Module Code & Module Title:

CS4051NI & Fundamental of Computing

Assessment Weightage & Type:

60% Individual Coursework

Year and Semester:

2022-2023 Autumn

Student Name: Kiran rai

London Met ID: 22072136

College ID: NP05CP4A220056

Assignment Due Date: 12/05/2023

Word Count:3540

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 mark of zero will be awarded.

Table of Contents

Table of Figures

Table of Tables

1 Introduction

1.1 Introduction to Python

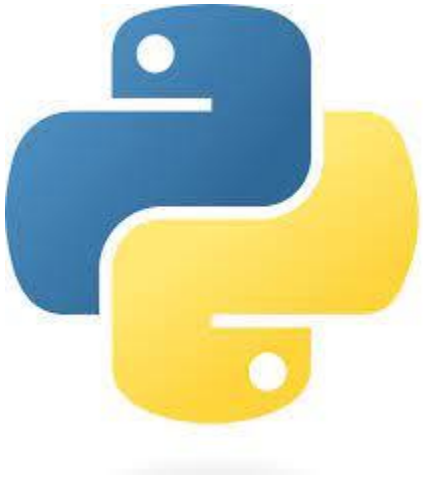

Figure 1: Python

Python is currently a popular programming language which was created by Guido van Rossum, and released in 1991. Since, then it has been optimized a lot and currently the latest version of python is 3.11. It is used **FOR** web development, software development, mathematics, etc. It works on cross plat**FOR**m like windows, Mac, Linux, etc. Unlike other programming languages, It's syntax is easy and very much similar to English language.

(W3schools, 2023)

1.2 Tools used

Figure 2: Visual Studio Code

Visual studio code also referred as Vs Code is a source-code editor. It was made by Microsoft with the Electron framework. It works on cross plat**FOR**m like windows, linux and macOS. It has various features which makes user time easy while using Vs Code. It has a debugger, error highlighter, snippets and embedded Git.

(Wikipidea, 2023)

Figure 3: Word

Word is a text editor but with enhanced features like spelling check, grammar check, text and **FOR**mat **FOR**matting. It also supports coping, pasting, moving and deleting text within the documentation.

Figure 4: Snipping tool

Snipping tool is a very useful app which is used **FOR** grabbing a part of a image as per desire. It is very easy to use and save photos.

2 Algorithm

Algorithm are set of steps which are carried out to find solution to certain problem. Algorithm are not language-specific, so they are used in many programming languages.

(Sharma, 2023)

Step 1: DEFINE a function available

Step 2: Create an empty dictionary av

Step 3: OPEN laptop.txt in read-only mode

Step 4: Run a loop in every line in the file

Step 5: Separate each line by commas and assign values to the variable

Step 6: Add all laptop information to the dictionary av

Step 7: PRINT all laptop details

Step 8: Return av

Step 9: DEFINE a new function companies

Step 10: CALL the function available to gain the details of laptops stored in dictionary

Step 11: Ask the user if they want to continue or not

Step 12: If user want to continue ask for name of the distributor and the name of laptop

Step 13: If the demanded laptop is not available, PRINT "Not Available" and return to function available

Step14: If the demanded laptop is available, get the details of the laptop and ask FOR the quantity of laptop to purchase

Step 15: CALCULATE total price without VAT and add VAT to get grand total

Step 16: Create a detailed invoice and save it to a file called retail.txt

Step 17: UPDATE the quantity of laptop in the dictionary and UPDATE the laptop.txt file as well

Step 18: If user don’t want to continue return to function options

Step 19: DEFINE a new function customer

Step 20: CALL the function available to gain the details of laptops stored in dictionary

Step 21: Ask the user if they want to continue or not

Step 22: If user want to continue ask for name of the customer and the name of laptop

Step 23: If the demanded laptop is not available, PRINT "Laptop not available" and call the customer function again

Step 24: If the demanded laptop is available get the details of the laptop from dictionary and ask FOR the quantity to purchase

Step 24: Ask the user if they want to get their laptop shipped or not

Step 25: If yes add shipping price to total price to get grand total

Step 26: Create a detailed invoice and save to a file called sell_laptop.txt

Step 27: UPDATE the quantity of laptop in the dictionary and UPDATE the laptop.txt file as well

Step 28: If user don’t want to continue return to function options

Step 29: Create a function details that lists the available options

Step 30: Create a function options which will ask the user for their choice and CALL appropriate functions based on the choice

Step 31: If user choose A, CALL function available

Step 32: If user choose B, CALL function companies

Step 33: If user choose C, CALL function customer

Step 34: If user choose D, exit the program

Step 35: Display an error message if user enter invalid option and CALL function options

Step 36: CALL function details to display all options

Step 37: CALL function options to get user choice

Step 38: End

3 Pseudocode

START

DEFINE a function available:

    CREATE an empty dictionary av

    PRINT "Name", "Brand", "Price", "Quantity", "Processor", "Graphics"

    OPEN laptop.txt file in read mode

    FOR each line in the file:

USING split and strip function, strip whitespace and split line into name, brand, price, quantity, processor, and graphics

CONVERT price to float and quantity to an integer

ADD name as key value and a dictionary containing brand, price, quantity, processor, and graphics as the value

PRINT contents of av

RETURN av

END FOR

DEFINE a function companies:

PRINT "List of Available Laptops: "

CREATE a variable laptopall and INITIALIZE it with function available

CREATE a String variable d

IF D is equals to "YES"

CREATE a variable namedis and store name of distributor

CREATE a variable namelap and store name of laptop

IF namelap not in laptopall

PRINT "Not Available"

RETURN function companies

CREATE a variable laptop and **SET** laptopall[namelap.upper()]

CREATE a Integer variable quantity

CREATE a Integer variable total_without_VAT and **CALCULATE** price of laptop price× quantity

CALCULATE quantity=quantity + number of purchased laptop

CREATE a variable Vat and **CALCULATE** 0.13×total_without_VAT

CREATE a variable grand_total and **CALCULATE** Vat + total_without_VAT

ELIF D is equals to "NO"

RETURN options()

ELSE

PRINT "Please enter appropriate choice:"

RETURN companies()

END IF

CREATE a detailed invoice

OPEN retail.txt file in write mode

WRITE the created invoice in retail.txt

PRINT "Your bill has been generated"

OPEN laptop.txt file in write mode

FOR each namelap, details in laptopall:

SET name to namelap

SET brand to details['brand']

SET price to details['quantity']

SET quantity to details['quantity']

SET processor to details['processor']

SET graphics to details['graphics']

UPDATE laptop.txt file with new values

DEFINE a function customer:

PRINT "List of Available Laptops: "

CREATE a variable laptopall and INITIALIZE it with function available

CREATE a String variable d

IF D is equals to "YES"

CREATE a variable name and ask **FOR** input from the user

CREATE a variable namelap and ask **FOR** input from the user

IF namelap is not in laptopall

PRINT "Laptop not available"

RETURN function customer

END IF

CREATE a variable laptop_upper and **SET** laptopall[namelap.upper()]

TRY

CREATE a variable quantity and ask input from user

EXCEPT

PRINT "Please enter valid value"

IF laptop_upper ['quantity'] is less than user inserted quantity

PRINT "Not enough stock"

RETURN function customer()

END IF

CREATE a variable price_without_shipping and **CALCULATE** price of laptop × quantity

CREATE a String variable, s and ask user whether to ship the laptop or not

IF s is equals to "YES"

CREATE a String variable and ask user **FOR** location

CREATE a variable, shipping and INITIALIZE 50 in it

ELSE

CREATE a variable, shipping and INITIALIZE 0 in it

CREATE a String variable and ASSIGN none

END IF

CALCULATE total = price_without_shipping + shipping

ELIF d is equals to "NO"

RETURN function options

END IF

CREATE a detailed invoice

OPEN sell_laptop.txt file in write mode

WRITE the created invoice in sell_laptop.txt

PRINT "Your bill has been generated"

CALCULATE quantity = quantity – number of laptop purchased

OPEN laptop.txt file in write mode

FOR each namelap, details in laptopall:

SET name to namelap

SET brand to details['brand']

SET price to details['quantity']

```
            SET quantity to details['quantity']
            SET processor to details['processor']
            SET graphics to details['graphics']
            UPDATE laptop.txt file with new values
        END FOR
PRINT welcome banner FOR the shop
DEFINE function details
    PRINT "A: Available details"
    PRINT "B: Place Order from Manufacturer"
    PRINT "C: Buy a laptop"
    PRINT "D: Exit"
    PRINT "Choose desired option"
DEFINE function options
    CREATE a String variable option
    IF option is equals to "A"
        CALL available function
    ELIF options equals to "B"
        CALL companies function
    ELIF options equals to "C"
        CALL customer function
    ELIF options equals to "D"
        PRINT "Thank you FOR shopping with us"
        CALL sys.exit()
```

```
        ELSE
            PRINT "Enter valid option"
            CALL options function
        END IF
CALL details function
CALL options function
END
```

4 Flowchart

Flowchart is a picture of steps which are to executed in program. They are arranged in sequential order and we use various shapes to execute different operations.

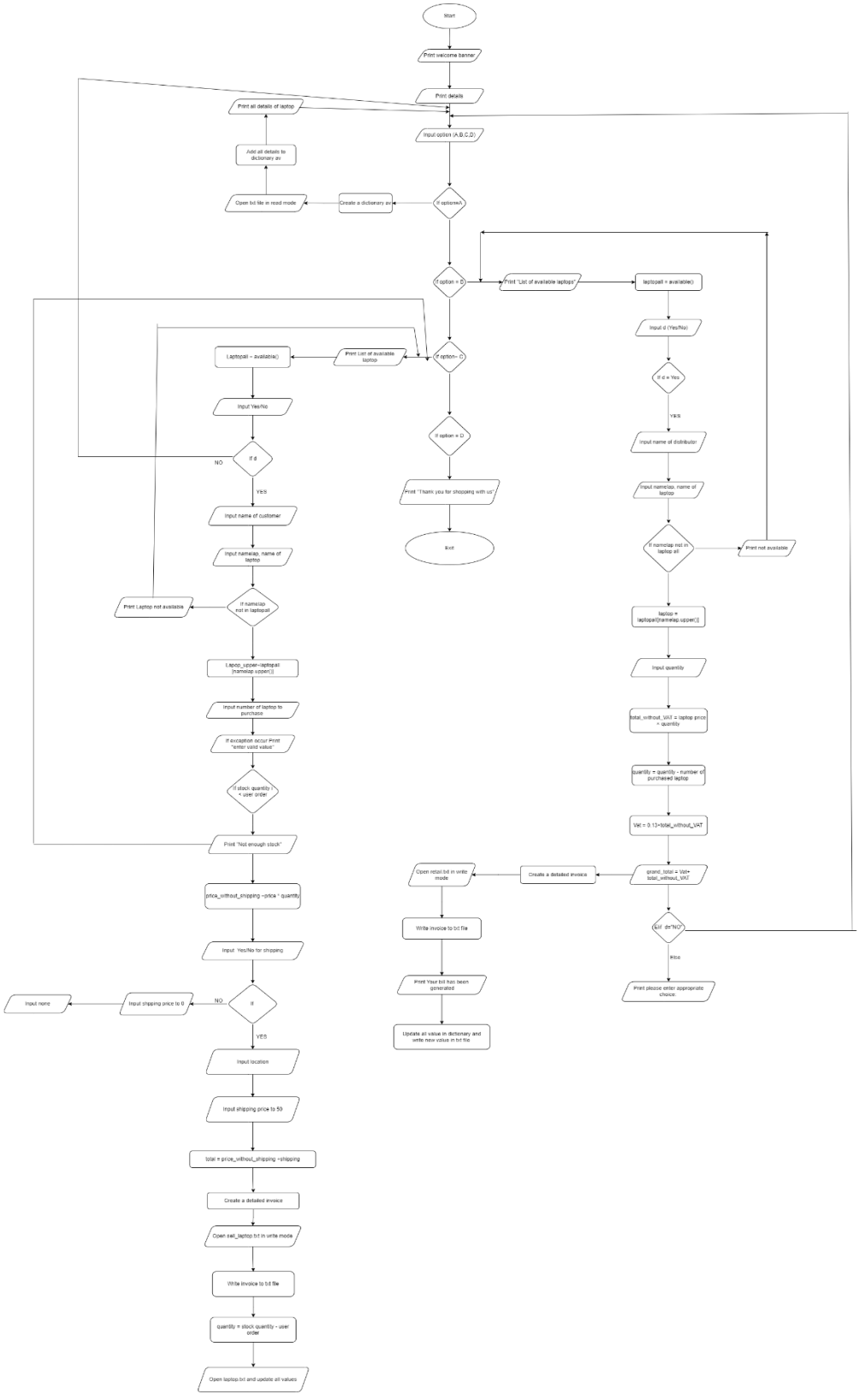

Figure 5: Flowchart of the program

5 Data Structures

Data structures are nothing but a way of organizing and storing data. They describe how data and various logical operations that can be applied to the data relate to one another. There are various ways in which data structures can be categorized. One way is to categorize them into primitive and non-primitive data types. Primitive data types include Integers, Strings, Boolean and Float whereas non-primitive data types include Tuples, Dictionary, Array, List, sets and Files.

(Sharma, 2023)

5.1 Primitive data types

```
import datetime
import sys
def available():
    av = {}
    print("Name\t\t   ","Brand\t\t","Price\t\t   ","Quantity\t\t      ","Processor\t     ","Graphics")
    with open('laptop.txt','r') as f:
        for line in f:
            name, brand, price, quantity, processor, graphics= line.strip().split(', ')
            av[str(name.upper())]={'brand': brand,'price': float(price.strip("$")),'quantity': int(quantity),'processor'
            print(f"{name:<20}{brand:<21}{price:<22}{quantity:<23}{processor:<24}{graphics}")
        return av
```

Figure 6: Integer data type

```
import datetime
import sys
def available():
    av = {}
    print("Name\t\t   ","Brand\t\t","Price\t\t   ","Quantity\t\t      ","Processor\t     ","Graphics")
    with open('laptop.txt','r') as f:
        for line in f:
            name, brand, price, quantity, processor, graphics= line.strip().split(', ')
            av[str(name.upper())]={'brand': brand,'price': float(price.strip("$")),'quantity': int(quantity),'processor'
            print(f"{name:<20}{brand:<21}{price:<22}{quantity:<23}{processor:<24}{graphics}")
        return av
```

Figure 7: Float data type

```
import datetime
import sys
def available():
    av = {}
    print("Name\t\t   ","Brand\t\t","Price\t\t   ","Quantity\t\t      ","Processor\t     ","Graphics")
    with open('laptop.txt','r') as f:
        for line in f:
            name, brand, price, quantity, processor, graphics= line.strip().split(', ')
            av[str(name.upper())]={'brand': brand,'price': float(price.strip("$")),'quantity': int(quantity),'processor'
            print(f"{name:<20}{brand:<21}{price:<22}{quantity:<23}{processor:<24}{graphics}")
        return av
```

Figure 8: String data type

```
    checked = False
    while checked == False:
        try:
            quantity   = int(input("Enter number of laptop to purchase: "))
            if (quantity <=0):
                print("Enter valid amount")
                checked = False
            else:
                checked = True
        except :
            print("Please enter valid value")
```

Figure 9: Boolean data type

5.2 Non-Primitive data types

```
import datetime
import sys
def available():
    av = {}
    print("Name\t\t   ","Brand\t\t","Price\t\t   ","Quantity\t\t      ","Processor\t     ","Graphics")
    with open('laptop.txt','r') as f:
        for line in f:
            name, brand, price, quantity, processor, graphics= line.strip().split(', ')
            av[str(name.upper())]={'brand': brand,'price': float(price.strip("$")),'quantity': int(quantity),'processor'
            print(f"{name:<20}{brand:<21}{price:<22}{quantity:<23}{processor:<24}{graphics}")
        return av
```

Figure 10: Dictionary

6 Program

This is a program that functions as a shop which sells and buys laptops. A distributor can buy laptops from the store using the "companies" function, which also creates an invoice, while a consumer can buy a laptop using the "customer" function, which also creates a receipt. The data for the available computers in the store is contained in a file named "laptop.txt," which is read by the application. The following details are listed for each laptop represented by a line in the file, each separated by a comma: name, brand, price, quantity, processor, and graphics. The "available" function reads the file and keeps a dictionary of the laptop's specifications. Both the "companies" and "customer" functions call the "available" function to obtain the list of laptops that are currently available. The distributor is to input the name and quantity of the laptop they wish to acquire in the "companies" feature. If a laptop is available, the function calculates the total cost, updates the quantity in the dictionary, and creates an invoice in a file called "retail.txt." The "laptop.txt" file is also updated by the function with the new number of laptops. The consumer is prompted to input their name, the name of the laptop they wish to purchase, and the quantity in the "customer" function. The function calculates the total cost and asks the customer if they want the laptop supplied if the laptop is available and the quantity is less than or equal to the available quantity. The function charges an additional $50 for shipping if the consumer choose to have the laptop shipped. A receipt is then produced by the function. The program has a number of problems. First, invalid input is not handled by an error mechanism. For instance, the software will fail if the user enters a quantity that is not a numeric value. Second, when input is provided in the incorrect format, the application does not manage the situation. For instance, the software will break if the user enters the price in the "laptop.txt" file without the dollar sign. Third, when a user provides an incorrect laptop name, the program does not manage the situation. Fourth, the scheme prohibits users from making simultaneous purchases of several laptops. Last but not least, the application is unable to handle situations in which the "laptop.txt" or "retail.txt" files cannot be located or opened for writing.

```
MACBOOK PRO 16      Apple
Do you want to proceed (Yes/No): yes
Enter name of distributor: HELLO WORLD
Enter name of laptop: xps
Enter number of laptop to purchase: 100
Your bill has been generated
```

Figure 11: Buying laptop from distributor

```
RAZER BLADE, Razer, $2000.0, 20, i7 7th Gen, GTX 3060
XPS, Dell, $1976.0, 29, i5 9th Gen, GTX 3070
ALIENWARE, Alienware, $1978.0, 20, i5 9th Gen, GTX 3070
SWIFT 7, Acer, $900.0, 22, i5 9th Gen, GTX 3070
MACBOOK PRO 16, Apple, $3500.0, 30, i5 9th Gen, GTX 3070
```

Figure 12: Before buying from distributor

```
RAZER BLADE, Razer, $2000.0, 20, i7 7th Gen, GTX 3060
XPS, Dell, $1976.0, 129, i5 9th Gen, GTX 3070
ALIENWARE, Alienware, $1978.0, 20, i5 9th Gen, GTX 3070
SWIFT 7, Acer, $900.0, 22, i5 9th Gen, GTX 3070
MACBOOK PRO 16, Apple, $3500.0, 30, i5 9th Gen, GTX 3070
```

Figure 13: After buying from distributor

```
*_*_*_*_*_*_*_*_*_*_*_*_*_*_*_*_*_*_*_*_*_*_*_*_*_*_*_*_*_*_*_*_*_*


*_*_*_*_*_*_*_*_*_*_*_*_*_*_*_*_*_*_*_*_*_*_*_*_*_*_*_*_*_*_*_*_*_*
Date: 2023-05-12 15:12:28
Retailer name: HELLO WORLD
____________________________________________________________________
Product: XPS
Quantity: 100
Per price: 1976.00
____________________________________________________________________
Total without VAT: 197600.00
Total: 223288.0
____________________________________________________________________
Thank you for doing business with us!
____________________________________________________________________
```

Figure 14: Generated bill in txt file

```
MACBOOK PRO 16       Apple                $3500.
Do yo want to proceed (Yes/No): yes
Enter your name: Kiran
Enter the name of laptop: xps
Enter number of laptop to purchase: 100
Do you want it to be shipped?yes
Enter your location: Itahari
Your bill has been generated
```

Figure 15: Selling to customer

```
RAZER BLADE, Razer, $2000.0, 20, i7 7th Gen, GTX 3060
XPS, Dell, $1976.0, 179, i5 9th Gen, GTX 3070
ALIENWARE, Alienware, $1978.0, 20, i5 9th Gen, GTX 3070
SWIFT 7, Acer, $900.0, 22, i5 9th Gen, GTX 3070
MACBOOK PRO 16, Apple, $3500.0, 30, i5 9th Gen, GTX 3070
```

Figure 16: Before selling it to customer

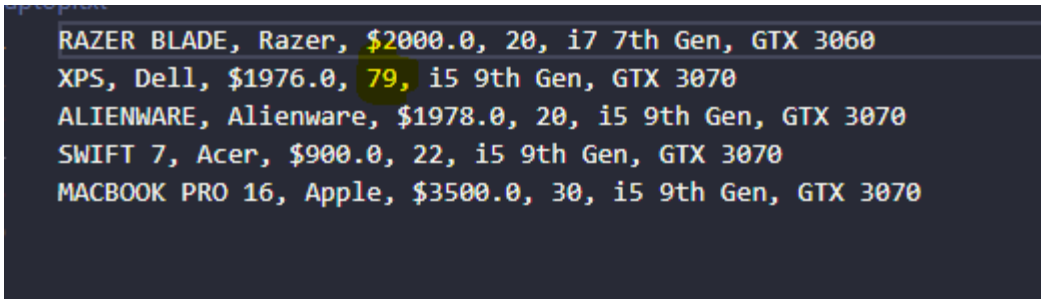

Figure 17: After selling to customer

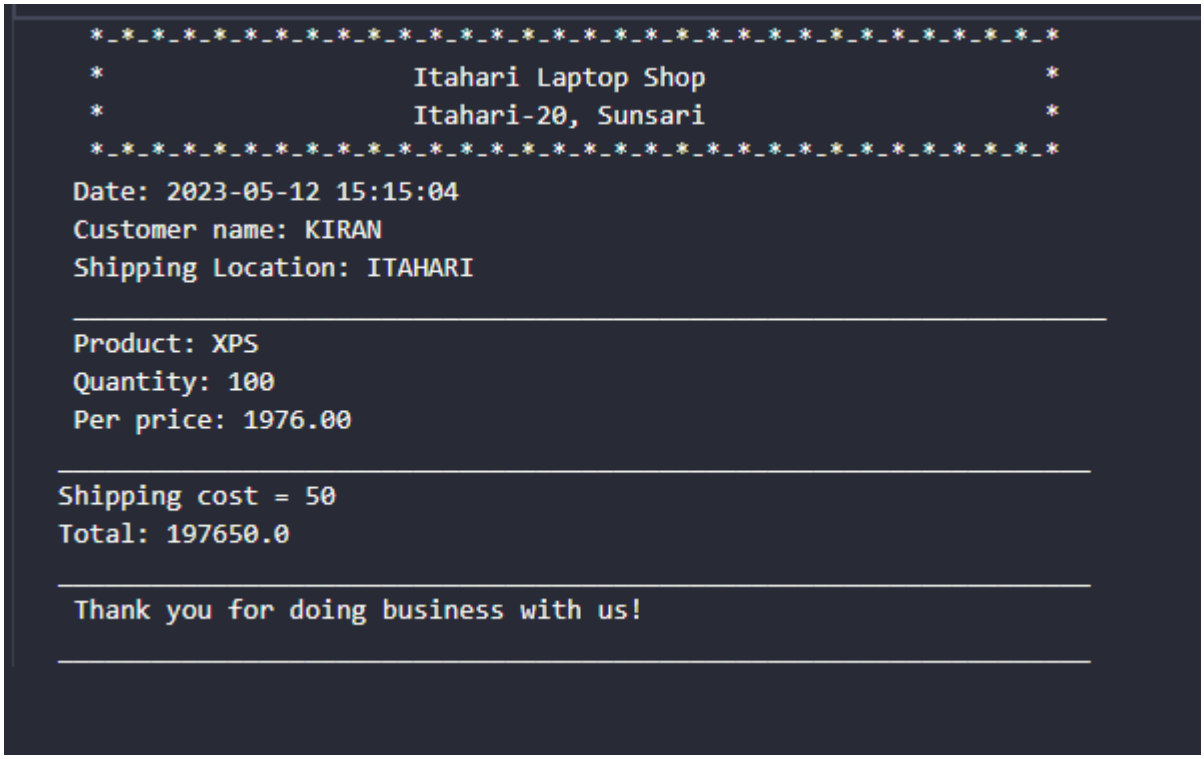

Figure 18: Generated bill in txt file

7 Testing

7.1 Test No.1

Table 1: Test No.1 Try catch block execution

Objective	To show if try and catch block is executed or not
Action	Insert String value and negative value in quantity
Expected result	Error message will occur
Actual result	Error message should occur
Conclusion	Test is successful

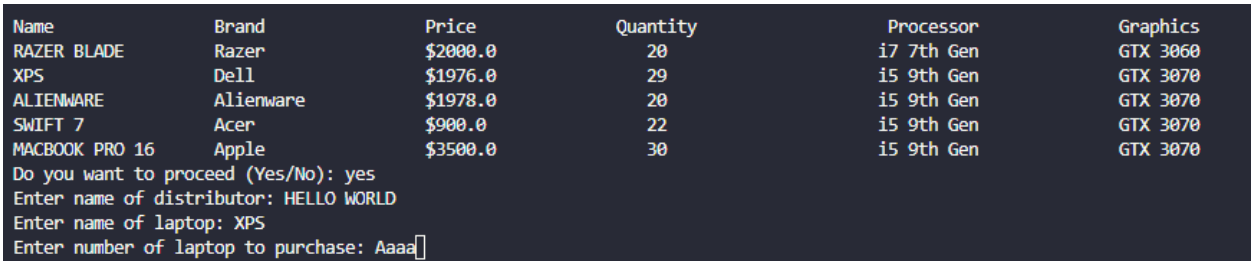

Figure 19: Inserting String value in quantity

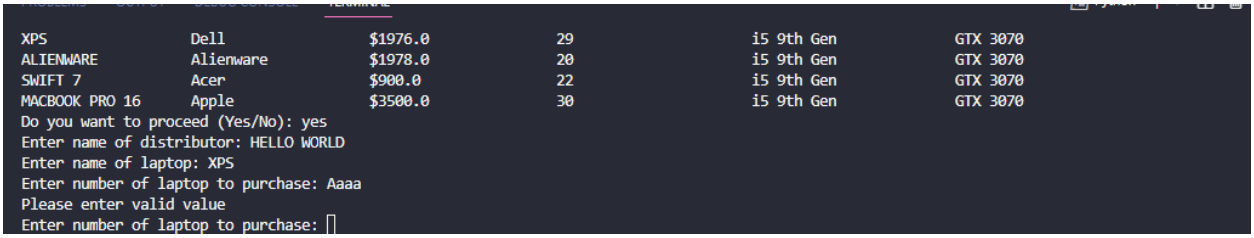

Figure 20: Error message generated

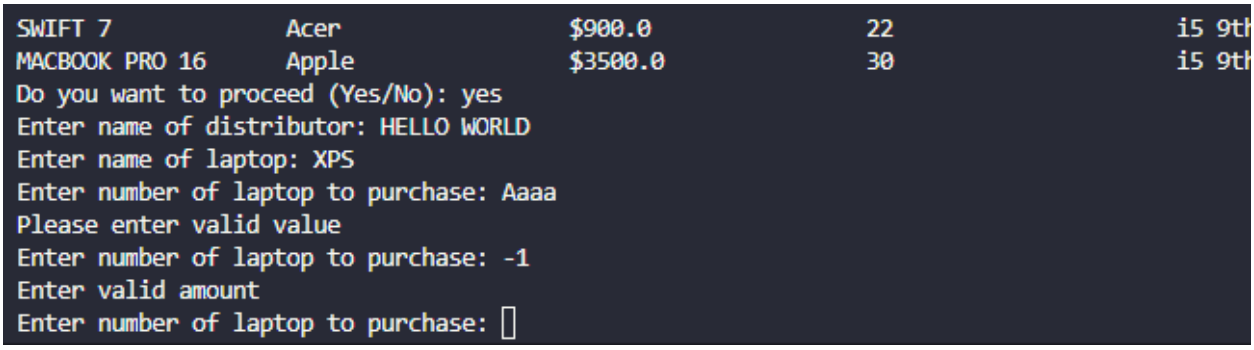

Figure 21: Inserting negative value

7.2 Test No.2

Table 2: Test No.2 Selecting option

Objective	To select available options
Action	Select all options and inspect what happens
Expected result	Options will be selected
Actual result	Options should be selected
Conclusion	Test is successful

```
------------------------------------------------------------
--------------------WELCOME TO OUR SHOP--------------------
------------------------------------------------------------
A: Available details
B: Place Order from Manufacturer
C: Buy a laptop:
D: Exit
Choose desired option:
```

Figure 22: List of options

```
Choose desired option:
a
Name            Brand           Price       Quantity        Processor       Graphics
RAZER BLADE     Razer           $2000.0        20           i7 7th Gen      GTX 3060
XPS             Dell            $1976.0        29           i5 9th Gen      GTX 3070
ALIENWARE       Alienware       $1978.0        20           i5 9th Gen      GTX 3070
SWIFT 7         Acer            $900.0         22           i5 9th Gen      GTX 3070
MACBOOK PRO 16  Apple           $3500.0        30           i5 9th Gen      GTX 3070
```

Figure 23: Choosing option A

```
b
List of Available Laptops:
Name            Brand           Price       Quantity        Processor       Graphics
RAZER BLADE     Razer           $2000.0        20           i7 7th Gen      GTX 3060
XPS             Dell            $1976.0        29           i5 9th Gen      GTX 3070
ALIENWARE       Alienware       $1978.0        20           i5 9th Gen      GTX 3070
SWIFT 7         Acer            $900.0         22           i5 9th Gen      GTX 3070
MACBOOK PRO 16  Apple           $3500.0        30           i5 9th Gen      GTX 3070
Do you want to proceed (Yes/No):
```

Figure 24: Choosing option B

```
c
List of Available Laptops:
Name                Brand               Price           Quantity            Processor           Graphics
RAZER BLADE         Razer               $2000.0           20                i7 7th Gen          GTX 3060
XPS                 Dell                $1976.0           29                i5 9th Gen          GTX 3070
ALIENWARE           Alienware           $1978.0           20                i5 9th Gen          GTX 3070
SWIFT 7             Acer                $900.0            22                i5 9th Gen          GTX 3070
MACBOOK PRO 16      Apple               $3500.0           30                i5 9th Gen          GTX 3070
Do yo want to proceed (Yes/No):
```

Figure 25: Choosing option C

```
d
Thank you for shopping with us.
```

Figure 26: Choosing option D

7.3 Test No.3

Table 3: Test No.3 Checking bill for purchasing

Objective	To check whether bill is generated while buying laptop from distributor
Action	To buy laptop from distributor
Expected result	Bill will be generated to the txt file
Actual result	Bill should be generated to the txt file
Conclusion	Test is successful

```
Name                Brand               Price           Quantity            Processor           Graphics
RAZER BLADE         Razer               $2000.0           20                i7 7th Gen          GTX 3060
XPS                 Dell                $1976.0           29                i5 9th Gen          GTX 3070
ALIENWARE           Alienware           $1978.0           20                i5 9th Gen          GTX 3070
SWIFT 7             Acer                $900.0            22                i5 9th Gen          GTX 3070
MACBOOK PRO 16      Apple               $3500.0           30                i5 9th Gen          GTX 3070
Do you want to proceed (Yes/No): yes
Enter name of distributor: Hello world
Enter name of laptop: Xps
Enter number of laptop to purchase: 10
```

Figure 27: Selecting laptop and quantity

```
Your bill has been generated
```

Figure 28: Bill has generated

```
*_*_*_*_*_*_*_*_*_*_*_*_*_*_*_*_*_*_*_*_*_*_*_*_*_*_*_*_*_*_*_*_*_*


*_*_*_*_*_*_*_*_*_*_*_*_*_*_*_*_*_*_*_*_*_*_*_*_*_*_*_*_*_*_*_*_*_*
Date: 2023-05-12 09:41:18
Retailer name: HELLO WORLD
_____________________________________________________________________
Product: XPS
Quantity: 10
Per price: 1976.00
_____________________________________________________________________
Total without VAT: 19760.00
Total: 22328.8
_____________________________________________________________________
Thank you for doing business with us!
_____________________________________________________________________
```

Figure 29: Generated bill in txt file

```
RAZER BLADE, Razer, $2000.0, 20, i7 7th Gen, GTX 3060
XPS, Dell, $1976.0, 29, i5 9th Gen, GTX 3070
ALIENWARE, Alienware, $1978.0, 20, i5 9th Gen, GTX 3070
SWIFT 7, Acer, $900.0, 22, i5 9th Gen, GTX 3070
MACBOOK PRO 16, Apple, $3500.0, 30, i5 9th Gen, GTX 3070
```

Figure 30: Stocks before buying

```
RAZER BLADE, Razer, $2000.0, 20, i7 7th Gen, GTX 3060
XPS, Dell, $1976.0, 39, i5 9th Gen, GTX 3070
ALIENWARE, Alienware, $1978.0, 20, i5 9th Gen, GTX 3070
SWIFT 7, Acer, $900.0, 22, i5 9th Gen, GTX 3070
MACBOOK PRO 16, Apple, $3500.0, 30, i5 9th Gen, GTX 3070
```

Figure 31: Stocks after buying

7.4 Test No.4

Table 4: Test No.4 Checking bill for selling

Objective	To check whether bill is generated while selling laptop to customer
Action	To sell laptop to customer
Expected result	Bill will be generated to the txt file
Actual result	Bill should be generated to the txt file
Conclusion	Test is successful

```
Name            Brand          Price          Quantity          Processor       Graphics
RAZER BLADE     Razer          $2000.0           20            i7 7th Gen       GTX 3060
XPS             Dell           $1976.0           39            i5 9th Gen       GTX 3070
ALIENWARE       Alienware      $1978.0           20            i5 9th Gen       GTX 3070
SWIFT 7         Acer           $900.0            22            i5 9th Gen       GTX 3070
MACBOOK PRO 16  Apple          $3500.0           30            i5 9th Gen       GTX 3070
Do yo want to proceed (Yes/No): yes
Enter your name: Kiran
Enter the name of laptop: xps
Enter number of laptop to purchase: 10
```

Figure 32: Selecting laptop and quantity

```
Do you want it to be shipped?Yes
Enter your location: Itahari
Your bill has been generated
```

Figure 33: Asking for shipping

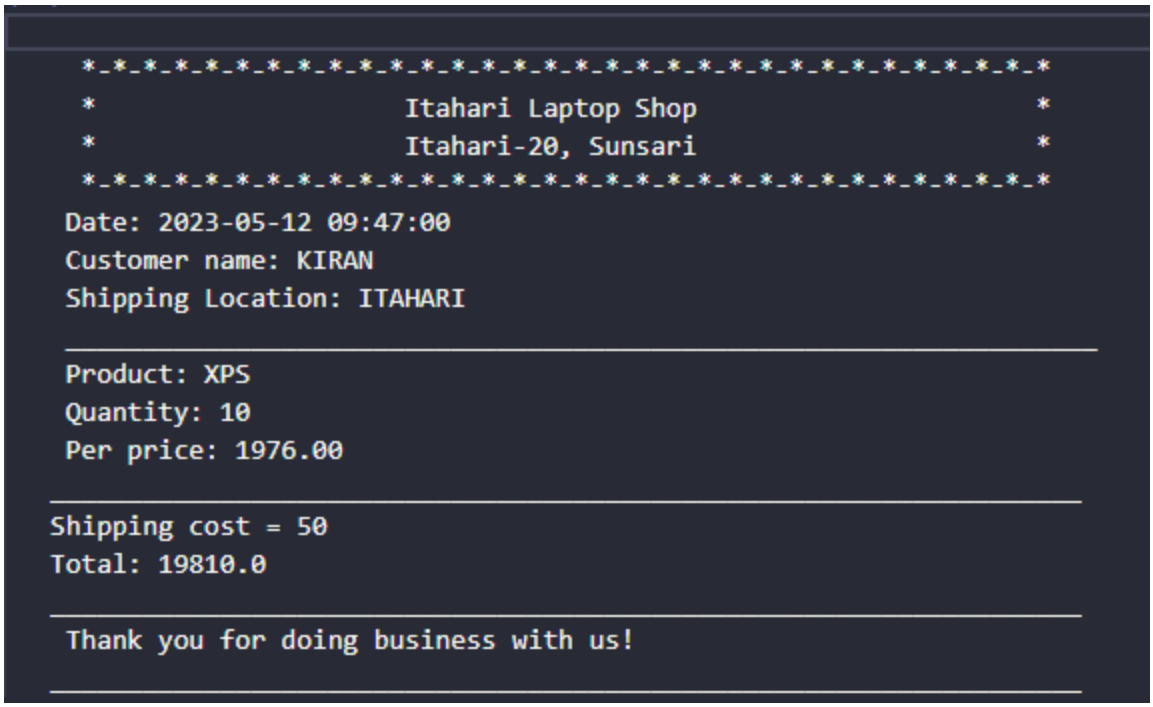

Figure 34: Generated bill in txt file

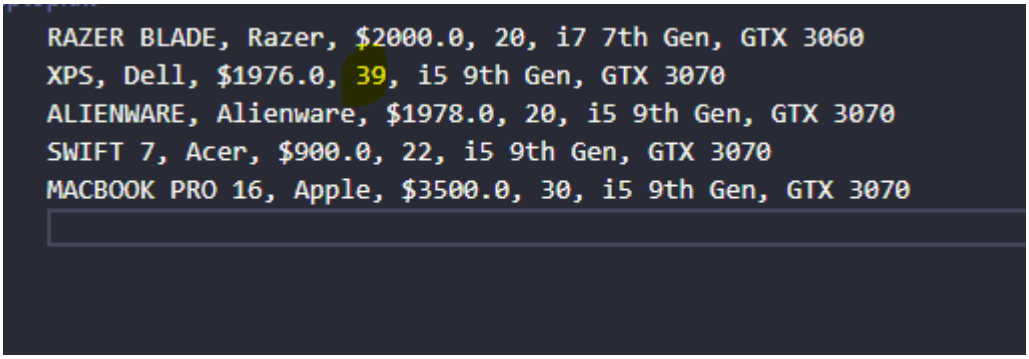

Figure 35: Before selling

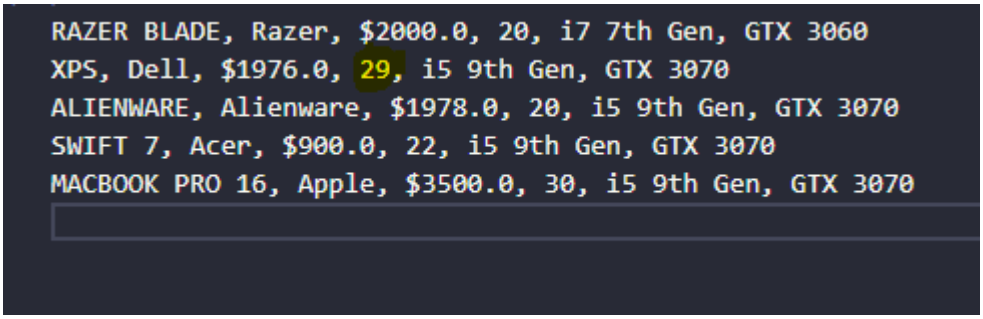

Figure 36: After selling

7.5 Test No.5

Table 5: Updating stocks

Objective	To update stocks to the txt file after buying or selling
Action	Buy laptop from distributor and sell laptop to customer
Expected result	Stocks will be updated in the txt file
Actual result	Stocks should be updated in the txt file
Conclusion	Test is successful

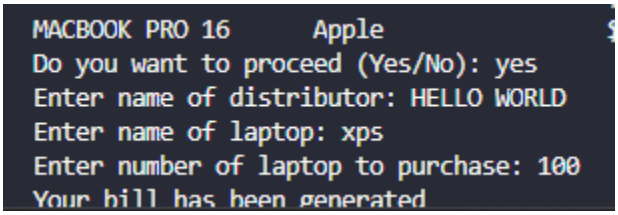

Figure 37: Selecting laptop and quantity for purchase

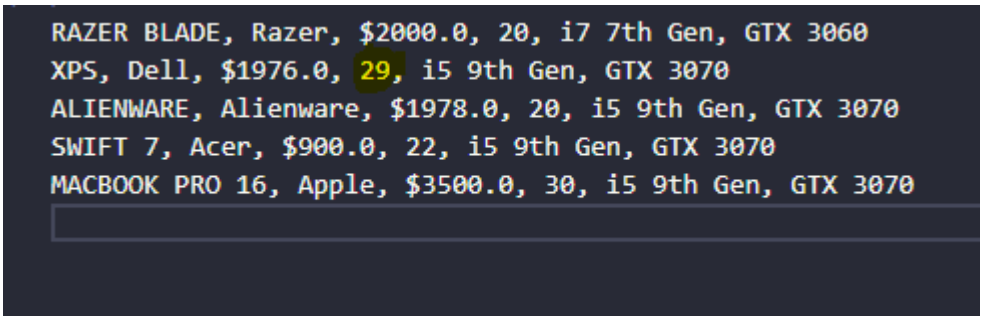

Figure 38: Before buying

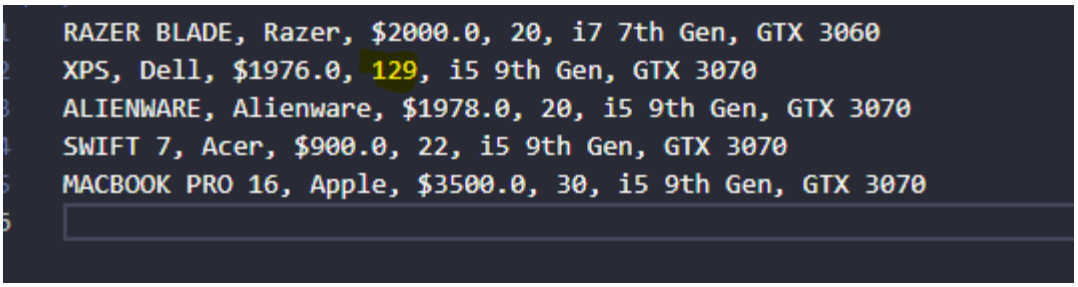

Figure 39: After buying

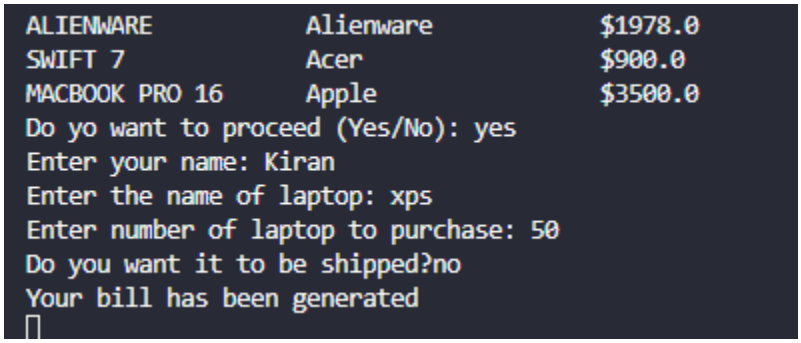

Figure 40: Selecting laptop and quantity for selling

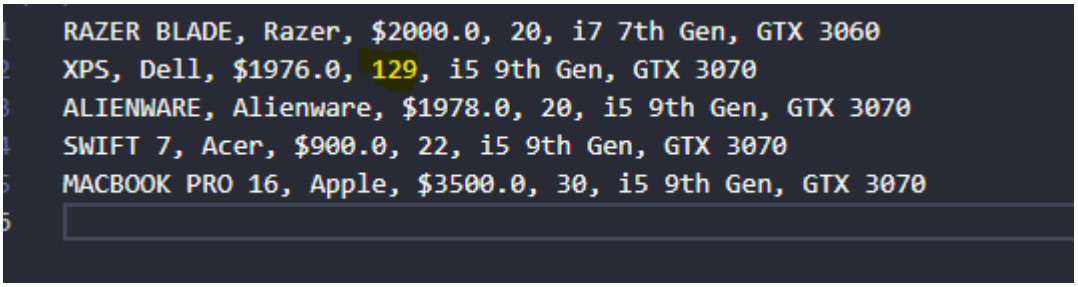

Figure 41: Before selling

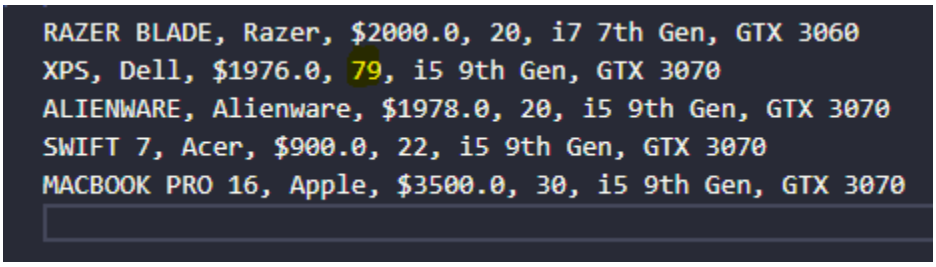

Figure 42: After selling

8 Conclusion

This coursework was very challenging and demanding. Specially flowchart and generating bill which was very time consuming. We were to create a shop with functions like buying a laptop from distributor, selling laptop to customer and update stocks to txt file. By, any means it was not an easy task we had to learn various things to execute our coursework. We had to learn to work with txt file. We had to have sound knowledge about file handling.

After completing this coursework, I have learnt many things about file handling, use of dictionary and bill generation. Which I think will be very helpful in my upcoming days while working with file. I have mostly learnt to utilize time and set daily goals. It has enhanced the experience which I had gained in my first coursework and made me more able than yesterday. It has boosted my confidence and now I feel confident to speak about python.

I may have committed many errors and I would sincerely like to apologize for that. I have tried my best to deliver this coursework. I will keep on improving in the upcoming days surely. And lastly, I would like to thank you for your precious time.

9 References

Sharma, R., 2023. *Data Structures & Algorithm in Python.* [Online]
Available at: https://www.upgrad.com/blog/data-structures-algorithm-in-python/#:~:text=are%20the%20differences%3F-,What%20are%20algorithms%20in%20Python%3F,guide%20the%20writing%20of%20algorithms.
[Accessed 09 05 2023].

Sharma, R., 2023. *Data Structures & Algorithm in Python.* [Online]
Available at: https://www.upgrad.com/blog/data-structures-algorithm-in-python/#:~:text=are%20the%20differences%3F-,What%20are%20algorithms%20in%20Python%3F,guide%20the%20writing%20of%20algorithms.
[Accessed 09 05 2023].

W3schools, 2023. *Python Introduction.* [Online]
Available at: https://www.w3schools.com/python/python_intro.asp
[Accessed 09 05 2023].

