Department of Computer Science and Engineering
Islamic University of Technology (IUT)
A subsidiary organ of OIC

Laboratory Report: Lab 0

CSE 4712: Artificial Intelligence Lab

Name: Tasfia Tasneem Annesha
Student ID: 190041220
Section: 2B
Semester: 7th
Academic Year: 2021-2022

Date of Submission: 8/8/2023

Task 1:

Working code:

```python
# addition.py
"""
Run python autograder.py
"""


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

    return a+b

```

Explanation:

This task was very easy. The goal was to only return the sum of the passed parameters. Returning a+b, where a and b are the arguments passed to the addition function, solves the problem in a straightforward manner.

```
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 ###
```

Task 2:

```
def buyLotsOfFruit(orderList):
    """ orderList: List of (fruit, numPounds) tuples
    Returns cost of order
    """
    totalCost = 0.0
    "*** YOUR CODE HERE ***"
    for fruit, numPounds in orderList:
        if fruit in fruitPrices:
            totalCost += numPounds * fruitPrices [fruit]
        else:
            print('Error')
            return None
    return totalCost
```

Analysis of the problem:

Here I had to figure out the overall price of all the fruits the customer purchased for this task. Here we are provided the function with an orderList list that contains the items and their corresponding quantities. Each item's price was listed in the fruitPrices dictionary.

Solution of the Problem:

The solution was as easy as iterating through each item in the **orderList** and using the **orderList** and **fruitPrices** to calculate the order's total cost. **Fruit** in the loop makes reference to the fruits listed in the **orderList**. **fruit** contains the fruit's name, which serves as the **fruitPrices** dictionary's key. The amount of the item in question is **numPounds**. **FruitPrices[fruit]** is used to find the price of a fruit, and its cost is calculated by multiplying it by the fruit's quantity. The **totalCost** value is returned after we calculate the price of each fruit and add them together.

```
Question q2
===========

Error
*** PASS: test_cases/q2/food_price1.test
***     buyLotsOfFruit correctly computes the cost of the order
*** PASS: test_cases/q2/food_price2.test
***     buyLotsOfFruit correctly computes the cost of the order
*** PASS: test_cases/q2/food_price3.test
***     buyLotsOfFruit correctly computes the cost of the order

### Question q2: 1/1 ###
```

Task 3:

```
def shopSmart(orderList, fruitShops):
    """   orderList: List of (fruit, numPound) tuples
        fruitShops: List of FruitShops """
    "*** YOUR CODE HERE ***"
    bestshop= None
    bestCost= float('inf')
    for shop in fruitShops:
        totalCost = 0.0
        totalCost += shop.getPriceOfOrder (orderList)
        if totalCost < bestCost:
            bestCost = totalCost
            bestshop=shop
    return bestshop
```

Explanation of the Question:

This question was very similar to the second. However, there was a further step. In this task, we had to choose the shop that would charge the least for our order from a list of shops.

Solution of the problem:

Here I first declared the **bestCost** variable infinity. Then using a for loop I have found the **totalCost** of each shop. If the **totalCost** is smaller than the **bestCost** then the bestCost is assigned totalCost. The **bestshop** value is returned after that.

```
Question q3
===========

Welcome to shop1 fruit shop
Welcome to shop2 fruit shop
*** PASS: test_cases/q3/select_shop1.test
***     shopSmart(order, shops) selects the cheapest shop
Welcome to shop1 fruit shop
Welcome to shop2 fruit shop
*** PASS: test_cases/q3/select_shop2.test
***     shopSmart(order, shops) selects the cheapest shop
Welcome to shop1 fruit shop
Welcome to shop2 fruit shop
Welcome to shop3 fruit shop
*** PASS: test_cases/q3/select_shop3.test
***     shopSmart(order, shops) selects the cheapest shop

### Question q3: 1/1 ###
```

Final Result: