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: Aayush Man Tuladhar

Group: C6

London Met ID: 22015636

College ID: NP01CP4S220066

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

1. Introduction

1.1. About the Project

The following coursework is presented to us for the purpose of learning and solving real life problems through the use of the programming language known as Python.

Tools used:

Python: Python is a high-level programming language used in various fields in the current era of technological advancements. Compared to other programming languages Python is more abstract and has easier syntaxes than any other programming languages which has allowed it to come to the forefront for beginner programmers trying to enter into the world of software development and many more fields.

Ms Word: Microsoft word is the most commonly used word processor developed by Microsoft. It is a versatile application that can be use in from writing screen plays to writing a report on any topic. It helps you to save these creations in a form of a documents to be shared or used for any purpose accordingly by the users. In this project this application has been used to create a report on the project created in python.

1.2. Goals and Objectives

The objective given to us is to create a program for a Costume Rental company. The program is used by the receptionist to rent the costumes for the customer or register the returned costume by the customer.

Here, all the details of the costumes are to be stored and managed in a text file. The program needs to read and write the text file as the user uses it to rent or returning the costumes and provide the information about the renting or returning

in displayed bill format. If a costume was rented or returned the appropriate invoice is created in the text file format to be viewed. The invoice should be generated for each transaction that has been made. Here you can only rent a costume for more than 5 days and if returned more than 5 days later a fine should be applied to the return bill. The program when submitted should not have any error and all the exception should be handled properly using a different logic at times or using the try except method.

A report is to be prepared detailing the functions of the program created for the project. The report should be in a pdf format and the programs should be described in terms of its structure and behaviour. It can be presented using text and structural charts, flowcharts, or other diagrams as needed.

1.3. Business Rules

The business of costume rental can be a complicated one if not known how to manage. Without knowing the intricacies of this business, it's going to be terribly hard. To make this easier here are some business rules for our pretend company.

- The Costumes can be rented for more 5 days and after that if not returned a certain amount of fine is to be paid.
- The program is used by the receptionist so we can trust here to enter the right number of days and amount returned by the customer.
- The business should have more the 1 costume up for rental business.
- The costume cannot be rented if the valid ID for the costume is not provided.
- After the customer rents the items, a Bill is to be printed in a tabular format.
- Only one customer can rent/ return multiple costumes at once, but multiple people cannot rent or return multiple costumes at once.s

2. Discussion & Analysis

2.1. Algorithm

Step 1: START

Step 2: Display options

Step 3: Input values 1, 2 or 3

Step 4: IF input is equal to 1 Go to step 5, else go to step 20

Step 5: Display the costume options

Step 6: Input sno and quantity of costume for Rent

Step 7: Assign sno as serial number of costume and quantity as quantity of coustumes rented

Step 8: IF sno is valid and quantity is valid Go to step 8, else Go to step 6

Step 9: Open txt file “costume.txt” and re-write the values of quantity and decrease it by the amount rented

Step 10: Assign variable rented_items for the list that stores the costumes rented by the customer.

Step 11: Add the costume info of rented costume in the list rented_items.

Step 12: Input value n for not wanting to Rent more

Step 13: IF input is equal to n Go to step 5, else Go to step 14

Step 14: Assign grand_total as total price of all the costumes rented

Step 15: Calculate grand total by multiplying the quantity and price of the rented costumes by the customer.

Step 16: Input name, contact

Step 17: Print the Bill in an appropriate format with name, contact, rented_items and grand total.

Step 18: Generate Rent Bills using name, contact, rented_items and grand total In a tabular format.

Step 19: Print Thank you for Renting, Go to step 2

Step 20: IF input is equal to 2 Go to step 21, else Go to step 21x`

Step 21: Assignment sno as serial number of the costumes

Step 22: Input values for sno

Step 23: IF sno is valid Go to step 24, else Go to step 22

Step 24: Assign variable quantity as quantity of costumes returned

Step 25: Input quantity of costume and number of days Rented for

Step 26: Open txt file “costume.txt” and re-write the values of quantity and add the appropriate quantity returned.

Step 27: Assign the variable returned_items for the list that stores all the costumes returned by the customer.

Step 28: Add the costume info of the returned costume in the Returned_items list.

Step 29: IF rented days is greater than 5 Go to step 18, else go to step 19

Step 30: Assign total fine as cumulative fine of the costumes the were returned.

Step 31: Calculate total_fine is equal to quantity*((rented_days-5) *0.25)

Step 32: Input value n for not wanting to Return more

Step 33: IF input is equal to n Go to step 22, else Go to step 32

Step 34: Input name, contact

Step 35: Print the Bill in an appropriate format with name, contact, returned_items and total_fine.

Step 36: Generate Rent Bills using name, contact, returned_items and total_fine In a tabular format.

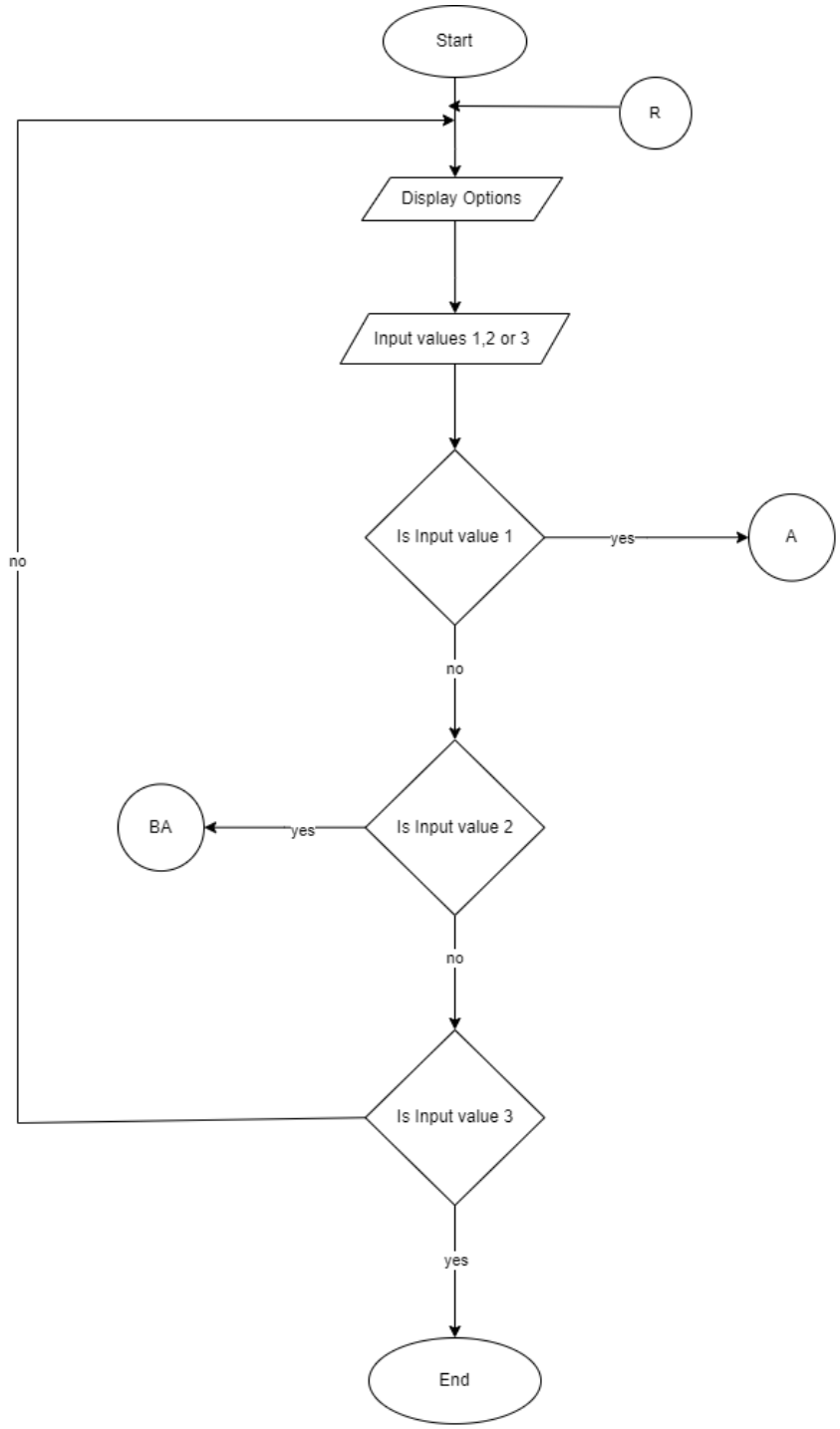
Step 37: Print Thank you for Returning, Go to step 2

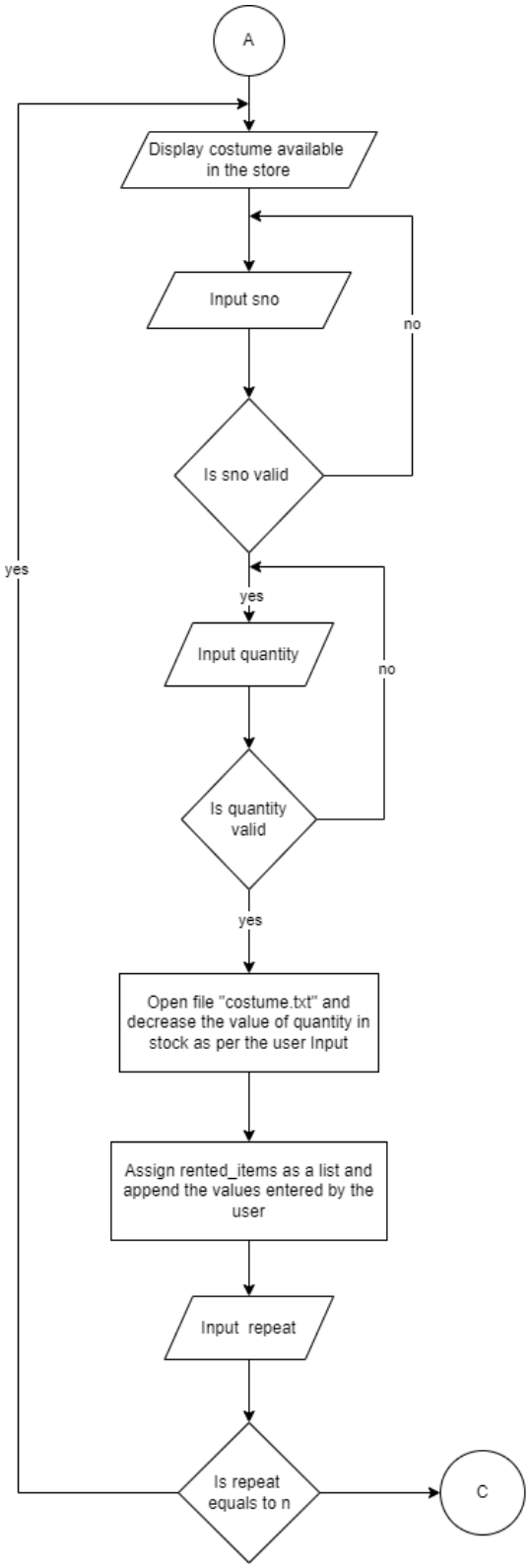
Step 38: If Input is equal to 3 Go to step 26 else, Go to step 3

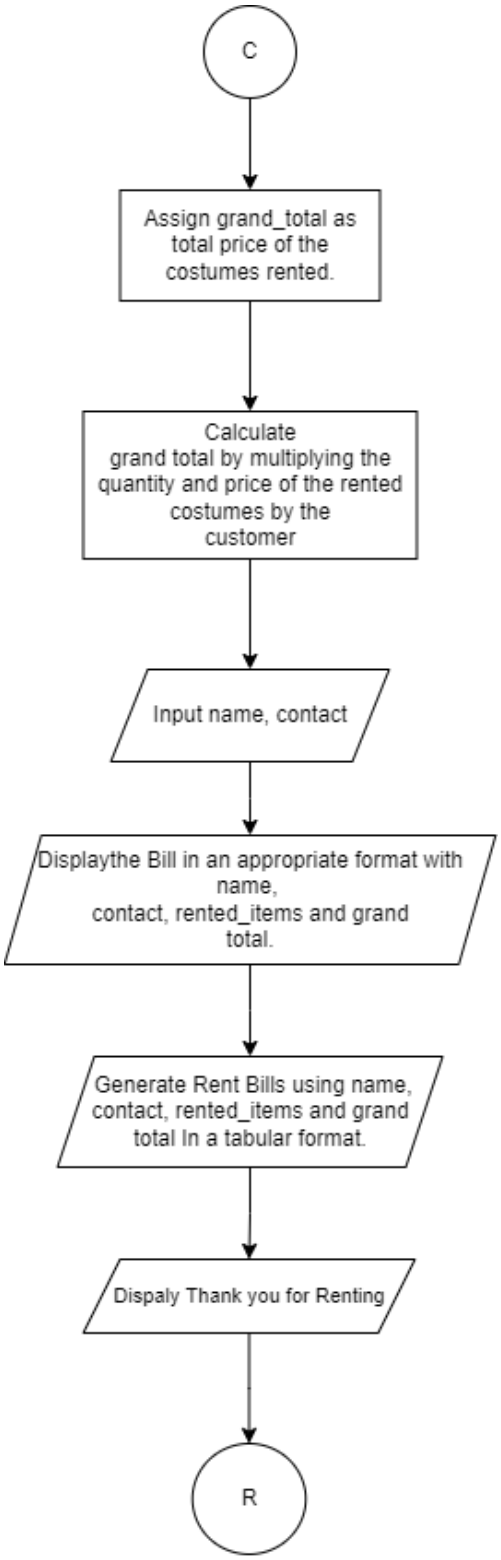
Step 39: Print Thank you for Visiting the Store

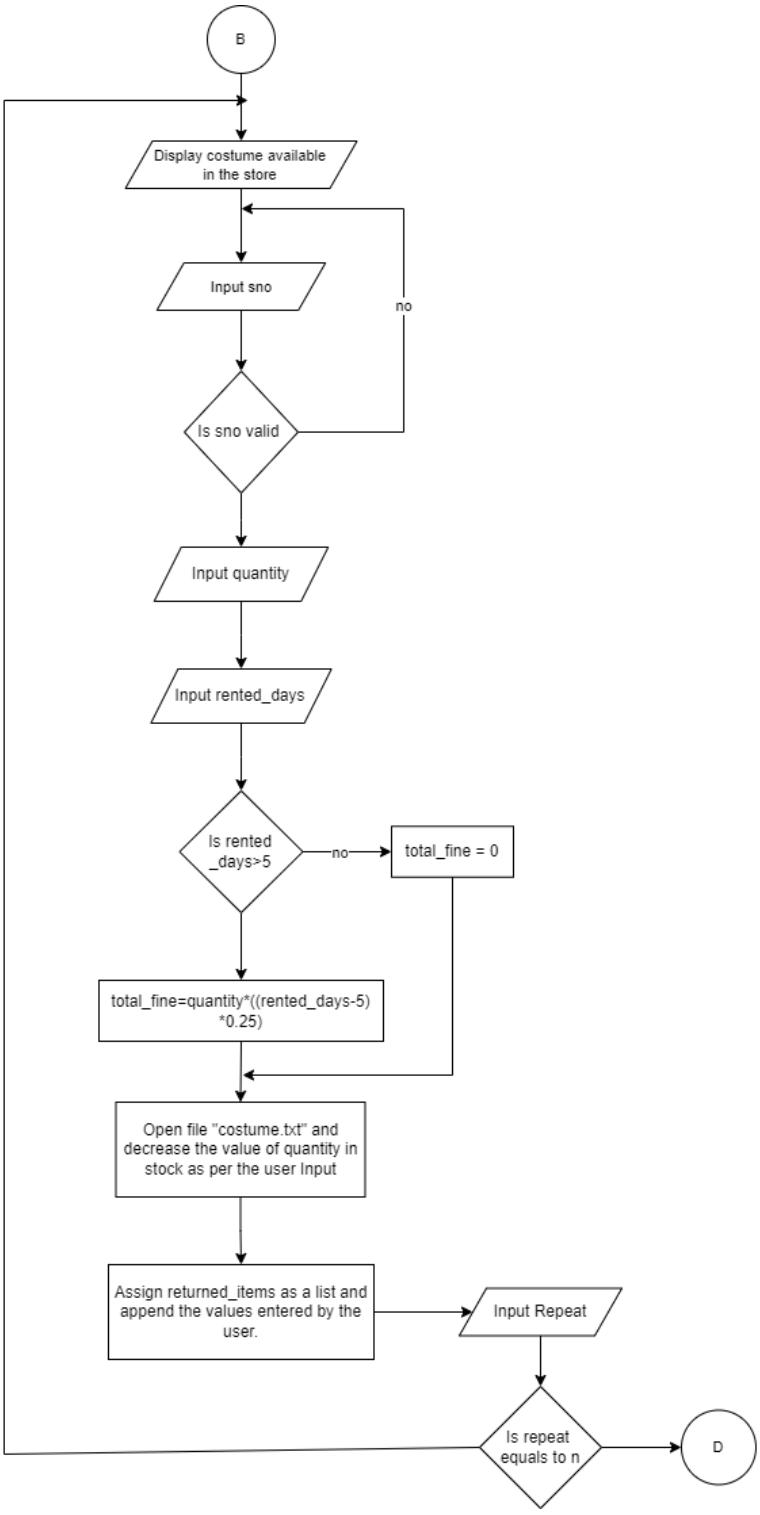
Step 40: END

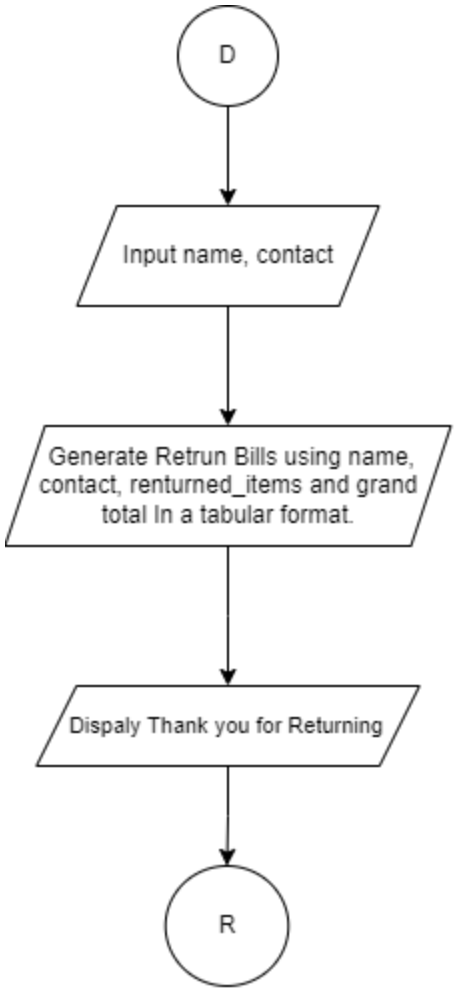
2.2. Flowchart




A
Display costume available in the store
Input sno
no
Is sno valid
yes
yes
Input quantity
no
Is quantity valid
yes
Open file "costume.txt" and decrease the value of quantity in stock as per the user Input
Assign rented_items as a list and append the values entered by the user
Input repeat
Is repeat equals to n
yes
C


