LONDON METROPOLITAN UNIVERSITY

islington college
(इस्लिङ्टन कलेज)

Module Code & Module Title

CS4051NI Fundamentals of Computing

Assessment Weightage & Type

60% Individual Coursework

Year and Semester

2021-22 Summer

Student Name: Utsarga K.C

Group:C9

London Met ID: 21049619

College ID: np01cp4a210280

Assignment Due Date: 26th August 2022

Assignment Submission Date: 26th August 2022

I confirm that I understand my coursework needs to be submitted online via Google Classroom under the relevant module page before the deadline in order for my assignment to be accepted and marked. I am fully aware that late submissions will be treated as non-submission and a marks of zero will be awarded.

Table of Contents

List of Figures

List of Tables

Introduction

In the project, we were given a task to create a python program for a costume rental shop that maintains information about the various available costumes in a text file.
An application needs to be created that will read the text file and display every costume that is available for renting. Then, after every transaction (such as renting a costume or costumes), a note or invoice should be created for that specific client and stored in a text file. Following each transaction, the costume stock should also be updated in the text file that has the costumes' details. If a costume needs to be returned, a new note or invoice should be created for the client returning the costume. Also, the stock is to be updated.
We used various concepts of Python programming like data structures, Object-oriented programming, Exception handling, use of lists and dictionaries, file handling, functions, etc. to complete this project. These concepts were taught to us during our semester and through this project we learned their implementation.

The tools used to complete this project are:

- IDLE
- MS-Word
- Draw.io

Algorithm

An algorithm is a finite set of instructions that is used to define a step-by-step solution to a problem. The use of an algorithm is especially very helpful in programming as also essentially a step-by-step process of problem solving. Algorithms are usually made before writing the code for a program.
In this program, the algorithm shows the complete structure and flow of the program for renting and returning different costumes.

Step 1: START
Step 2: Display the three available options for renting, returning or exiting the program and ask the user to choose one of the options.
Step 3: If the user enters “1” go to step 6.
Step 4: If the user enters “2” go to step 34.
Step 5: If the user enters “3” go to step 69.
Step 6: Start the renting process by calling the rent function from rentFunction class.
Step 7: Call the function to create list from the .txt file and then call the method to created dictionary for the list.
Step 8: Ask the user for the serial number of the costume they want to rent.
Step 9: Check if the input serial number is above 0 and within the existing serial numbers.
Step 10: If the above condition is false go to step 11 else go to step 8.
Step 11: Use try and except to check if the serial number is input in integer format or not.
Step 12: If an exception occurs go to step 8 else go to step 13.
Step 13: Ask the user to input the quantity of the costume they want to rent.
Step 14: Check if the input quantity is greater than 0 and less than or equal to the available quantity.
Step 15: If the above statement is true go to step 16 else go to step 13.
Step 16: Use try and except to check if the quantity is input in integer format or not.
Step 17: If an exception occurs go to step 12 else go to step 16.

Step 18: Update the value of quantity in the dictionary by subtracting the user's input from the quantity in the dictionary.
Step 19: Add the name of the costume to a list called CostumeName.
Step 20: Add the price of the costume from the main dictionary to a variable called totalPrice.
Step 21: Open the .txt file of costumes and update the new quantity.
Step 22: Ask the user if they want to rent more costumes or not.
Step 23: while "yes" go to step 24 else break and go to step 26.
Step 24: Ask the user to input the quantity of the costume they want to rent.
Step 25: Do the same as from Step 14 to Step 21.
Step 26: Ask the user to input their name.
Step 27: Ask the user to input their phone number.
Step 28: Check if the costumer's name input is empty and phone number is 0 or not.
Step 29: If above statement is true go to step 26 else go to step 30.
Step 30: Use try and except to check if the phone number is in int format or not.
Step 31: If an exception occurs go to step 26 else go to step 32.
Step 32: Open a new .txt file in which the invoice will be written.
Step 33: Write the necessary invoice details in the file such as; Costumer's name and number, costumes' names, total price, etc.
Step 34: Start the returning process by calling the return function from returnFunction class.
Step 35: Call the function to create list from the .txt file and then call the method to created dictionary for the list.
Step 36: Ask the user for the serial number of the costume they want to return.
Step 37: Check if the input serial number is above 0 and within the existing serial numbers.
Step 38: If the above condition is true go to step 39 else go to step 36.
Step 39: Use try and except to check if the serial number is input in integer format or not.
Step 40: If an exception occurs go to step 36 else go to step 41.
Step 41: Ask the user to input the quantity of the costume they want to return.

Step 42: Check if the input quantity is greater than 0.

Step 43: If the above statement is true go to step 44 else go to step 41.

Step 44: Use try and except to check if the quantity is input in integer format or not.

Step 45: If an exception occurs go to step 41 else go to step 46.

Step 46: Update the value of quantity in the dictionary by adding the user's input to the quantity in the dictionary.

Step 47: Ask the user to input the number of days.

Step 48: If number of days is equal to zero go to step 47 else go to step 49.

Step 49: Use try and except to check if input number of days is in correct format or not.

Step 50: If an exception occurs go to step 47 else go to step 51.

Step 51: Add the name of the costume to a list called CostumeName.

Step 52: Add the price of the costume from the main dictionary to a variable called totalPrice.

Step 54: Open the .txt file of costumes and update the new quantity.

Step 55: If number of days is greater the 5 days add fine for those days.

Step 56: Add the fine to total price to calculate total price with fine.

Step 57: Ask the user if they want to return more costumes or not.

Step 58: while "yes" go to step 57 else break and go to step 59.

Step 59: Ask the user to input the quantity of the costume they want to rent.

Step 60: Do the same as from Step 42 to Step 56.

Step 61: Ask the user to input their name.

Step 62: Ask the user to input their phone number.

Step 63: Check if the costumer's name input is empty and phone number is 0 or not.

Step 64: If above statement is true go to step 61 else go to step 65.

Step 65: Use try and except to check if the phone number is in int format or not.

Step 66: If an exception occurs go to step 61 else go to step 67.

Step 67: Open a new .txt file in which the invoice will be written.

Step 68: Write the necessary invoice details in the file such as; Costumer's name and number, costumes' names, total price, fine, price with fine, date and time, etc.

Step 69: Display Thank You Message and exit the program.