Wikipidea, 2023. *Visual Studio Code.* [Online]
Available at: https://en.wikipedia.org/wiki/Visual_Studio_Code
[Accessed 09 05 2023].

10 Appendix

```
import datetime
import sys
#  defining a function available
def available():
    # creating a dictionary
    av = {}
    # print heading of laptop details
    print("Name\t\t   ","Brand\t\t","Price\t\t   ","Quantity\t\t      ","Processor\t     ","Graphics")
    # openeing txt file in read mode
    with open('laptop.txt','r') as f:
        for line in f:
            name, brand, price, quantity, processor, graphics= line.strip().split(', ')
            av[str(name.upper())]={'brand': brand,'price': float(price.strip("$")),'quantity':
int(quantity),'processor':processor,'graphics':graphics}
            # printing all laptop details

print(f"{name:<20}{brand:<21}{price:<22}{quantity:<23}{processor:<24}{graphics}")
        return av
    # creating a function companies
def companies():
```

```
    print("List of Available Laptops: ")
    laptopall = available()
    d= str(input("Do you want to proceed (Yes/No): "))
    # checking whether user want to continue or not
    if d.upper()=='YES':
        namedis=str(input("Enter name of distributor: "))
        namelap= str(input("Enter name of laptop: "))
        # checking whether demanded laptop is available or not
        if namelap.upper() not in laptopall:
            print("Not available")
            companies()
        else:
            laptop = laptopall[namelap.upper()]
        checked = False
        # checking condition is true or false
        while checked == False:
            # try block to catch invalid input like string
            try:
                quantity  = int(input("Enter number of laptop to purchase: "))
                # if quantity less than 0 print error message
                if (quantity <=0):
                    print("Enter valid amount")
                    checked = False
```

```
            else:
                checked = True
        except :
            print("Please enter valid value")

    total_without_VAT=laptop['price']*quantity
    laptop['quantity'] +=quantity
    Vat = (0.13*total_without_VAT)
    grand_total=Vat+total_without_VAT
elif d.upper()=='NO':
    options()
else :
    print("Please enter appropriate choice:")

# creating a invoice
invoice = f'''
*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*



*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
Date: {datetime.datetime.now().strftime('%Y-%m-%d %H:%M:%S')}
Retailer name: {namedis.upper()}
```

```
    Product: {namelap.upper()}
    Quantity: {quantity}
    Per price: {laptop['price']:.2f}
_____________________________________________________________________________
    Total without VAT: {total_without_VAT:.2f}
    Total: {grand_total}
_____________________________________________________________________________
    Thank you for doing business with us!
_____________________________________________________________________________
    '''
    #  setting invoice to txt file
    with open('retail.txt','w') as f:
        f.write(invoice)
        print("Your bill has been generated")
        #  updating value in txt file
    with open('laptop.txt', 'w') as f:
        for namelap, details in laptopall.items():
            name=namelap
            brand = details ['brand']
            price = details ['price']
            quantity=details ['quantity']
            processor=details ['processor']
            graphics=details ['graphics']
```

```
            f.write(f'{name}, {brand}, ${price}, {quantity}, {processor}, {graphics}\n')



def customer():
    print("List of Available Laptops: ")
    laptopall = available()
    d= str(input("Do yo want to proceed (Yes/No): "))
    if d.upper()=='YES':
        name = str(input("Enter your name: "))
        namelap = input("Enter the name of laptop: ")
        # checking whether demanded laptop is available or not
        if namelap.upper() not in laptopall:
            print("Laptop not available")
            return customer()
        laptop_upper=laptopall[namelap.upper()]
        checked = False
        while checked == False:
            try:
                quantity  = int(input("Enter number of laptop to purchase: "))
                if (quantity <=0):
                    print("Enter valid amount")
                    checked = False
                else:
```

```
                checked = True
            except ValueError:
                print("Please enter valid value")
            if laptop_upper['quantity']<quantity:
                print("Not enough stock")
                return customer()
        # laptop_upper['quantity']-=quantity
        price_without_shipping =laptop_upper['price']*quantity
        s = str(input("Do you want it to be shipped?"))
        if(s.upper()=='YES'):
            d = str(input("Enter your location: "))
            shipping = 50
        else:
            shipping =0
            d="none"
        total =price_without_shipping+shipping
    elif d.upper()=='NO':
        return options()


    invoice =f"'
     *-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
     *                  Itahari Laptop Shop                    *
     *                  Itahari-20, Sunsari                    *
```

```
*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*

Date: {datetime.datetime.now().strftime('%Y-%m-%d %H:%M:%S')}

Customer name: {name.upper()}

Shipping Location: {d.upper()}

______________________________________________________________

Product: {namelap.upper()}

Quantity: {quantity}

Per price: {laptop_upper['price']:.2f}

______________________________________________________________

Shipping cost = {shipping}

Total: {total}

______________________________________________________________

Thank you for doing business with us!

______________________________________________________________

'''


with open ('sell_laptop.txt', 'w') as f:

    f.write(invoice)

    print("Your bill has been generated")

laptop_upper['quantity']-= quantity


with open('laptop.txt','w') as f:

    for namelap, details in laptopall.items():
```

```
            name = namelap
            brand = details['brand']
            price = details['price']
            quantity = details['quantity']
            processor = details['processor']
            graphics = details['graphics']
            f.write(f"{name}, {brand}, ${price}, {quantity}, {processor}, {graphics}\n")



print("-------------------------------------------------------")
print("--------------------WELCOME TO OUR SHOP--------------------")
print("-------------------------------------------------------")
def details():
    print("A: Available details")
    print("B: Place Order from Manufacturer")
    print("C: Buy a laptop: ")
    print("D: Exit")
    print("Choose desired option:")
def options():
    option= str(input())
    if option.upper()=='A':
        available()
    elif option.upper()=='B':
```

```
        companies()
    elif option.upper()=='C':
        customer()
    elif option.upper()=='D':
        print("Thank you for shopping with us.")
        sys.exit()
    else:
        print("Enter valid option")
    options()
details()
options()
```

Originality report

COURSE NAME

11 Fundamental of Computing Plagiarism Checker

STUDENT NAME

12 KIRAN RAI

FILE NAME

13 22072136 Kiran rai

REPORT CREATED

14 May 12, 2023

Summary

Flagged passages	0	0%
Cited/quoted passages	1	0.4%

Web matches

geekedu.org	1	0.4%

1 passage

Student passage CITED

…data types include Integers, Strings, Boolean and Float whereas **non-primitive data types include Tuples, Dictionary**, Array, List, **sets and Files**.

Top web match

Non-primitive data types include List, Array, **Tuples, Dictionary, Sets, and Files**, while primitive data types include Integers, Float, Strings, and Boolean.

Data Structures in Python: Everything you need to know! - Geekedu
https://www.geekedu.org/blogs/data- structures-in-python-everything-you-need-to-know

---