CS4051NI/CC4059NI Fundamentals of Computing

60% Individual Coursework

2022-2023 Autumn Intake

Student Name: Angana Bhattarai

London Met ID: 22067110

College ID: NP01NT4A220074

Assignment Due Date: Friday, July 28, 2023

Assignment Submission Date: Friday, July 28, 2023

Word Count: 6416

I confirm that I understand my coursework needs to be submitted online via MySecondTeacher 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

Table of Figures

Table of tables

1. Introduction

Python is a potent high-level, general-purpose programming language with a wide range of capabilities. Python is a network-centric, object-oriented, functional, and cross-platform language that is quick, secure, and dependable. Python is a very productive language in modern times that is easy to learn, has clear syntax, and is easy to read. In Python, every step from creating a program to identifying errors—is incredibly simple. (Python-Org, n.d.)

For this coursework, IDLE was the recommended IDE. In this coursework, we are required to create a program for a Laptop Shop. A system is to created which reads a text file, that contains information about the brands of laptops the shop where the shop buys laptops/computers from manufacturers and sells it to customers which might either be individual or company. The Anganas's Computer Store is able to place orders to the manufacturing company and the customers are able to place orders to the shop.

1.1. Goals and Objectives.

- Allow the user to choose the desired choice whether they want to buy, order or quit the program.
- If the user chooses to order a computer, they get to view the stock of the computers and then choose the required computer. They are also able to choose multiple computers.
- To generate an invoice which must consist of the user's name, computer's name, brand name, price, quantity, processor, graphics card. If multiple computers have been purchased, then the invoice must include all the specified information with the total cost.
- If the user chooses to sell a computer, they get to view the list of computers the user might want to sell. They are also able to sell multiple computers.

- To generate an invoice which must consist of the user's name, computer's name, brand name, quantity, processor, graphics card and the price. If multiple computers are being sold, then the invoice must include all the specified information with the total cost.

This program is built to make help accuracy, sales management, time saver, scalability and inventory management for the user.

2. Discussion and Analysis.

2.1. Algorithm.

An algorithm is a group of instructions with clear objectives created to do a certain task. Performing a task could be as simple as multiplying two numbers or as difficult as playing a compressed video file. Algorithm for this project is given below:

STEP 1: START

STEP 2: PRINT Welcome Message.

STEP 3: INPUT your option.

STEP 4: IF n=1, go to Step 5

IF n=2, go to Step 15

IF n=3, END the program.

STEP 5: READ the text file and add it to the dictionary.

STEP 6: DISPLAY the computers present in the dictionary.

STEP 7: INPUT the ID and QUANTITY for desired computer.

STEP 8: INCREASE the quantity of the ordered computer.

STEP 9: ASK if the orderer wants to order more computers.

STEP 10: IF the orderer chooses YES, GO TO Step 5.

STEP 11: IF the orderer chooses NO, GO TO Step 12.

STEP 12: INPUT Customer's Name and Contact Number.

STEP 13: CREATE an invoice with the name, contact number, brand, price, quantity, generation, graphics, current date and time, and the total price with addition VAT amount

STEP 14: DISPLAY the computers have been successfully ordered and the Invoice has been successfully generated.

STEP 15: GO TO Step 2

STEP 16: INPUT option 2.

STEP 17: READ the text file and add it to the dictionary.

STEP 18: DISPLAY the computers present in the dictionary.

STEP 19: INPUT the ID and QUANTITY for desired computer**.**

STEP 20: DECREASE the quantity of the desired computer.

STEP 21: ASK if the buyer wants to buy more computers.

STEP 22: IF the buyer chooses YES, GO TO Step 17.

STEP 23: IF the buyer chooses NO, GO TO Step 24.

STEP 24: INPUT Customer's Name and Contact Number.

STEP 25: CREATE an invoice with the name, contact number, brand, price, quantity, generation, graphics, current date and time, and the total price with addition of shipping cost.

STEP 26: DISPLAY the computers have been successfully bought and the Invoice has been successfully generated.

STEP 27: GO TO Step 3.

STEP 28: END

.

2.2. Flowchart

A flowchart is a pictorial representation of sequential orders required to complete a certain process, system, or computer algorithm. It is used to document, study, and improve the current plans and change ideas to different ones. It informs how a certain process starts and reaches the end. The steps of the processes are written in a shape that provides a certain meaning. For example, A rectangle is used to write the process that is required to be done, a parallelogram is used for either input or output and many other shapes are used that represent their meanings. Each step/shape is connected using connecting arrows. (Chart, 2020)

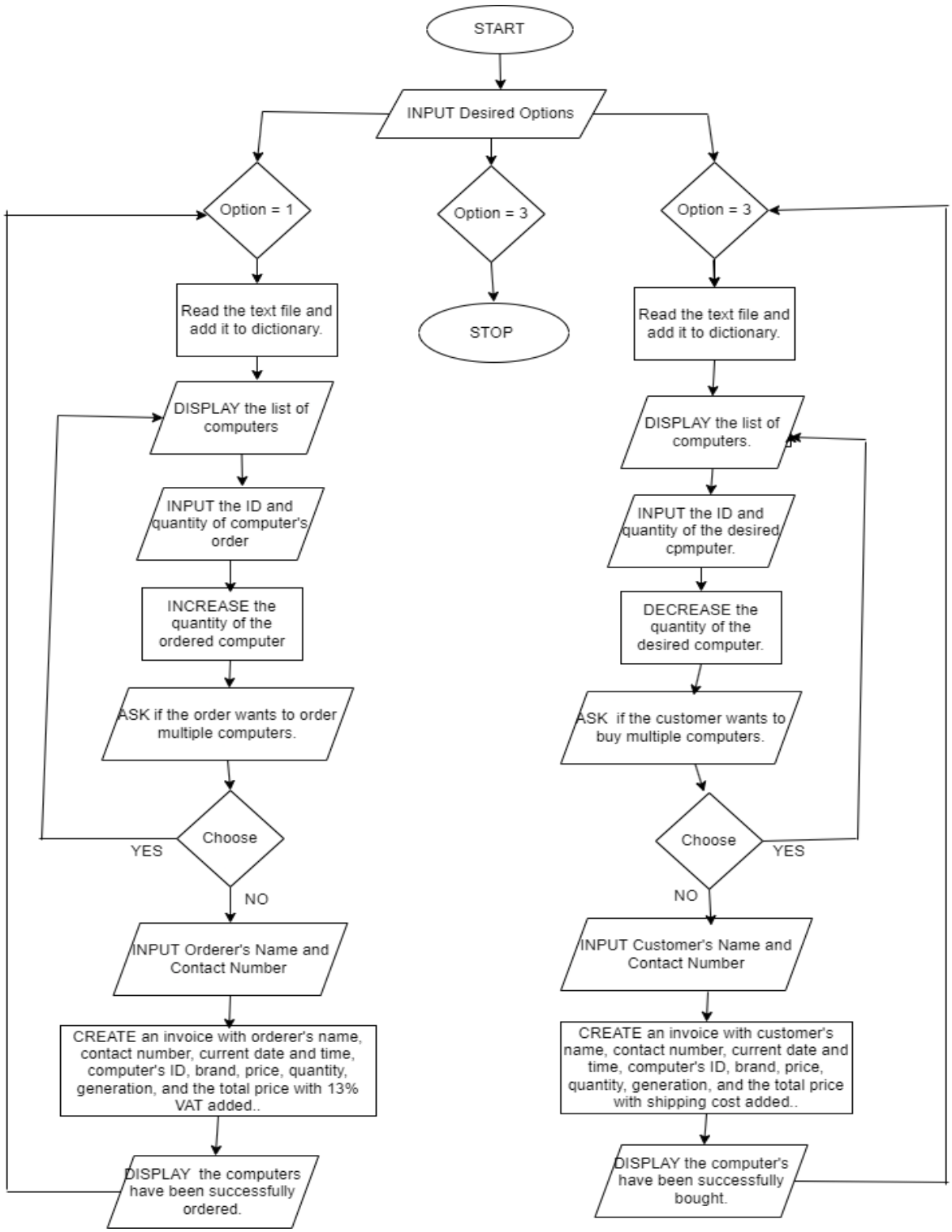

Figure 1: Flowchart of the program.

