Project 0: Python, Setup & 3Qs

Due: **17:00, July 14th, 2025**

TABLE OF CONTENTS

Introduction

The projects for this class assume you use Python 3.6 or higher.

Project 0 will cover the following:

- Instructions on how to set up the right Python version
- Workflow examples
- A mini-Python tutorial

Files to Edit and Submit: You will fill in portions of `addition.py`, `buyLotsOfFruit.py`, and `shopSmart.py` in tutorial.zip . Please do not change **the other files** in this distribution or submit any of the files; do not change **the names** of any provided functions or classes within the code.

Python Installation

Many of you may not have Python 3.6 or higher already installed on your computers. Conda is an easy way to manage many different environments, each with its own Python versions and dependencies. This allows us to avoid conflicts between our preferred Python version and that of other classes.

Workflow/ Setup Examples

You are not expected to use a particular code editor or anything, but here are some suggestions on convenient workflows (you can skim and choose the one that looks better to you):

- Getting Started with Python in VS Code.

Python Basics

If you're new to Python or need a refresher, we recommend going through the Python basics tutorial.

Q1: Addition

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

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

The tests called this 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
```

Q2: buyLotsOfFruit function

Implement the `buyLotsOfFruit(orderList)` function in `buyLotsOfFruit.py` which takes a list of `(fruit,numPounds)` 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 several tests. 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
```

Q3: shopSmart function

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. 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.

Run several tests. 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`

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

Submission

Submit your Project 0 assignment (Python files) in one zip file. File name is your student number, for example 1234567.zip if your student ID is s1234567. Do not submit unnecessary files.