Step 70: END

Flowchart

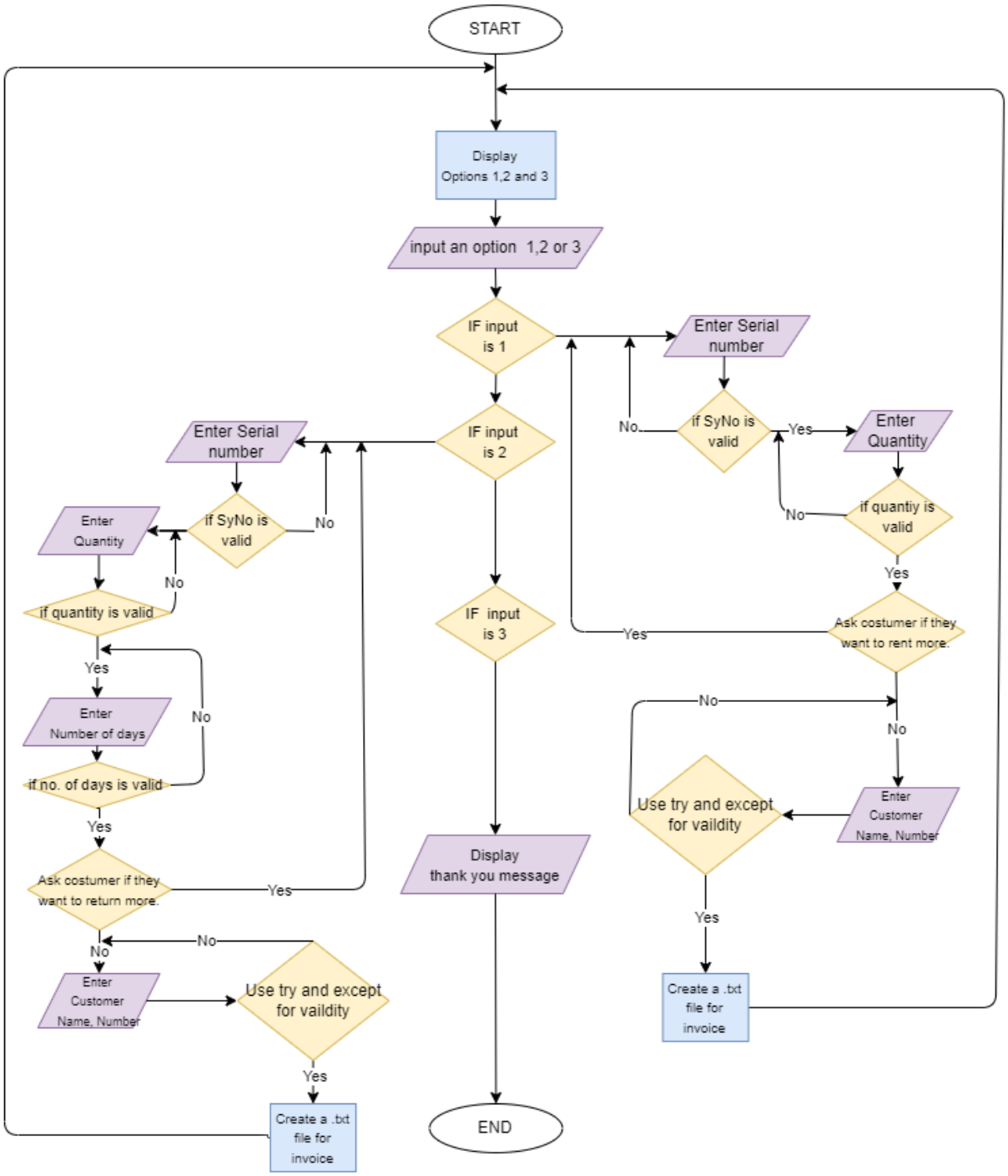

Figure 1: Flowchart

Pseudocode

Pseudo code is essentially an informal approach to write the code in order to properly grasp the program. Pseudocode is written in a simplified language and lacks a specific syntax. Before we even begin writing the code, it helps to outline the program's flow and final outcome. It can also be used to spot any errors in the program.

MainInterpace.py

```
from rentFunction IMPORT rent
from ReturnFunction IMPORT Return


OUTPUT('''

++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

        Welcome to costume rental application

++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++


    ''')

SET Exit TO False
WHILE Exit==False:
    OUTPUT("Select a desireable option")
    OUTPUT('''
  Press 1 to rent costumes.
  Press 2 to RETURN costumes.
  Press 3 to exit.''')
    SET option TO INPUT("Enter an option: ")

    IF option EQUALS "1":
        rent()

    ELSEIF option EQUALS "2":
        Return()

    ELSEIF option EQUALS "3":
        SET Exit TO True
        OUTPUT('''

++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

```
        Thank You FOR using the rental service

++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

''')
    ELSE:
        OUTPUT('''
Invalid INPUT!!!!
Please select a value as per the provided options.
''')
```

Cusfunctions.py

```
DEFINE FUNCTION getCostumesInFile():
SET file TO open("costume.txt","r")
    SET costumeData TO file.readlines()
    file.close()
    RETURN costumeData


DEFINE FUNCTION costumeDictionary(costumesInFile):
    SET costumeData TO {}
    FOR index IN range(len(costumesInFile)):
        SET costumeData[index+1] TO costumesInFile[index].replace("\n","").split(",")
    RETURN costumeData


DEFINE FUNCTION costumesTable():
    SET costumesInFile TO getCostumesInFile()
    SET tableData TO costumeDictionary(costumesInFile)
    OUTPUT("S.No.","\t","Costume Name","\t\t","Brand","\t\t","Price","\t","Stock")

OUTPUT("==========================================================
====")
    FOR key, costume IN tableData.items():
        OUTPUT(key,"\t",costume[0],"\t\t",costume[1],"\t",costume[2],"\t",costume[3])
    OUTPUT("")


DEFINE FUNCTION Get_dateTime():
    IMPORT datetime
    SET datetime TO datetime.datetime.now()
    RETURN str(datetime)
```

RentFunction.py

```
from datetime IMPORT datetime
from Cusfunctions IMPORT getCostumesInFile, costumeDictionary, costumesTable,
Get_dateTime

DEFINE FUNCTION selectCosToRent():
   SET costumesInFile TO getCostumesInFile()
   SET tableData TO costumeDictionary(costumesInFile)
   SET validSyNo TO False
   WHILE validSyNo EQUALS False:
      TRY:
         SET SyNo TO int(INPUT("Enter the serial number of the Costume you want to
rent: "))
         IF SyNo is greater than 0 and SyNo less than or equal to len(tableData):
            SET validSyNo TO True
            SET a TO tableData [SyNo]
            OUTPUT("S.No.","\t","Costume Name","\t\t","Brand","\t\t","Price","\t","Stock")

OUTPUT("==========================================================
====")
            OUTPUT(SyNo,"\t",a[0],"\t\t",a[1],"\t",a[2],"\t",a[3])
            OUTPUT("")
            RETURN SyNo
         ELSE:
            OUTPUT("Invalid Symbol Number!!!")
      EXCEPT:
         OUTPUT("")
         OUTPUT("Please INPUT Serial number IN valid format.")
         OUTPUT("")

DEFINE FUNCTION quantityToRent(SyNo):
   SET costumesInFile TO getCostumesInFile()
   SET tableData TO costumeDictionary(costumesInFile)
   SET validstock TO False
   WHILE validstock EQUALS False:
      TRY:
         SET quantity TO int(INPUT("Enter the quantity of the item you have selected: "))
         IF quantity is greater than 0 and quantity less than or equal to int(tableData
[SyNo][3]):
            RETURN quantity
         ELSEIF quantity EQUALS 0:
            OUTPUT("Costume not available FOR rent")
         ELSE:
```

```
            OUTPUT("Quantity limit out of stock!!!")
        EXCEPT:
            OUTPUT("")
            OUTPUT("Please INPUT quantity IN valid format.")
            OUTPUT("")


DEFINE FUNCTION rent():
    OUTPUT('''
    Let's rent a costume.''')
    SET CostumeName TO an empty list
    SET totalPrice TO 0
    SET costumesInFile TO getCostumesInFile()
    SET tableData TO costumeDictionary(costumesInFile)
    costumesTable()
    SET SyNo TO selectCosToRent()
    SET quantity TO quantityToRent(SyNo)

    SET tableData [SyNo] [3] TO str(int(tableData[SyNo] [3]) - quantity)
    Open “Costume.txt” in Cosfile in wrie mode
    FOR key, costume IN tableData.items():
        SET write_data TO
str(costume[0])+","+str(costume[1])+","+str(costume[2])+","+str(costume[3])+"\n"
        Write in lines of .txt file Cosfile(write_data)
    Close the file Cosfile

    CostumeName.append (tableData [SyNo] [0])
    SET price TO tableData [SyNo] [2]