2.3. Pseudocode

Pseudocode is a language that artificially and informally lets programmers build algorithms. It is an informal means of describing programming without rigid syntaxes and technical concerns. It summarizes the flow of a program to an understanding level for even people out of the technical field increasing its readability which allows programmers to recognize and align code according to the specification of a software project. However, it is not an actual programming language. It also provides rough documentation allowing you to quickly understand the development process for the software. Documentation is an important part that tells what has been done, what can be changed, and what not to do, and here pseudocode is important for explaining precisely what each line of the program can do and making it simpler for the programmer to create the codes. (Times, n.d.)

Listed below are the rules that needed to be followed while writing the pseudocode for any of the programs:

- Initial keywords are capitalized and bold.
- Statements by-lines are printed and not jumbled.
- The DO and ENDDO keyword contain a block of code.
- Multiline form ends with its suffix with END keywords.
- To keep it concise and readable, simple, and plain vocabulary is to be used.
- Indentation and separation shall be handled correctly.

Pseudocode for program to manage laptops in a store and generate invoices for purchases and sales.

Pseudocode of main.py

```
OUTPUT("+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++")

OUTPUT("|-----------------Anganas's Laptop Store-------------------------|")

OUTPUT("+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++")


DEFINE FUNCTION inv_message():

    OUTPUT("\n\t\t***Invalid Input!! Please follow the instructions properly***\n")


DEFINE FUNCTION main():

    WHILE(True):

        OUTPUT("\nPlease select your option........\n")

        OUTPUT("(1) | | Press 1 to view and order a computer........")

        OUTPUT("(2) | | Press 2 to view and buy a computer........")

        OUTPUT("(3) | | Press 3 to exit........\n")

        INPUT("\nPlease enter your option: ")

        IF option EQUALS '1':

            order.order_main()

        ELSEIF option EQUALS '2':

            buy.buy_main()
```

```
            ELSEIF option EQUALS '3':
                BREAK
            ELSE
                inv_message()
        main()
```

Pseudocode of view.py

```
DEFINE FUNCTION inv_message():
    OUTPUT("\n\t\t***Invalid Input!! Please follow the instructions properly***\n")


DEFINE FUNCTION display_computers():
    SET count TO 0
    OUTPUT("-----------------------------------------------------------------------------------------------")
    OUTPUT('ID. \t Computer Name \tBrand \tPrice\tQuantity\tGeneration\t Graphics')
    OUTPUT("-----------------------------------------------------------------------------------------------")
    SET textdoc TO open("computer.txt","r")
```

```
SET lines TO textdoc.read()

SET lines TO lines.split("\n")

WHILE ("" IN lines):

    lines.remove("")

FOR i IN range(len(lines)):

    SET count TO count+1

    SET d_computer[count] TO lines[i].split(",")

FOR key,value IN d_computer.items():

    OUTPUT(key,end="\t")

    FOR i IN value:

        OUTPUT(i,end="\t")

    OUTPUT("\n")

OUTPUT("-----------------------------------------------------------------------------------------")

RETURN d_computer
```

Pseudocode for buy.py

```
DEFINE FUNCTION inv_message():

    OUTPUT("\n\t\t***Invalid Input!! Please follow the instructions properly***\n")



DEFINE FUNCTION buy_main():

    Loop=True

    WHILE loop EQUALS True:

        view.display_computers()

        SET computer_id TO buy_computer(d_computer)

        SET quantity TO check_quantity(d_computer)

        SET enter TO False

        WHILE enter EQUALS False:

            up_quantity(d_computer, computer_id, quantity)

            update_stock(d_computer)
```

```
        SET option TO INPUT("Do you want to buy more computers?\n Choose 'Y' FOR
Yes and 'N' FOR No: " ).upper()

        IF option EQUALS "Y":

            SET enter TO True

        ELSEIF option EQUALS "N":

            gen_invoice(d_computer)

            SET loop TO False

            SET enter TO True

            BREAK

        ELSE

            inv_message()


DEFINE FUNCTION buy_computer(d_computer):

    WHILE True

        TRY

            computer_id=(int(INPUT("\nEnter the ID of Commputer you want to buy: ")))
```

```
        IF computer_id>0 and computer_id<=len(d_computer):
            OUTPUT ("You can now buy the Computer")
            BREAK
        ELSE
            OUTPUT ("Please enter a valid ID")
    EXCEPT
        inv_message()
  RETURN computer_id


DEFINE FUNCTION check_quantity(d_computer):
  TRY
    SET quantity TO int(INPUT("Enter the quantity of the Computers: "))
    IF quantity<0:
        OUTPUT("Please enter the desired quantity IN a proper format! ")
```

```
        ELSE
            RETURN quantity
    EXCEPT
        inv_message()
DEFINE FUNCTION up_quantity(d_computer,computer_id,quantity):
    SET quantity_left TO int(d_computer[computer_id][3]) - quantity
    SET d_computer[computer_id][3] TO str(quantity_left)
    buy.append([computer_id,quantity])
    RETURN buy
DEFINE FUNCTION update_stock(d_computer):
    SET s TO open("computer.txt","w")
    FOR key,value IN d_computer.items():
        s.write(",".join(value))
        s.write("\n")
```

```
    s.close()

DEFINE FUNCTION gen_invoice(d_computer):

    SET Datetime TO datetime.datetime.now()

    Year=str(Datetime.year)

    Month=str(Datetime.month)

    Day=str(Datetime.day)

    Hour=str(Datetime.hour)

    Minutes=str(Datetime.minute)

    Sec=str(Datetime.second)

    SET Date TO Year+"-"+Month+"-"+Day

    SET Time TO Hour+":"+Minutes+':'+Sec

    SET user_name TO INPUT("Enter your Name: ")

    TRY

        SET ph_number TO INPUT("Enter your contact number: ")
```

```
        SET ph TO str(ph_number)

    EXCEPT

        OUTPUT("Please INPUT correct phone number")


OUTPUT("\n*****************************************************************************************")

    OUTPUT("\t\t\t\tInvoice")


OUTPUT("\n*****************************************************************************************")

    OUTPUT("Name: ",user_name)

    OUTPUT("Contact number: ",ph_number)

    OUTPUT("Date: ",Date , "\t\t\t","Time: ",Time)

    OUTPUT("Status: Paid")


OUTPUT("*****************************************************************************************")

    OUTPUT("SN.\tID\tName\tBrand\tPrice\tQuantity\tGeneration\tGraphics")
```

```
SET Total TO 0

FOR i IN range(len(buy)):

    SET c_id TO int(buy[i][0])

    SET csr_id TO str(c_id)

    SET c_name TO d_computer[c_id][0]

    SET c_brand TO d_computer[c_id][1]

    SET c_price TO float(d_computer[c_id][2].replace("$",""))

    SET c_quantity TO int(buy[i][1])

    SET cr_price TO float(d_computer[c_id][2].replace("$",""))*c_quantity

    SET c_generation TO d_computer[c_id][4]

    SET c_graphics TO d_computer[c_id][5]

    Total += cr_price

    SET Tprice TO str(Total)

    SET Fprice TO Total + 125
```

```
        SET Fwprice TO str(Fprice)


OUTPUT(str(i+1),"\t",csr_id,"\t",c_name,"\t",c_brand,"\t","$",str(c_price),"\t",str(c_quantit
y),"t",c_generation,"t",c_graphics)

    OUTPUT("\n")

    OUTPUT("Initial Amount: ",str(Tprice))


OUTPUT("*********************************************************************************
*******")

    OUTPUT("Note: The shipping cost FOR all transaction is 125$")


OUTPUT("*********************************************************************************
*******")

    OUTPUT("Shipping Cost: 125$")

    OUTPUT("Final Amount: ",str(Fprice))

CREATE notefile

    buyFilename="bought_by_"+user_name+".txt"

    WITH open(buyFilename,"w") as file:

        file.write("\nName: " + user_name)
```

```
    file.write("\nPhone number: " + ph_number)

    file.write("\nDate of Buy: " + Date + "\t\tTime of Buy: " + Time)

    file.write("\nStatus: Paid")

    file.write("\nSN.\tID\tCostume
Name\tBrand\tPrice\tQuantity\tGeneration\tGraphics")

    FOR j IN range(len(buy)):

        SET cw_id TO int(buy[j][0])

        cw_quantity= int(buy[j][1])

        SET cw_name TO d_computer[cw_id][0]

        SET cw_brand TO d_computer[cw_id][1]

        SET cw_price TO float(d_computer[cw_id][2].replace("$",""))

        SET crw_price TO float(d_computer[cw_id][2].replace("$",""))*c_quantity

        cw_generation =d_computer[cw_id][4]

        cw_graphics =d_computer[cw_id][5]
```

```
file.write("\n"+str(j+1)+"\t"+str(cw_id)+"\t"+cw_name+"\t"+cw_brand+"\t"+"$"+str(cw_price)+"\t"+str(cw_quantity)+"\t"+cw_generation+"\t"+cw_graphics)

    file.write("\nTotal Price: " + Tprice + "$")

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

    file.write("\nNote: The shipping cost FOR all transaction is 125$")

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

    file.write("\nShipping Cost: 125$")

    file.write("\nFinal Amount: " + Fwprice + "$")

    file.close()
```

Pseudocode of order.py

```
DEFINE FUNCTION inv_message():

    OUTPUT("\n\t\t***Invalid Input!! Please follow the instructions properly***\n")



DEFINE FUNCTION order_main():

    loop=True

    WHILE loop EQUALS True:

        view.display_computers()

        SET computer_id TO order_computer(d_computer)

        SET quantity TO check_quantity(d_computer)

        SET enter TO False

        WHILE enter EQUALS False:

            up_quantity(d_computer, computer_id, quantity)

            update_stock(d_computer)
```

```
        SET option TO INPUT("Do you want to buy more computers?\n Choose 'Y' FOR
Yes and 'N' FOR No: " ).upper()

        IF option EQUALS "Y":

            SET enter TO True

        ELSEIF option EQUALS "N":

            gen_invoice(d_computer)

            SET loop TO False

            SET enter TO True

            BREAK

        ELSE

            inv_message()


DEFINE FUNCTION order_computer(d_computer):


    WHILE True:

        TRY
```

```
        computer_id=(int(INPUT("\nEnter the ID of Commputer you want to buy: ")))

        IF computer_id>0 and computer_id<=len(d_computer):

            OUTPUT ("You can now order the Computer")

            BREAK

        ELSE

            OUTPUT ("Please enter a valid ID")

    EXCEPT

        inv_message()

  RETURN computer_id



DEFINE FUNCTION check_quantity(d_computer):

  TRY

    SET quantity TO int(INPUT("Enter the quantity of the Computers: "))

    IF quantity<0:
```

```
        OUTPUT("Please enter the desired quantity IN a proper format! ")
      ELSE
        RETURN quantity
    EXCEPT
      inv_message()


CREATE list
DEFINE FUNCTION up_quantity(d_computer,computer_id,quantity):

    SET quantity_left TO int(d_computer[computer_id][3]) + quantity
    SET d_computer[computer_id][3] TO str(quantity_left)
    order.append([computer_id,quantity])
    RETURN order
```

```
DEFINE FUNCTION update_stock(d_computer):

    SET s TO open("computer.txt","w")

    FOR key,value IN d_computer.items():

        s.write(",".join(value))

        s.write("\n")

    s.close()


DEFINE FUNCTION gen_invoice(d_computer):

    SET Datetime TO datetime.datetime.now()

    Year=str(Datetime.year)

    Month=str(Datetime.month)

    Day=str(Datetime.day)

    Hour=str(Datetime.hour)

    Minutes=str(Datetime.minute)

    Sec=str(Datetime.second)
```

```
    SET Date TO Year+"-"+Month+"-"+Day

    SET Time TO Hour+":"+Minutes+':'+Sec

    SET user_name TO INPUT("Enter your Name: ")

    TRY

        SET ph_number TO INPUT("Enter your contact number: ")

        SET ph TO str(ph_number)

        SET ph_length TO len(ph)

    EXCEPT

        OUTPUT("Please INPUT correct phone number")


OUTPUT("\n*****************************************************************************
*********")

    OUTPUT("\t\t\t\tInvoice")


OUTPUT("\n*****************************************************************************
*********")

    OUTPUT("Name: ",user_name)
```

```
    OUTPUT("Contact number: ",ph_number)

    OUTPUT("Date: ",Date , "\t\t\t","Time: ",Time)

    OUTPUT("Status: Paid")


OUTPUT("*****************************************************************************************
*******")

    OUTPUT("SN.\tID\tName\tBrand\tPrice\tQuantity\tGeneration\tGraphics")

    SET Total TO 0

    FOR i IN range(len(order)):

        SET c_id TO int(order[i][0])

        SET csr_id TO str(c_id)

        SET c_name TO d_computer[c_id][0]

        SET c_brand TO d_computer[c_id][1]

        SET c_price TO float(d_computer[c_id][2].replace("$",""))

        c_quantity= int(order[i][1])
```

```
    SET cr_price TO float(d_computer[c_id][2].replace("$",""))*c_quantity

    c_generation =d_computer[c_id][4]

    c_graphics =d_computer[c_id][5]

    Total += cr_price

    VAT EQUALS (13/100)*Total

    SET VATw TO str(VAT)

    SET Tprice TO str(Total)

    SET FwPrice TO str (Total +VAT)


OUTPUT(str(i+1),"\t",csr_id,"\t",c_name,"\t",c_brand,"\t","$",str(c_price),"\t",str(c_quantit
y),"t",c_generation,"t",c_graphics)

   OUTPUT("\n")

   OUTPUT("Intial Amount: ",Tprice)

   OUTPUT("VAT Amount: ", str(VAT))

   OUTPUT("Total Amount", str(Total+VAT))
```

```
OUTPUT("*****************************************************************************************
*******")

    orderFilename="ordered_by_"+user_name+".txt"

    WITH open(orderFilename,"w") as file:

        file.write("\nName: " + user_name)

        file.write("\nPhone number: " + ph_number)

        file.write("\nDate of Buy: " + Date + "\t\tTime of Buy: " + Time)

        file.write("\nStatus: Paid")

        file.write("\nSN.\tID\tCostume
Name\tBrand\tPrice\tQuantity\tGeneration\tGraphics")

        FOR j IN range(len(order)):

            SET cw_id TO int(order[j][0])

            cw_quantity= int(order[j][1])

            SET cw_name TO d_computer[cw_id][0]

            SET cw_brand TO d_computer[cw_id][1]
```

```
        SET cw_price TO float(d_computer[cw_id][2].replace("$",""))

        SET crw_price TO float(d_computer[cw_id][2].replace("$",""))*cw_quantity

        cw_generation =d_computer[cw_id][4]

        cw_graphics =d_computer[cw_id][5]


file.write("\n"+str(j+1)+"\t"+str(cw_id)+"\t"+cw_name+"\t"+cw_brand+"\t"+"$"+str(cw_pric
e)+"\t"+str(cw_quantity)+"\t"+cw_generation+"\t"+cw_graphics)

    file.write("\nIntial Amount: " +Tprice+"$")

    file.write("\nVAT Amount: "+ VATw+"$")

    file.write("\nFinal Amount: " + FwPrice + "$")

    file.close()
```

2.4. Data Structures.

Data structures are a method for gathering and arranging data so that it is simple to change and retrieve. They make it simple for the user to gather, relate, and manipulate the data. Any computer language's foundation is its data structures. List, dictionary, set, and tuples are the four main and pre-built data structures in Python. The mutability and order of these data structures vary from one another. The ability to modify a state or piece of information after it has been created is referred to as mutability. After they have been formed, mutable objects can be changed, added to, or removed. The order of the elements determines whether or not a particular element may be accessed from its place and is represented by the order.

Lists

One of the fundamental data structures in Python is the list, which is an ordered collection of items. Each item in lists is presented in a different sequence for identification. Because list entries can be changed, added, or removed after the list has been established, lists are changeable. Lists can expand or contract, making them dynamic. A list's element order is a fixed property that endures throughout the list's existence. Indexes are used to access lists. Duplicate values are permitted in lists. Square brackets are used to make lists, and commas are used to divide the entries.

Dictionary

Dictionaries are used to store data values in key: value pairs. A dictionary is mutable. The collection of data can be ordered, changeable but it does not allow duplicates. Dictionaries are dynamic as they can grow and shrink as needed. Dictionaries are created inside curly brackets; the dictionary items are presented in key: value pairs and can be referred using key name. The elements in a dictionary are accessed using the keys. Dictionaries are also generally known as associative array. Each key in a

dictionary is associated with a specific value. A value is retrieved from a dictionary by specifying its corresponding key in square brackets ([]). (Python, 2022)

String

A string is a sort of data that typically consists of a collection of letters that either form a word or a statement. There are numerous characters in it. Values for strings are always encapsulated in quotation marks.

