Department of Computer Science and Engineering

Islamic University of Technology (IUT)

A subsidiary organ of OIC

CSE 4618: Artificial Intelligence

Lab Report

Name	**:**	**M M Nazmul Hossain**
Student ID	**:**	**200042118**
Semester	**:**	**6th**
Academic Year	**:**	**2022-2023**
Date of Submission	**:**	**30.01.2024**

Problem-1 (Addition)

Analysis

The first problem, addition.py, was to define a function that takes in two parameters, a and b. It was instructed to return the summation of a and b.

Solution

Just return a+b.

Explanation

The first problem was just introductory and it was also solved by Sir.

Challenges

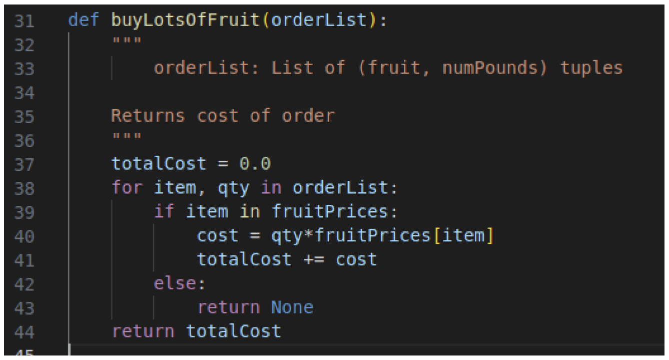
No challenges were faced while doing this as it was solved by Sir.

Problem-2 (Buy Lots of Fruits)

Analysis

The second problem, buyLotsOfFruit.py, was to define a function that takes in an orderList which is an array of pairs, item_name, and their quantity. A fruitPrices list was also provided which is a list of item_names and their prices. It was instructed to return the cost of the order. It was also instructed to return None in cases where the fruitPrices list does not contain the fruit item in the orderList.

Solution



```python
def buyLotsOfFruit(orderList):
    """
        orderList: List of (fruit, numPounds) tuples

    Returns cost of order
    """
    totalCost = 0.0
    for item, qty in orderList:
        if item in fruitPrices:
            cost = qty*fruitPrices[item]
            totalCost += cost
        else:
            return None
    return totalCost
```

Explanation

The basic implementation of the problem is to traverse the orderList using a loop and using the prices of the item from the fruitPrices array to save the cost of the provided amount/quantity of that item in a previously initiated to 0 variable which will contain the totalCost of the order which is updated after each iteration of the loop.

Challenges

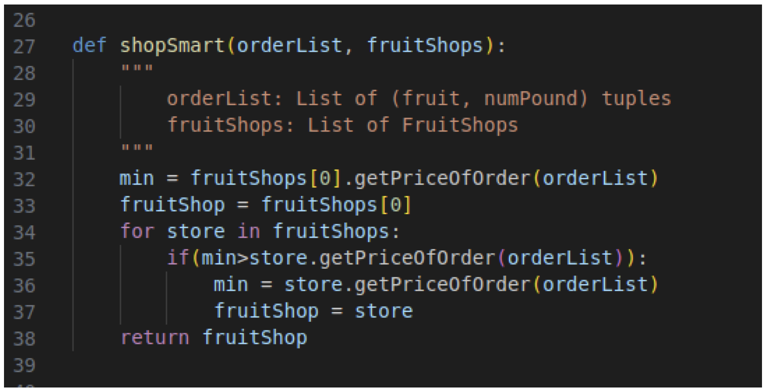
The main challenge of the problem was that in one of the test cases, the fruitPrices list didn’t contain one of the fruits in the orderList. In those cases, None had to be returned. It was actually mentioned in the document, which I neglected to read at first, which led to me taking too long to solve the problem. After looking through the test cases, I saw that one of the cases was different from the rest, and from there I knew what had to be done.

Problem-3 (Shop Smart)

Analysis

The third and final problem of the lab, shopSmart.py, was to define a function that takes in an orderList and a fruitShops List as input parameters. The orderList is the same as before, a list of pairs, item_name, and quantity. The fruitShops were a list of shops, which contained their list of fruit item_names and their prices.

Solution



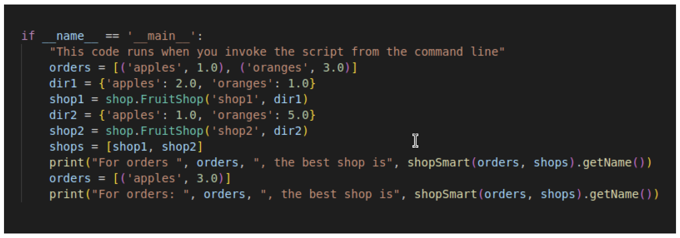
```
def shopSmart(orderList, fruitShops):
    """
        orderList: List of (fruit, numPound) tuples
        fruitShops: List of FruitShops
    """
    min = fruitShops[0].getPriceOfOrder(orderList)
    fruitShop = fruitShops[0]
    for store in fruitShops:
        if(min>store.getPriceOfOrder(orderList)):
            min = store.getPriceOfOrder(orderList)
            fruitShop = store
    return fruitShop
```

Explanation

This problem was immensely easy if approached correctly. First, the shop had to be examined, which disclosed that it contained predefined functions for calculating the cost of the order using the orderList “getPriceOfOrder” using which, the process becomes much simpler. Next, there can be two approaches. Saving Int_Max in a min variable and iterating across all the shops to replace the minFruitShop with the shop that had the minimum cost and the min variable with that cost. However, to counter that, some test cases had immensely large integer values. So, that approach failed. After that, the second approach was attempted. In the min variable, the priceOfOrder of the first fruitShop was inserted and the minFruitShop was set to the first fruit shop. After that, the same iteration was carried out and finally, the minFruitShop was returned.

Challenges

The challenge I faced with this problem, which led to me not being able to submit my answer despite solving the problem in its entirety during the lab, was not reading the main function.



```python
if __name__ == '__main__':
    "This code runs when you invoke the script from the command line"
    orders = [('apples', 1.0), ('oranges', 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]
    print("For orders ", orders, ", the best shop is", shopSmart(orders, shops).getName())
    orders = [('apples', 3.0)]
    print("For orders: ", orders, ", the best shop is", shopSmart(orders, shops).getName())
```

I thought we had to return the shop name, so I was returning fruitShop.getName() which was constantly returning an error that I could not solve in time. When I saw the error later, I wanted to give myself a facepalm. All I had to do differently was just return to the shop and the problem was solved all by itself. Truly, one of the dumbest moments. To be fair, python doesn’t specify the return type, so I can not be blamed.