



Fundamentals of Computing

60% of Individual Coursework

2022-2023 Autumn

Student Name: Dikshya Sharma

London Met ID: 22067520

College ID:NP01CP4A220029

Assignment Due Date: Friday, May 12, 2023

Assignment Submission Date: Friday, May 12, 2023

Word Count: 9002

Project File Links:

Google Drive Link:	https://drive.google.com/drive/folders/1gm_Rw2Dd1NLTRxjTctdogc0lpQmIUl6p?usp=share_link
--------------------	---

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.

ACKNOWLEDGEMENT

I would like to take this opportunity to express my sincere gratitude and appreciation to everyone who helped in completion of this coursework. First and foremost, I am so grateful for my module leader Mr.Hrishav Tandukar, my tutor Miss.Pranchee Singh, and all my respected teachers including Mr.Navraj Kafle and Mr.Aayush Pandey, for their continuous guidance and all helpful advice and assistance they gave while working on the coursework. Their expertise and feedback and constant encouragement were immensely beneficial. In order for me to complete this coursework on time, my lecturers and tutors were of the greatest possible help. Whether it was in person or online through google hangouts, they were always willing to assist.

Furthermore, I am thankful to the university and all the members of Islington College for assigning this coursework. I can undoubtedly say that the abilities and knowledge that I learned from this coursework will be beneficial to me in the days to come.

Finally, I extend my gratitude to all my friends for helping me overcome the obstacles with their skills. I greatly appreciate it. Thank you all so much.

ABSTRACT

This report provides a documentation of a program developed for the Fundamental of Computing module's coursework using IDLE (Integrated Development and Learning Environment). The report is structured into a different section, each of which satisfies the standards and objectives of the coursework. An outline of the coursework is given in the report's introduction which includes a description of the program's key features, the purpose of the program and how it was developed. The report then delves into goals and objectives of coursework and then various tools that were used to complete this coursework along with their role in the development process. It then includes algorithm and pseudocode for the program, to understand their real-world applications. The flowchart in this program plays vital role to visualize the program's logic and understand its practical uses. Furthermore, it explains about data structures. It also covers the testing process, including the different testing methods that were employed to detect and address any error. To complement the written descriptions of the program the report includes several screenshots of the program which helps reader to visualize the program and how they function. Finally, the report concludes with a summary of the project, its objectives and outcomes which moreover includes list of references and citations used throughout the report.

Table of Contents

INTRODUCTION	1
1.1. About the Project	1
1.2. GOALS AND OBJECTIVES	2
1.3. TOOLS USED FOR COURSEWORK	3
1.3.1. Python IDLE	3
1.3.2. Microsoft Word	3
1.3.3. Draw.io	4
2. DISCUSSION AND ANALYSIS	5
2.1. Algorithms	5
2.2. Flowchart	7
2.3. Pseudo Code	11
2.3.1. main module	11
2.3.2. operations module	13
2.3.3. read module	22
2.3.4. write module	25
3. Data Structures	32
3.1. Lists	33
3.2. Dictionary	34
4. PROGRAM	36
4.1 Implementation of Program	36
4.2. Purchase And Sale of Laptop	37
4.2.1. Purchase of Laptop	37
4.2.2 Sell to Customers	42
4.3. Termination of a Program	47
5. TESTING	48
5.1. TEST 1: Implementation of Try Except	48
5.2. TEST 2: Selection Purchase and Sale of Laptop	51
5.3. Test 3 : To Generate a File for Purchase of Laptop	54
5.4. Test 4 : To Generate a File for sale of Laptop	58
5.5. Test 3 : To Update the Stock of Laptop	62