Integer

Whole numbers without any decimals are known as integer data types in python. They can represent both positive or negative numbers. Integers are a crucial data type in python and they are widely used in programming and coding tasks ranging from simple calculation to more complex algorithms and data manipulation.

Boolean

Logical values which are either True or False are known as Boolean data types in python. Boolean values are used in conditional statements and logical operations to control the flow of a program. They control the flow of the program, making decisions, and determining the behavior of the code based on it’s conditions.

Characters

Individual characters in python are represented using the string data type.
A single character is just a string of length 1. The concept of characters is fundamental in dealing with text and strings.

3. Program

In this coursework, we are required to create a program for a Computer Store. An application has been created which reads a text file that contains information about the computers available in the store and displays it to the consumer. The consumer can choose if they want to buy or sell the computers. The consumer can also choose the quantity and multiple brands which they want to buy or sell. After the consumer chooses their desired option, the information is to be recorded of the consumer i.e., name, contact number. Then, the invoice is generated in the shell and also in a text file format. The text file format invoice generated will automatically be stored in the folder itself.

When the user runs the application, they get to choose on what they would like to do. main.py runs at the start of the program.

```
>>>
= RESTART: C:\Users\Lenovo\Documents\Python FOC\FOC_22067110_AnganaBhattarai\Angana\main.py
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
|-----------------Anganas's Laptop Store-------------------------|
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

Please select your option........

(1) | | Press 1 to view and order a computer........
(2) | | Press 2 to view and buy a computer........
(3) | | Press 3 to exit........


Please enter your option: |
```

Figure 2: Welcome Message

After the input value is entered, the application reads the text file and displays the list of computers stored in the dictionary. Then the program asks the user to enter the ID of the computer they desire to order.

```
Please enter your option: 1
--------------------------------------------------------------------------------
ID.      Computer Name  Brand    Price    Quantity        Generation       Graphics
--------------------------------------------------------------------------------
1        Razer Blade      Razer  $2000    40      i7 7th Gen      GTX 3060

2        XPS              Dell    $1976   50      i5 9th Gen      GTX 3070

3        Alienware       Alienware       $1978   34      i5 9th Gen      GTX 3070

4        Swift 7      Acer         $900   33      i5 9th Gen       GTX 3070

5        MacbookPro16    Apple   $3500   30      i5 9th Gen      GTX 3070

--------------------------------------------------------------------------------

Enter the ID of Commputer you want to buy:
```

Figure 3: Display of list of computers.

After the display and input of one particular ID, the application asks the user about their desired quantity for the computer. Then the user is also asked if they want to order other computers.

```
Enter the ID of Commputer you want to buy: 2
You can now order the Computer
Enter the quantity of the Computers: 3
Do you want to buy more computers?
 Choose 'Y' for Yes and 'N' for No:
```

Figure 4: Choice of Yes and No

If the user chooses to order more computers, the application will display the list of computers available too. The loop continues with the same set of information.

```
Do you want to buy more computers?
 Choose 'Y' for Yes and 'N' for No: Y
--------------------------------------------------------------------------------
ID.      Computer Name  Brand   Price   Quantity        Generation       Graphics
--------------------------------------------------------------------------------
1       Razer Blade      Razer  $2000   40      i7 7th Gen      GTX 3060

2       XPS              Dell    $1976  53      i5 9th Gen      GTX 3070

3       Alienware       Alienware       $1978   34      i5 9th Gen      GTX 3070

4       Swift 7      Acer         $900  33      i5 9th Gen       GTX 3070

5       MacbookPro16    Apple   $3500   30      i5 9th Gen      GTX 3070

--------------------------------------------------------------------------------

Enter the ID of Commputer you want to buy: 1
You can now order the Computer
Enter the quantity of the Computers: 2
```

Figure 5: Display of computers to order more computers.

After the input of desired ID and quantity again, the application works on the same loop asking for the user to choose if they still want to order more or not. If the user chooses YES, the loop continues back to displaying the list of computers. If the user chooses NO, the loop breaks and the application ask for the input of name and contact number.

```
Enter the quantity of the Computers: 2
Do you want to buy more computers?
 Choose 'Y' for Yes and 'N' for No: N
Enter your Name: PJK
Enter your contact number: 4437225
```

Figure 6: Input of user's information

After the information is given, the invoice is generated in the shell.

```
*****************************************************************************************
                                Invoice

*****************************************************************************************
Name:  PJK
Contact number:  4437225
Date:  2023-7-28                        Time:  8:50:45
Status: Paid
*****************************************************************************************
SN.     ID      Name    Brand   Price   Quantity        Generation      Graphics
1        2       XPS             Dell   $ 1976.0        3 t i5 9th Gen t GTX 3070
2        1       Razer Blade     Razer          $ 2000.0        2 t i7 7th Gen t GTX 3060


Intial Amount:  9928.0
VAT Amount:  1290.64
Total Amount 11218.64
*****************************************************************************************
```

Figure 7: Generating invoice or order in the shell

The invoice is also generated is text file format which automatically is saved in the folder.

ordered_by_PJK	7/28/2023 8:51 AM	Text Document	1 KB

Figure 8: Invoice text file in folder.

ordered_by_PJK - Notepad

File Edit Format View Help

```
Name: PJK
Phone number: 4437225
Date of Buy: 2023-7-28          Time of Buy: 8:50:45
Status: Paid
SN.     ID      Costume Name    Brand   Price   Quantity        Generation      Graphics
1       2       XPS              Dell   $1976.0 3       i5 9th Gen      GTX 3070
2       1       Razer Blade      Razer  $2000.0 2       i7 7th Gen      GTX 3060
Intial Amount: 9928.0$
VAT Amount: 1290.64$
Final Amount: 11218.64$
```

Figure 9: Display of text file invoice for order.

The quantity increases when the user orders computers for the store.

```
-----------------------------------------------------------------------------------
ID.      Computer Name  Brand    Price    Quantity        Generation       Graphics
-----------------------------------------------------------------------------------
1        Razer Blade      Razer  $2000    40      i7 7th Gen      GTX 3060

2        XPS              Dell    $1976   50      i5 9th Gen      GTX 3070

3        Alienware       Alienware       $1978   34      i5 9th Gen      GTX 3070

4        Swift 7      Acer         $900   33      i5 9th Gen       GTX 3070

5        MacbookPro16    Apple   $3500   30      i5 9th Gen      GTX 3070
```

Figure 10: Quantity before the order.

```
-----------------------------------------------------------------------------------
ID.      Computer Name  Brand    Price    Quantity        Generation       Graphics
-----------------------------------------------------------------------------------
1        Razer Blade      Razer  $2000    42      i7 7th Gen      GTX 3060

2        XPS              Dell    $1976   53      i5 9th Gen      GTX 3070

3        Alienware       Alienware       $1978   34      i5 9th Gen      GTX 3070

4        Swift 7      Acer         $900   33      i5 9th Gen       GTX 3070

5        MacbookPro16    Apple   $3500   30      i5 9th Gen      GTX 3070

-----------------------------------------------------------------------------------
```

Figure 11: Increase of quantity after the success of order.

The loop goes back to the input of option from the start.

```
*****************************************************

Please select your option........

(1) | | Press 1 to view and order a computer........
(2) | | Press 2 to view and buy a computer........
(3) | | Press 3 to exit........


Please enter your option: |
```

Figure 12: Loop to start.