C
Assign grand_total as total price of the costumes rented.
Calculate grand total by multiplying the quantity and price of the rented costumes by the customer
Input name, contact
Displaythe Bill in an appropriate format with name, contact, rented_items and grand total.
Generate Rent Bills using name, contact, rented_items and grand total In a tabular format.
Dispaly Thank you for Renting
R


B
Display costume available in the store
Input sno
no
Is sno valid
Input quantity
Input rented_days
Is rented _days>5
no
total_fine = 0
total_fine=quantity*((rented_days-5) *0.25)
Open file "costume.txt" and decrease the value of quantity in stock as per the user Input
Assign returned_items as a list and append the values entered by the user.
Input Repeat
Is repeat equals to n
D


D
Input name, contact
Generate Retrun Bills using name, contact, renturned_items and grand total In a tabular format.
Dispaly Thank you for Returning
R

2.3. Pseudocode

a. Main.py Pseudocode

IMPORT Rent_Validation

IMPORT Return_Validation

DEFINE FUNCTION with no parameters options

OUTPUT 'Choose your desired task'

OUTPUT '1 || Press 1 to rent a costume.'

OUTPUT '2 || Press 2 to RETURN a costume.'

OUTPUT '3 || Press 3 to exit.'

OUTPUT
RETURN TYPE NONE

SET continueLoop **TO** True

WHILE continueLoop **EQUALS** True

OUTPUT Welcome to Costume Rental Shop

CALL function options()

SET choice **TO** int **INPUT** "Enter your choice: "

IF choice **EQUALS** 1

Rent_Validation.Renting

ELSEIF choice **EQUALS** 2:

Return_Validation.Returning

ELSEIF choice **EQUALS** 3:

SET continueLoop **TO** False

OUTPUT " ||Thank You FOR Renting our Costumes||"

ELSE:

OUTPUT "Error!! Enter available options!!"

OUTPUT

b. Get_Info.py Pseudocode

```
DEFINE FUNCTION get_file_info
    SET file TO OPEN"costume.txt" in read mode
    SET data TO function readlines()
    CLOSE file
    RETURN data

DEFINE FUNCTION get_dict_info with PARAMETER( file_info)
    SET data_dict TO {}
    FOR index IN range len file_info
        SET data_dict[index+1] TO file_info[index].replace ,"" .split ","
    RETURN data_dict

DEFINE FUNCTION costumes_info
    SET file_info TO get_file_info
    SET main_data TO get_dict_info file_info
    OUTPUT
    OUTPUT "ID","    ","Name","\t","  ","Brand","\t","            ","Price","\t","Quantity"


    FOR key,value IN main_data.items  :
        OUTPUT key,"      ",value[0],"\t","  ",value[1],"\t",value[2],"\t","  ",value[3]
        OUTPUT
```

c. Date_Time.py

```
IMPORT datetime


DEFINE FUNCTION get_datetime
    SET year TO FUNCTION datetime.now().year
    SET month TO FUNCTION datetime.now().month
    SET day TO FUNCTION datetime.now().day
    SET hour TO FUNCTION datetime.now().hour
    SET minute TO FUNCTION datetime.now().minute
    SET second TO FUNCTION datetime.now().second

    SET date_time TO year+month+day+hour+minute+second
    RETURN date_time

DEFINE FUNCTION dates()
    SET year TO FUNCTION datetime.now().year
    SET month TO FUNCTION datetime.now().month
    SET day TO FUNCTION datetime.now().day

    SET date_only TO day+"/"+month+"/"+year
    RETURN date_only
```

d. Rent_Validation.py Pseudocode

```
IMPORT Get_Info

IMPORT Rent_Bill


DEFINE FUNCTION validation_of_id():
    SET file_info TO Get_Info.get_file_info()
    SET main_data TO Get_Info.get_dict_info(file_info)
    SET valid_INPUT TO False
    WHILE valid_INPUT EQUALS False:
        ExceptionLoop =True
        WHILE  ExceptionLoop EQUALS True:
            TRY:
                SET sno TO int(INPUT("Costume ID needed: "))
                SET ExceptionLoop TO False
            EXCEPT:
                OUTPUT lines
                OUTPUT Error!!Invalid Input!
                OUTPUT lines
        IF sno>0 and sno<=len(main_data):
            IF  int(main_data[sno][3]) EQUALS 0:
                OUTPUT lines
                OUTPUT("This costume is out of Stock")
                OUTPUT lines
```

```
            OUTPUT
            SET valid_INPUT TO False
        ELSE:
            OUTPUT lines
            OUTPUT("Your costume are available to be rented.")
            OUTPUT lines
            OUTPUT
            SET valid_INPUT TO True
    ELSE:
            OUTPUT lines
            OUTPUT("-------Error!!Invalid Input!!------")
            OUTPUT lines
            OUTPUT
            SET valid_INPUT TO False
    RETURN sno


DEFINE FUNCTION validation_of_quantity(valid_id):
    SET file_info TO Get_Info.get_file_info()
    SET main_data TO Get_Info.get_dict_info(file_info)
    SET quantity TO int(main_data[valid_id][3])
    SET valid_INPUT TO False
    WHILE valid_INPUT EQUALS False:
```

```
ExceptionLoop =True
WHILE  ExceptionLoop EQUALS True:
   TRY:
      SET INPUT_quantity TO int(INPUT("Amount you would like to Rent: "))
      ExceptionLoop =False
   EXCEPT:
      OUTPUT lines
      OUTPUT("-------Error!!Invalid Input!!------")
      OUTPUT lines
      OUTPUT
IF INPUT_quantity >0 and INPUT_quantity <=quantity:
   OUTPUT lines
   OUTPUT("Costume has been Rented sucessfully!!")
  OUTPUT lines
   OUTPUT
   SET valid_INPUT TO True
ELSE:
   OUTPUT lines
   OUTPUT("-------Error!!Invalid Input!!------")
   OUTPUT lines
   OUTPUT

RETURN INPUT_quantity
```

```
DEFINE FUNCTION Renting():
    SET grand_total TO 0
    SET price  TO 0
    SET rented_items TO []
    SET continueLoop TO True
    WHILE continueLoop EQUALS True:
        Get_Info.costumes_info()
        OUTPUT
        SET validID TO validation_of_id()
        SET available TO validation_of_quantity(validID)
        SET file_info TO Get_Info.get_file_info()
        SET main_data TO Get_Info.get_dict_info(file_info)
        SET no TO main_data[validID][2].replace("$","")
        SET price TO float(no)*int(available)
        grand_total += float(price) TO
        SET main_data[validID][3] str(int(main_data[validID][3]) - available)

        SET file TO open("costume.txt","w")
        FOR value IN main_data.values():
            OUTPUT lines
rewrite_data TO value[0]+","+value[1]+","+value[2]+","+value[3]+"\n"
            file.write(rewrite_data)
        file.close()
```

```
rented_items.append([validID,main_data[validID][0],available])
SET x TO False
WHILE x EQUALS False:
    SET repeat TO INPUT("Would you like to rent more: ")

    IF repeat EQUALS "n":
        OUTPUT()
        SET name TO INPUT("Enter your name: ")
        SET contact TO INPUT ("Phone no: ")
        OUTPUT()
        Rent_Bill.bill_for_rent(name,contact,grand_total,rented_items)
        OUTPUT()
        Rent_Bill.generate_bill(name, contact,grand_total,rented_items)
        SET continueLoop TO False
        SET x TO True
    ELSE:
        SET continueLoop TO True
        SET x TO True
```

e. Rent_Bill.py Pseudocode

```
IMPORT Date_Time

IMPORT Get_Info

DEFINE FUNCTION bill_for_rent(name,contact,grand_total,rented_items):
    OUTPUT lines
    OUTPUT("                    Your Rent Bill                    ")
    OUTPUT lines
    OUTPUT("Customer Name: ", name)
    OUTPUT("Contact : ",contact)
    OUTPUT("Rented Date: ", Date_Time.dates())
    OUTPUT("Your Items:")
    FOR items IN rented_items:
        OUTPUT(items)
    OUTPUT lines
    OUTPUT("Grand total: $",grand_total)
    OUTPUT lines


DEFINE FUNCTION generate_bill(name, contact,
    grand_total,rented_items):
    SET file_info TO Get_Info.get_file_info()
    SET main_data TO Get_Info.get_dict_info(file_info)
    SET file TO open("Rent"+"_"+Date_Time.get_datetime()+"_"+name+".txt","w")
```

```
        file.write("_____
_____"+"\n")

    file.write("                    Costume Rental Bill                                    "+"\n")

file.write("_____
_____"+"\n")

    file.write("Customer name: "+name+"\n")

    file.write("Contact : "+contact+"\n")

    file.write("Rented Date: "+Date_Time.dates()+"\n")

    file.write("-----------------------------------------------------------------------"+"\n")

    file.write("ID"+"    "+"Name"+"\t"+"          "+"Brand"+"\t"+"       "+"Price"+"\t"+"
"+"Quantity"+"\n")

    file.write("-----------------------------------------------------------------------"+"\n")

    FOR index IN range(len(rented_items)):

        SET c_sno TO int(rented_items[index][0])

        SET c_quantity TO int(rented_items[index][2])

        SET c_name TO main_data[c_sno][0]

        SET c_brand TO main_data[c_sno][1]

        SET c_price TO float(main_data[c_sno][2].replace("$","")) * c_quantity

        file.write(str(index+1)+"\t"+c_name+"\t"+c_brand+"\t"+"  "+str(c_price)+"\t"+"
"+str(c_quantity)+"\n")

        file.write("-----------------------------------------------------------------------"+"\n")

file.write("_____
_____"+"\n")

    file.write("Grand total: $"+str(grand_total)+"\n")
```

```
file.write("_____
_____"+"\n")

    file.close()
```

f. Return_Validation.py Pseudocode

```
IMPORT Get_Info

IMPORT Return_Bill


DEFINE FUNCTION get_valid_id():
    SET file_info TO Get_Info.get_file_info()
    SET main_data TO Get_Info.get_dict_info(file_info)
    SET valid_INPUT TO False
    WHILE valid_INPUT EQUALS False:
        SET ExceptionLoop TO True
        WHILE ExceptionLoop EQUALS True:
            TRY:
                SET sno TO int(INPUT("Costume ID needed: "))
                SET ExceptionLoop TO False
            EXCEPT:
                OUTPUT lines
                OUTPUT("-------Error!!Invalid Input!!------")
                OUTPUT lines
                OUTPUT()
        IF sno>0 and sno<=len(main_data):
```

```
            OUTPUT lines
            OUTPUT("Your costume are available to be yet to be RETURNed")
            OUTPUT lines
            OUTPUT()
            SET valid_INPUT TO True
        ELSE:
            OUTPUT lines
            OUTPUT("-------Error!!Invalid Input!!------")
            OUTPUT lines
            OUTPUT()
    RETURN sno


DEFINE FUNCTION get_valid_quantity(valid_id):
    SET file_info TO Get_Info.get_file_info()
    SET main_data TO Get_Info.get_dict_info(file_info)
    SET quantity TO int(main_data[valid_id][3])
    SET valid_INPUT TO False
    WHILE valid_INPUT EQUALS False:
        SET ExceptionLoop TO True
        WHILE ExceptionLoop EQUALS True:
            TRY:
                SET INPUT_quantity TO int(INPUT("Amount you would like to Return:
"))
```

```
            SET ExceptionLoop TO False
        EXCEPT:
            OUTPUT lines
            OUTPUT("-------Error!!Invalid Input!!------")
            OUTPUT lines
            OUTPUT()
    IF INPUT_quantity >0 :
        OUTPUT lines
        OUTPUT("Costume has been Returned sucessfully!!")
        OUTPUT lines
        OUTPUT()
        SET valid_INPUT TO True
    ELSE:
        OUTPUT lines
        OUTPUT("-------Error!!Invalid Input!!------")
        OUTPUT lines
        OUTPUT()
  RETURN INPUT_quantity


DEFINE FUNCTION Days_Checker():
  SET y TO True
  WHILE y EQUALS True:
```

```
        SET rented_days TO int(INPUT("How many days since Rented: "))
        IF rented_days > 5:
            OUTPUT lines
            OUTPUT("You will be Fined FOR Returning Late")
            OUTPUT lines
            OUTPUT()
            SET y TO False
        ELSE:
            OUTPUT lines
            OUTPUT("Thank you FOR Returning IN time")
            OUTPUT lines
            OUTPUT()
            SET y TO False
        RETURN rented_days


SET RETURNed_items TO []
DEFINE FUNCTION Returning():
    SET fine TO 0.25
    SET total_fine TO 0


    SET continueLoop TO True
    WHILE continueLoop EQUALS True:
        Get_Info.costumes_info()
```

```
OUTPUT()
SET validID TO get_valid_id()
SET available TO get_valid_quantity(validID)
SET days TO Days_Checker()
SET file_info TO Get_Info.get_file_info()
SET main_data TO Get_Info.get_dict_info(file_info)

SET total_fine TO total_fine*((days-5)*fine)
total_fine += available

SET no TO main_data[validID][2].replace("$","")


SET main_data[validID][3] TO str(int(main_data[validID][3]) + available)

SET file TO open("costume.txt","w")
FOR value IN main_data.values():
    SET rewrite_data TO value[0]+","+value[1]+","+value[2]+","+value[3]+"\n"
    file.write(rewrite_data)
file.close()

RETURNed_items.append([validID,main_data[validID][0],available])
```

```
        SET x TO False
        WHILE x EQUALS False:
            ExceptionLoop =True
            WHILE ExceptionLoop EQUALS True:
                TRY:
                    SET repeat TO INPUT("Would you like to RETURN more: ")
                    SET ExceptionLoop TO False
                EXCEPT:
                    OUTPUT()
                    OUTPUT lines
                    OUTPUT("-------Error!!Invalid Input!!------")
                    OUTPUT lines

                IF repeat EQUALS "n":
                    OUTPUT()
                    SET name TO INPUT("Enter your name: ")
                    SET contact TO INPUT("Phone no: ")
                    OUTPUT()

    Return_Bill.bill_for_RETURN(name,contact,total_fine,RETURNed_items)
                    OUTPUT()
                    Return_Bill.generate_bill(name, contact,total_fine
    ,RETURNed_items)
```

SET continueLoop **TO** False

SET x **TO** True

ELSE:

SET continueLoop **TO** True

SET x **TO** True

g. Return_Bill.py Pseudocode

IMPORT Date_Time

IMPORT Get_Info

DEFINE FUNCTION bill_for_RETURN(name,contact,total_fine,RETURNed_items):

OUTPUT lines

OUTPUT(" Your Return Bill ")

OUTPUT lines

OUTPUT("Customer Name: ", name)

OUTPUT("Contact: ", contact)

OUTPUT("Returned Date: ", Date_Time.dates())

OUTPUT("List of Items Returned: ")

FOR items **IN** RETURNed_items:

OUTPUT(items)

OUTPUT lines

OUTPUT("Fine: $",total_fine)

OUTPUT lines

```
DEFINE FUNCTION generate_bill(name, contact,total_fine ,RETURNed_items):

    SET file_info TO Get_Info.get_file_info()

    SET main_data TO Get_Info.get_dict_info(file_info)


    SET file TO
open("Return"+"_"+Date_Time.get_datetime()+"_"+name+".txt","w")


file.write("_____
_____"+"\n")

    file.write("                    Costume Returned Bill
"+"\n")


file.write("_____
_____"+"\n")

    file.write("Customer name: "+name+"\n")

    file.write("Contact: "+contact+"\n")

    file.write("Rented Date: "+Date_Time.dates()+"\n")

    file.write("-----------------------------------------------------------------------"+"\n")

    file.write("ID"+"    "+"Name"+"\t"+"          "+"Brand"+"\t"+"       "+"Price"+"\t"+"
"+"Quantity"+"\n")

    file.write("-----------------------------------------------------------------------"+"\n")

    FOR index IN range(len(RETURNed_items)):

        SET r_sno TO int(RETURNed_items[index][0])

        SET r_quantity TO int(RETURNed_items[index][2])

        SET r_name TO main_data[r_sno][0]

        SET r_brand TO main_data[r_sno][1]
```

```
        SET r_price TO float(main_data[r_sno][2].replace("$","")) * r_quantity

        file.write(str(index+1)+"\t"+r_name+"\t"+r_brand+"\t"+"  "+str(r_price)+"\t"+"
"+str(r_quantity)+"\n")

        file.write("---------------------------------------------------------------------"+"\n")


file.write("+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++"+"\n")

    file.write("Fine: $"+str(total_fine)+"\n")


file.write("+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++"+"\n")

    file.close()
```

2.4. Data Structures

While programming you need to store data, extract it, and manage it as efficiently as possible. Data structure allows you to organize your data by letting you store collections of data and perform operation regarding your purposes accordingly. The data structures that were used during the development of this program were Lists and Dictionaries which are built-in Python data structures. These were used to manage the costumes that were available and input values of the customer for renting and returning the costumes. To be more specific the dictionary has been used to store the extracted values from the costume txt file in a usable manner which helps to manage the costumes in a dynamic and mutable way. Equally the list used here is a 2D list which stores the data of the rented and returned costume to create a bill for the customer and generate an invoice accordingly. The below images are examples of where the aforementioned data structures have been used.