    SET totalPrice TO totalPrice + float(price.replace('$','')) * quantity

    OUTPUT("S.No.","\t","Costume Name","\t\t","Brand","\t\t","Price","\t","Stock")

OUTPUT("==========================================================
====")
    FOR key, costume IN tableData.items():
        OUTPUT(key,"\t",costume[0],"\t\t",costume[1],"\t",costume[2],"\t",costume[3])
    OUTPUT("")

    SET continueRenting TO True
    WHILE continueRenting EQUALS True:
        SET addCus TO INPUT("Press 'y' IF you want to rent another costume press any
other key to continue.. ")
        IF addCus EQUALS "y":
            SET SyNo TO selectCosToRent()
            SET quantity TO quantityToRent(SyNo)
```

```
        SET tableData [SyNo] [3] TO str(int(tableData[SyNo] [3]) - quantity)

Open “Costume.txt” in Cosfile in wrie mode
   FOR key, costume IN tableData.items():
      SET write_data TO
str(costume[0])+","+str(costume[1])+","+str(costume[2])+","+str(costume[3])+"\n"
      Write in lines of .txt file Cosfile(write_data)
   Close the file Cosfile

        CostumeName.append (tableData [SyNo] [0])
        SET price TO tableData [SyNo] [2]
        SET totalPrice TO totalPrice + float(price.replace('$','')) * quantity
        OUTPUT("S.No.","\t","Costume Name","\t\t","Brand","\t\t","Price","\t","Stock")

OUTPUT("============================================================
====")
        FOR key, costume IN tableData.items():
           OUTPUT(key,"\t",costume[0],"\t\t",costume[1],"\t",costume[2],"\t",costume[3])
        OUTPUT("")
      ELSE:
        break

   SET redo TO True
   WHILE redo EQUALS True:
      TRY:
           SET CustomerName TO INPUT("Enter the customer's name: ")
           SET CustomerPhone TO int(INPUT("Enter the customer's phone number: "))
           IF CustomerName is empty or CustomerPhone EQUALS 0:
              SET redo TO True
              OUTPUT("Please fill the Customer's name and phone number.")
           ELSE:
              SET redo TO False
      EXCEPT:
        OUTPUT("Invalid phone number!!")
        SET redo TO True

   OUTPUT("'

=================================================================
===========

        *Invoice has been generated FOR Rented Costumes*

=================================================================
===========
```

```
    ''')

    #writing the invoice
    SET filename TO  "Invoice for-" + CustomerName +".txt"
    file= open(r"RentInvoices\+" + filename, "w+")
    file.write('''
=====================================================
          *Invoice  FOR Rented Costumes*
=====================================================
''')
    file.write("Customer Name: " + CustomerName + "\n")
    file.write("Customer Phone: " + str(CustomerPhone) + "\n")
    file.write("Costumes Rented: ")
    FOR x IN range(len(CostumeName)):
        file.write(CostumeName[x] + ",")
    file.write("\n" + "Total Price: "+ str(totalPrice)+"\n")

    SET DateTime TO Get_dateTime()
    file.write("Date and time of Rent: "+ DateTime+"\n")
    file.write("==============================x========================")
```

ReturnFunction.py

```
from Cusfunctions IMPORT getCostumesInFile, costumeDictionary,costumesTable,
Get_dateTime

DEFINE FUNCTION selectCosToReturn():
    SET costumesInFile TO getCostumesInFile()
    SET tableData TO costumeDictionary(costumesInFile)
    SET validSyNo TO False
    WHILE validSyNo EQUALS False:
        TRY:
            SET SyNo TO int(INPUT("Enter the serial number of the Costume you want to
RETURN: "))
            IF SyNo is greater than 0 and SyNo less than or equal to len(tableData)
                SET validSyNo TO True
                SET a TO tableData [SyNo]
                OUTPUT("S.No.","\t","Costume Name","\t\t","Brand","\t\t","Price","\t","Stock")

OUTPUT("==========================================================
====")
                OUTPUT(SyNo,"\t",a[0],"\t\t",a[1],"\t",a[2],"\t",a[3])
                OUTPUT("")
                RETURN SyNo
            ELSE:
                OUTPUT("Invalid Symbol Number!!!")
```

```
        EXCEPT:
            OUTPUT("")
            OUTPUT("Please INPUT serial number IN correct format.")
            OUTPUT("")

DEFINE FUNCTION quantityToReturn(SyNo):
    SET costumesInFile TO getCostumesInFile()
    SET tableData TO costumeDictionary(costumesInFile)
    SET validstock TO False
    WHILE validstock EQUALS False:
        TRY:
            SET quantity TO int(INPUT("Enter the quantity of the costume you want to
RETURN: "))
            IF quantity is grater than 0
                RETURN quantity
            ELSEIF quantity EQUALS 0:
                OUTPUT("Costume not available FOR rent")
            ELSE:
                OUTPUT("Quantity limit out of stock!!!")
        EXCEPT:
            OUTPUT("")
            OUTPUT("Please INPUT quantity IN correct format.")
            OUTPUT("")


DEFINE FUNCTION Return():
    OUTPUT("Let’s return the costumes below.")
    SET CostumeName TO an empty list
    SET fine TO 0
    SET totalPrice TO 0
    SET totalPriceWithFine TO 0
    SET costumesInFile TO getCostumesInFile()
    SET tableData TO costumeDictionary(costumesInFile)
    costumesTable(
    SET SyNo TO selectCosToReturn()
    SET quantity TO quantityToReturn(SyNo)

    SET validNodays TO False
    WHILE validNodays EQUALS False:
        TRY:
            SET noOfDays TO int(INPUT("Enter the number of days the costume has been
rented: "))
            IF noOfDays EQUALS 0:
                OUTPUT("Number of days cannot be zero. Please enter correct number of
days.")
            ELSE:
```

```
            SET validNodays TO True
        EXCEPT:
            OUTPUT("Please enter number of days IN correct format.")
    SET tableData [SyNo] [3] TO str(int(tableData[SyNo] [3]) + quantity)

    Open “Costume.txt” in Cosfile in wrie mode
    FOR key, costume IN tableData.items():
        SET write_data TO
str(costume[0])+","+str(costume[1])+","+str(costume[2])+","+str(costume[3])+"\n"
        Write in lines of .txt file Cosfile(write_data)
    Close the file Cosfile

    CostumeName.append (tableData [SyNo] [0])
    SET price TO tableData [SyNo] [2]
    SET totalPrice TO totalPrice + float(price.replace('$',"")) * quantity
    IF noOfDays is greater than 5:
        SET fine TO (noOfDays - 5) * ((3/100) * totalPrice)
        SET totalPriceWithFine TO totalPrice + fine
    ELSEIF noOfDays less than or equal to 5:
            SET totalPriceWithFine TO totalPrice
    OUTPUT("S.No.","\t","Costume Name","\t\t","Brand","\t\t","Price","\t","Stock")

OUTPUT("==========================================================
====")
    FOR key, costume IN tableData.items():
        OUTPUT(key,"\t",costume[0],"\t\t",costume[1],"\t",costume[2],"\t",costume[3])
    OUTPUT("")
    SET continueReturning TO True
    WHILE continueReturning EQUALS True:
        SET addCus TO INPUT("Press 'y' IF you want to RETURN another costume press
any other key to continue.. ")
        IF addCus EQUALS "y":
            SET SyNo TO selectCosToReturn()
            SET quantity TO quantityToReturn(SyNo)
            SET validNodays TO False
            WHILE validNodays EQUALS False:
                TRY:
                    SET noOfDays TO int(INPUT("Enter the number of days the costume has
been rented: "))
                    IF noOfDays EQUALS 0:
                        OUTPUT("Number of days cannot be zero. Please enter correct number
of days.")
                    ELSE:
                        SET validNodays TO True
                EXCEPT:
                    OUTPUT("Please enter number of days IN correct format.")
```