The user selects 2.

```
Please enter your option: 2
--------------------------------------------------------------------------------
ID.      Computer Name  Brand    Price    Quantity        Generation       Graphics
--------------------------------------------------------------------------------
1       Razer Blade      Razer  $2000    42      i7 7th Gen      GTX 3060

2       XPS              Dell    $1976   53      i5 9th Gen      GTX 3070

3       Alienware       Alienware       $1978   34      i5 9th Gen      GTX 3070

4       Swift 7      Acer         $900   33      i5 9th Gen       GTX 3070

5       MacbookPro16    Apple   $3500   30      i5 9th Gen      GTX 3070

--------------------------------------------------------------------------------
```

Figure 13: Display of the list of computers to buy

After the display and input of one particular ID, the application asks the user about their desired quantity for the computer. Then the user is also asked if they want to buy other computers.

```
4       Swift 7      Acer         $900   33      i5 9th Gen       GTX 3070

5       MacbookPro16    Apple   $3500   30      i5 9th Gen      GTX 3070

--------------------------------------------------------------------------

Enter the ID of Commputer you want to buy: 4
You can now buy the Computer
Enter the quantity of the Computers: 3
```

Figure 14: Input of ID and Quantity

If the user chooses to buy more computers, the application will display the list of computers available too. The loop continues with the same set of information.

```
Enter the quantity of the Computers: 3
Do you want to buy more computers?
 Choose 'Y' for Yes and 'N' for No: Y
```

Figure 15: Choice of Yes and No

After the input of desired ID and quantity again, the application works on the same loop asking for the user to choose if they still want to buy more or not. If the user chooses YES, the loop continues back to displaying the list of computers. If the user chooses NO, the loop breaks and the application ask for the input of name and contact number.

```
 Choose 'Y' for Yes and 'N' for No: Y
--------------------------------------------------------------------------------
ID.      Computer Name  Brand    Price   Quantity        Generation       Graphics
--------------------------------------------------------------------------------
1       Razer Blade      Razer  $2000   42      i7 7th Gen      GTX 3060

2       XPS              Dell    $1976  53      i5 9th Gen      GTX 3070

3       Alienware       Alienware       $1978   34      i5 9th Gen      GTX 3070

4       Swift 7      Acer         $900  30      i5 9th Gen       GTX 3070

5       MacbookPro16    Apple   $3500   30      i5 9th Gen      GTX 3070

--------------------------------------------------------------------------------

Enter the ID of Commputer you want to buy: 3
You can now buy the Computer
Enter the quantity of the Computers: 4
Do you want to buy more computers?
```

Figure 16: Display of list of computers.

With the choice of NO, the application asks the user's information.

```
 Choose 'Y' for Yes and 'N' for No: N
Enter your Name: SRK
Enter your contact number: 9813192791
```

Figure 17: Input of user's information

After the information is given, the invoice is generated in the shell.

```
Enter your Name: SRK
Enter your contact number: 9813192791

*****************************************************************************************
                                Invoice

*****************************************************************************************
Name:  SRK
Contact number:  9813192791
Date:  2023-7-28                        Time:  9:4:53
Status: Paid
*****************************************************************************************
SN.     ID      Name    Brand   Price   Quantity        Generation      Graphics
1        4       Swift 7             Acer       $ 900.0         3 t i5 9th Gen t  GTX 3070
2        3       Alienware       Alienware       $ 1978.0        4 t i5 9th Gen t GTX 3070


Initial Amount:  10612.0
*****************************************************************************************
Note: The shipping cost for all transaction is 125$
*****************************************************************************************
Shipping Cost: 125$
Final Amount:  10737.0
```

Figure 18: Invoice for buy generated in the shell.

The invoice is also generated is text file format which automatically is saved in the folder.

bought_by_SRK	7/28/2023 9:05 AM	Text Document	1 KB

Figure 19: Invoice text file in folder.

bought_by_SRK - Notepad

File Edit Format View Help

```
Name: SRK
Phone number: 9813192791
Date of Buy: 2023-7-28          Time of Buy: 9:4:53
Status: Paid
SN.     ID      Costume Name    Brand   Price   Quantity        Generation      Graphics
1       4       Swift 7      Acer       $900.0  3       i5 9th Gen       GTX 3070
2       3       Alienware       Alienware       $1978.0 4       i5 9th Gen      GTX 3070
Total Price: 10612.0$
*****************************************************************************************
Note: The shipping cost for all transaction is 125$
*****************************************************************************************
Shipping Cost: 125$
Final Amount: 10737.0$
```

Figure 20: Display of invoice for buy in text file.

After generating the invoice, the application runs back to loop for the user to select if they want to order, buy or end the program.

```
*****************************************************
Shipping Cost: 125$
Final Amount:  10737.0

Please select your option........

(1) | | Press 1 to view and order a computer........
(2) | | Press 2 to view and buy a computer........
(3) | | Press 3 to exit........


Please enter your option: |
```

Figure 21: Select of options

```
Please select your option........

(1) | | Press 1 to view and order a computer........
(2) | | Press 2 to view and buy a computer........
(3) | | Press 3 to exit........


Please enter your option: 3
>>>
```

Figure 22: Exit of program:

If the user selects option 3, the user exits from the application and the program ends.

4. Testing

4.1. Test 1

```
-------------------------------------------------------------------------------------------
ID.      Computer Name  Brand   Price   Quantity        Generation       Graphics
-------------------------------------------------------------------------------------------
1       Razer Blade      Razer  $2000   40      i7 7th Gen      GTX 3060

2       XPS              Dell    $1976  52      i5 9th Gen      GTX 3070

3       Alienware       Alienware       $1978   29      i5 9th Gen      GTX 3070

4       Swift 7      Acer         $900  36      i5 9th Gen       GTX 3070

5       MacbookPro16    Apple   $3500   30      i5 9th Gen      GTX 3070

-------------------------------------------------------------------------------------------

Enter the ID of Commputer you want to buy: 2
You can now order the Computer
Enter the quantity of the Computers: -3
Please enter the desired quantity in a proper format!
```

Figure 23: Test 1

• Objective	To get an error message when a negative value as input when entering the desired quantity.
• Action	ID entered = 2 Quantity = -3
• Expected Result	Error message saying “Please enter the desired quantity in a proper format.”
• Actual Result	Error message saying invalid input is displayed.
• Conclusion	The test is successful.

Table 1: Analysis table for test 1

4.2. Test 2

```
-----------------------------------------------------------------------------------------
ID.      Computer Name  Brand   Price   Quantity        Generation       Graphics
-----------------------------------------------------------------------------------------
1       Razer Blade      Razer  $2000   40      i7 7th Gen      GTX 3060

2       XPS              Dell    $1976  52      i5 9th Gen      GTX 3070

3       Alienware       Alienware       $1978   -3      i5 9th Gen      GTX 3070

4       Swift 7      Acer         $900  36      i5 9th Gen       GTX 3070

5       MacbookPro16    Apple   $3500   25      i5 9th Gen      GTX 3070

-----------------------------------------------------------------------------------------

Enter the ID of Commputer you want to buy: #

                ***Invalid Input!! Please follow the instructions properly***
```

Figure 24 : Test 2

• Objective	Implementation of try and except.
• Action	Inputting a character ("#") instead of the desired computer's ID.
• Expected Result	Error message saying "Invalid Input" should be displayed.
• Actual Result	Error message with 'invalid input' is displayed.
• Conclusion	The test is successful.

Table 2: Analysis table for Test 2

4.3. Test 3

```
(2) | | Press 2 to view and buy a computer........
(3) | | Press 3 to exit........


Please enter your option: 2
--------------------------------------------------------------------------------
ID.      Computer Name  Brand    Price    Quantity        Generation       Graphics
--------------------------------------------------------------------------------
1        Razer Blade      Razer  $2000    40      i7 7th Gen      GTX 3060

2        XPS              Dell    $1976   52      i5 9th Gen      GTX 3070

3        Alienware       Alienware       $1978   33      i5 9th Gen      GTX 3070

4        Swift 7      Acer         $900   36      i5 9th Gen       GTX 3070

5        MacbookPro16    Apple   $3500   25      i5 9th Gen      GTX 3070

--------------------------------------------------------------------------------

Enter the ID of Commputer you want to buy: 2
You can now buy the Computer
Enter the quantity of the Computers: 2
Do you want to buy more computers?
 Choose 'Y' for Yes and 'N' for No: N
```

Figure 25: Test 3 i.

```
Please enter your option: 2
--------------------------------------------------------------------------------
ID.      Computer Name  Brand    Price    Quantity        Generation       Graphics
--------------------------------------------------------------------------------
1        Razer Blade      Razer  $2000    40      i7 7th Gen      GTX 3060

2        XPS              Dell    $1976   50      i5 9th Gen      GTX 3070

3        Alienware       Alienware       $1978   33      i5 9th Gen      GTX 3070

4        Swift 7      Acer         $900   36      i5 9th Gen       GTX 3070

5        MacbookPro16    Apple   $3500   25      i5 9th Gen      GTX 3070

--------------------------------------------------------------------------------
```

Figure 26: Test 3 ii

• Objective	To show the quantity being deducted after the selling the computers to buyer.
• Action	1. Entering the ID = 2 2. Entering the quantity = 2
• Expected Result	Quantity of the computer is reduced after sales.
• Actual Result	Quantity of the computer has been reduced after selling.
• Conclusion	The test is successful.

