Homework 1

To get you familiarized with the autograder, Python and PyCharm you will code, test, and submit solutions for three questions.

You can download all of the files associated with this assignment as a zip archive: homework1.zip. Unzip this file and examine the contents of the folder:

addition.py
autograder.py
buyLotsOfFruit.py
grading.py
projectParams.py
shop.py
shopSmart.py
testClasses.py
testParser.py
test_cases
tutorialTestClasses.py
util.py

The folder contains a number of files you'll edit or run:

addition.py: source file you will edit for question 1
buyLotsOfFruit.py: source file you will edit for question 2
shop.py: source file you need to understand for question 3
shopSmart.py: source file you will edit for question 3
autograder.py: autograding script (see below)

and others you can ignore:

test_cases: directory contains the test cases for each question
grading.py: autograder code
testClasses.py: autograder code
tutorialTestClasses.py: test classes for this particular project
projectParams.py: project parameters

Running the Python program autograder.py grades your solution to all three questions. If we run it before editing any files we get a page or two of output:

```
Starting on 1-24 at 11:11:45

Question q1
===========

*** FAIL: test_cases\q1\addition1.test
***        add(a,b) must return the sum of a and b
***        student result: "0"
***        correct result: "2"
*** FAIL: test_cases\q1\addition2.test
***        add(a,b) must return the sum of a and b
***        student result: "0"
***        correct result: "5"
*** FAIL: test_cases\q1\addition3.test
***        add(a,b) must return the sum of a and b
***        student result: "0"
***        correct result: "7.9"
*** Tests failed.

### Question q1: 0/1 ###


Question q2
===========

*** FAIL: test_cases\q2\food_price1.test
***        buyLotsOfFruit must compute the correct cost of the order
***        student result: "0.0"
***        correct result: "12.25"
*** FAIL: test_cases\q2\food_price2.test
***        buyLotsOfFruit must compute the correct cost of the order
***        student result: "0.0"
***        correct result: "14.75"
*** FAIL: test_cases\q2\food_price3.test
***        buyLotsOfFruit must compute the correct cost of the order
```

```
***     student result: "0.0"
***     correct result: "6.4375"
*** Tests failed.

### Question q2: 0/1 ###


Question q3
===========

Welcome to shop1 fruit shop
Welcome to shop2 fruit shop
*** FAIL: test_cases\q3\select_shop1.test
***     shopSmart(order, shops) must select the cheapest shop
***     student result: "None"
***     correct result: "<FruitShop: shop1>"
Welcome to shop1 fruit shop
Welcome to shop2 fruit shop
*** FAIL: test_cases\q3\select_shop2.test
***     shopSmart(order, shops) must select the cheapest shop
***     student result: "None"
***     correct result: "<FruitShop: shop2>"
Welcome to shop1 fruit shop
Welcome to shop2 fruit shop
Welcome to shop3 fruit shop
*** FAIL: test_cases\q3\select_shop3.test
***     shopSmart(order, shops) must select the cheapest shop
***     student result: "None"
***     correct result: "<FruitShop: shop3>"
*** Tests failed.

### Question q3: 0/1 ###


Finished at 11:11:45

Provisional grades
==================
Question q1: 0/1
```

```
Question q2: 0/1
Question q3: 0/1
------------------
Total: 0/3

Your grades are NOT yet registered.  To register your grades, make sure
to follow your instructor's guidelines to receive credit on your project.
```

For each of the three questions, this shows the results of that question's tests, the questions grade, and a final summary at the end. Because you haven't yet solved the questions, all the tests fail. As you solve each question you may find some tests pass while other fail. When all tests pass for a question, you get full marks.

Looking at the results for question 1, you can see that it has failed three tests with the error message "add(a,b) must return the sum of a and b". The answer your code gives is always 0, but the correct answer is different. We'll fix that next.