CONCLUSION	67
References	68
APPENDIX.....	69

Table of Figures

Figure 1:Python IDLE logo	3
Figure 2: Microsoft Word logo	3
Figure 3:draw.io logo	4
Figure 4: Main Flowchart.....	8
Figure 5:flowchart of purchase of laptop	9
Figure 6:flowchart for Selling of laptop	10
Figure 7:Data Structure	32
Figure 8:Implementation of List	33
Figure 9: Screenshot Implementation of List 1.1	33
Figure 10:Screenshot of Implementation of dictionary	35
Figure 11:Screenshot of Implementation of Dictionary 1.1	35
Figure 12:Screenshot of Implementation of Program; Welcome Message	36
Figure 13:Screenshot of when user inputs option 1, and inputs name and number	37
Figure 14:Screenshot of id and quantity input	37
Figure 15:screenshot of program if user wants to buy more	38
Figure 16:Screenshot of Printed Invoice when user buys more.....	38
Figure 17:screenshot of Generated invoice when user buys more	39
Figure 18:Screenshot of Printed Invoice when user only buys one product at a time ...	40
Figure 19:Screenshot of Generated invoice when user buys only one product at a time	40
Figure 20:Screenshot of Program iteration.....	41
Figure 21:Screenshot of program When user inputs 2 from option and inputs name and number	42
Figure 22:screenshot of valid ID and quantity input.....	42
Figure 23: screenshot of program when customer wants to buy more	43
Figure 24: Shipping Details Input	43
Figure 25:screenshot of Printed invoice with shipping details.....	44
Figure 26:screenshot of Generated Invoice with shipping details	44
Figure 27:screenshot of Printed invoice when no shipping required	45

Figure 28:Screenshot of Generated invoice when no shipping required.....	45
Figure 29:Screenshot of Iteration of a program	46
Figure 30:Screenshot of Termination of Program.....	47
Figure 31:Screenshot of Welcome Message.....	49
Figure 32:iScreemshot of nvalid option input	49
Figure 33:Screenshot of valid option input	49
Figure 34:Screenshot of :Invalid Phone Number Input	50
Figure 35:Screenshot of Negative Input.....	52
Figure 36:Screenshot of Message displayed on invalid input	52
Figure 37:Screenshot of Non-existing input	53
Figure 38:Screenshot of Message displayed on non-existing input	53
Figure 39:Screenshot of name and phone input.....	55
Figure 40:screenshot of system when user wants to buy more	55
Figure 41:Screenshot of New id and quantity input	56
Figure 42:Screenshot of Printed Bill of more than one product	56
Figure 43:Screenshot of Generated Bill of more than one product	57
Figure 44:Screenshot of a system when user inputs option 2	59
Figure 45:Screenshot of name and phone input.....	59
Figure 46:Screenshot of program when customers wants to buy more	60
Figure 47: Screenshot of Shipping Details	60
Figure 48:Screenshot of printed bill when customers buy more than one laptop	61
Figure 49:Screenshot of Generated Bill when customers buy more than one laptop....	61
Figure 50:Screenshot of name and number input	63
Figure 51:Screenshot of Id and quantity input	63
Figure 52:Screenshot of Printed invoice.....	64
Figure 53:Screenshot of Updated stock shown on system.....	64
Figure 54:Screenshot of Updated stock on text file	64
Figure 55:Screenshot When customer buys from store.....	65
Figure 56:Screenshot of Printed Invoice	65
Figure 57:Screenshot of Updated stock shown on system	66
Figure 58:Screenshot of Updated stock on text file	66

Table of Tables

Table 1: Symbols of flowchart	7
Table 2: Test to check if system runs on invalid input	48
Table 3: Test to check if system accepts negative values or not existing values	51
Table 4: Test to generate a file for purchase of laptop	54
Table 5: To test to generate file for sale of laptop	58
Table 6: Test to update stock of laptop	62

INTRODUCTION

Python is a high-level programming language which is interactive, interpreted language (gumster). It is a readable, dynamic, pleasant, flexible, fast, and powerful language which supports multiple programming paradigms such as object-oriented, functional, and procedural. Python is a popular choice for general-purpose programming, Scientific computing, data science, artificial intelligence, web development and more. It is known for its simplicity and readability, as its syntax is easy to understand and code blocks are separated using indentation rather than brackets which makes python a great language for beginners to learn (vegibit).

Additionally, Python has large and comprehensive standard library which includes so many built-in modules for many common tasks and reduce the need for external libraries. Being a cross-Platform language, it can run on variety of operating system including Windows, Linus, macOS.