Table 3: Analysis table of Test 3

4.4. Test 4

```
Please select your option........

(1) | | Press 1 to view and order a computer........
(2) | | Press 2 to view and buy a computer........
(3) | | Press 3 to exit........


Please enter your option: 2
--------------------------------------------------------------------------------
ID.      Computer Name  Brand    Price    Quantity        Generation       Graphics
--------------------------------------------------------------------------------
1        Razer Blade      Razer  $2000    40      i7 7th Gen      GTX 3060

2        XPS              Dell    $1976   50      i5 9th Gen      GTX 3070

3        Alienware       Alienware       $1978   34      i5 9th Gen      GTX 3070

4        Swift 7      Acer         $900   35      i5 9th Gen       GTX 3070

5        MacbookPro16    Apple   $3500   30      i5 9th Gen      GTX 3070

--------------------------------------------------------------------------------

Enter the ID of Commputer you want to buy: 4
You can now buy the Computer
Enter the quantity of the Computers: 2
Do you want to buy more computers?
 Choose 'Y' for Yes and 'N' for No: N
Enter your Name: Priya
Enter your contact number: 9841424806
```

Figure 27: Test 4 i.

```
*********************************************************************************************
                                Invoice

*********************************************************************************************
Name:  Priya
Contact number:  9841424806
Date:  2023-7-28                        Time:  7:39:27
Status: Paid
*********************************************************************************************
SN.     ID      Name    Brand   Price   Quantity        Generation      Graphics
1        4       Swift 7             Acer       $ 900.0         2 t i5 9th Gen t  GTX 3070


Initial Amount:  1800.0
*********************************************************************************************
Note: The shipping cost for all transaction is 125$
*********************************************************************************************
Shipping Cost: 125$
Final Amount:  1925.0
```

Figure 28: Test 4 ii.

```
Name: Priya
Phone number: 9841424806
Date of Buy: 2023-7-28          Time of Buy: 7:39:27
Status: Paid
SN.     ID      Costume Name    Brand   Price   Quantity        Generation      Graphics
1       4       Swift 7      Acer       $900.0  2       i5 9th Gen       GTX 3070
Total Price: 1800.0$
*****************************************************************************************
Note: The shipping cost for all transaction is 125$
*****************************************************************************************
Shipping Cost: 125$
Final Amount: 1925.0$
```

Figure 29: Test 4 iii.

• Objective	1. To show the complete buy process. 2. To show the output in the shell 3. To show the invoice generation of the process in text file format.
• Action	1. Please enter your option: 2 2. Enter the ID of the computer: 4 3. Enter the quantity for the computer:1 4. Do you want to buy more? =N 5. Enter your name: Priya 6. Enter your contact number: 9841424806
• Expected Result	A complete buying process should be shown, a computer should be sold to the consumer, and an invoice text file should be created.
• Actual Result	Expected message and note is displayed on the shell. An invoice text file has been successful created.
• Conclusion	The test is successful.

Table 4: Analysis table of Test 4

4.5. Test 5

```
(1) | | Press 1 to view and order a computer........
(2) | | Press 2 to view and buy a computer........
(3) | | Press 3 to exit........

Please enter your option: 1
--------------------------------------------------------------------------------
ID.      Computer Name  Brand    Price    Quantity        Generation       Graphics
--------------------------------------------------------------------------------
1        Razer Blade      Razer  $2000    40      i7 7th Gen      GTX 3060

2        XPS              Dell    $1976   50      i5 9th Gen      GTX 3070

3        Alienware       Alienware       $1978   33      i5 9th Gen      GTX 3070

4        Swift 7      Acer         $900   35      i5 9th Gen       GTX 3070

5        MacbookPro16    Apple   $3500   30      i5 9th Gen      GTX 3070

--------------------------------------------------------------------------------

Enter the ID of Commputer you want to buy: 3
You can now order the Computer
Enter the quantity of the Computers: 1
Do you want to buy more computers?
 Choose 'Y' for Yes and 'N' for No: N
Enter your Name: Saroj
Enter your contact number: 9841017090
```

Figure 30: Test 5 i.

```
********************************************************************************
                                Invoice

********************************************************************************
Name:  Saroj
Contact number:  9841017090
Date:  2023-7-28                        Time:  7:38:22
Status: Paid
********************************************************************************
SN.     ID      Name    Brand   Price   Quantity        Generation      Graphics
1        3       Alienware       Alienware       $ 1978.0        1 t i5 9th Gen t GTX 3070


Intial Amount:  1978.0
VAT Amount:  257.14
Total Amount 2235.14
********************************************************************************
```

Figure 31: Test 5 ii.

```
Name: Saroj
Phone number: 9841017090
Date of Buy: 2023-7-28          Time of Buy: 7:38:22
Status: Paid
SN.     ID      Costume Name    Brand   Price   Quantity        Generation      Graphics
1       3       Alienware       Alienware       $1978.0 1       i5 9th Gen      GTX 3070
Intial Amount: 1978.0$
VAT Amount: 257.14$
Final Amount: 2235.14$
```

Figure 32: Test 5 iii.

• Objective	1. To show the complete order process. 2. To show the output in the shell 3. To show the invoice generation of the process in text file format.
• Action	1. Please enter your option: 1 2. Enter the ID of the computer: 3 3. Enter the quantity for the computer: 1 4. Do you want to order more? = N 5. Enter your name: Saroj 6. Enter your contact number: 9841017090
• Expected Result	A complete ordering process should be shown, a computer should be bought by the store owner, and an invoice text file should be created.
• Actual Result	Expected message and note is displayed on the shell. An invoice text file has been successful created.
• Conclusion	The test is successful.

Table 5: Analysis table of Test 5

4.6. Test 6

```
Please enter your option: 1
--------------------------------------------------------------------------------
ID.      Computer Name  Brand    Price    Quantity        Generation       Graphics
--------------------------------------------------------------------------------
1        Razer Blade      Razer  $2000    40      i7 7th Gen      GTX 3060

2        XPS              Dell    $1976   50      i5 9th Gen      GTX 3070

3        Alienware       Alienware       $1978   34      i5 9th Gen      GTX 3070

4        Swift 7      Acer         $900   33      i5 9th Gen       GTX 3070

5        MacbookPro16    Apple   $3500   30      i5 9th Gen      GTX 3070

--------------------------------------------------------------------------------

Enter the ID of Commputer you want to buy:
```

Figure 33: Test 6 i.

```
Enter the ID of Commputer you want to buy: 2
You can now order the Computer
Enter the quantity of the Computers: 3
Do you want to buy more computers?
 Choose 'Y' for Yes and 'N' for No:
```

Figure 34: Test 6 ii.

```
Do you want to buy more computers?
 Choose 'Y' for Yes and 'N' for No: Y
--------------------------------------------------------------------------------
ID.      Computer Name  Brand    Price    Quantity        Generation       Graphics
--------------------------------------------------------------------------------
1        Razer Blade      Razer  $2000    40      i7 7th Gen      GTX 3060

2        XPS              Dell    $1976   53      i5 9th Gen      GTX 3070

3        Alienware       Alienware       $1978   34      i5 9th Gen      GTX 3070

4        Swift 7      Acer         $900   33      i5 9th Gen       GTX 3070

5        MacbookPro16    Apple   $3500   30      i5 9th Gen      GTX 3070

--------------------------------------------------------------------------------

Enter the ID of Commputer you want to buy: 1
You can now order the Computer
Enter the quantity of the Computers: 2
```

Figure 35: Test 6 iii.

• Objective	To have the increment changes in quantity on the list of computers when ordered.
• Action	1. Choose to order laptop. 2. Select the desired quantity. 3. Input name, contact number. 4. Generate Invoice. 5. Go back to order other computers
• Expected Result	The quantity should increase of the computer after the purchase of order.
• Actual Result	We can see the quantity increase after the purchase.
• Conclusion	The test is successful.

Table 6: Analysis Table of Test 6

5. Conclusion

It was assumed that students would have intermediate level Python knowledge and experience when this course material was assigned. a whole new area of real-world project development where we had to create an application that anyone without technical knowledge could use. The fact that the issue was based on the creation of a real-world scenario made the implementation a little challenging because you had to know how to simplify and make the user interface appealing and simple to use. We had a decent amount of time to finish the project. As a result, it was feasible to conduct research using a variety of YouTube teaching videos, review previous lessons, as well as check Internet blogs and materials for more information. Additionally, the goal is to make it simple and unique so that you can think outside the box and try out new ideas. I am appreciative of my teachers for helping me realize my vision and for recognizing the various points of view I wanted to share and experiment with.