Figure 1: Using Dictionary to manage data from costume.txt file

```
rented_items = []
continueLoop = True
while continueLoop == True:
    Get_Info.costumes_info()
    print()
    validID = validation_of_id()
    available = validation_of_quantity(validID)
    file_info = Get_Info.get_file_info()
    main_data = Get_Info.get_dict_info(file_info)
    no = main_data[validID][2].replace("$","")
    price = float(no)*int(available)
    grand_total += float(price)
    main_data[validID][3] = str(int(main_data[validID][3]) - available)

    file = open("costume.txt","w")
    for value in main_data.values():
        rewrite_data = value[0]+","+value[1]+","+value[2]+","+value[3]+"\n"
        file.write(rewrite_data)
    file.close()

    rented_items.append([validID,main_data[validID][0],available])
```

Figure 2: Creating a 2D list and storing input values into the list

```
lef generate_bill(name, grand_total,rented_items):
    file_info = Get_Info.get_file_info()
    main_data = Get_Info.get_dict_info(file_info)

    file = open("Rent"+"_"+Date_Time.get_datetime()+"_"+name+".txt","w")
    file.write("_____"+"\n")
    file.write("                Costume Rental Bill                "+"\n")
    file.write("_____"+"\n")
    file.write("Customer name: "+name+"\n")
    file.write("Rented Date: "+Date_Time.dates()+"\n")
    file.write("-----"+"\n")
    file.write("ID"+"    "+"Name"+"\t"+"          "+"Brand"+"\t"+"      "+"Price"+"\t"+"    "+"Quantity"+"\n")
    file.write("-----"+"\n")
    for index in range(len(rented_items)):
        c_sno = int(rented_items[index][0])
        c_quantity = int(rented_items[index][2])
        c_name = main_data[c_sno][0]
        c_brand = main_data[c_sno][1]
        c_price = float(main_data[c_sno][2].replace("$","")) * c_quantity
        file.write(str(index+1)+"\t"+c_name+"\t"+c_brand+"\t"+"  "+str(c_price)+"\t"+"      "+str(c_quantity)
        file.write("-----"+"\n")
    file.write("++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++")
    file.write("Grand total: $"+str(grand_total)+"\n")
    file.write("++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++")
    file.close()
```

Figure 3: Using the 2D list to generate bills by extracting data from list

As mentioned before there are two types of data structures in Python. These are Built-in Data Structures and User-Defined Data Structure, and examples are List, Dictionary, Tuples, Set and Stack, Queue, Tree, Graph, HashMap respectively. Below are the Built-in Data structures of python and their properties and uses.

LIST: This is used to store the data in a sequential manner and can have data of multiple data types. Here, each value is associated with an index starting from the number 0 from front to back. If data is to be extracted from the end of the list the index starts from -1 from back to front. This data structure is mutable which means it is dynamic so, you can add, remove, and edit data stored in this data structure. This is created by using [ ].

DICTIONARY: A Dictionary store data in pairs of keys and values. Here, you assign a value with the appropriate key so you can manage the data more efficiently. This is also a mutable data structure which make it dynamic in the sense that values in this collection can be edited by the developer. This can be created by using {key1:value1, key2:value2}.

TUPLES: This data structure is very similar to the List. The values here are also associated with the index starting from the number 0. The main differences are that Tuples are not mutable which means it is not dynamic. You cannot append new values or remove value from Tuples, but it is faster than a List. This is created by using ().

SET: This is an unordered collection of unique elements which means that if multiple data of the similar types are store it will only store one of it. You can perform functions like intersections, differences, union, and others. This data structure is also mutable in nature. This is created by using {} only.

PRIMARY DATA TYPES:

The data types that are already pre-existing and pre-defined by the python language are called primary data types. These are used almost every time you try to code anything on python. Some of the Data types used in this project are as follows.

INT: This is a data type that is used for integers. It accepts values that are whole numbers. Integers are are zero, positive or negative whole numbers without a fractional part and having unlimited precision. Integers can be binary, octal, and hexadecimal values.

FLOAT: A float is a data type composed of a number that is not an integer, because it includes a fraction represented in decimal format. It can be used to calculate a more accurate result.

BOOL: A Boolean value is having only two possible states, them being true and false. It can be used to create conditions and control how a program behaves when certain things happen. (Falconer, 2021)

STR: Traditionally, a string data type consists of a series of characters, either in the form of a literal constant or a variable. The latter can either be constant in length or allow its elements to alter. (Braunschweig, 2022)

3. Program

3.1. Implementation

The program has been created in a modular fashion with multiple ".**py**" files containing common as well as unique functions as per the requirement of the application. The main file that runs the whole program is appropriately names "**main.py**." it imports the function of rent and return validation files and uses them as per their function if the valid input is given. Here, option 1 is for renting the costumes, option 2 is for returning the costumes and finally option 3 is for the exiting the program.

Since the data and information of the costumes are stored in the test file "costume.txt" the data needs to be extracted for the purposes of running the program. Here, the file "**Get_Info.py**" is used to open the text file and read to each line and separate each data store it into a dictionary to manage the data of the costume. A dictionary was chosen here because due to its mutable nature and also because it stores the info of the costume in key and value format which makes the data more easily accessible. For validation of renting the available costumes the file "**Rent_Validation.py**" has been created. It contains that the functionality needed to rent a costume and also overrides the "**costume.txt**" file and updates the value of the quantity of the items accordingly. For validation of returning the costumes the file "**Return_Validation.py**" has been created. It contains that the functionality needed to return a costume and also overrides the "**costume.txt**" file and updates the value of the quantity of the items accordingly.

For each of these functions' bills are generated and printed through the files "**Rent_Bill.py**" and "**Return_Bill.py**" and accordingly the date for the bills are provided by the file "**Date_Time.py**" file which imports the date and time module which provides accurate time calculation down to micro seconds.

3.2. Renting and Returning

The renting and returning is the main function of the program. For returning it option 1 needs to be entered. After this the program will ask for the costume ID of the costume you want to rent. If the costumes are rented the grand total is calculated by the adding the and multiplying the price each item in the List after being appended. Here after renting is complete the program asks for your details on your name and contact for the purposes of bill generation.

```
= RESTART: C:\Users\tulad\OneDrive\Documents\2nd Semester\Func
sh Man Tuladhar\main.py
++++++++++++++++++++++++++++++++++++++++++++++++++
            Welcome to Costume Rental Shop
++++++++++++++++++++++++++++++++++++++++++++++++++
Choose your desired task
1) || Press 1 to rent a costume.
2) || Press 2 to return a costume.
3) || Press 3 to exit.

Enter your choice: 1
```

Figure 4: Entering value 1 to rent a costume

```
Enter your choice: 1
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
ID    Name             Brand            Price      Quantity
----------------------------------------------------------------------------------
1     Cop Costumes     Cartmax          $15.5      13
----------------------------------------------------------------------------------
2     Formal Suite     Megaplex         $14        35
----------------------------------------------------------------------------------
3     Fairy Costume    DollarS          $18        30
----------------------------------------------------------------------------------
4     Shrek Costume    DreaMs           $17        30
----------------------------------------------------------------------------------
5     Donkey Suite     DREAMs           $20        30
----------------------------------------------------------------------------------
6     Pikachu Dress    PokeInc.         $18        30
----------------------------------------------------------------------------------
7     Demon Suite      MonInc.          $13        30
----------------------------------------------------------------------------------
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

Figure 5: Displaying all the costumes available in the store

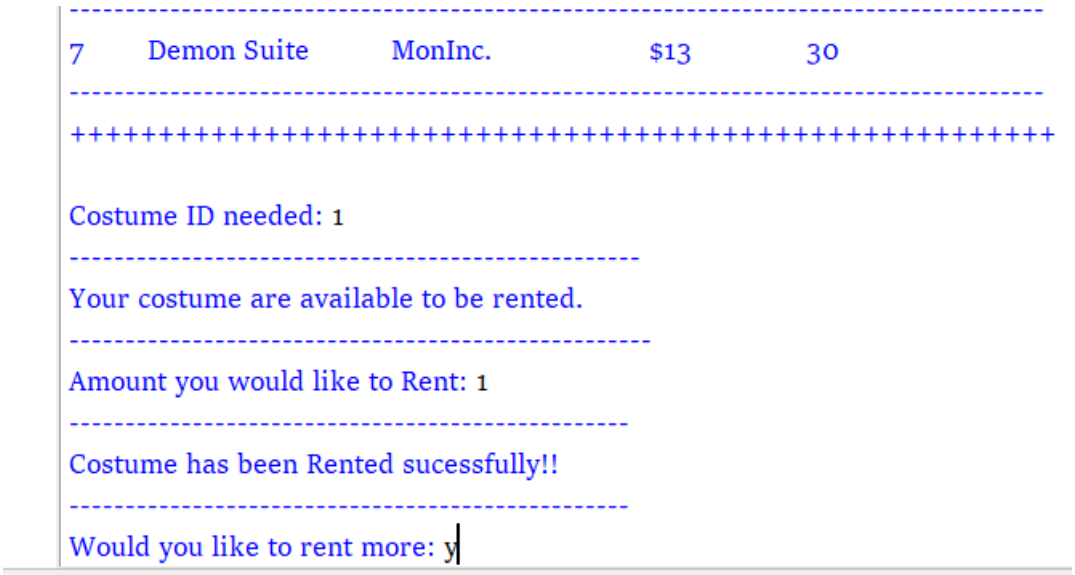


Figure 6: Renting the costume and saying yes to renting more.

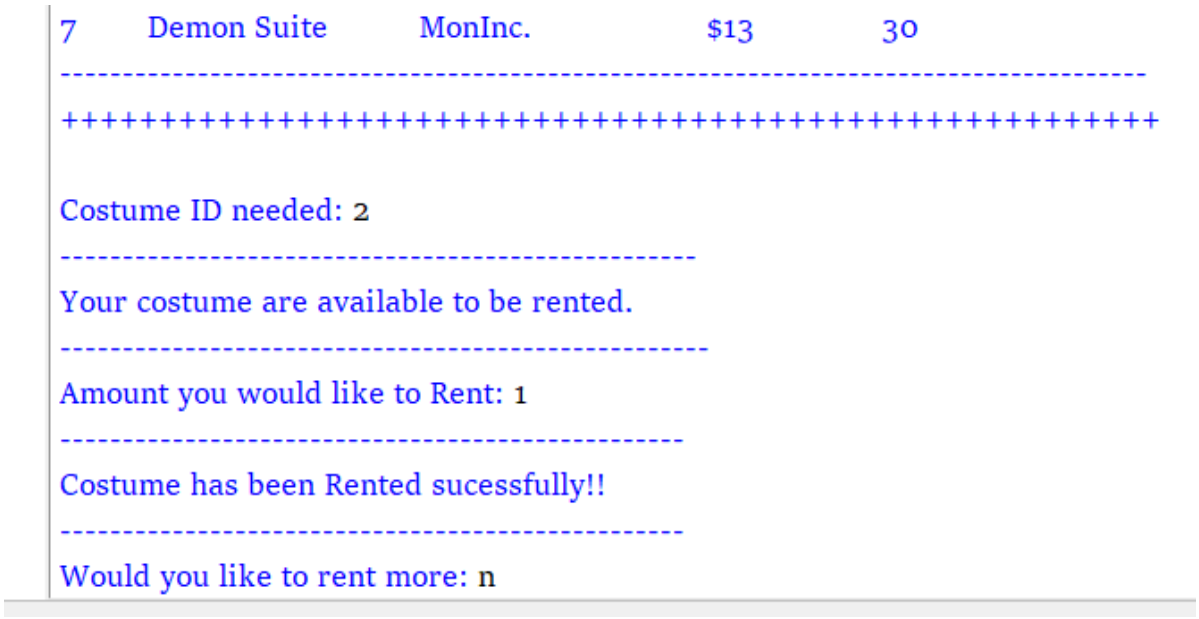


Figure 7: Renting one more costume and saying no to renting more

After renting the amount of costume desired the user is prompted to enter their name and contact number and if they are valid values, bills are printed accordingly with the help of data stored in the 2D list "**rented_items**".

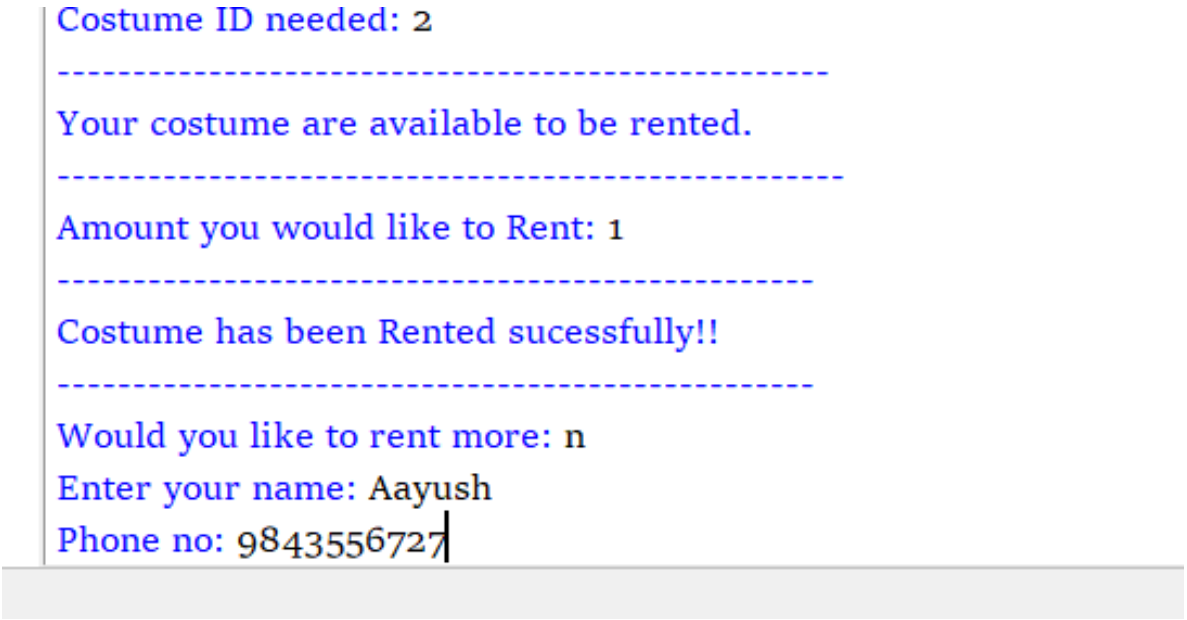


Figure 8: Entering personal information after the renting process is complete

Figure 9: Printed bill after renting a costume

For returning the costume similar process are used. The extra step in rent is calculation of days it was rented for and calculation of file. The fine is applied if the costume is rented for more than 5 days, and the amount fined is $0.25 for each costume and extra day. If any invalid values are entered by the user here the program loops and asks for the users input again.

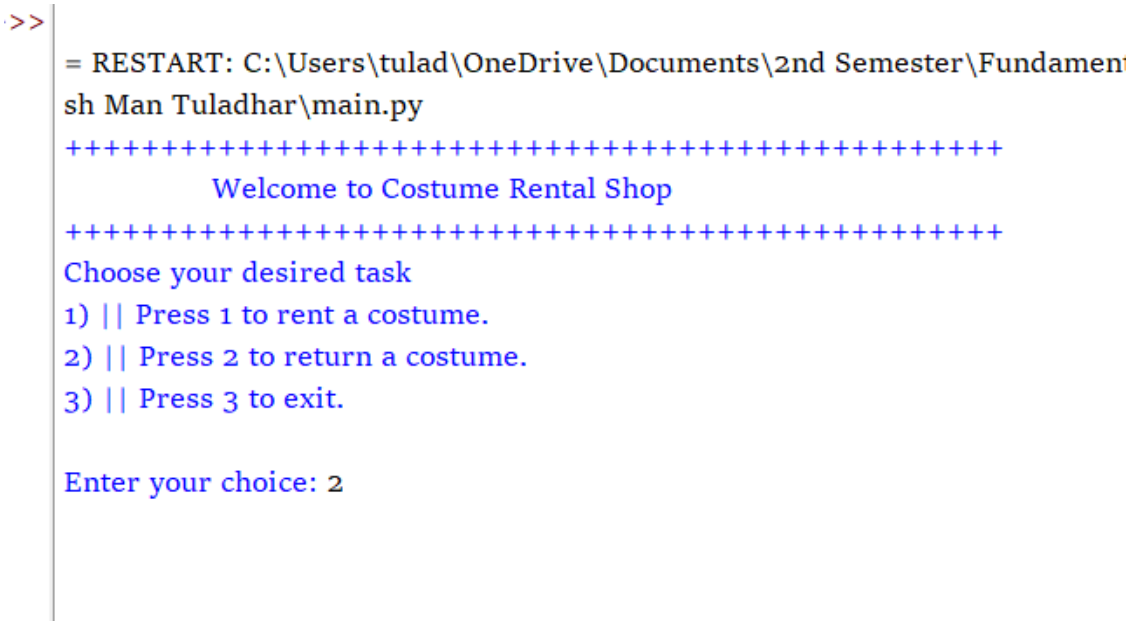


Figure 10: Entering value 2 to return a rented costume

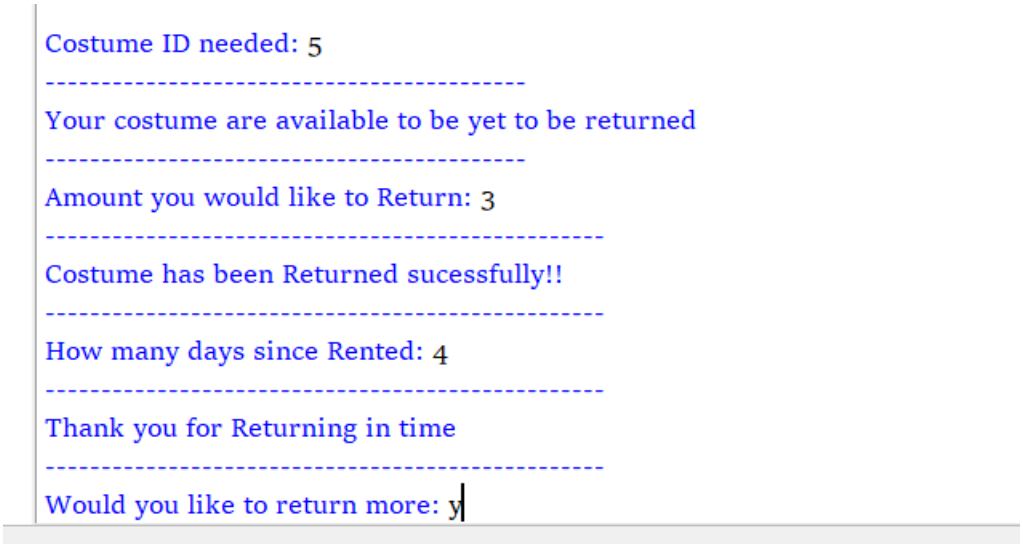


Figure 11: Entering appropriate values to return the costume and saying yes to returning more