```
        SET tableData [SyNo] [3] TO str(int(tableData[SyNo] [3]) + quantity)

        Open “Costume.txt” in Cosfile in wrie mode
    FOR key, costume IN tableData.items():
      SET write_data TO
str(costume[0])+","+str(costume[1])+","+str(costume[2])+","+str(costume[3])+"\n"
      Write in lines of .txt file Cosfile(write_data)
    Close the file Cosfile

        CostumeName.append (tableData [SyNo] [0])
        SET price TO tableData [SyNo] [2]
        SET totalPrice TO totalPrice + float(price.replace('$','')) * quantity
        IF noOfDays is greater than 5:
          SET fine TO (noOfDays - 5) * ((3/100) * totalPrice)
          SET totalPriceWithFine TO totalPrice + fine
        ELSEIF noOfDays less than or equal to 5:
          SET totalPriceWithFine TO totalPrice
        OUTPUT("S.No.","\t","Costume Name","\t\t","Brand","\t\t","Price","\t","Stock")

OUTPUT("==========================================================
====")
        FOR key, costume IN tableData.items():
          OUTPUT(key,"\t",costume[0],"\t\t",costume[1],"\t",costume[2],"\t",costume[3])
        OUTPUT("")
      ELSE:
        break
    SET redo TO True
    WHILE redo EQUALS True:
      TRY:
        SET CustomerName TO INPUT("Enter the customer's name: ")
        SET CustomerPhone TO int(INPUT("Enter the customer's phone number: "))
        IF CustomerName is empty or CustomerPhone EQUALS 0:
          SET redo TO True
          OUTPUT("Please fill the Customer's name and phone number.")
        ELSE:
          SET redo TO False
      EXCEPT:
        OUTPUT("Invalid phone number!!")
        SET redo TO True


    OUTPUT("'
==================================================================
===============
        *Invoice has been generated  FOR Costumes RETURNed*
```

```
==================================================================
===============
    ''')
    #writing the invoice
    SET filename TO  "Invoice for-" + CustomerName +".txt"
    file= open(r"ReturnInvoices\+" + filename, "w+")
    file.write('''
=====================================================
            *Invoice FOR Costumes RETURNed*
=====================================================
''')
    file.write("Customer Name: " + CustomerName + "\n")
    file.write("Customer Phone: " + str(CustomerPhone) + "\n")
    file.write("Costumes Rented: ")
    FOR x IN range(len(CostumeName)):
        file.write(CostumeName[x] + ",")
    #file.write("Costumes Rented: " + CostumeName + "\n")
    file.write("\n" + "Total Fine: " + str(fine) + "\n")
    file.write("Total Price with fine: " + str(totalPriceWithFine) + "\n")
    SET DateTime TO Get_dateTime()
    file.write("Date and time of Return: "+ DateTime+"\n")
    file.write("==============================x========================")
```

Data Structures

In python, there are two types of data structures, primitive and non-primitive. Primitive data types are those data types that are relatively simpler and can be used to create other sophisticated data types. Integer, String, Boolean and Float are the primitive data types in Python. Non-primitive data types more sophisticated as they are not pre-defined and are defined by the user usually referring to an object.

Primitive Data types

Integer

The integer data type takes integer input and only stores non-fractional integer values. Particularly, an integer can be used to represent whole integers from negative infinity to infinity, such as 4, 5, or -1.

```
ry:
    SyNo = int(input("Enter the serial number of the Costume you want to rent: "))
    if SyNo > 0 and SyNo <= len(tableData):
```

Figure 2: Use of integer data type in the program

Float

The float (floating point number) data type takes fractional number also as input and stores with the decimal points. It can be used for rational values that often end in a decimal, such 1.11 or 3.14.

```
price = tableData [SyNo] [2]
totalPrice = totalPrice + float(price.replace('$','')) * quantity
print("S.No ","\t","Costume Name","\t\t","Brand","\t\t","Price","\t","Stock
```

Figure 3: Use of Float data type in the program

String

The string data type stores alphanumeric characters. In Python, strings are made by enclosing a group of characters in a pair of single or double quotes. Consider the words "cake," "cookie," etc.

```
  write_data = str(costume[0])+","+str(costume[1])+","+str(costume[2])+","+str(costume[3])+"\n"
  Cosfile.writelines(write_data)
```

Figure 4: Use of String data type in the program

Boolean

The Boolean data type takes input and stores only in “True” or “False”. Booleans are useful for comparisons, conditional expressions and looping statements.

```
print("")
continueRenting = True
while continueRenting == True:
    addCus = input("Press 'y' if you want to rent another costume press any other key to continue.. ")
```

Figure 5: Use of Boolean data type in the program

Non-primitive Data types

Arrays

Arrays in python are a compact collection of basic data types and are not used very often. The elements stored in an array have a constrained data type which is specified during array creation.

Lists

In python, lists are groups or collections of heterogenous items. Lists can be identified by the square brackets “[ ]” that contain the elements of the list. Lists are mutable, meaning that their content can be changed whilst not changing their identity.

```
print('''
Let's rent a costume.''')
CostumeName = []
totalPrice = 0
```

Figure 6: Use of lists in the program

Dictionaries

In dictionaries in python, the data are stored in key, value pairs. Dictionaries are ordered and mutable as well. Dictionaries can be recognized by curly brackets “{ }” that enclose their elements.

```
def costumeDictionary(costumesInFile):
    costumeData = {}
    for index in range(len(costumesInFile)):
        costumeData[index+1] = costumesInFile[index].replace("\n","").split(",")
```

Figure 7: Use of Dictionary in the program

Tuples

Tuples are used in python to store multiple items in a single variable. Tuple are ordered and non-mutable.

Testing

Here five distinctive tests have been carried out in order to check the functionality and exception handling of the program.

Test 1

Show implementation of try, except:

- Provide invalid input and show the message.

Objective	To show the proper implementation of try, except in the program.
Action	A string value was provided as input in the serial number of the costume to be rented (which takes integer values only.)
Expected Result	A proper error message must be displayed and the user must be asked to input the serial number again.
Actual Result	A proper error message was displayed and the user was asked to input the serial number again.
Conclusion	Test Successful.

Table 1: Test table for try except

```
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
         Welcome to costume rental application
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++



Select a desireable option

  Press 1 to rent costumes.
  Press 2 to return costumes.
  Press 3 to exit.
Enter an option: 1

    Let's rent a costume.
S.No.    Costume Name            Brand           Price   Stock
================================================================
1        Cop Costume             MaxWalters      $15     23
2        Thief Costume           MaxWalters      $12     31
3        Formal Suit             SegaSmart       $14.5   27
4        Angel Costume           Funkywears      $18     27
5        Black Panther           Cozplayy        $28     22
6        Ghillie Suit            Tommmers        $19.5   24
7        Prince Dress            Funkywears      $29     31
8        Princess Robe           Funkywears      $35     32
9        Thor Costume            Cozplayy        $45     33
10       Captain Price           MaxWalters      $18     32
11       Vampire Dress           Cozplayy        $25     45

Enter the serial number of the Costume you want to rent: w

Please input Serial number in valid format.

