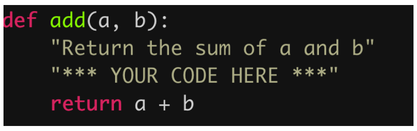
```
micromamba activate aihw
```

2 Q1: Addition (0 pt)



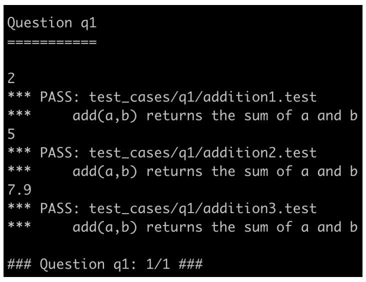
```
def add(a, b):
    "Return the sum of a and b"
    "*** YOUR CODE HERE ***"
    return 0
```

(a)



```
def add(a, b):
    "Return the sum of a and b"
    "*** YOUR CODE HERE ***"
    return a + b
```

(b)



```
Question q1
===========

2
*** PASS: test_cases/q1/addition1.test
***     add(a,b) returns the sum of a and b
5
*** PASS: test_cases/q1/addition2.test
***     add(a,b) returns the sum of a and b
7.9
*** PASS: test_cases/q1/addition3.test
***     add(a,b) returns the sum of a and b

### Question q1: 1/1 ###
```

(c)

Open `addition.py` and look at the definition of `add` as in Figure 1a. The tests called this with a and b set to different values, but the code always returned zero. Modify this definition to read as in Figure 1b to implement the addition. You can run `python autograder.py` to test your code. The output should be like Figure 1c. Ignore SyntaxWarning on Windows.

3 Q2: buyLotsOfFruit function (0.5 pt)

Implement the `buyLotsOfFruit(orderList)` function in `buyLotsOfFruit.py` which takes a list of `(fruit,numPounds)` tuples and returns the cost of your list. The `fruitPrices` is a dictionary with the fruit names as keys and their prices (per pound). Please do not change the `fruitPrices` variable.

Run python `autograder.py` until question 2 passes all tests, and you get full marks. Each test will confirm that `buyLotsOfFruit(orderList)` returns the correct answer given various possible inputs. For example, `test-cases/q2/food-price1.test` tests whether: Cost of `[('apples', 2.0), ('pears', 3.0), ('limes', 4.0)]` is 12.25

4 Q3: shopSmart function (0.5 pt)

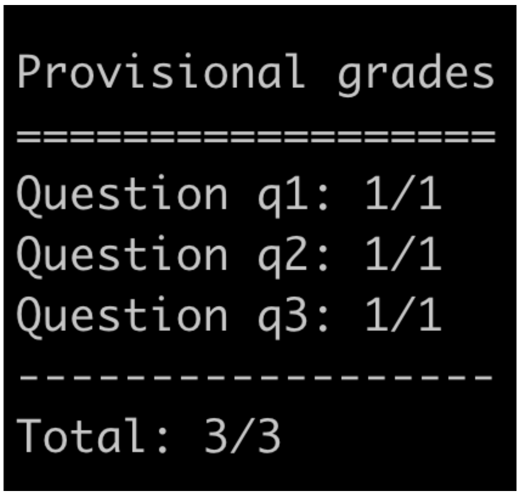
Fill in the function `shopSmart(orderList, fruitShops)` in `shopSmart.py`, which takes an orderList (like the kind passed in to `FruitShop.getPriceOfOrder`) and a list of `FruitShop` and returns the `FruitShop` where your order costs the least amount in total. Note that we will provide the `shop.py` implementation as a “support” file, where we define the class `FruitShop`, so you don’t need to submit yours. Don’t change the file name or variable names, please.

Run `autograder.py` until question 3 passes all tests, and you get full marks. Each test will confirm that `shopSmart(orderList,fruitShops)` returns the correct answer given various possible inputs. For example, with the following variable definitions:

```
orders1 = [('apples', 1.0), ('oranges', 3.0)]
orders2 = [('apples', 3.0)]
dir1 = {'apples': 2.0, 'oranges': 1.0}
shop1 =  shop.FruitShop('shop1',dir1)
dir2 = {'apples': 1.0, 'oranges': 5.0}
shop2 = shop.FruitShop('shop2', dir2)
shops = [shop1, shop2]
```

- `test_cases/q3/select_shop1.test` tests whether:
  `shopSmart.shopSmart(orders1, shops) == shop1`

- `test_cases/q3/select_shop2.test` tests whether:
  `shopSmart.shopSmart(orders2, shops) == shop2`



```
Provisional grades
==================
Question q1: 1/1
Question q2: 1/1
Question q3: 1/1
------------------
Total: 3/3
```

Figure 2

5 Submission

After finishing this assignment, please run `python autograder.py`. Save the screenshot of output like Figure 2 in the folder. Please zip the whole folder, including your modification to the questions' source codes. Then name your zip archive as `${student id}.zip`, e.g., `2023012345.zip`. Please submit it to the homework 0 page on Web Learning before **October 8, 22:00, UTC+08:00**.

6 Grading

Homework 0 is worth 2 points. Once you successfully submit your assignment to Web Learner before the deadline, you will get 1 point. Q2 and Q3 are worth 0.5 point each, constituting the remaining 1 point. The TAs will run your codes to check if all test cases pass.

We will actively be checking for code plagiarism. If plagiarism is found, your grade **for this course** will be **F**.

7 Acknowledgment

This assignment references UC Berkely courses. The programs and autograder are developed based on Pacman AI Project by UC Berkely.