```
7       Demon Suite         MonInc.              $13          30
-------------------------------------------------------------------------------
+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

Costume ID needed: 5
-----------------------------------------
Your costume are available to be yet to be returned
-----------------------------------------
Amount you would like to Return: 2
-------------------------------------------------
Costume has been Returned sucessfully!!
-------------------------------------------------
How many days since Rented: 10
-------------------------------------------------
You will be Fined for Returning Late
-------------------------------------------------
Would you like to return more: n
```

Figure 12: Entering the values to return a costume and saying no to returning more costumes

```
-------------------------------------------------
You will be Fined for Returning Late
-------------------------------------------------
Would you like to return more: n
Enter your name: Aayush
Phone no: 9976636522
-------------------------------------------------
```

Figure 13: Entering personal info for the purposes of Billing

Figure 14: Bill being printed for the return of costumes

3.3. Rent Bill and Return Bill

One of the requirements of the course work is to generate a bill in a different text file with the name, contact and date of the transaction that occurred. For renting a small bill is printed in the IDLE with the name, contact, grand total, and the details of the costume. With the same info gathered an external text file is created and is overridden and an invoice is printed in the in the text file accordingly. Below are the images of the Rent Bill generated by the file **"Rent_Bill.py"** and **"Rent_Validation"**.

```
------------------------------------------------
Costume has been Rented sucessfully!!
------------------------------------------------
Would you like to rent more: n
Enter your name: Aayush
Phone no: 9847473281
------------------------------------------------------------------
                        Your Rent Bill
------------------------------------------------------------------
Customer Name:  Aayush
Contact :  9847473281
Rented Date:  25/8/2022
Your Items:
[3, 'Fairy Costume', 1]
[2, 'Formal Suite', 1]
------- ----------------------------------------------------------
Grand total: $ 32.0
------------------------------------------------------------------
```

Figure 15: Printing Bill after renting costume successfully

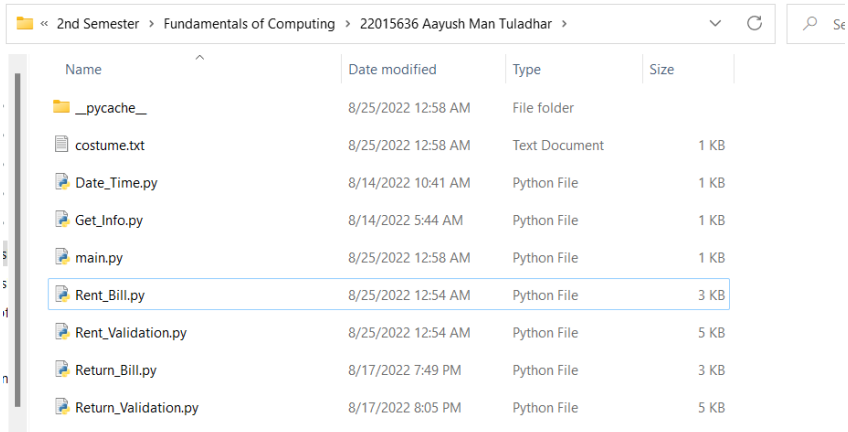
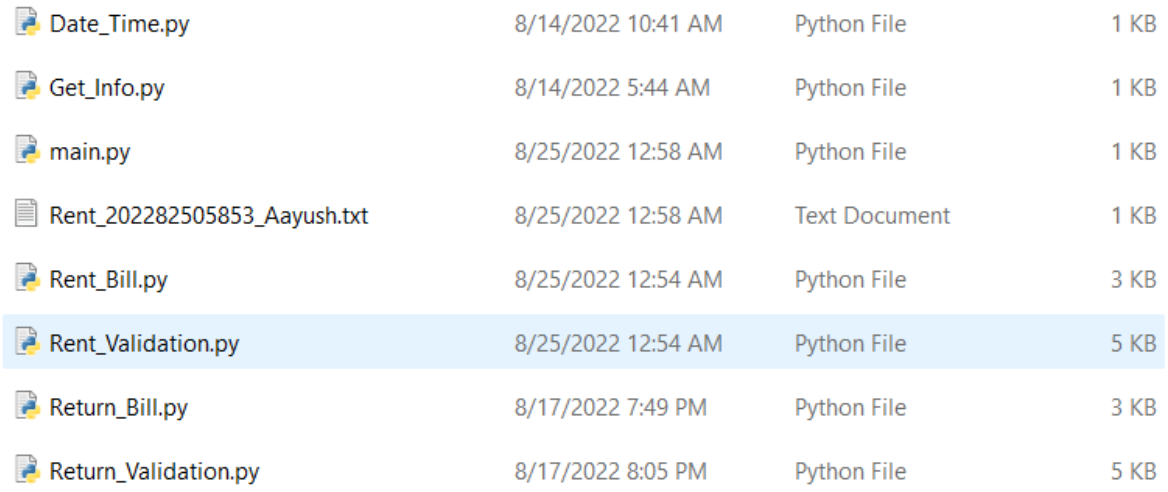
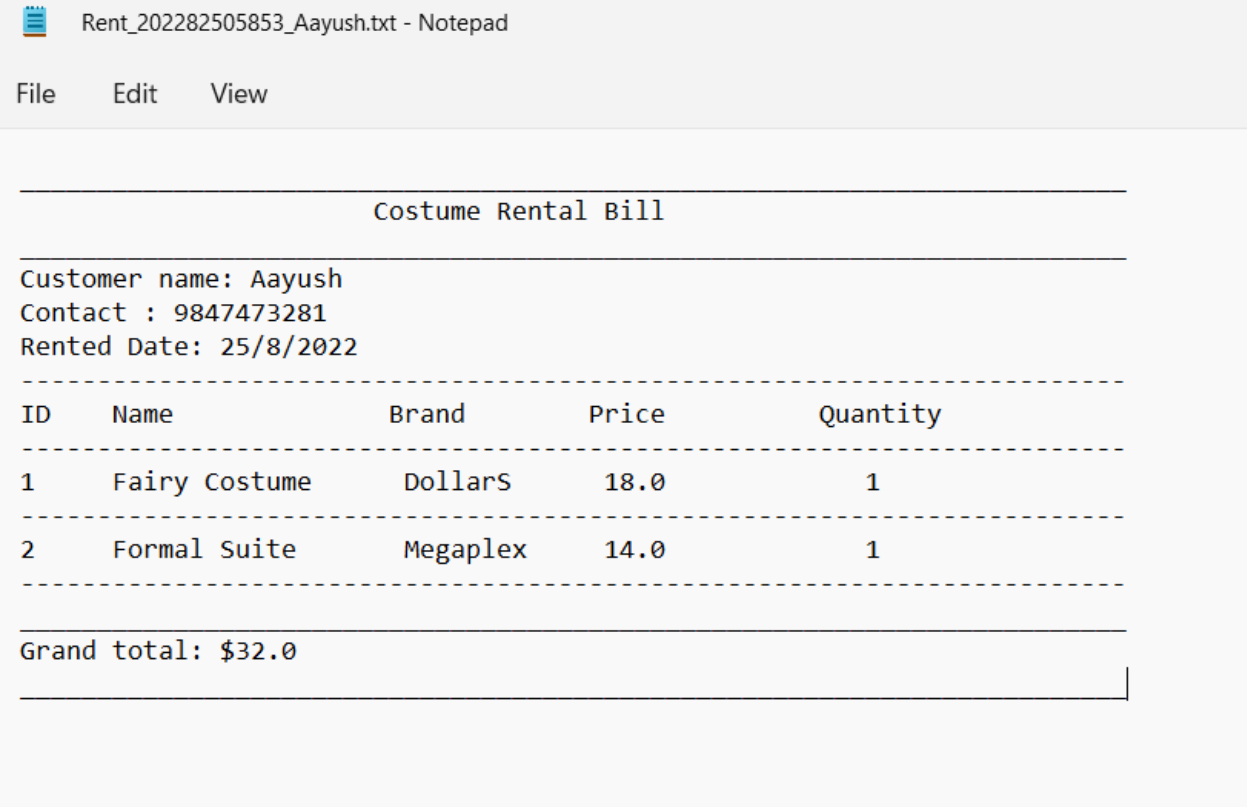


Figure 16: Before the bill was generated



Date_Time.py	8/14/2022 10:41 AM	Python File	1 KB
Get_Info.py	8/14/2022 5:44 AM	Python File	1 KB
main.py	8/25/2022 12:58 AM	Python File	1 KB
Rent_202282505853_Aayush.txt	8/25/2022 12:58 AM	Text Document	1 KB
Rent_Bill.py	8/25/2022 12:54 AM	Python File	3 KB
Rent_Validation.py	8/25/2022 12:54 AM	Python File	5 KB
Return_Bill.py	8/17/2022 7:49 PM	Python File	3 KB
Return_Validation.py	8/17/2022 8:05 PM	Python File	5 KB

Figure 17: After the Bill is Generated



Rent_202282505853_Aayush.txt - Notepad

File Edit View

```
_____________________________________________________________________________
                      Costume Rental Bill
_____________________________________________________________________________
Customer name: Aayush
Contact : 9847473281
Rented Date: 25/8/2022
-----------------------------------------------------------------------------
ID    Name              Brand        Price          Quantity
-----------------------------------------------------------------------------
1     Fairy Costume      DollarS      18.0             1
-----------------------------------------------------------------------------
2     Formal Suite       Megaplex     14.0             1
-----------------------------------------------------------------------------
_____________________________________________________________________________
Grand total: $32.0
_____________________________________________________________________________|
```

Figure 18: Bill generated in Notepad

The bill for returning the costumes is also very similar in nature but with some key differences. It also asks for the same user info as name and contact but also calculates the days for which the items have been rented for. If any costume has been rented for more than 5 days appropriate amount of fine is needed to be paid. The files that are used for this process are "**Return_Vaidation.py**" and "**Return_Bill.py**". Below are the images of bill being printed and generated in a new text file as well as in IDLE.