Question 1: Addition

Open addition.py and look at the definition of the *add* function:

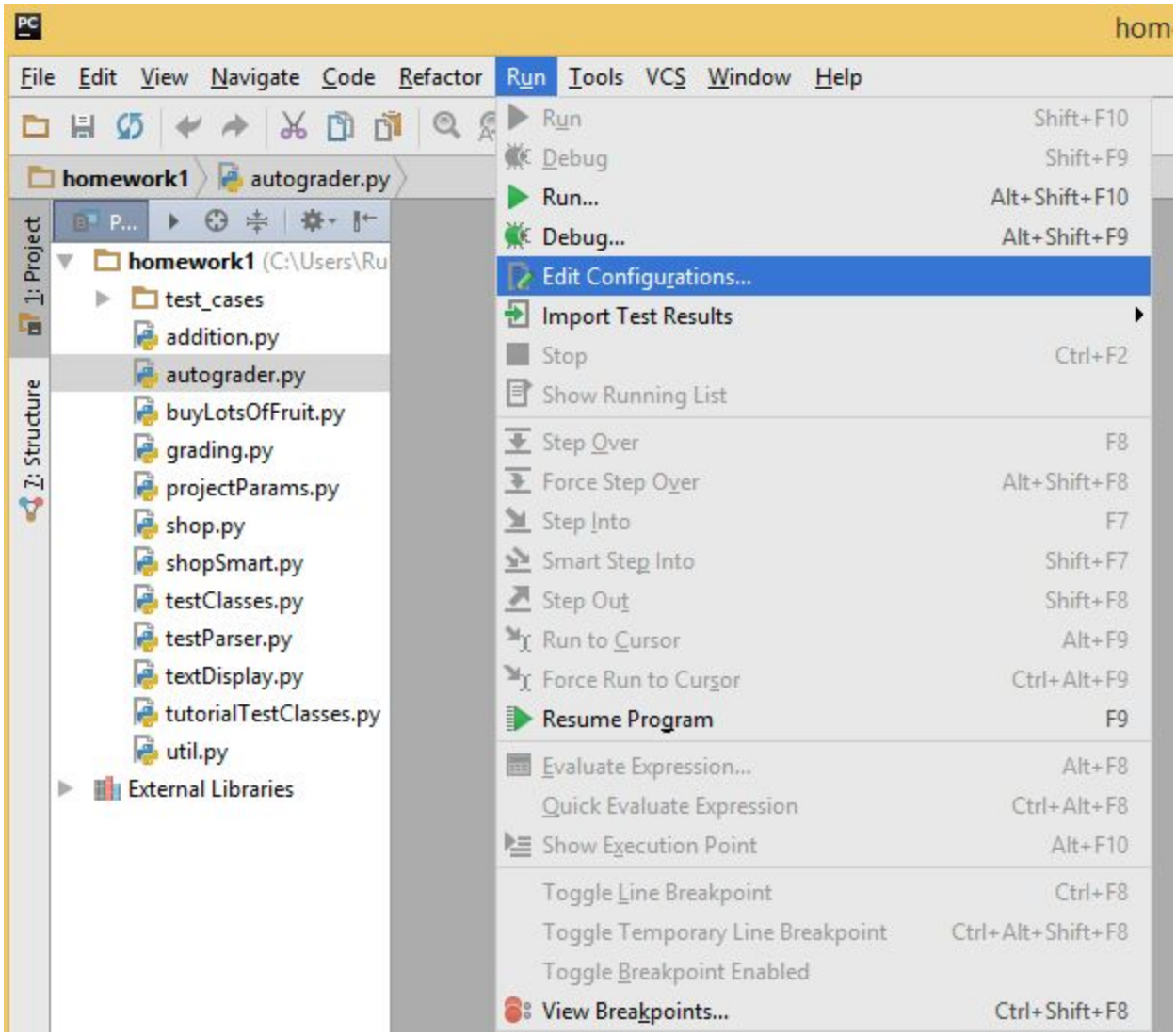
```
def add(a, b):
  "Return the sum of a and b"
  "*** YOUR CODE HERE ***"
  return 0
```