Enter the serial number of the Costume you want to rent: |
```

Figure 8: Test 1 Screenshot

Test 2

Selection renting and returning of costumes

- Provide negative value as input
- Provide non-existed value as input

Objective	To show that the program doesn't take negative and non-existed values as input.
Action	First a negative number was input as the serial number of the costume to be rented and then a non-existed serial number was input.
Expected Result	A proper error message must be displayed and the user must be asked to input the serial number again.
Actual Result	A proper error message was displayed and the user was asked to input the serial number again.
Conclusion	Test Successful.

Table 2: Table for selection of renting and returning of costumes

```
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
         Welcome to costume rental application
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++



Select a desireable option

  Press 1 to rent costumes.
  Press 2 to return costumes.
  Press 3 to exit.
Enter an option: 1

    Let's rent a costume.
S.No.    Costume Name            Brand           Price   Stock
==============================================================
1        Cop Costume             MaxWalters      $15     23
2        Thief Costume           MaxWalters      $12     31
3        Formal Suit             SegaSmart       $14.5   27
4        Angel Costume           Funkywears      $18     27
5        Black Panther           Cozplayy        $28     22
6        Ghillie Suit            Tommmers        $19.5   24
7        Prince Dress            Funkywears      $29     31
8        Princess Robe           Funkywears      $35     32
9        Thor Costume            Cozplayy        $45     33
10       Captain Price           MaxWalters      $18     32
11       Vampire Dress           Cozplayy        $25     45

Enter the serial number of the Costume you want to rent: -1
Invalid Symbol Number!!!
Enter the serial number of the Costume you want to rent: |
```

Figure 9: Test 2 Screenshot (negative input)

```
+++++++++++++++++++++++++++++++++++++++++++++++++++++++++
         Welcome to costume rental application
+++++++++++++++++++++++++++++++++++++++++++++++++++++++++



Select a desireable option

  Press 1 to rent costumes.
  Press 2 to return costumes.
  Press 3 to exit.
Enter an option: 1

    Let's rent a costume.
S.No.    Costume Name            Brand           Price   Stock
=================================================================
1        Cop Costume             MaxWalters      $15     23
2        Thief Costume           MaxWalters      $12     31
3        Formal Suit             SegaSmart       $14.5   27
4        Angel Costume           Funkywears      $18     27
5        Black Panther           Cozplayy        $28     22
6        Ghillie Suit            Tommmers        $19.5   24
7        Prince Dress            Funkywears      $29     31
8        Princess Robe           Funkywears      $35     32
9        Thor Costume            Cozplayy        $45     33
10       Captain Price           MaxWalters      $18     32
11       Vampire Dress           Cozplayy        $25     45

Enter the serial number of the Costume you want to rent: 12
Invalid Symbol Number!!!
Enter the serial number of the Costume you want to rent: |
```

Figure 10: Test 2 Screenshot (non-existed input)

Test 3

File generation of renting costume (Renting multiple costumes)

- Show complete renting costume
- Show output in the shell as well
- Finally show the renting of costume in renting note in txt file

Objective	To show that the process of renting runs properly.
Action	The program is run and all the necessary inputs are provided for the renting of multiple costumes.
Expected Result	The renting process must be executed and an invoice with proper information must be generated in a new .txt file. The stock must also be updated in the .txt file with details of all costumes.
Actual Result	The renting process was executed and an invoice with proper information was generated in a new .txt file. The stock was updated in the .txt file with details of all costumes.
Conclusion	Test Successful.

Table 3: Test table for testing of the renting process

```
Select a desireable option

  Press 1 to rent costumes.
  Press 2 to return costumes.
  Press 3 to exit.
Enter an option: 1

    Let's rent a costume.
S.No.    Costume Name            Brand           Price   Stock
==============================================================
1        Cop Costume             MaxWalters      $15     22
2        Thief Costume           MaxWalters      $12     31
3        Formal Suit             SegaSmart       $14.5   27
4        Angel Costume           Funkywears      $18     25
5        Black Panther           Cozplayy        $28     25
6        Ghillie Suit            Tommmers        $19.5   24
7        Prince Dress            Funkywears      $29     31
8        Princess Robe           Funkywears      $35     31
9        Thor Costume            Cozplayy        $45     32
10       Captain Price           MaxWalters      $18     31
11       Vampire Dress           Cozplayy        $25     42

Enter the serial number of the Costume you want to rent: 2
S.No.    Costume Name            Brand           Price   Stock
==============================================================
2        Thief Costume           MaxWalters      $12     31

Enter the quantity of the item you have selected: 3
S.No.    Costume Name            Brand           Price   Stock
==============================================================
1        Cop Costume             MaxWalters      $15     22
2        Thief Costume           MaxWalters      $12     28
3        Formal Suit             SegaSmart       $14.5   27
4        Angel Costume           Funkywears      $18     25
5        Black Panther           Cozplayy        $28     25
6        Ghillie Suit            Tommmers        $19.5   24
7        Prince Dress            Funkywears      $29     31
8        Princess Robe           Funkywears      $35     31
9        Thor Costume            Cozplayy        $45     32
10       Captain Price           MaxWalters      $18     31
11       Vampire Dress           Cozplayy        $25     42

Press 'y' if you want to rent another costume press any other key to continue.. y
Enter the serial number of the Costume you want to rent: 3
```

```
Press 'y' if you want to rent another costume press any other key to continue.
Enter the serial number of the Costume you want to rent: 3
S.No.    Costume Name            Brand           Price   Stock
==============================================================
3        Formal Suit             SegaSmart       $14.5   27

Enter the quantity of the item you have selected: 2
S.No.    Costume Name            Brand           Price   Stock
==============================================================
1        Cop Costume             MaxWalters      $15     22
2        Thief Costume           MaxWalters      $12     28
3        Formal Suit             SegaSmart       $14.5   25
4        Angel Costume           Funkywears      $18     25
5        Black Panther           Cozplayy        $28     25
6        Ghillie Suit            Tommmers        $19.5   24
7        Prince Dress            Funkywears      $29     31
8        Princess Robe           Funkywears      $35     31
9        Thor Costume            Cozplayy        $45     32
10       Captain Price           MaxWalters      $18     31
11       Vampire Dress           Cozplayy        $25     42

Press 'y' if you want to rent another costume press any other key to continue.
Enter the customer's name: Sanam
Enter the customer's phone number: 9863122445

==========================================================================
            *Invoice has been generated for Rented Costumes*
==========================================================================

Select a desireable option

  Press 1 to rent costumes.
  Press 2 to return costumes.
  Press 3 to exit.
Enter an option:
```

Figure 11: Test 3 Screenshot (renting in shell)

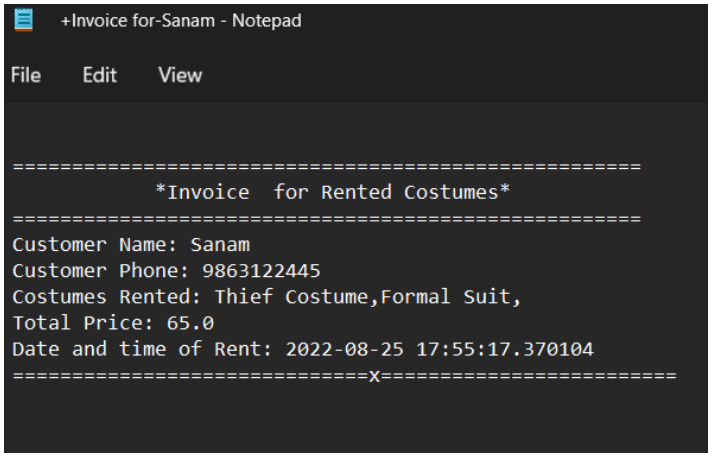

Figure 12: Test 3 Screenshot (invoice generated in new .txt file)

Test 4

File generation of returning process of costume (returning multiple costumes)

- Show complete returning process of costume
- Show output in the shell as well
- Finally show the returning of costume in returning note in txt file

Objective	To show that the process of returning runs properly.
Action	The program is run and all the necessary inputs are provided for the returning of multiple costumes.
Expected Result	The returning process must be executed and an invoice with proper information must be generated in a new .txt file. The stock must also be updated in the .txt file with details of all costumes.
Actual Result	The returning process was executed and an invoice with proper information was generated in a new .txt file. The stock was updated in the .txt file with details of all costumes.
Conclusion	Test Successful.

Table 4: Test table for testing of the returning process