```
------------------------------------------------
Would you like to return more: n
Enter your name: Aayush
Phone no: 9876525353
---------------------------------------------------------------------
                    Your Return Bill
---------------------------------------------------------------------
Customer Name:  Aayush
Contact:  9876525353
Returned Date:  25/8/2022
List of Items Returned:
[2, 'Formal Suite', 8]
[2, 'Formal Suite', 4]
------- -------------------------------------------------------------
Fine: $ 20.0
---------------------------------------------------------------------
```

Figure 19: Bill printed after costume is returned successfully

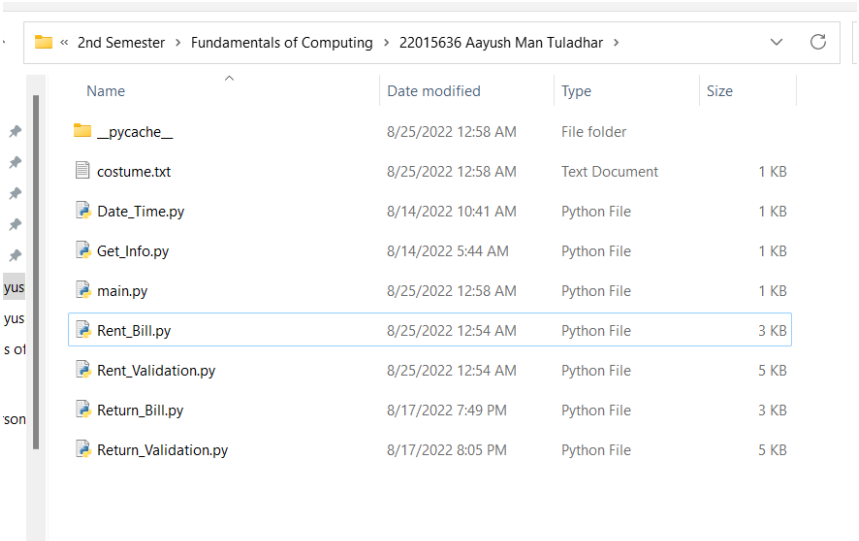


Figure 20: Before the text file for the bill was generated

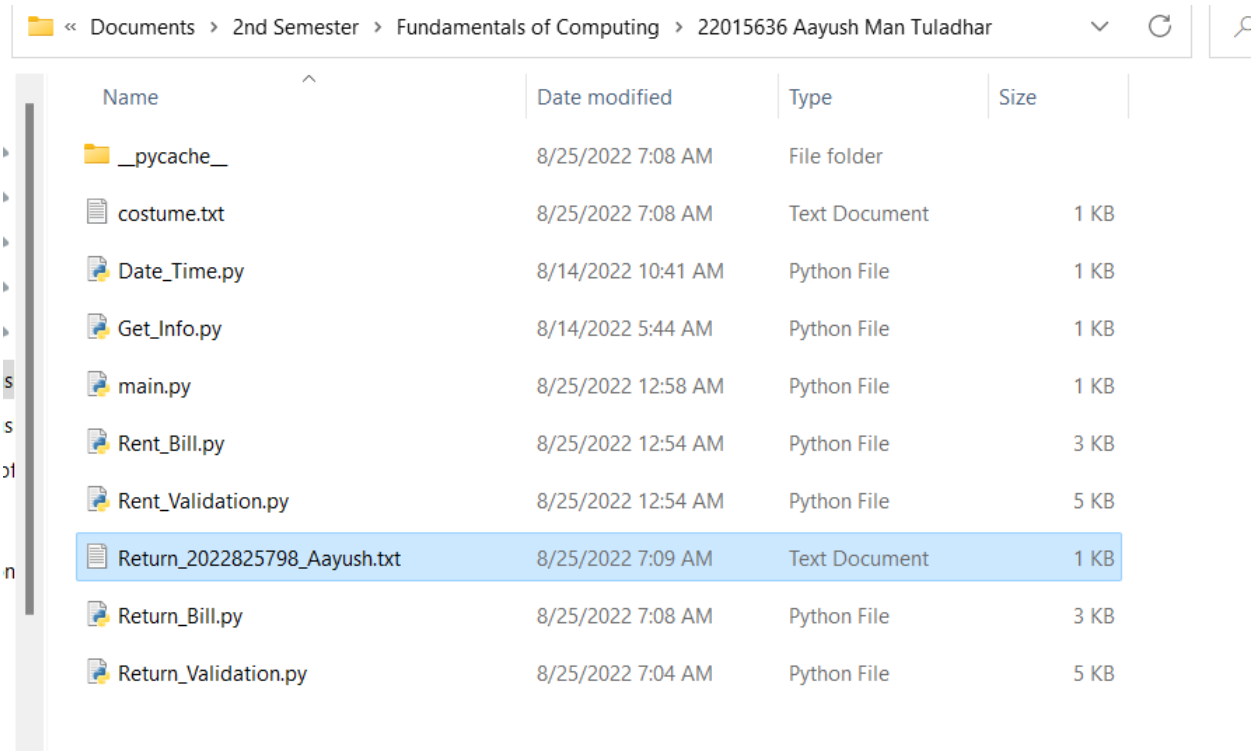


Figure 21: Text file being generated for invoice of returned items

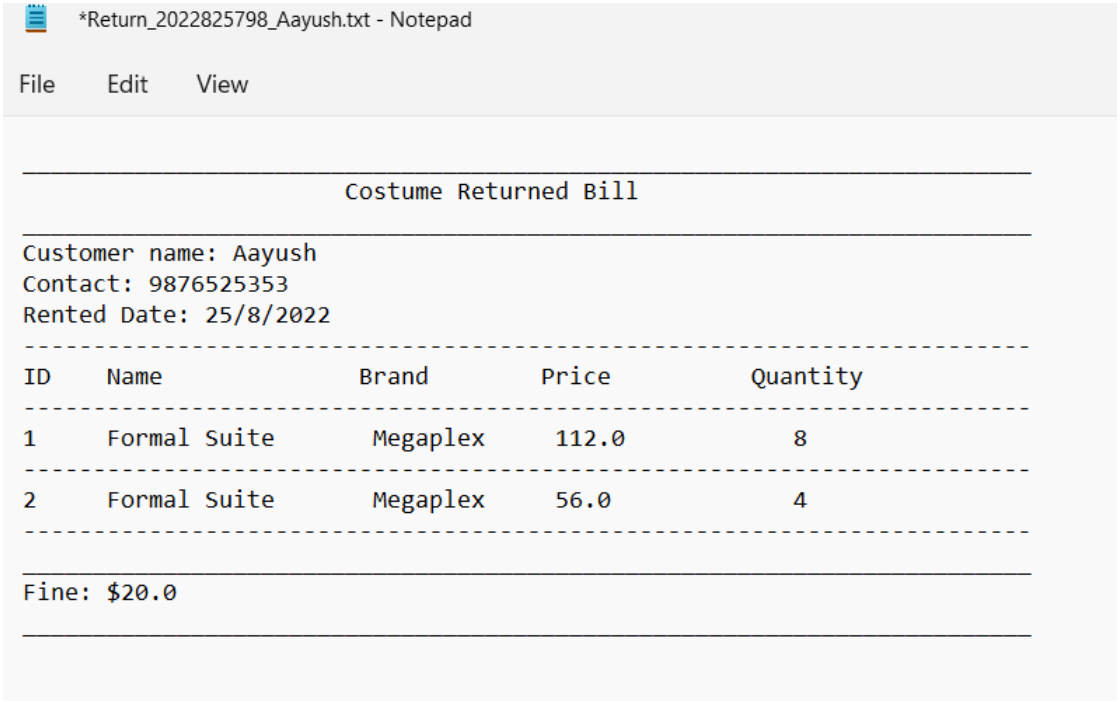


Figure 22: Invoice generated for the returned items

3.4. Exiting the Program

This an essential part of the program as a program needs to be terminated after it has already been used. Here there are 3 options available as shown before number 1 performs the act of renting whereas number 2 performs the act of returning the rented costume. So, at last the number 3 option is used to terminate the program.

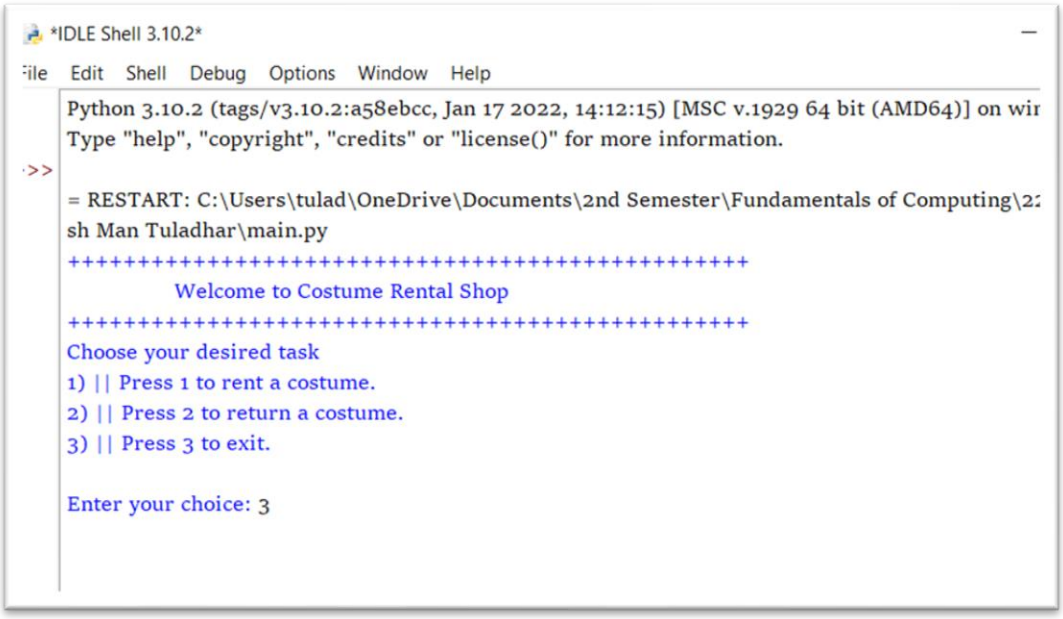


Figure 24: Entering the value to exit the program

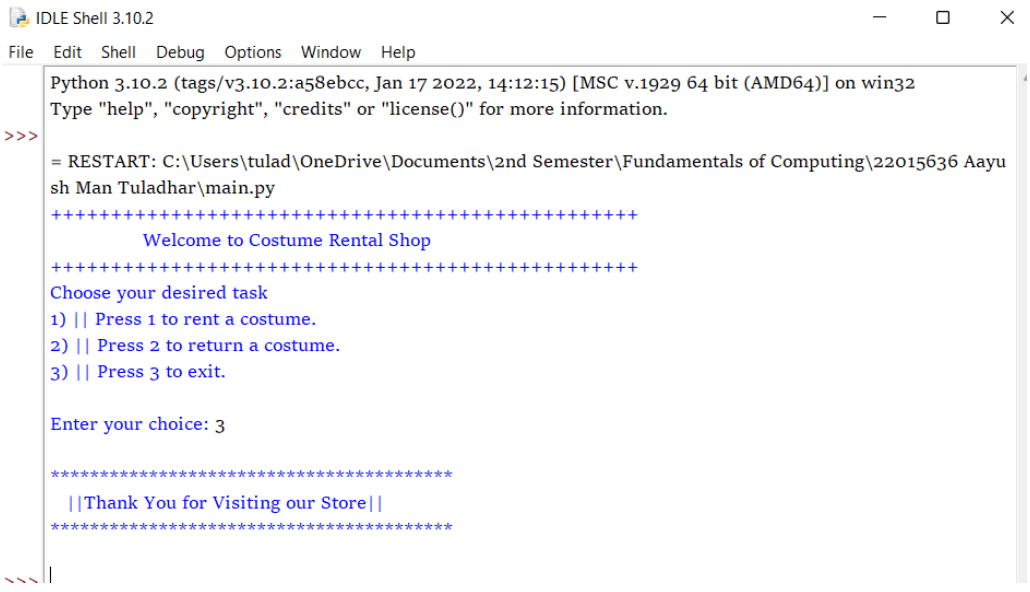


Figure 23: Exiting the program and displaying appropriate dialog

4. Testing

4.1. Test 1

Objective	To handle invalid input.
Action	• Run the program • Input alphabet as option • Input invalid number as option.
Expected Result	The invalid input is to be entered, and the appropriate message was displayed and loop for input was continued until the right values were entered.
Actual Result	The invalid input was entered, and the appropriate message was displayed and loop for input was continued until the right values were entered.
Conclusion	Objective completed.

Table 1: Testing invalid and alphabet as your choice

Figure 25: Entering unavailable option in input

Figure 26: Entering alphabets in input area

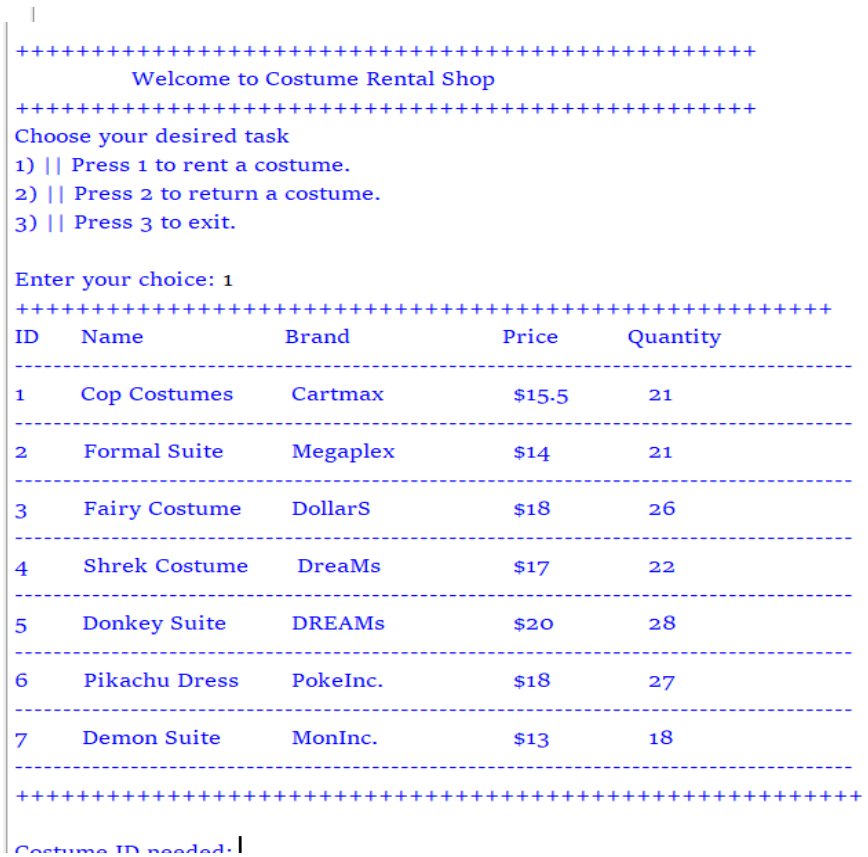


Figure 27: Entering the correct values

4.2. Test 2

Objective	To test the renting and returning capabilities.
Action	• Run the program • Input negative value while renting • Input non-existing value while renting • Input negative value while returning • Input non-existing value while returning.
Expected Result	When the appropriate values are entered the costumes are rented or returned as per the choice of the customer.
Actual Result	When the appropriate values were entered the costumes are rented or returned as per the choice of the customer.
Conclusion	Objective completed

Table 2: Testing selecting Renting and Returning

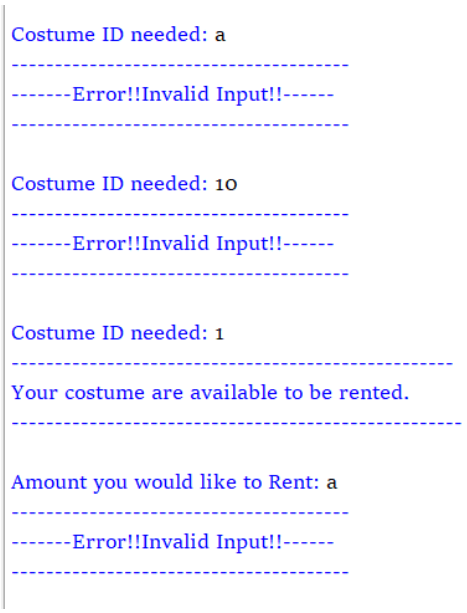


Figure 28: Entering negative values in Rent

Figure 29: Entering unavailable values in rent options

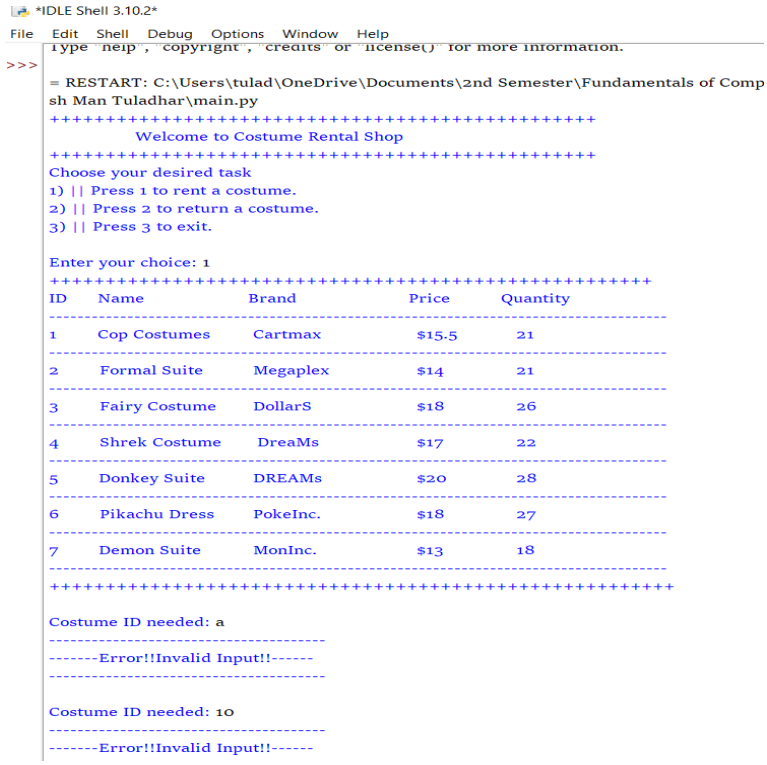


Figure 30: Entering alphabets values in Returning options

Figure 31: Entering negative values in Returning options

Figure 32: Entering non-existent values in Returning options

4.3. Test 3

Objective	Generate bill after renting a costume.
Action	• Run the Program • Input option “1” • Rent any costume no. 1 with amount entered as 2 • Enter your name and contact • Exit Program • Open “Rent_202282661411_Aayush.txt” and invoice should be printed.
Expected Result	If the costume was rented a bill or an invoice should be generated with an appropriate format and information.
Actual Result	After the costume was rented a bill or an invoice was generated with an appropriate format and information.
Conclusion	Objective completed

Table 3: Testing File generation of Renting

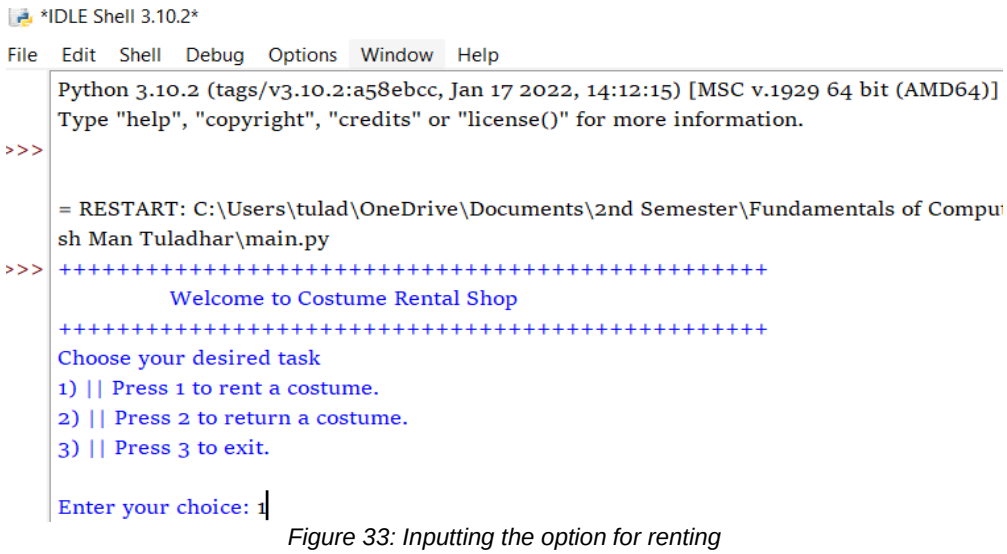


Figure 33: Inputting the option for renting