Therefore, this coursework has enlarged our horizons and taught us how to develop fundamental real-world projects. Since we now have a solid understanding of the subject, working on projects that are similar to or distinct from this one will be relatively easier. Understanding each component that was needed to make the software simple, lightweight, and functional took a lot of time. There are a lot of work to be done before becoming great and making fewer mistakes. We practiced everything from creating lists and dictionaries to exception handling and its implementation, which is heavily used in real life. Since research was one of the top goals for finishing the coursework, several approaches beyond the requirements for the coursework have also been learned.

I have already gained a lot of knowledge from the coursework by overcoming a number of real-life and coding challenges. Since there was initially no rush, much of my coursework was completed at the last minute, but this time I was able to manage my time well enough to complete the project and learn and review Python from scratch, having a significant impact on my experience, knowledge, and ethics. In the end, I think I have developed the skills to create and implement many programs of a similar level, and in the next days, I am confident that I can complete similar tasks like these.

6. Bibliography

Chart, 2020. [Online]
Available at: https://www.lucidchart.com/pages/what-is-a-flowchart-tutorial

Python, 2022. [Online]
Available at: https://realpython.com/python-dicts/

Python-Org, n.d. [Online]
Available at: https://www.python.org/doc/essays/blurb/

Times, n.d. [Online]
Available at: https://economictimes.indiatimes.com/definition/pseudocode

7. Appendix

Main.py

```
1#importing order.py

import order

#importing buy.py

import buy

print("+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++")

print("|-----------------Anganas's Laptop Store-------------------------|")

print("+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++")

#Creating a function for invalid message

def inv_message():

    print("\n\t\t***Invalid Input!! Please follow the instructions properly***\n")

#This is the main page that allows an user to choose three options either for ordering or buying a computer or exit the program all together.

def main():

    '''Creating a function that holds the key components of the main page.

        It uses match-case to allow and link the three options the user might choose. '1' for viewing and

        ordering a computer, '2' for buying a computer and '3' to exit the program. For any other

        user input that doesn't satisfy any of the conditions an invalid message is printed.

        Till the condition is true the program keeps on looping'''

    while(True):

        print("\nPlease select your option........\n")

        print("(1) | | Press 1 to view and order a computer........")

        print("(2) | | Press 2 to view and buy a computer........")
```

```
    print("(3) | | Press 3 to exit........\n")
    option= input("\nPlease enter your option: ")
    if option == '1':
        order.order_main()
    elif option == '2':
        buy.buy_main()
    elif option == '3':
        break
    else:
        inv_message()
main()
```

View.py

```
#Creating a dictionary
d_computer={}


def inv_message():
    print("\n\t\t***Invalid Input!! Please follow the instructions properly***\n")


def display_computers():
    '''Creating a function that reads the text file containing information about the computers with
    their ID number, name, brand, price and quantity. It then removes the commas and lines
    and stores the data into a dictionary and then returns the given data for viewing
    when function is called'''
    count = 0
    print("-----------------------------------------------------------------------------------------")
    print('ID. \t Computer Name \tBrand \tPrice\tQuantity\tGeneration\t Graphics')
    print("-----------------------------------------------------------------------------------------")
    textdoc = open("computer.txt","r")
    lines = textdoc.read()
    lines = lines.split("\n")
    while ("" in lines):
        lines.remove("")
    for i in range(len(lines)):
        count = count+1
        d_computer[count] = lines[i].split(",")
    for key,value in d_computer.items():
        print(key,end="\t")
```

```
        for i in value:
            print(i,end="\t")
        print("\n")
    print("-----------------------------------------------------------------------------------------")
    return d_computer
```

Buy.py

```
#importing view.py
import view
# import datetime
import datetime
#importing the dictionary from view
from view import d_computer
#Creating a function for invalid message
def inv_message():
    print("\n\t\t***Invalid Input!! Please follow the instructions properly***\n")


def buy_main():
    loop=True
    '''The main function that calls other functions for buying the computers from the manufacturer when run. This
    functions runs in a loop until the process is complete or the user chooses to end it.
    This function is responsible in calling all other functions to start and end the computer buying process'''
    while loop == True:
        view.display_computers()
        computer_id = buy_computer(d_computer)
        quantity = check_quantity(d_computer)
        enter = False
        while enter == False:
            up_quantity(d_computer, computer_id, quantity)
            update_stock(d_computer)
            option = input("Do you want to buy more computers?\n Choose 'Y' for Yes and 'N' for No: " ).upper()
```

```
        if option == "Y":
            enter = True
        elif option == "N":
            gen_invoice(d_computer)
            loop = False
            enter = True
            break
        else:
            inv_message()


#function resposible for validating Computer ID
def buy_computer(d_computer):
    """This function checks the inputted Computer ID compares it to the length in dictionary,
    checks if the inputted value is less than zero or not. This function takes the dictionary as the
    parameter and returns the Computer ID. In case an invalid ID is given this function will
    show an output of invalid ID"'
    while True:
        try:
            computer_id=(int(input("\nEnter the ID of Commputer you want to buy: ")))
            if computer_id>0 and computer_id<=len(d_computer):
                print ("You can now buy the Computer")
                break
            else:
                print ("Please enter a valid ID")
        except:
            inv_message()
```

```
    return computer_id


#function responsible for validating quantity
def check_quantity(d_computer):
    '''Asks the user to input a quantity. Checks if the quantity is less tha zero or not. If it is
less than zero shows invalid input. This function takes the dictionary as the parameter
and returns the
quantity'''
    try:
        quantity = int(input("Enter the quantity of the Computers: "))
        if quantity<0:
            print("Please enter the desired quantity in a proper format! ")
        else:
            return quantity
    except:
        inv_message()


#function responsible in updating the quantity
buy=[]#creating a list
def up_quantity(d_computer,computer_id,quantity):
    '''This function takes the dictionary, the computer id and the quantity as the
parameter. It calls
the data of the third index of the called computer id converts into integer and allows to
add
bought from the desired quantity of the user and increase the number of computers
after the
user has bought them. It then again converts the increased quantity to a string and
update
it to the list. While it is updated to the list the text file hasnot been overwritten'''
    quantity_left = int(d_computer[computer_id][3]) - quantity
```

```
    d_computer[computer_id][3] = str(quantity_left)
    buy.append([computer_id,quantity])
    return buy

#function responsible in updating the quantity in the text file
'''Opens the text file with write function and channges the assigned value adapted
at the dictionary'''
def update_stock(d_computer):
    s = open("computer.txt","w")
    for key,value in d_computer.items():
        s.write(",".join(value))
        s.write("\n")
    s.close()

#Function reponsible for generating the invoices
def gen_invoice(d_computer):
    Datetime = datetime.datetime.now()
    Year=str(Datetime.year)
    Month=str(Datetime.month)
    Day=str(Datetime.day)
    Hour=str(Datetime.hour)
    Minutes=str(Datetime.minute)
    Sec=str(Datetime.second)
    Date = Year+"-"+Month+"-"+Day
    Time = Hour+":"+Minutes+':'+Sec
    user_name = input("Enter your Name: ")
    try:
        ph_number = input("Enter your contact number: ")
```

```
        ph = str(ph_number)
    except:
        print("Please input correct phone number")

print("\n*****************************************************************************************
***")
    print("\t\t\t\tInvoice")

print("\n*****************************************************************************************
***")
    print("Name: ",user_name)
    print("Contact number: ",ph_number)
    print("Date: ",Date , "\t\t\t","Time: ",Time)
    print("Status: Paid")

print("******************************************************************************************
*")
    print("SN.\tID\tName\tBrand\tPrice\tQuantity\tGeneration\tGraphics")
    Total = 0
    for i in range(len(buy)):
        c_id = int(buy[i][0])
        csr_id = str(c_id)
        c_name = d_computer[c_id][0]
        c_brand = d_computer[c_id][1]
        c_price = float(d_computer[c_id][2].replace("$",""))
        c_quantity = int(buy[i][1])
        cr_price = float(d_computer[c_id][2].replace("$",""))*c_quantity
        c_generation = d_computer[c_id][4]
        c_graphics = d_computer[c_id][5]
        Total += cr_price
        Tprice = str(Total)
```

```
        Fprice = Total + 125
        Fwprice = str(Fprice)

print(str(i+1),"\t",csr_id,"\t",c_name,"\t",c_brand,"\t","$",str(c_price),"\t",str(c_quantity),"t",
c_generation,"t",c_graphics)
    print("\n")
    print("Initial Amount: ",str(Tprice))

print("*****************************************************************************************
*")
    print("Note: The shipping cost for all transaction is 125$")

print("*****************************************************************************************
*")
    print("Shipping Cost: 125$")
    print("Final Amount: ",str(Fprice))


   #Creating the notepad file
    buyFilename="bought_by_"+user_name+".txt"
    with open(buyFilename,"w") as file:
        file.write("\nName: " + user_name)
        file.write("\nPhone number: " + ph_number)
        file.write("\nDate of Buy: " + Date + "\t\tTime of Buy: " + Time)
        file.write("\nStatus: Paid")
        file.write("\nSN.\tID\tCostume
Name\tBrand\tPrice\tQuantity\tGeneration\tGraphics")
        for j in range(len(buy)):
            cw_id = int(buy[j][0])
            cw_quantity= int(buy[j][1])
            cw_name = d_computer[cw_id][0]
            cw_brand = d_computer[cw_id][1]
```

```
        cw_price = float(d_computer[cw_id][2].replace("$",""))
        crw_price = float(d_computer[cw_id][2].replace("$",""))*c_quantity
        cw_generation =d_computer[cw_id][4]
        cw_graphics =d_computer[cw_id][5]

file.write("\n"+str(j+1)+"\t"+str(cw_id)+"\t"+cw_name+"\t"+cw_brand+"\t"+"$"+str(cw_price)+"\t"+str(cw_quantity)+"\t"+cw_generation+"\t"+cw_graphics)
    file.write("\nTotal Price: " + Tprice + "$")

file.write("\n*****************************************************************************************")
    file.write("\nNote: The shipping cost for all transaction is 125$")

file.write("\n*****************************************************************************************")
    file.write("\nShipping Cost: 125$")
    file.write("\nFinal Amount: " + Fwprice + "$")
    file.close()
```