```
Select a desireable option

  Press 1 to rent costumes.
  Press 2 to return costumes.
  Press 3 to exit.
Enter an option: 2
returned
S.No.    Costume Name            Brand           Price   Stock
==============================================================
1        Cop Costume             MaxWalters      $15     22
2        Thief Costume           MaxWalters      $12     28
3        Formal Suit             SegaSmart       $14.5   25
4        Angel Costume           Funkywears      $18     25
5        Black Panther           Cozplayy        $28     25
6        Ghillie Suit            Tommmers        $19.5   24
7        Prince Dress            Funkywears      $29     31
8        Princess Robe           Funkywears      $35     31
9        Thor Costume            Cozplayy        $45     32
10       Captain Price           MaxWalters      $18     31
11       Vampire Dress           Cozplayy        $25     42

Enter the serial number of the Costume you want to return: 4
S.No.    Costume Name            Brand           Price   Stock
==============================================================
4        Angel Costume           Funkywears      $18     25

Enter the quantity of the costume you want to return: 6
Enter the number of days the costume has been rented: 12
S.No.    Costume Name            Brand           Price   Stock
==============================================================
1        Cop Costume             MaxWalters      $15     22
2        Thief Costume           MaxWalters      $12     28
3        Formal Suit             SegaSmart       $14.5   25
4        Angel Costume           Funkywears      $18     31
5        Black Panther           Cozplayy        $28     25
6        Ghillie Suit            Tommmers        $19.5   24
7        Prince Dress            Funkywears      $29     31
8        Princess Robe           Funkywears      $35     31
9        Thor Costume            Cozplayy        $45     32
10       Captain Price           MaxWalters      $18     31
11       Vampire Dress           Cozplayy        $25     42

Press 'y' if you want to return another costume press any other key to continue.. y
Enter the serial number of the Costume you want to return: 6
```

```
Press 'y' if you want to return another costume press any other key to continue.. y
Enter the serial number of the Costume you want to return: 6
S.No.    Costume Name            Brand           Price   Stock
==============================================================
6        Ghillie Suit            Tommmers        $19.5   24

Enter the quantity of the costume you want to return: 1
Enter the number of days the costume has been rented: 3
S.No.    Costume Name            Brand           Price   Stock
==============================================================
1        Cop Costume             MaxWalters      $15     22
2        Thief Costume           MaxWalters      $12     28
3        Formal Suit             SegaSmart       $14.5   25
4        Angel Costume           Funkywears      $18     31
5        Black Panther           Cozplayy        $28     25
6        Ghillie Suit            Tommmers        $19.5   25
7        Prince Dress            Funkywears      $29     31
8        Princess Robe           Funkywears      $35     31
9        Thor Costume            Cozplayy        $45     32
10       Captain Price           MaxWalters      $18     31
11       Vampire Dress           Cozplayy        $25     42

Press 'y' if you want to return another costume press any other key to continue.. n
Enter the customer's name: Roman
Enter the customer's phone number: 983322187

==========================================================================
            *Invoice has been generated  for Costumes returned*
==========================================================================

Select a desireable option

  Press 1 to rent costumes.
  Press 2 to return costumes.
  Press 3 to exit.
Enter an option:
```

Figure 13: Test 4 Screenshot (returning process in shell)

```
+Invoice for-Roman - Notepad
File    Edit    View

=====================================================
            *Invoice for Costumes returned*
=====================================================
Customer Name: Roman
Customer Phone: 983322187
Costumes Rented: Angel Costume,Ghillie Suit,
Total Fine: 22.68
Total Price with fine: 127.5
Date and time of Return: 2022-08-25 18:13:48.488244
==============================x=========================
```

Figure 7: Test 4 Screenshot (invoice generated in new .txt file)

Test 5

Show the update in stock of costume

- Show the quantity being deducted while renting costume (Update should be reflected in .txt file as well)
- Show the quantity being added while returning costume (Update should be reflected in .txt file as well)

Objective	To show that the costume stock value is being updated after renting and returning.
Action	The program is run and all the necessary inputs are provided for the renting and returning of multiple costumes (this was done in the tests above). Then, the .txt file for costumes is checked for updates.
Expected Result	The renting and returning process must be executed. Then, the stock must be updated in the .txt file with details of all costumes.
Actual Result	The renting and returning process was executed. Then, the stock was updated in the .txt file with details of all costumes.
Conclusion	Test Successful.

Table 5: Test table for testing of the updated values in .txt file of costumes

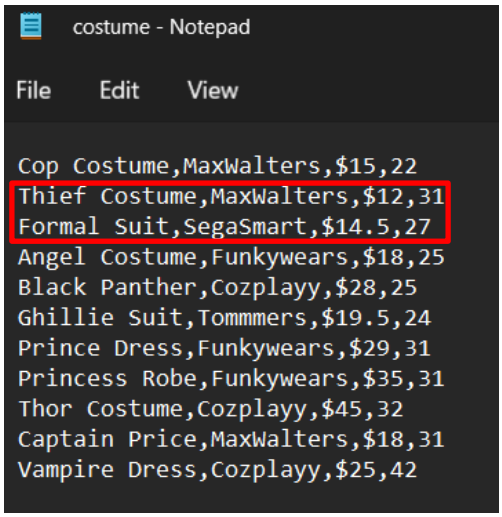

Figure 8: Test 3 Screenshot (stock before renting)

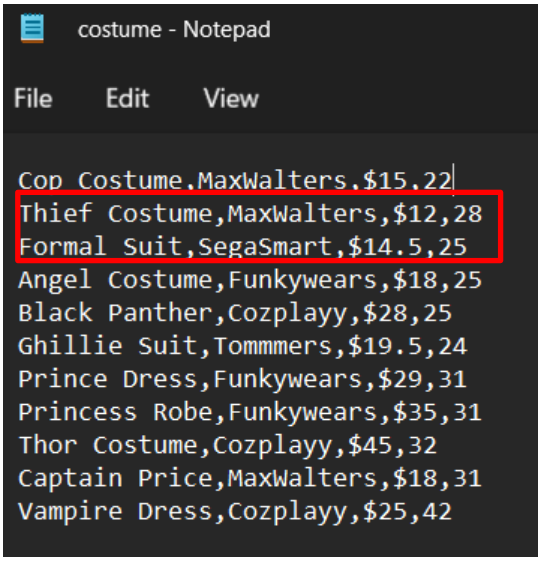

Figure 9: Test 3 Screenshot (stock after renting)

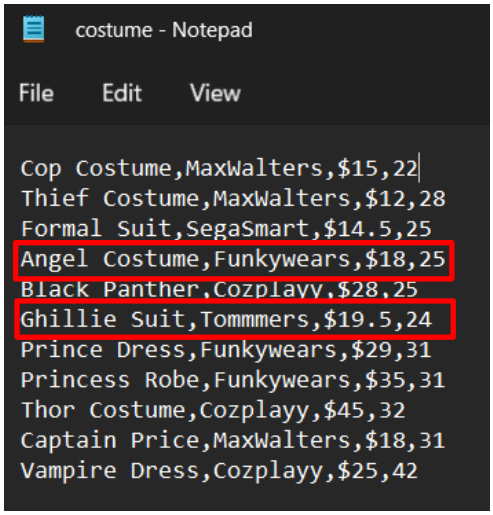

Figure 10: Test 4 Screenshot (stock before returning)

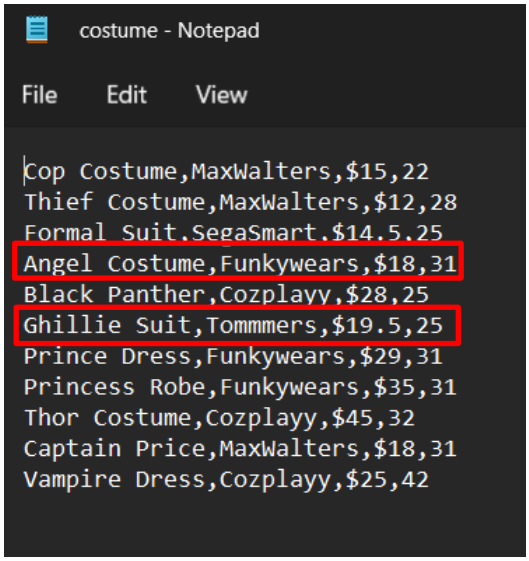

Figure 11: Test 4 Screenshot (stock after returning)

Conclusion

To sum up, this project has been time-consuming, difficult, fascinating, and intriguing all at once. It required a great deal of commitment and work. After completing this project, I felt a great sense of satisfaction and excitement. I want to thank all of my friends, teachers, and lecturers for their assistance in getting this assignment done.

I got to learn a lot through this project and all the contents of the module that were cover in this project seem very familiar to me now and I believe that I have gained a better understanding of the subject. We used various concepts of Python programming like data structures, Object-oriented programming, Exception handling, use of lists and dictionaries, file handling, functions, etc. to complete this project. These concepts that were taught to us during our semester are implemented in this project to solve a real-life problem. So, this project has improved my problem-solving skills and also helped me get used to the concept of conditional statements and looping.

I want to thank all of my teachers, professors, and friends once more for their assistance with this project.

Appendix

MainInterface.py