```
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
ID    Name              Brand              Price     Quantity
----------------------------------------------------------------------------
1     Cop Costumes      Cartmax            $15.5     21
----------------------------------------------------------------------------
2     Formal Suite      Megaplex           $14       21
----------------------------------------------------------------------------
3     Fairy Costume     DollarS            $18       26
----------------------------------------------------------------------------
4     Shrek Costume     DreaMs             $17       22
----------------------------------------------------------------------------
5     Donkey Suite      DREAMs             $20       28
----------------------------------------------------------------------------
6     Pikachu Dress     PokeInc.           $18       27
----------------------------------------------------------------------------
7     Demon Suite       MonInc.            $13       18
----------------------------------------------------------------------------
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

Costume ID needed: 1
-------------------------------------------------
Your costume are available to be rented.
-------------------------------------------------

Amount you would like to Rent: 2
```

Figure 34: Renting costume no. 1 and 2 of them

```
Enter your name: Aayush
Phone no: 9876651556

-----------------------------------------------------------------------
                        Your Rent Bill
-----------------------------------------------------------------------
Customer Name:  Aayush
Contact :  9876651556
Rented Date:  26/8/2022
Your Items:
[1, 'Cop Costumes', 2]
------- ---------------------------------------------------------------
Grand total: $ 31.0
-----------------------------------------------------------------------
```

Figure 35: Bill being printed after renting

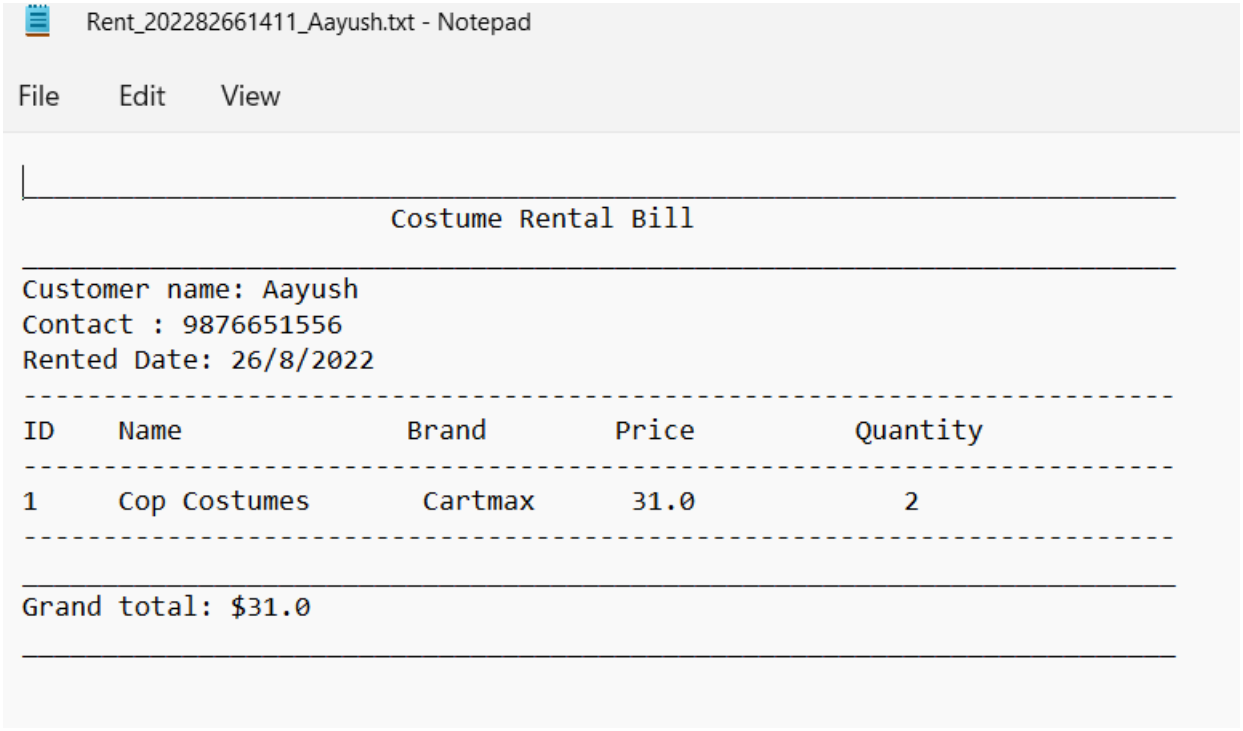


Figure 36: Opening Rent Bill printing in a .txt file

4.4. Test 4

Objective	Generate bill after returning a costume.
Action	• Run the Program • Input option “2” • Return any costume no. 1 with amount entered as 2 • Enter your name and contact • Exit Program • Open “Return_202282662553_Aayush.txt” and invoice should be printed.
Expected Result	If the costume was returned a bill or an invoice should be generated with an appropriate format and information.
Actual Result	After the costume was returned a bill or an invoice was generated with an appropriate format and information.
Conclusion	Objective completed

Table 4: Testing File generation of Returning

```
++++++++++++++++++++++++++++++++++++++++++++++++++
          Welcome to Costume Rental Shop
++++++++++++++++++++++++++++++++++++++++++++++++++
Choose your desired task
1) || Press 1 to rent a costume.
2) || Press 2 to return a costume.
3) || Press 3 to exit.

Enter your choice: 2
```

Figure 37: Entering value to return the costume

```
Enter your choice: 2
+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
ID    Name              Brand              Price      Quantity
-------------------------------------------------------------------------------
1     Cop Costumes      Cartmax            $15.5      19
-------------------------------------------------------------------------------
2     Formal Suite      Megaplex           $14        21
-------------------------------------------------------------------------------
3     Fairy Costume     DollarS            $18        26
-------------------------------------------------------------------------------
4     Shrek Costume     DreaMs             $17        22
-------------------------------------------------------------------------------
5     Donkey Suite      DREAMs             $20        28
-------------------------------------------------------------------------------
6     Pikachu Dress     PokeInc.           $18        27
-------------------------------------------------------------------------------
7     Demon Suite       MonInc.            $13        18
-------------------------------------------------------------------------------
+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

Costume ID needed: 1
----------------------------------------
Your costume are available to be yet to be returned
----------------------------------------

Amount you would like to Return: 2
--------------------------------------------------
Costume has been Returned sucessfully!!
--------------------------------------------------

How many days since Rented: 7
```

Figure 38: Entering what and which costume are to be rented

```
Would you like to return more: n

Enter your name: Aayush
Phone no: 9978266199

----------------------------------------------------------------------
                    Your Return Bill
----------------------------------------------------------------------
Customer Name:  Aayush
Contact:  9978266199
Returned Date:  26/8/2022
List of Items Returned:
[1, 'Cop Costumes', 2]
------- --------------------------------------------------------------
```

Figure 39: Entering name and contact to print Return Bill.

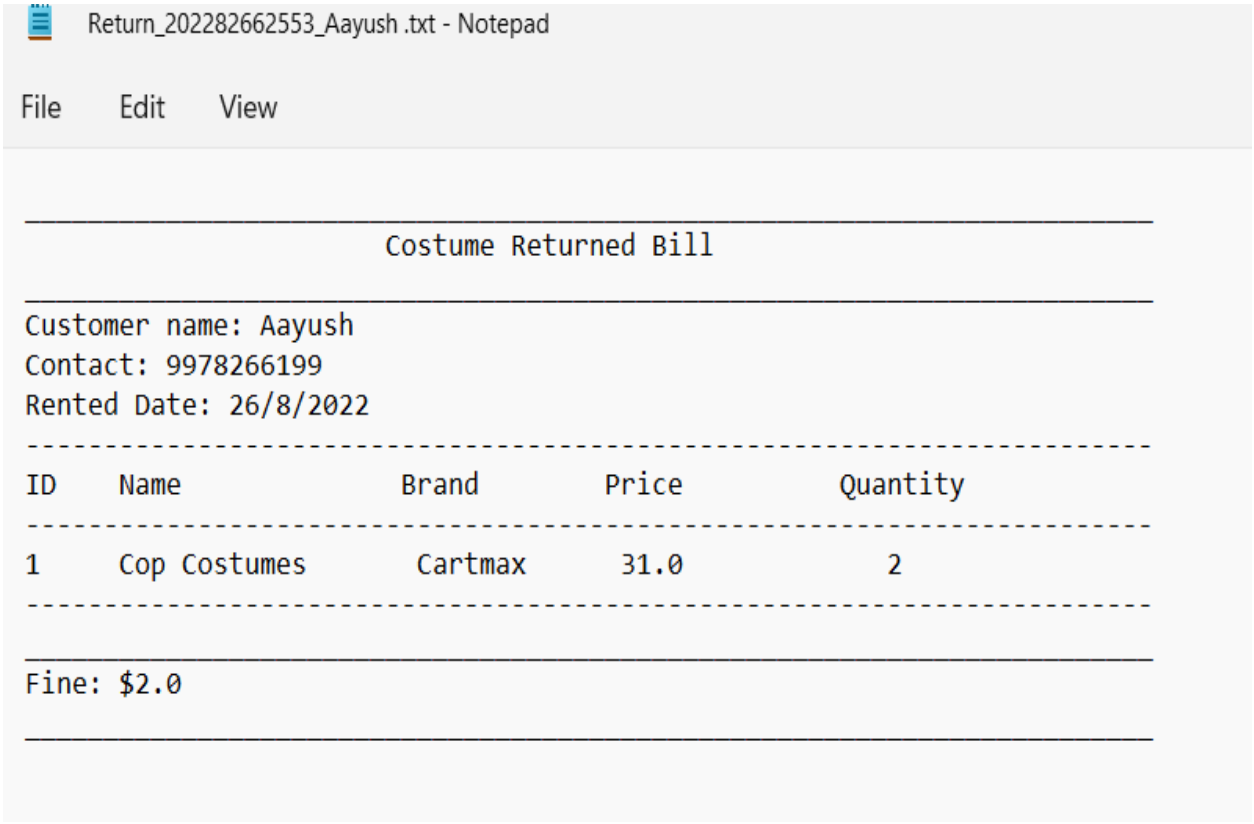


Figure 40: Return Bill Generated in a separated text file

4.5. Test

Objective	To re-write the stock of the costume after renting or returning.
Action	• Run the Program • Input option “1” • Rent any costume no. 1 with amount entered as 2 • Enter your name and contact • Exit Program • Open “costume.txt” and check if quantity decreased by 2 • Again, Run the Program • Input option “2” • Return any costume no. 1 with amount entered as 2 • Enter your name and contact • Exit Program • Open “costume.txt” and check if quantity increased by 2
Expected Result	If the costume was rented the stock should decrease and if the costumes were returned the stock should increase.
Actual Result	If the costume was rented the stock should decrease and if the costumes were returned the stock should increase.
Conclusion	Objective completed

Table 5: Testing the update of stocks

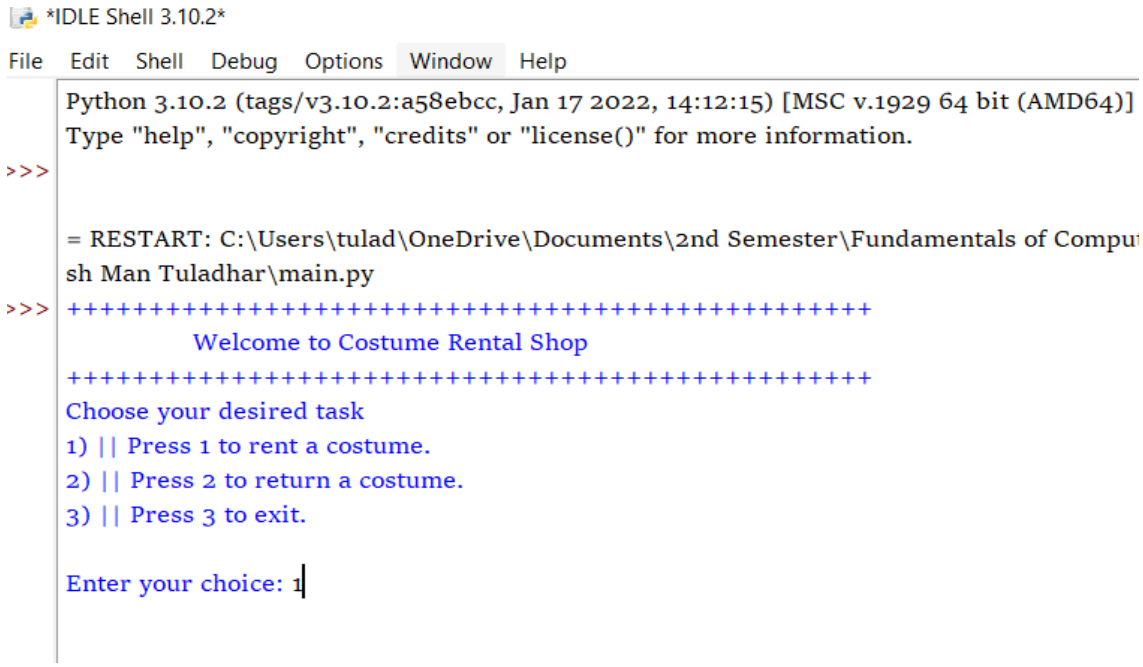


Figure 41: Entering 1 to go to Renting options

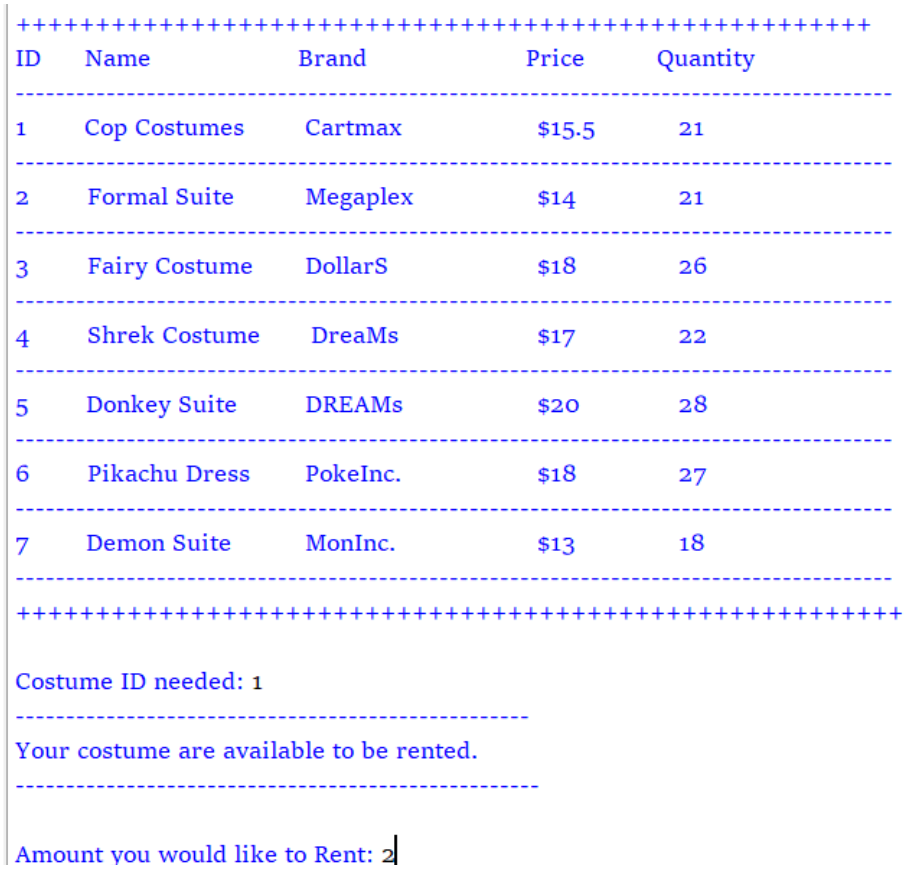


Figure 42: Entering the sno and amount to Rent

