

Muhammad Rizwan Khalid

BSCS – 6A

180459

Task: 01:

The image shows two windows side-by-side. The left window is a code editor titled "addition.py - C:\Users\M Rizwan\Desktop\tutorial\addition.py". It contains Python code with comments explaining licensing and attribution. The right window is a Python Shell titled "Python 2.7.15 Shell" with the command "python addition.py" run, showing the script's output.

```
# addition.py
# -----
# Licensing Information: You are free to use or
# educational purposes provided that (1) you
# retain this notice, and (2) you do not
# change or remove this notice.
# Attribution Information: The Pacman AI project
# The core projects and autograders were primarily created by John DeNero
# (denero@cs.berkeley.edu) and Dan Klein (klein@cs.berkeley.edu).
# Student side autograding was added by Brad Miller, Nick Hay, and
# Pieter Abbeel (pabbeel@cs.berkeley.edu).

"""
Run python autograder.py
"""

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


print add('this', ' that')


```

```
Python 2.7.15 (v2.7.15:ca079a3ea3, Apr 30 2018, 16:30:26) [MSC v.1500 64 bit (AM
D64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:\Users\M Rizwan\Desktop\tutorial\addition.py ======
this that
>>>
```

Task: 02:

The image shows two windows side-by-side. The left window is a code editor titled "buyLotsOfFruit.py - C:\Users\M Rizwan\Desktop\tutorial\buyLotsOfFruit.py (2.7.15)". It contains Python code for calculating the cost of a fruit order. The right window is a Python Shell titled "Python 2.7.15 Shell" with the command "python buyLotsOfFruit.py" run, showing the script's output.

```
# Attribution Information: The Pacman AI projects were developed at UC Berkeley.
# The core projects and autograders were primarily created by John DeNero
# (denero@cs.berkeley.edu) and Dan Klein (klein@cs.berkeley.edu).
# Student side autograding was added by Brad Miller, Nick Hay, and
# Pieter Abbeel (pabbeel@cs.berkeley.edu).

"""
To run this script, type
python buyLotsOfFruit.py

Once you have correctly implemented the buyLotsOfFruit function,
the script should produce the output:
"""

Cost of [('apples', 2.0), ('pears', 3.0), ('limes', 4.0)] is 12.25

fruitPrices = {'apples':2.00, 'oranges': 1.50, 'pears': 1.75,
               'limes':0.75, 'strawberries':1.00}

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

    Returns cost of order
    """
    totalCost = 0.0
    for fruit , no in orderList:
        totalCost += fruitPrices[fruit] * no
    *** YOUR CODE HERE ***
    return totalCost

# Main Method
if __name__ == '__main__':
    "This code runs when you invoke the script from the command line"
    orderList = [ ('apples', 2.0), ('pears', 3.0), ('limes', 4.0) ]
    print 'Cost of', orderList, 'is', buyLotsOfFruit(orderList)
```

```
Python 2.7.15 (v2.7.15:ca079a3ea3, Apr 30 2018, 16:30:26) [MSC v.1500 64 bit (AM
D64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:\Users\M Rizwan\Desktop\tutorial\buyLotsOfFruit.py ======
Cost of [('apples', 2.0), ('pears', 3.0), ('limes', 4.0)] is 12.25
>>>
```

Task: 3:

The screenshot shows a Windows desktop with two open windows. On the left is a Notepad window titled "shopSmart.py - C:\Users\M Rizwan\Desktop\tutorial\shopSmart.py (2.7.15)". It contains Python code for a script named shopSmart. The code includes comments, imports, function definitions, and a main block. On the right is a "Python 2.7.15 Shell" window. It shows the Python interpreter running the script, displaying welcome messages, order details, and the output of the shopSmart function, which identifies the best shop based on price.

```
shopSmart.py - C:\Users\M Rizwan\Desktop\tutorial\shopSmart.py (2.7.15)
File Edit Format Run Options Window Help

"""
Here's the intended output of this script, once you fill it in.

Welcome to shop1 fruit shop
Welcome to shop2 fruit shop
For orders: [('apples', 1.0), ('oranges', 3.0)] best shop is shop1
For orders: [('apples', 3.0)] best shop is shop2
"""

import shop

def shopSmart(orderList, fruitShops):
    """
        orderList: List of (fruit, numPound) tuples
        fruitShops: List of FruitShops
    """
    *** YOUR CODE HERE ***
    best = dict()
    for shop in fruitShops:
        price = 0
        for fruit, no in orderList:
            price += shop.fruitPrices[fruit] * no
        best[shop] = price
    return min(best, key=best.get)

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]
    shopSmart(orders, shops)
    print "For orders ", orders, ", the best shop is", shopSmart(orders, shops)
    orders = [('apples',3.0)]
    print "For orders: ", orders, ", the best shop is", shopSmart(orders, shops)

Python 2.7.15 Shell
File Edit Shell Debug Options Window Help
Python 2.7.15 (v2.7.15:ca079a3ea3, Apr 30 2018, 16:30:26) [MSC v.1500 64 bit (AMD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:\Users\M Rizwan\Desktop\tutorial\shopSmart.py ======
Welcome to shop1 fruit shop
Welcome to shop2 fruit shop
For orders  [('apples', 1.0), ('oranges', 3.0)], the best shop is shop1
For orders:  [('apples', 3.0)], the best shop is shop2
>>> |
```