University of Windsor
School of Computer Science
https://cs.uwindsor.ca

Master of Applied Computing
COMP-8347
Internet Applications and Distributed Systems

Dr. Adel Abusitta – adel.abusitta@uwindsor.ca

LAB 2 – Writing a Complete Script in Python

Part 1 – Working with date and time types in Python

Tasks:

a. Write a Python program that prompts the user to enter a date in the format "MM/DD/YYYY", and then converts it to a datetime object.

Answer:

```
import datetime

#Take input from user
date_in=input("Enter date in MM/DD/YYYY format: ")

#convert to datetime object

date_obj=datetime.datetime.strptime(date_in, "%m/%d/%Y")

print("date_object: ",date_obj)
```

b. Write a Python function that takes a datetime object and formats it as a string in the format "YYYY-MM-DDTHH:MM:SSZ".

Answer:

```
#convert format of datetime_obj
date_obj_update=date_obj.strftime("%Y-%m-%dT%H:%M:%SZ")

print("updated datetime object: ",date_obj_update)
```

c. Write a Python function that takes a datetime object and returns the date and time as separate strings.

Answer:

```
from datetime import datetime,timedelta
datestr = input('enter datetime as MM/DD/YYYY HH:MM:ss ')

x2 = datetime.strptime(datestr, '%m/%d/%Y %H:%M:%S')
print('date :'+x2.strftime("%x")+'\ntime :'+x2.strftime("%X"))
```

d. Write a Python program that prompts the user to enter two dates in the format "MM/DD/YYYY", and calculates the number of days between them using the datetime library.

Answer:

```
from datetime import datetime,timedelta

date1 = input("Enter first date MM/DD/YYYY")
date2 = input("Enter second date MM/DD/YYYY")

x1 = datetime.strptime(date1, '%m/%d/%Y').date()
x2 = datetime.strptime(date2, '%m/%d/%Y').date()

diff = x1 - x2

print('number of days between two dates is {}'.format(diff.days))
```

e. Write a Python function that takes a datetime object and adds a specified number of days to it.

Answer:

```
from datetime import datetime,timedelta

date_str = input('Enter date in format MM/DD/YYYY')
x = datetime.strptime(date_str, '%m/%d/%Y').date()
print(x)

add = int(input('Number of days to add') )
new_date = x + timedelta(days=add)
print(new_date)
```

Part 2: Write a complete Python script, with comments, to do the following:

a. Open a text file called "*catalog.txt*", attached with this lab, for reading. The file contains the items available in a fitness studio, the items categories/classes, and their quantities.

b. Define a list of strings called ***fit_items***. The list should contain at least 10 strings and each string represent a specific fitness item, e.g., treadmill, lifting bars, weights, etc.

c. Loop over each element in ***fit_items*** and check if that element matches any of the products in the file.

d. If there is a match, save the category and the quantity corresponding to that item in some variables.

e. Create a dict ***d1*** with entries ***item:category*** where ***item*** **(key)** is the item (string) found in *catalog.txt* and ***category*** (**value**) is the category of that item. Add the item and its category to ***d1*** as {***item:category***}. Create another dict ***d2*** with entries ***item:quantity*** and add the item found and its quantity to ***d2***.

f. Next the program should ask the user to enter a string ***s, representing a fitness item,*** as an input and retrieve the *category* of ***s*** from ***d1*** and the *quantity* from ***d2*** .

• After displaying the category and quantity corresponding to item ***s***, the program asks if the user would like to do another search with (***yes***/***no***) options.

• If the user enters ***yes***, another category and quantity retrieval should be done for another item.

• If the answer is ***no***, the program should exit.

g. If the item's name entered by the user does not correspond to a valid key, the program should catch an exception. When the exception occurs, display an appropriate error message then prompt the user to input another item's name.

Answer script for part 2:

```
# File open and handelled exeception
try:
  with open('catalog.txt', 'r') as catalogFile:
    catalogLines = catalogFile.readlines()
except FileNotFoundError:
  print("Unable to Open the file")
  quit()

# declaring list to search items
fitItems = ['treadmill', 'lifting bars', 'exercise bikes', 'weights', 'rigs', 'dumbbells', 'gym rats', 'steppers',
'exercise balls', 'rowing machine']

# declaring d1 and d2 for category and quantity
d1 = {}
d2 = {}

# Loop over each fitness item
for item in fitItems:
    # Loop over each line in the catalog file
    for index, line in enumerate(catalogLines, start=0):
        # Check if the item matches the item name in the catalog file
        itemSearched = line.replace("\n", "")
        if item == itemSearched:
            # Save the category and quantity
            category = catalogLines[index+1]
            quantity = catalogLines[index+2]
            category = category.replace("\n", "")
            quantity = quantity.replace("\n", "")
            d1.update({item: category})
            d2.update({item: quantity})
            break

# Section to get user input
continueFlag = True
# continue till user enter no
while continueFlag:
  userInput = input("Enter the Item you want to search: \n")
  # searching key in category and quantity
  try:
    category = d1[userInput]
    quantity = d2[userInput]
    print(f"The Category for {userInput} is '{category}' and the quantity is '{quantity}'\n")
  except KeyError:
    print("Item Not Found\n")
  # check if user wants to continue
  checkExecute = input("Do you want to continue ? Enter yes or no \n")
  if(checkExecute == "yes"):
    continueFlag = True
  else:
    continueFlag = False
```

Part 3: Answer the following questions:
a. Assume that you have a list named L, e.g., L = [19, 52, 87, 2, 8, 11, 18, 22], write a Python script to count the number of odd numbers in L?

Answer:

```
def count_number_odd_numbers_in_list(L):
    count_odd_numbers = 0
    for element in L:
        if element % 2 != 0:
            count_odd_numbers += 1
    return count_odd_numbers


print(count_number_odd_numbers_in_list([19, 52, 87, 2, 8, 11, 18, 22]))
```

b. Assume that you have a list named L, e.g., L = [38, 5, 7, 2, 8,112,18,400], write a Python script to calculate the average of all even numbers in L?

Answer:

```
def calculate_avg_even_numbers_in_list(L):
    count_even_numbers, sum = 0, 0
    for element in L:
        if element % 2 == 0:
            count_even_numbers += 1
            sum += element
    return sum / count_even_numbers


print(calculate_avg_even_numbers_in_list([38, 5, 7, 2, 8, 112, 18, 400]))
```

c. Assume that you have a list named L, e.g., L = [38, 5, 7, 2, 8,112,18,400], write a Python script to find the largest and the smallest number in L?

Answer:

```
def calculate_largest_and_smallest_numbers_in_list(L):
    smallest, largest = L[0], L[0]
    for element in L:
        if element > largest:
            largest = element
        if element < smallest:
            smallest = element
    print(f"Smallest number: {smallest}")
    print(f"Largest number: {largest}")
    return smallest, largest


print(calculate_largest_and_smallest_numbers_in_list([38, 5, 7, 2, 8, 112, 18, 400]))
```