```
Enter your name: Aayush
Phone no: 9876651556

---------------------------------------------------------------------
                        Your Rent Bill
---------------------------------------------------------------------
Customer Name:  Aayush
Contact :  9876651556
Rented Date:  26/8/2022
Your Items:
[1, 'Cop Costumes', 2]
------- -------------------------------------------------------------
Grand total: $ 31.0
---------------------------------------------------------------------
```

Figure 43: Printing Rent Bill in Idle

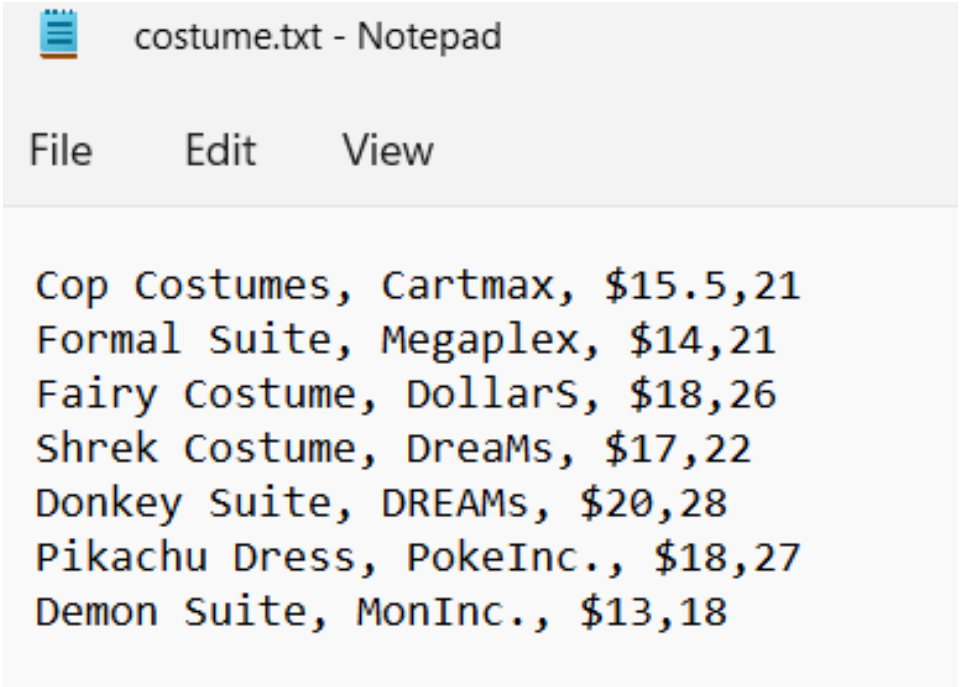


Figure 44: Checking value before renting

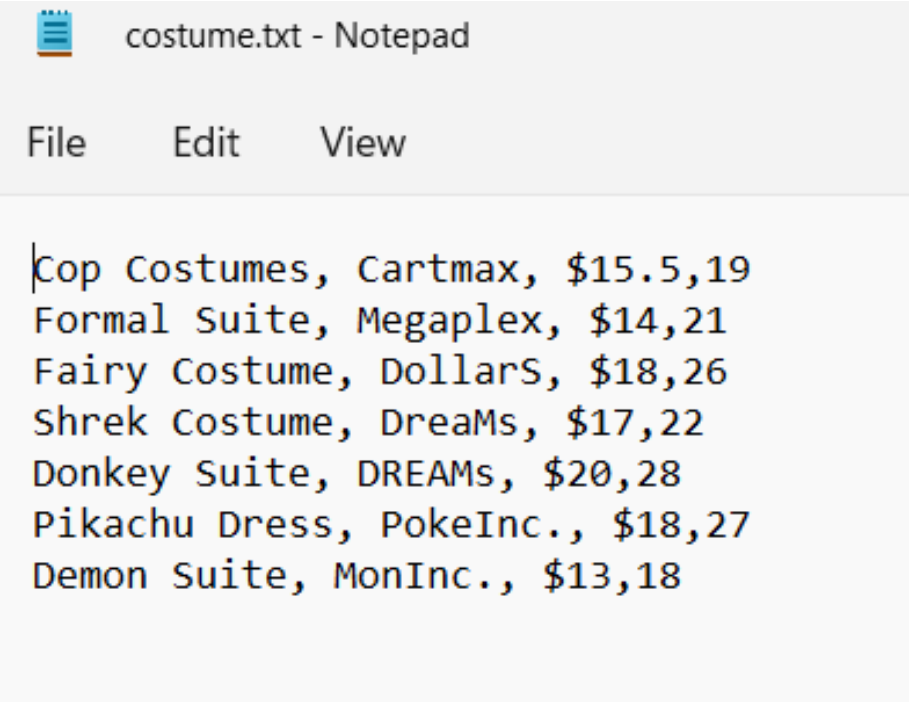


Figure 45: Values after renting

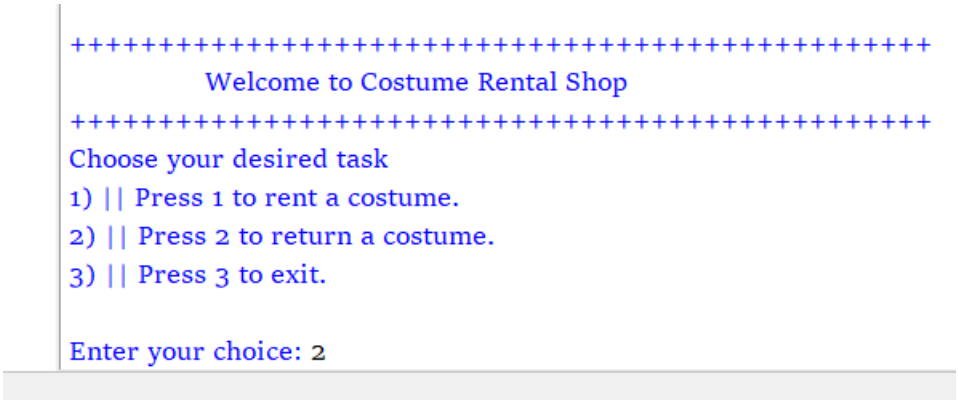


Figure 46: Entering value 2 to go to Returning options

```
Enter your choice: 2
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
ID    Name              Brand              Price      Quantity
-------------------------------------------------------------------------------
1     Cop Costumes      Cartmax            $15.5      19
-------------------------------------------------------------------------------
2     Formal Suite      Megaplex           $14        21
-------------------------------------------------------------------------------
3     Fairy Costume     DollarS            $18        26
-------------------------------------------------------------------------------
4     Shrek Costume     DreaMs             $17        22
-------------------------------------------------------------------------------
5     Donkey Suite      DREAMs             $20        28
-------------------------------------------------------------------------------
6     Pikachu Dress     PokeInc.           $18        27
-------------------------------------------------------------------------------
7     Demon Suite       MonInc.            $13        18
-------------------------------------------------------------------------------
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

Costume ID needed: 1
-----------------------------------------
Your costume are available to be yet to be returned
-----------------------------------------

Amount you would like to Return: 2
-------------------------------------------------
Costume has been Returned sucessfully!!
-------------------------------------------------

How many days since Rented: 7
```

Figure 47: Returning the same costume previously rented

```
Enter your name: Aayush
Phone no: 9876651556

---------------------------------------------------------------------
                        Your Rent Bill
---------------------------------------------------------------------
Customer Name:  Aayush
Contact :  9876651556
Rented Date:  26/8/2022
Your Items:
[1, 'Cop Costumes', 2]
------- -------------------------------------------------------------
Grand total: $ 31.0
---------------------------------------------------------------------
```

Figure 48: Printing the Returning Bill

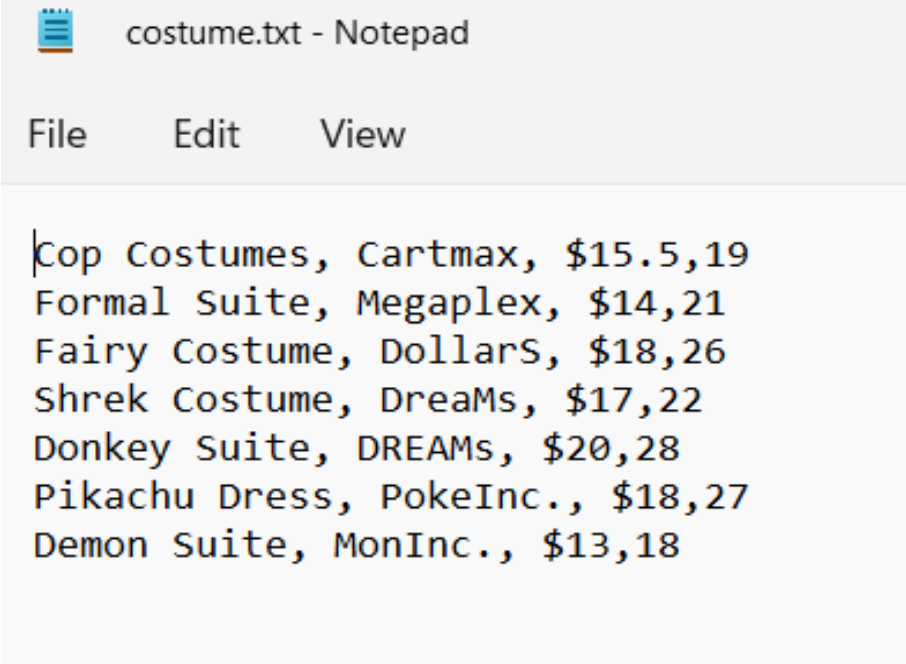


Figure 49: Values before returning

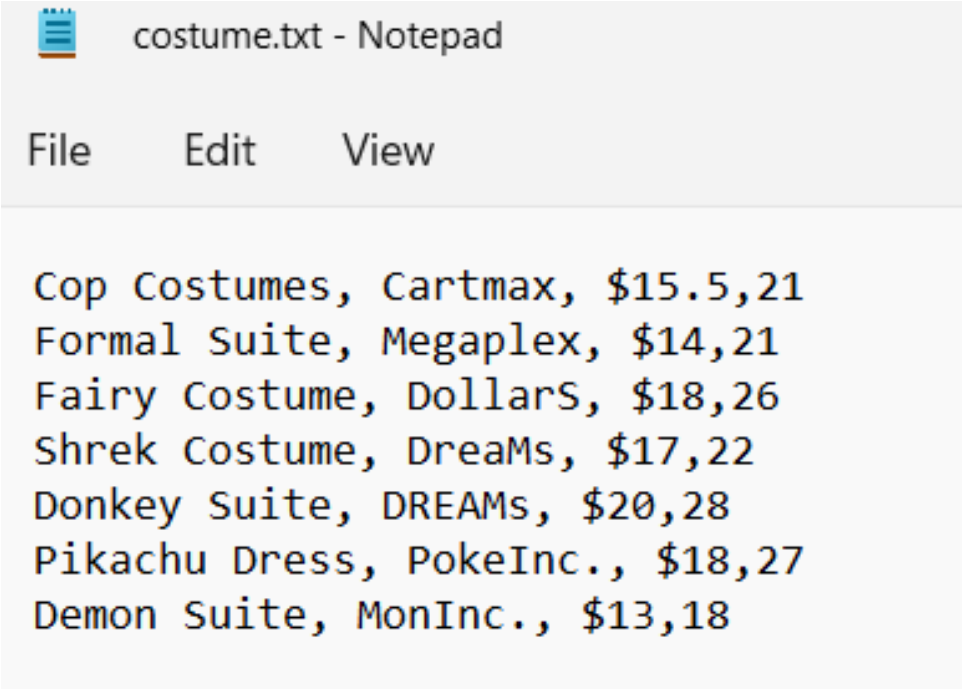


Figure 50: value after Returning

Conclusion

Throughout the course of the this report I have gone through many mishaps even though I have a good grip on the concepts of programming in Python. The goal of the project is to create a costume rental system that records the renting and returning of the costumes by the customer and displays the appropriate details about their purchase. Each transaction should come with an invoice with all the details of the purchase and or the costume returned. If returned late appropriate amount of fine should be applied to the customer.

In the beginning, I was not very confident in my ability to complete this project but as I progressed through the course, I understood the concepts and the nuances of programming in python. Since, python is a very abstract programming language it has a more understandable syntax and is easy to remember. My peers that helped me through this ordeal were helpful and the lecturers were good at conveying the ideas and concept in a way the was digestible for me. One of the key things I learned during this project is data management and how you can execute it in a realistic and modular manner. While programming data is a huge part of the equation and the more you can manage and sort data efficiently the better, you'll be at solving real-world problems. The practical and real-world problem provided to us through the coursework will help us adapt and problem solve other problems well within our future. Due to this my motivation for the next coursework has increased and I. looking forward to more of the same or even different challenges. During this coursework I learned many things like time management, critical thinking and problem solving.

The material provided was clear, concise and to the point. The teachers have been a great guide while going through this unfamiliar terrain. I've learned all that I can here and will continue to do so. The learning for this module has not been a rough one as the content is not very to grasp. It has been a lot of understanding the material or looking at the material and trying to do it myself to understand.

Currently in the module I have not had a problem that has been a roadblock in my learning. There has been some minor in conveniences in the way, but I have understood it better. The part I had the most problem with was not understanding the questions provided to me in the beginning. Although it was something I struggled with it at the beginning I have become more accustomed to it.

Throughout this journey for the coursework, I had a lot of positive experience. The benefit of this coursework far outweighs the cons. This project has reignited my joy of programming. I felt like I was learning something new every step of the way. I feel that I am more ready than ever to progress forward in the field of programming and learning more nuanced ideas about programming.

References

Braunschweig, K. L. B. a. D., 2022. *Rebus Community.* [Online]
Available at: https://press.rebus.community/programmingfundamentals/chapter/string-data-type/

[Accessed Friday August 2022].
Falconer, J., 2021. *Sitepoint.* [Online]
Available at: https://www.sitepoint.com/boolean-data-type/
[Accessed Friday August 2022].

Appendix

Main.py

```
import Rent_Validation
import Return_Validation

def options():
    print('Choose your desired task')
    print('1) || Press 1 to rent a costume.')
    print('2) || Press 2 to return a costume.')
    print('3) || Press 3 to exit.')
    print()

ExceptionLoop = True
while  ExceptionLoop == True:
    try:
        continueLoop  = True
        while continueLoop == True:
            print('++++++++++++++++++++++++++++++++++++++++++++++++++\n
Welcome                    to                    Costume                    Rental
Shop\n++++++++++++++++++++++++++++++++++++++++++++++++++')
            options()
            choice = int(input("Enter your choice: "))
            if choice == 1:
                Rent_Validation.Renting()

            elif choice == 2:
                Return_Validation.Returning()

            elif choice == 3:
```

```
            continueLoop = False
            print()
            print("****************************************")
            print("   ||Thank You for Visiting our Store||")
            print("****************************************")
            print()

        else:
            print("--------------------------------------")
            print("    Enter the available options")
            print("--------------------------------------")
            print()
        ExceptionLoop = False
    except:
        print()
        print("--------------------------------------")
        print("-------Error!!Invalid Input!!------")
        print("--------------------------------------")
        print()
```

Get_Info.py

```
def get_file_info():
    file = open("costume.txt","r")
    data = file.readlines()
    file.close()
    return data

def get_dict_info(file_info):
```

```
    data_dict = {}
    for index in range(len(file_info)):
        data_dict[index+1] = file_info[index].replace("\n","").split(",")
    return data_dict

def costumes_info():
    file_info = get_file_info()
    main_data = get_dict_info(file_info)
    print("+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++")
    print("ID","    ","Name","\t","  ","Brand","\t","               ","Price","\t","Quantity")
    print("-------------------------------------------------------------------------------------")
    for key,value in main_data.items():
        print(key,"      ",value[0],"\t","  ",value[1],"\t",value[2],"\t","  ",value[3])
        print("-------------------------------------------------------------------------------------")
    print("+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++")
```

Date_Time.py

```
import datetime

def get_datetime():
    year = str(datetime.datetime.now().year)
    month = str(datetime.datetime.now().month)
    day = str(datetime.datetime.now().day)
    hour = str(datetime.datetime.now().hour)
    minute = str(datetime.datetime.now().minute)
    second = str(datetime.datetime.now().second)

    date_time = year+month+day+hour+minute+second
    return date_time

def dates():
    year = str(datetime.datetime.now().year)
    month = str(datetime.datetime.now().month)
    day = str(datetime.datetime.now().day)

    date_only = day+"/"+month+"/"+year
    return date_only
```

Rent_Validation.py

```
import Get_Info
import Rent_Bill

def validation_of_id():
    file_info = Get_Info.get_file_info()
```

```
    main_data = Get_Info.get_dict_info(file_info)
    valid_input = False
    while valid_input == False:
        ExceptionLoop =True
        while  ExceptionLoop == True:
            try:
                sno = int(input("Costume ID needed: "))
                ExceptionLoop = False
            except:
                print("---------------------------------------")
                print("-------Error!!Invalid Input!!------")
                print("---------------------------------------")
                print()
        if sno>0 and sno<=len(main_data):
            if  int(main_data[sno][3]) == 0:
                print("-------------------------------------------------")
                print("This costume is out of Stock")
                print("-------------------------------------------------")
                print()
                valid_input = False
            else:
                print("-------------------------------------------------")
                print("Your costume are available to be rented.")
                print("-------------------------------------------------")
                print()
                valid_input = True
        else:
                print("---------------------------------------")
                print("-------Error!!Invalid Input!!------")
                print("---------------------------------------")
```

```
            print()
            valid_input = False
    return sno

def validation_of_quantity(valid_id):
    file_info = Get_Info.get_file_info()
    main_data = Get_Info.get_dict_info(file_info)
    quantity = int(main_data[valid_id][3])
    valid_input = False
    while valid_input == False:
        ExceptionLoop =True
        while  ExceptionLoop == True:
            try:
                input_quantity = int(input("Amount you would like to Rent: "))
                ExceptionLoop =False
            except:
                print("----------------------------------------")
                print("-------Error!!Invalid Input!!------")
                print("----------------------------------------")
                print()
        if input_quantity >0 and input_quantity <=quantity:
            print("-------------------------------------------------")
            print("Costume has been Rented sucessfully!!")
            print("-------------------------------------------------")
            print()
            valid_input = True
        else:
            print("----------------------------------------")
            print("-------Error!!Invalid Input!!------")
            print("----------------------------------------")
```