Order.py

```
#importing view.py
import view
# import datetime
import datetime
#importing the dictionary from view
from view import d_computer
#Creating a function for invalid message
def inv_message():
    print("\n\t\t***Invalid Input!! Please follow the instructions properly***\n")


def order_main():
    loop=True
    '''The main function that calls other functions for ordering the computers from the manufacturer when run. This
    functions runs in a loop until the process is complete or the user chooses to end it.
    This function is responsible in calling all other functions to start and end the comouter ordering process'''
    while loop == True:
        view.display_computers()
        computer_id = order_computer(d_computer)
        quantity = check_quantity(d_computer)
        enter = False
        while enter == False:
            up_quantity(d_computer, computer_id, quantity)
            update_stock(d_computer)
            option = input("Do you want to buy more computers?\n Choose 'Y' for Yes and 'N' for No: " ).upper()
```

```
        if option == "Y":
            enter = True
        elif option == "N":
            gen_invoice(d_computer)
            loop = False
            enter = True
            break
        else:
            inv_message()


#function resposible for validating Computer ID
def order_computer(d_computer):
    """This function checks the inputted Computer ID compares it to the length in dictionary,
    checks if the inputted value is less than zero or not. This function takes the dictionary as the
    parameter and returns the Computer ID. In case an invalid ID is given this function will
    show an output of invalid ID"'
    while True:
        try:
            computer_id=(int(input("\nEnter the ID of Commputer you want to buy: ")))
            if computer_id>0 and computer_id<=len(d_computer):
                print ("You can now order the Computer")
                break
            else:
                print ("Please enter a valid ID")
        except:
            inv_message()
```

```
    return computer_id


#function responsible for validating quantity
def check_quantity(d_computer):
    """Asks the user to input a quantity. Checks if the quantity is less tha zero or not. If it is
less than zero shows invalid input. This function takes the dictionary as the parameter
and returns the
quantity"""
    try:
        quantity = int(input("Enter the quantity of the Computers: "))
        if quantity<0:
            print("Please enter the desired quantity in a proper format! ")
        else:
            return quantity
    except:
        inv_message()


#function responsible in updating the quantity
order=[]#creating a list
def up_quantity(d_computer,computer_id,quantity):
    '''This function takes the dictionary, the computer id and the quantity as the
parameter. It calls
the data of the third index of the called computer id converts into integer and allows to
add
order from the desired quantity of the user and increase the number of computers after
the
user has ordered them. It then again converts the increased quantity to a string and
update
it to the list. While it is updated to the list the text file hasnot been overwritten'''
    quantity_left = int(d_computer[computer_id][3]) + quantity
```

```
    d_computer[computer_id][3] = str(quantity_left)
    order.append([computer_id,quantity])
    return order

#function responsible in updating the quantity in the text file
'''Opens the text file with write function and channges the assigned value adapted
at the dictionary'''
def update_stock(d_computer):
    s = open("computer.txt","w")
    for key,value in d_computer.items():
        s.write(",".join(value))
        s.write("\n")
    s.close()

#Function reponsible for generating the invoices
def gen_invoice(d_computer):
    Datetime = datetime.datetime.now()
    Year=str(Datetime.year)
    Month=str(Datetime.month)
    Day=str(Datetime.day)
    Hour=str(Datetime.hour)
    Minutes=str(Datetime.minute)
    Sec=str(Datetime.second)
    Date = Year+"-"+Month+"-"+Day
    Time = Hour+":"+Minutes+':'+Sec
    user_name = input("Enter your Name: ")
    try:
        ph_number = input("Enter your contact number: ")
```

```
        ph = str(ph_number)
        ph_length = len(ph)
    except:
        print("Please input correct phone number")

print("\n*****************************************************************************************
***")
    print("\t\t\t\tInvoice")

print("\n*****************************************************************************************
***")
    print("Name: ",user_name)
    print("Contact number: ",ph_number)
    print("Date: ",Date , "\t\t\t","Time: ",Time)
    print("Status: Paid")

print("*****************************************************************************************
*")
    print("SN.\tID\tName\tBrand\tPrice\tQuantity\tGeneration\tGraphics")
    Total = 0
    for i in range(len(order)):
        c_id = int(order[i][0])
        csr_id = str(c_id)
        c_name = d_computer[c_id][0]
        c_brand = d_computer[c_id][1]
        c_price = float(d_computer[c_id][2].replace("$",""))
        c_quantity= int(order[i][1])
        cr_price = float(d_computer[c_id][2].replace("$",""))*c_quantity
        c_generation =d_computer[c_id][4]
        c_graphics =d_computer[c_id][5]
        Total += cr_price
```

```
        VAT= (13/100)*Total
        VATw = str(VAT)
        Tprice = str(Total)
        FwPrice = str (Total +VAT)

print(str(i+1),"\t",csr_id,"\t",c_name,"\t",c_brand,"\t","$",str(c_price),"\t",str(c_quantity),"t",
c_generation,"t",c_graphics)
    print("\n")
    print("Intial Amount: ",Tprice)
    print("VAT Amount: ", str(VAT))
    print("Total Amount", str(Total+VAT))

print("**********************************************************************************************
*")
    orderFilename="ordered_by_"+user_name+".txt"
    with open(orderFilename,"w") as file:
        file.write("\nName: " + user_name)
        file.write("\nPhone number: " + ph_number)
        file.write("\nDate of Buy: " + Date + "\t\tTime of Buy: " + Time)
        file.write("\nStatus: Paid")
        file.write("\nSN.\tID\tCostume
Name\tBrand\tPrice\tQuantity\tGeneration\tGraphics")
        for j in range(len(order)):
            cw_id = int(order[j][0])
            cw_quantity= int(order[j][1])
            cw_name = d_computer[cw_id][0]
            cw_brand = d_computer[cw_id][1]
            cw_price = float(d_computer[cw_id][2].replace("$",""))
            crw_price = float(d_computer[cw_id][2].replace("$",""))*cw_quantity
            cw_generation =d_computer[cw_id][4]
```

```
        cw_graphics =d_computer[cw_id][5]

file.write("\n"+str(j+1)+"\t"+str(cw_id)+"\t"+cw_name+"\t"+cw_brand+"\t"+"$"+str(cw_pric
e)+"\t"+str(cw_quantity)+"\t"+cw_generation+"\t"+cw_graphics)
    file.write("\nIntial Amount: " +Tprice+"$")
    file.write("\nVAT Amount: "+ VATw+"$")
    file.write("\nFinal Amount: " + FwPrice + "$")
    file.close()
```