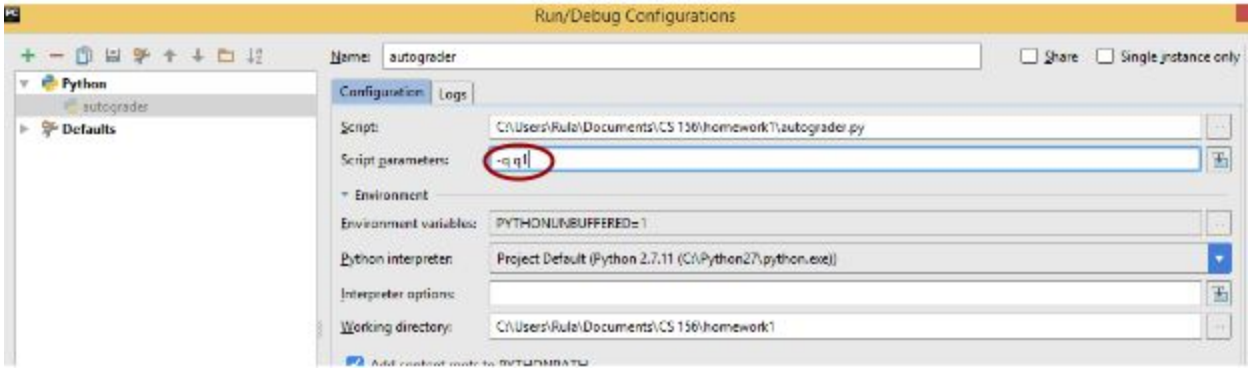
The tests called this function with a and b set to different values, but the code always returned zero. Modify this definition to read:

```
def add(a, b):
    "Return the sum of a and b"
    print "Passed a=%s and b=%s, returning a+b=%s" % (a,b,a+b)
    return a+b
```

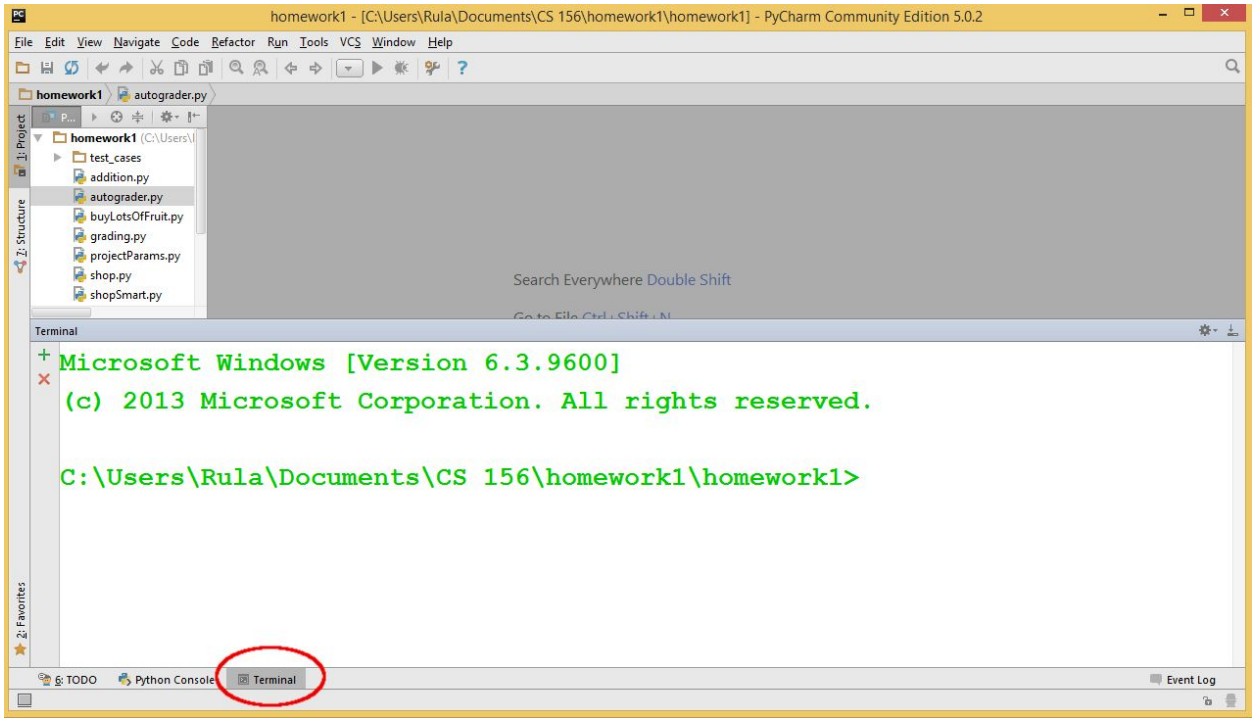
Now rerun the autograder. We can omit the results of questions 2 and 3 by specifying -q q1 as arguments to autograder.py. We can do that in two ways:

1. By editing the Run Configuration corresponding to autograder.py and specifying -q q1 as the script parameters in PyCharm.





2. By invoking autograder from the terminal window and typing:
   python autograder.py -q q1
   The easiest way to open a terminal window from PyCharm is by clicking on 'Terminal' at the bottom of the PyCharm window as shown below. You can also press ALT-F12 on Windows or press option-fn-F12 on a Mac OS.



Here's the expected output:

```
Starting on 1-21 at 23:52:05

Question q1
===========
Passed a=1 and b=1, returning a+b=2
*** PASS: test_cases/q1/addition1.test
***      add(a,b) returns the sum of a and b
Passed a=2 and b=3, returning a+b=5
*** PASS: test_cases/q1/addition2.test
***      add(a,b) returns the sum of a and b
Passed a=10 and b=-2.1, returning a+b=7.9
*** PASS: test_cases/q1/addition3.test
***      add(a,b) returns the sum of a and b

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

Finished at 23:41:01

Provisional grades
==================
Question q1: 1/1
Question q2: 0/1
Question q3: 0/1
------------------
Total: 1/3
```

You now pass all tests, getting full marks for question 1. Notice the new lines "Passed a=..." which appear before "*** PASS: ...". These are produced by the print statement in add. You can use print statements like that to output information useful for debugging. You can also run the autograder with the option --mute to temporarily hide such lines, as follows:

```
python autograder.py -q q1 --mute
Starting on 1-22 at 14:15:33

Question q1
===========
*** PASS: test_cases/q1/addition1.test
```

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

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

Question 2: buyLotsOfFruit function

Add a buyLotsOfFruit(orderList) function to buyLotsOfFruit.py which takes a list of (fruit,pound) tuples and returns the cost of your list. If there is some fruit in the list which doesn't appear in fruitPrices it should print an error message and return None. 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

Question 3: shopSmart function

Fill in the function shopSmart(orders,shops) in shopSmart.py, which takes an orderList (like the kind passed in toFruitShop.getPriceOfOrder) and a list of FruitShop and returns the FruitShop where your order costs the least amount in total. Don't change the file name or variable names, please. Note that we will provide the shop.py implementation as a "support" file, so you don't need to submit yours. Use the methods defined in shop.py to simplify your task.

Run python autograder.py until question 3 passes all tests and you get full marks. Each test will confirm that shopSmart(orders,shops)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

and test_cases/q3/select_shop2.test tests whether:

shopSmart.shopSmart(orders2, shops) == shop2

Submission

You're not done yet! Upload the following modified files to get credit on your project:

Question 1: addition.py
Question 2: buyLotsOfFruit.py
Question 3: shopSmart.py