```
            print()
    return input_quantity



def Renting():
    grand_total = 0
    price  = 0
    rented_items = []
    continueLoop = True
    while continueLoop == True:
        Get_Info.costumes_info()
        print()
        validID = validation_of_id()
        available = validation_of_quantity(validID)
        file_info = Get_Info.get_file_info()
        main_data = Get_Info.get_dict_info(file_info)
        no = main_data[validID][2].replace("$","")
        price = float(no)*int(available)
        grand_total += float(price)
        main_data[validID][3] = str(int(main_data[validID][3]) - available)

        file = open("costume.txt","w")
        for value in main_data.values():
            rewrite_data = value[0]+","+value[1]+","+value[2]+","+value[3]+"\n"
            file.write(rewrite_data)
        file.close()

        rented_items.append([validID,main_data[validID][0],available])
```

```
    x = False
    while x == False:
        repeat = input("Would you like to rent more: ")

        if repeat == "n":
            print()
            name = input("Enter your name: ")
            contact = input ("Phone no: ")
            print()
            Rent_Bill.bill_for_rent(name,contact,grand_total,rented_items)
            print()
            Rent_Bill.generate_bill(name, contact,grand_total,rented_items)
            continueLoop = False
            x = True
        else:
            continueLoop = True
            x = True
```

Rent_Bill.py

```
import Date_Time
import Get_Info


def bill_for_rent(name,contact,grand_total,rented_items):
    print("------------------------------------------------------------------------")
    print("                          Your Rent Bill                                ")
    print("------------------------------------------------------------------------")
    print("Customer Name: ", name)
    print("Contact : ",contact)
```

```
    print("Rented Date: ", Date_Time.dates())
    print("Your Items:")
    for items in rented_items:
        print(items)
    print("------- ------------------------------------------------------------")
    print("Grand total: $",grand_total)
    print("------------------------------------------------------------------")

def generate_bill(name, contact,
    grand_total,rented_items):
    file_info = Get_Info.get_file_info()
    main_data = Get_Info.get_dict_info(file_info)

    file = open("Rent"+"_"+Date_Time.get_datetime()+"_"+name+".txt","w")

file.write("_____
__________"+"\n")
    file.write("                    Costume Rental Bill                                "+"\n")

file.write("_____
__________"+"\n")
    file.write("Customer name: "+name+"\n")
    file.write("Contact : "+contact+"\n")
    file.write("Rented Date: "+Date_Time.dates()+"\n")
    file.write("------------------------------------------------------------------------"+"\n")
    file.write("ID"+"      "+"Name"+"\t"+"                "+"Brand"+"\t"+"          "+"Price"+"\t"+"
"+"Quantity"+"\n")
    file.write("------------------------------------------------------------------------"+"\n")
    for index in range(len(rented_items)):
        c_sno = int(rented_items[index][0])
```

```
        c_quantity = int(rented_items[index][2])
        c_name = main_data[c_sno][0]
        c_brand = main_data[c_sno][1]
        c_price = float(main_data[c_sno][2].replace("$","")) * c_quantity
        file.write(str(index+1)+"\t"+c_name+"\t"+c_brand+"\t"+"            "+str(c_price)+"\t"+"
"+str(c_quantity)+"\n")
        file.write("-------------------------------------------------------------------------"+"\n")

file.write("_____
_____"+"\n")

    file.write("Grand total: $"+str(grand_total)+"\n")

file.write("_____
_____"+"\n")
file.close()
```

Return_Validation.py

```
import Get_Info
import Return_Bill

def get_valid_id():
    file_info = Get_Info.get_file_info()
    main_data = Get_Info.get_dict_info(file_info)
    valid_input = False
    while valid_input == False:
        ExceptionLoop = True
        while ExceptionLoop == True:
```

```
            try:
                sno = int(input("Costume ID needed: "))
                ExceptionLoop = False
            except:
                print("---------------------------------------")
                print("-------Error!!Invalid Input!!------")
                print("---------------------------------------")
                print()
        if sno>0 and sno<=len(main_data):
            print("-----------------------------------------")
            print("Your costume are available to be yet to be returned")
            print("-----------------------------------------")
            print()
            valid_input = True
        else:
            print("---------------------------------------")
            print("-------Error!!Invalid Input!!------")
            print("---------------------------------------")
            print()
    return sno

def get_valid_quantity(valid_id):
    file_info = Get_Info.get_file_info()
    main_data = Get_Info.get_dict_info(file_info)
    quantity = int(main_data[valid_id][3])
    valid_input = False
    while valid_input == False:
        ExceptionLoop = True
        while ExceptionLoop == True:
            try:
```

```
            input_quantity = int(input("Amount you would like to Return: "))
            ExceptionLoop = False
        except:
            print("---------------------------------------")
            print("-------Error!!Invalid Input!!------")
            print("---------------------------------------")
            print()
    if input_quantity >0 :
        print("-------------------------------------------------")
        print("Costume has been Returned sucessfully!!")
        print("-------------------------------------------------")
        print()
        valid_input = True
    else:
        print("---------------------------------------")
        print("-------Error!!Invalid Input!!------")
        print("---------------------------------------")
        print()
  return input_quantity

def Days_Checker():
  y = True
  while y == True:
    rented_days = int(input("How many days since Rented: "))
    if rented_days > 5:
        print("-------------------------------------------------")
        print("You will be Fined for Returning Late")
        print("-------------------------------------------------")
        print()
        y = False
```

```
        else:
            print("-------------------------------------------------")
            print("Thank you for Returning in time")
            print("-------------------------------------------------")
            print()
            y = False
        return rented_days

returned_items = []
def Returning():
    fine = 0.25
    total_fine = 0

    continueLoop = True
    while continueLoop == True:
        Get_Info.costumes_info()
        print()
        validID = get_valid_id()
        available = get_valid_quantity(validID)
        days = Days_Checker()
        file_info = Get_Info.get_file_info()
        main_data = Get_Info.get_dict_info(file_info)

        total_fine = total_fine*((days-5)*fine)
        total_fine += available

        no = main_data[validID][2].replace("$","")


        main_data[validID][3] = str(int(main_data[validID][3]) + available)
```

```
file = open("costume.txt","w")
for value in main_data.values():
    rewrite_data = value[0]+","+value[1]+","+value[2]+","+value[3]+"\n"
    file.write(rewrite_data)
file.close()

returned_items.append([validID,main_data[validID][0],available])

x = False
while x == False:
    ExceptionLoop =True
    while ExceptionLoop == True:
        try:
            repeat = input("Would you like to return more: ")
            ExceptionLoop = False
        except:
            print()
            print("---------------------------------------")
            print("-------Error!!Invalid Input!!------")
            print("---------------------------------------")

        if repeat == "n":
            print()
            name = input("Enter your name: ")
            contact = input("Phone no: ")
            print()
            Return_Bill.bill_for_return(name,contact,total_fine,returned_items)
            print()
            Return_Bill.generate_bill(name, contact,total_fine ,returned_items)
```

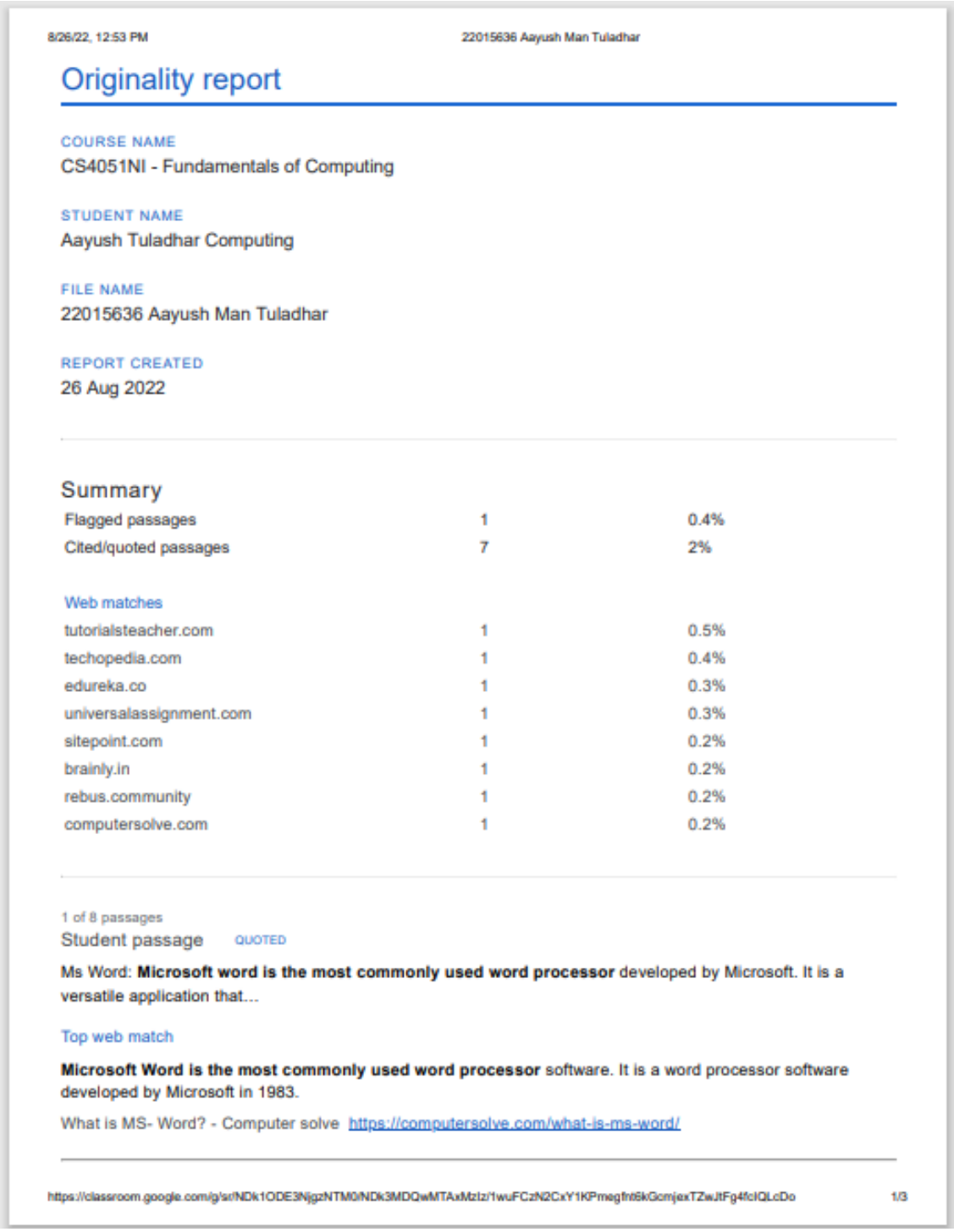
```
            continueLoop = False
            x = True
        else:
            continueLoop = True
            x = True
```

Return_Bill.py

```
import Date_Time
import Get_Info
def bill_for_return(name,contact,total_fine,returned_items):
    print("--------------------------------------------------------------------------")
    print("                          Your Return Bill                                ")
    print("--------------------------------------------------------------------------")
    print("Customer Name: ", name)
    print("Contact: ", contact)
    print("Returned Date: ", Date_Time.dates())
    print("List of Items Returned: ")
    for items in returned_items:
        print(items)
    print("------- ----------------------------------------------------------------")
    print("Fine: $",total_fine)
    print("--------------------------------------------------------------------------")

def generate_bill(name, contact,total_fine ,returned_items):
    file_info = Get_Info.get_file_info()
    main_data = Get_Info.get_dict_info(file_info)

    file = open("Return"+"_"+Date_Time.get_datetime()+"_"+name+".txt","w")
```

```
file.write("_____
_____"+"\n")
    file.write("                    Costume Returned Bill                                "+"\n")

file.write("_____
_____"+"\n")
    file.write("Customer name: "+name+"\n")
    file.write("Contact: "+contact+"\n")
    file.write("Rented Date: "+Date_Time.dates()+"\n")
    file.write("-----------------------------------------------------------------------"+"\n")
    file.write("ID"+"      "+"Name"+"\t"+"                "+"Brand"+"\t"+"          "+"Price"+"\t"+"
"+"Quantity"+"\n")
    file.write("-----------------------------------------------------------------------"+"\n")
    for index in range(len(returned_items)):
        r_sno = int(returned_items[index][0])
        r_quantity = int(returned_items[index][2])
        r_name = main_data[r_sno][0]
        r_brand = main_data[r_sno][1]
        r_price = float(main_data[r_sno][2].replace("$","")) * r_quantity
        file.write(str(index+1)+"\t"+r_name+"\t"+r_brand+"\t"+"              "+str(r_price)+"\t"+"
"+str(r_quantity)+"\n")
        file.write("-----------------------------------------------------------------------"+"\n")

file.write("++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++"+"\n")
    file.write("Fine: $"+str(total_fine)+"\n")
file.write("++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++"+"\n")
    file.close()
```

Originality Test



8/26/22, 12:53 PM 22015636 Aayush Man Tuladhar

Originality report

COURSE NAME
CS4051NI - Fundamentals of Computing

STUDENT NAME
Aayush Tuladhar Computing

FILE NAME
22015636 Aayush Man Tuladhar

REPORT CREATED
26 Aug 2022

Summary

Flagged passages	1	0.4%
Cited/quoted passages	7	2%

Web matches

tutorialsteacher.com	1	0.5%
techopedia.com	1	0.4%
edureka.co	1	0.3%
universalassignment.com	1	0.3%
sitepoint.com	1	0.2%
brainly.in	1	0.2%
rebus.community	1	0.2%
computersolve.com	1	0.2%

1 of 8 passages

Student passage QUOTED

Ms Word: **Microsoft word is the most commonly used word processor** developed by Microsoft. It is a versatile application that...

Top web match

Microsoft Word is the most commonly used word processor software. It is a word processor software developed by Microsoft in 1983.

What is MS- Word? - Computer solve https://computersolve.com/what-is-ms-word/

https://classroom.google.com/g/sc/NDk1ODE3NjgzNTM0/NDk3MDQwMTAxMzIz/1wuFCzN2CxY1KPmegfnt6kGcmjexTZwJtFg4fclQLcDo 1/3

2 of 8 passages

Student passage QUOTED

…be described in terms of its structure and behaviour. **It can be presented using text and structural charts, flowcharts, or other diagrams as needed**.

Top web match

The program must be described in terms of its structure and behaviour. **It can be presented using text and structural charts, flowcharts or other diagrams as needed**.

CS4051 Fundamentals of Computing Coursework https://universalassignment.com/cs4051-fundamentals-of-computing/

---

3 of 8 passages

Student passage QUOTED

Data structure **allows you to organize your data** by letting **you store collections of data and perform** operation regarding your purposes **accordingly**

Top web match

Data Structures **allows you to organize your data** in such a way that enables **you** to **store collections of data**, relate them **and perform** operations on them **accordingly**.

Data Structures in Python | List, Tuple, Dict, Sets, Stack, Queue https://www.edureka.co/blog/data-structures-in-python/

---

4 of 8 passages

Student passage QUOTED

Integers are **are zero, positive or negative whole numbers without a fractional part and having unlimited precision. Integers can be binary, octal, and hexadecimal values**.

Top web match

In Python, **integers are zero, positive or negative whole numbers without a fractional part and having unlimited precision**, e.g. 0, 100, -10…**Integers can be binary, octal, and hexadecimal values**

Python Number Types: int, float, complex - TutorialsTeacher https://www.tutorialsteacher.com/python/python-number-type

---

5 of 8 passages

Student passage FLAGGED

FLOAT: **A float is a data type composed of a number that is not an integer, because it includes a fraction represented in decimal format**. It can be used to calculate a more accurate…

Top web match

In computer science, **a float is a data type composed of a number that is not an integer, because it includes a fraction represented in decimal format**.

What is a Float? (Computer Science) - Definition from Techopedia https://www.techopedia.com/definition/23980/float-computer-science

---

6 of 8 passages

Student passage CITED

...states, them being true and false. It can be **used to create conditions and control how a program behaves when certain things happen**. (Falconer, 2021)

Top web match

A Boolean value is **used to create conditions and control how a program behaves when certain things happen** (e.g. if a condition is true, then do something).

What Is a Boolean Data Type, and What Are Some Uses? - SitePoint https://www.sitepoint.com/boolean-data-type/

---

7 of 8 passages

Student passage CITED

STR: Traditionally, **a string data type** consists of **a** series **of characters, either** in the form of **a literal constant or** a **variable**. The latter can either be constant in length or...

Top web match

A string data type is traditionally **a** sequence **of characters, either** as **a literal constant or** as some kind of **variable**. The latter may allow its elements to be mutated and the length changed, or it...

String Data Type – Programming Fundamentals - Rebus Press https://press.rebus.community/programmingfundamentals/chapter/string-data-type/

---

8 of 8 passages

Student passage CITED

...the form of a literal constant or a variable. **The latter can either be constant in length or allow its elements to alter**. (Braunschweig, 2022)

Top web match

The latter can either be constant in length or allow its elements to alter (after creation). A string is sometimes implemented as an array data structure of bytes (or words) that contains a succession...

a set of characters is called a. - Brainly.in https://brainly.in/question/23818325