```
from rentFunction import rent
from returnFunction import Return
#Welcome Message
print('''


++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
        Welcome to costume rental application
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++


    ''')
#User Interface

Exit = False
while Exit==False:
    print("Select a desireable option")
    print('''
  Press 1 to rent costumes.
  Press 2 to return costumes.
  Press 3 to exit.''')
    option = input("Enter an option: ")
    #rent
    if option == "1":
        rent()

    #return
    elif option == "2":
        Return()

    #exit
    elif option == "3":
        Exit = True
        print('''
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
        Thank You for using the rental service
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
''')

    #error
    else:
```

```
        print('''
Invalid input!!!!
Please select a value as per the provided options.

''')
```

Cusfonctions.py

```
#creating a list containing the costumes and details in the .txt file
def getCostumesInFile():
    file = open("costume.txt","r")
    costumeData = file.readlines()
    file.close()
    return costumeData

'''Crearing a dictonary of the costumes using the lists' index as key and costume details
as value.
This is done so that the key can be used to access the lists of each costume and each
detail like price,stock,etc. can be accessed through their index within that list.'''
def costumeDictionary(costumesInFile):
    costumeData = {}
    for index in range(len(costumesInFile)):
        costumeData[index+1] = costumesInFile[index].replace("\n","").split(",")
    return costumeDictionary


#Function to display all costumes in a table.
def costumesTable():
    costumesInFile = getCostumesInFile()
    tableData = costumeDictionary(costumesInFile)
    print("S.No.","\t","Costume Name","\t\t","Brand","\t\t","Price","\t","Stock")

print("==============================================================
")
    for key, costume in tableData.items():
        print(key,"\t",costume[0],"\t\t",costume[1],"\t",costume[2],"\t",costume[3])
    print("")


#writing date and time in file
def Get_dateTime():
    import datetime
    datetime = datetime.datetime.now()
```

```
    return str(datetime)
```

rentFunction.py

```
from datetime import datetime
from Cusfunctions import getCostumesInFile, costumeDictionary, costumesTable,
Get_dateTime


#Function to select serial number of the costume to be rented.
def selectCosToRent():
    costumesInFile = getCostumesInFile()
    tableData = costumeDictionary(costumesInFile)
    validSyNo = False
    while validSyNo == False:
        try:
            SyNo = int(input("Enter the serial number of the Costume you want to rent: "))
            if SyNo > 0 and SyNo <= len(tableData):
                validSyNo = True
                a = tableData [SyNo]
                print("S.No.","\t","Costume Name","\t\t","Brand","\t\t","Price","\t","Stock")

print("==============================================================
")
                print(SyNo,"\t",a[0],"\t\t",a[1],"\t",a[2],"\t",a[3])
                print("")
                return SyNo
            else:
                print("Invalid Symbol Number!!!")
        except:
            print("")
            print("Please input Serial number in valid format.")
            print("")


#Function to select the quantity of the costume to be rented.
def quantityToRent(SyNo):
    costumesInFile = getCostumesInFile()
    tableData = costumeDictionary(costumesInFile)
    validstock = False
    while validstock == False:
        try:
            quantity = int(input("Enter the quantity of the item you have selected: "))
            if quantity > 0 and quantity <= int(tableData [SyNo][3]):
```

```
            return quantity
        elif quantity == 0:
            print("Costume not available for rent")
        else:
            print("Quantity limit out of stock!!!")
    except:
        print("")
        print("Please input quantity in valid format.")
        print("")


#Function to Rent a coustume
def rent():
    print('''
    Let's rent a costume.''')
    CostumeName = []
    totalPrice = 0
    costumesInFile = getCostumesInFile()
    tableData = costumeDictionary(costumesInFile)
    costumesTable()
    SyNo = selectCosToRent()
    quantity = quantityToRent(SyNo)

    tableData [SyNo] [3] = str(int(tableData[SyNo] [3]) - quantity)

    Cosfile = open("costume.txt","w")
    for key, costume in tableData.items():
        write_data =
str(costume[0])+","+str(costume[1])+","+str(costume[2])+","+str(costume[3])+"\n"
        Cosfile.writelines(write_data)
    Cosfile.close()

    CostumeName.append (tableData [SyNo] [0])
    price = tableData [SyNo] [2]
    totalPrice = totalPrice + float(price.replace('$','')) * quantity
    print("S.No.","\t","Costume Name","\t\t","Brand","\t\t","Price","\t","Stock")

print("================================================================
")
    for key, costume in tableData.items():
        print(key,"\t",costume[0],"\t\t",costume[1],"\t",costume[2],"\t",costume[3])
    print("")
    continueRenting = True
    while continueRenting == True:
        addCus = input("Press 'y' if you want to rent another costume press any other key
to continue.. ")
```

```
        if addCus == "y":
            SyNo = selectCosToRent()
            quantity = quantityToRent(SyNo)

            tableData [SyNo] [3] = str(int(tableData[SyNo] [3]) - quantity)

            Cosfile = open("costume.txt","w")
            for key, costume in tableData.items():
                write_data =
str(costume[0])+","+str(costume[1])+","+str(costume[2])+","+str(costume[3])+"\n"
                Cosfile.writelines(write_data)
            Cosfile.close()

            CostumeName.append (tableData [SyNo] [0])
            price = tableData [SyNo] [2]
            totalPrice = totalPrice + float(price.replace('$',"")) * quantity
            print("S.No.","\t","Costume Name","\t\t","Brand","\t\t","Price","\t","Stock")

print("==================================================================
")
            for key, costume in tableData.items():
                print(key,"\t",costume[0],"\t\t",costume[1],"\t",costume[2],"\t",costume[3])

            print("")
        else:
            break
    redo = True
    while redo == True:
        try:
                CustomerName = input("Enter the customer's name: ")
                CustomerPhone = int(input("Enter the customer's phone number: "))
                if CustomerName == "" or CustomerPhone ==0:
                    redo = True
                    print("Please fill the Customer's name and phone number.")
                else:
                    redo = False
        except:
            print("Invalid phone number!!")
            redo = True

    #rent invoice
    print('''
=====================================================================
===========
            *Invoice has been generated for Rented Costumes*
```

```
==================================================================
===========
    ''')

    #writing the invoice
    filename =  "Invoice for-" + CustomerName +".txt"
    file= open(r"RentInvoices\+" + filename, "w+")
    file.write('''
=====================================================
            *Invoice  for Rented Costumes*
=====================================================
''')
    file.write("Customer Name: " + CustomerName + "\n")
    file.write("Customer Phone: " + str(CustomerPhone) + "\n")
    file.write("Costumes Rented: ")
    for x in range(len(CostumeName)):
        file.write(CostumeName[x] + ",")
    file.write("\n" + "Total Price: "+ str(totalPrice)+"\n")
    #for date and time
    DateTime = Get_dateTime()
    file.write("Date and time of Rent: "+ DateTime+"\n")
    file.write("==============================x========================")
```

returnFunction.py

```
from Cusfunctions import getCostumesInFile, costumeDictionary,costumesTable,
Get_dateTime

def selectCosToReturn():
    costumesInFile = getCostumesInFile()
    tableData = costumeDictionary(costumesInFile)
    validSyNo = False
    while validSyNo == False:
        try:
            SyNo = int(input("Enter the serial number of the Costume you want to return: "))
            if SyNo > 0 and SyNo <= len(tableData):
                validSyNo = True
                a = tableData [SyNo]
                print("S.No.","\t","Costume Name","\t\t","Brand","\t\t","Price","\t","Stock")

print("==================================================================
")
                print(SyNo,"\t",a[0],"\t\t",a[1],"\t",a[2],"\t",a[3])
                print("")
```

```
            return SyNo
        else:
            print("Invalid Symbol Number!!!")
    except:
        print("")
        print("Please input serial number in correct format.")
        print("")


#Function to select the quantity of the costume to be rented.
def quantityToReturn(SyNo):
    costumesInFile = getCostumesInFile()
    tableData = costumeDictionary(costumesInFile)
    validstock = False
    while validstock == False:
        try:
            quantity = int(input("Enter the quantity of the costume you want to return: "))
            if quantity > 0:
                return quantity
            elif quantity == 0:
                print("Costume not available for rent")
            else:
                print("Quantity limit out of stock!!!")
        except:
            print("")
            print("Please input quantity in correct format.")
            print("")


def Return():
    print("Let’s return the costumes below.")
    CostumeName = []
    fine = 0
    totalPrice = 0
    totalPriceWithFine = 0
    costumesInFile = getCostumesInFile()
    tableData = costumeDictionary(costumesInFile)
    costumesTable()
    SyNo = selectCosToReturn()
    quantity = quantityToReturn(SyNo)
    validNodays = False
    while validNodays == False:
        try:
            noOfDays = int(input("Enter the number of days the costume has been rented:
"))
            if noOfDays == 0:
```

```
            print("Number of days cannot be zero. Please enter correct number of days.")
        else:
            validNodays = True
    except:
        print("Please enter number of days in correct format.")

tableData [SyNo] [3] = str(int(tableData[SyNo] [3]) + quantity)

Cosfile = open("costume.txt","w")
for key, costume in tableData.items():
    write_data =
str(costume[0])+","+str(costume[1])+","+str(costume[2])+","+str(costume[3])+"\n"
    Cosfile.writelines(write_data)
Cosfile.close()

CostumeName.append (tableData [SyNo] [0])
price = tableData [SyNo] [2]
totalPrice = totalPrice + float(price.replace('$','')) * quantity
if noOfDays > 5:
    fine = (noOfDays - 5) * ((3/100) * totalPrice)
    totalPriceWithFine = totalPrice + fine
elif noOfDays <= 5:
            totalPriceWithFine = totalPrice
print("S.No.","\t","Costume Name","\t\t","Brand","\t\t","Price","\t","Stock")

print("================================================================
")
for key, costume in tableData.items():
    print(key,"\t",costume[0],"\t\t",costume[1],"\t",costume[2],"\t",costume[3])
print("")
continueReturning = True
while continueReturning == True:
    addCus = input("Press 'y' if you want to return another costume press any other
key to continue.. ")
    if addCus == "y":
        SyNo = selectCosToReturn()
        quantity = quantityToReturn(SyNo)
        validNodays = False
        while validNodays == False:
            try:
                noOfDays = int(input("Enter the number of days the costume has been
rented: "))
                if noOfDays == 0:
                    print("Number of days cannot be zero. Please enter correct number of
days.")
                else:
```

```
                    validNodays = True
                except:
                    print("Please enter number of days in correct format.")

            tableData [SyNo] [3] = str(int(tableData[SyNo] [3]) + quantity)

            Cosfile = open("costume.txt","w")
            for key, costume in tableData.items():
                write_data =
str(costume[0])+","+str(costume[1])+","+str(costume[2])+","+str(costume[3])+"\n"
                Cosfile.writelines(write_data)
            Cosfile.close()

            CostumeName.append (tableData [SyNo] [0])
            price = tableData [SyNo] [2]
            totalPrice = totalPrice + float(price.replace('$','')) * quantity
            if noOfDays > 5:
                fine = (noOfDays - 5) * ((3/100) * totalPrice)
                totalPriceWithFine = totalPrice + fine
            elif noOfDays <= 5:
                totalPriceWithFine = totalPrice
            print("S.No.","\t","Costume Name","\t\t","Brand","\t\t","Price","\t","Stock")

print("================================================================
")
            for key, costume in tableData.items():
                print(key,"\t",costume[0],"\t\t",costume[1],"\t",costume[2],"\t",costume[3])
            print("")
        else:
            break
    redo = True
    while redo == True:
        try:
            CustomerName = input("Enter the customer's name: ")
            CustomerPhone = int(input("Enter the customer's phone number: "))
            if CustomerName == "" or CustomerPhone ==0:
                redo = True
                print("Please fill the Customer's name and phone number.")
            else:
                redo = False
        except:
            print("Invalid phone number!!")
            redo = True

    #return invoice
    print('''
```

```
==================================================================
===============
        *Invoice has been generated  for Costumes returned*
==================================================================
===============
    ''')

    #writing the invoice
    filename =  "Invoice for-" + CustomerName +".txt"
    file= open(r"ReturnInvoices\+" + filename, "w+")
    file.write('''
=====================================================
        *Invoice for Costumes returned*
=====================================================
''')
    file.write("Customer Name: " + CustomerName + "\n")
    file.write("Customer Phone: " + str(CustomerPhone) + "\n")
    file.write("Costumes Rented: ")
    for x in range(len(CostumeName)):
        file.write(CostumeName[x] + ",")
    #file.write("Costumes Rented: " + CostumeName + "\n")
    file.write("\n" + "Total Fine: " + str(fine) + "\n")
    file.write("Total Price with fine: " + str(totalPriceWithFine) + "\n")
    #for date and time
    DateTime = Get_dateTime()
    file.write("Date and time of Return: "+ DateTime+"\n")
    file.write("==============================x========================")
```