1.1. About the Project

The following coursework involves developing a program for a laptop rental shop that keeps the track of laptops in a text file. The system is designed to work as a laptop store that purchases laptops from manufacturers and then sells them to customers. This program keeps track of available laptop's Information in a text file. It reads the text file to display every laptop that is currently available and modifies it based on the type of transaction i.e., purchasing from manufacturer or selling it to the customer. For instance, when a laptop rental shop orders laptops from manufacturer, the stock of the respective laptop increases and when a customer purchases a laptop, the stock of that particular laptop reduces. For each order or sale that is carried out, a unique invoice with the details of the transaction is generated. Whenever user wants to buy more, the updated invoice is generated. Moreover, the program is designed to be user-friendly, with easy-to-use interfaces and clear instruction.

1.2. GOALS AND OBJECTIVES

This project aims to develop a system for a laptop rental shop for making it easier to handle the information regarding the purchase of laptop. It seeks to provide accurate and up-to-date Information on the availability of laptops, to improve the efficiency of the laptop rental shop and to enhance customer satisfaction by providing user-friendly interface and reliable service.

The goal and objectives of this coursework are given below:

- To apply a modular and function-based programming approach.
- To use Python's file handling functionality to efficiently create, write, add, and handle files.
- To utilize data structures, such as dictionaries and lists to store and manage information related to laptops in the system.
- To ensure the program's effectiveness by incorporating Try-except blocks to manage any potential errors that might occur during execution.
- To use looping and control-flow statements correctly in the program logic.
- To develop algorithm and pseudocode for the program, to understand their real-world applications.
- To create a flowchart to visualize the program's logic and understand its practical uses.
- To design a detailed testing plan that thoroughly evaluates the program's functionality and verifies that it meets the specified requirements.

1.3. TOOLS USED FOR COURSEWORK

1.3.1. Python IDLE

IDLE which stands FOR Integrated Development and Learning Environment is a quite simple and sophisticated IDE which offers features like multi-window text editor, syntax highlighting, code autocompletion intelligent indenting and so many more (Sharma, 2020). In this coursework, IDLE was used as IDE for writing Python program as it is a very beginner friendly editor.



Figure 1: Python IDLE logo

1.3.2. Microsoft Word

Microsoft Word simply called as MS-Word is a word-processor produced by Microsoft which lets user to create professional-quality documents, reports, letters, resume etc (hope, 2021). Since it provides an easy to use and familiar interface for creating, editing and formatting text documents, MS-word was used to write a report for this coursework.



Figure 2: Microsoft Word logo

1.3.3. Draw.io

Draw.io is an online diagramming tool to create flowcharts, diagrams, mind maps, organisation charts, and much more (Paraschiv, 2023). Draw.io was used in creating flowcharts FOR this coursework to provide a better understanding of project's structure.

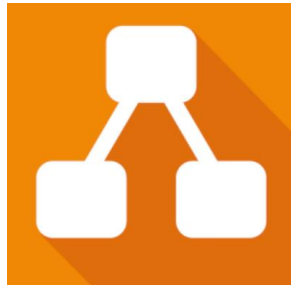


Figure 3:draw.io logo

2. DISCUSSION AND ANALYSIS

2.1. Algorithms

Algorithm is simply a set of steps or a finite rule and instructions to be followed for problem-solving operations (rishabPrabhu). Algorithms are necessary for solving complex problems efficiently and effectively. It is a blueprint to write a program. Algorithms are written in steps and can be further described using flowcharts and pseudocode. To understand the problem in real life-based scenario algorithm for this project is below.

Step 1: Start

Step 2: Call function welcome_message

Step 3: Display options to the user

Step 4: Take user input for options

Step 5: If user input is equal to 1 then call function buy_from_manufacturer

Step 6: Take user input for name and check IF it is valid

Step 7: If valid, take input for phone number

Step 8: Else display error message and go to step 6

Step 9: Check If phone number is valid

Step 10: If valid go to step 12

Step 11: If not valid display error message and go to step 7

Step 12: display all the stocks in tabular form

Step 13: Take user input for valid id

Step 14: Check If id is valid

Step 15: If valid go to step 17

Step 16: If not valid display error message and go to step 13

Step 17: Take user input for quantity

Step 18: Check If quantity is valid,

Step 19: If valid go to step 20

Step 20: If not valid display error message go to step 17

Step 21: Ask If the user want to buy more

Step 22: If yes, go to step 13

Step 23: If no, update the laptopDetails.txt file

Step 24: Display the invoice

Step 25: Go to step 3

Step 26: Go to step 4

Step 27: If user input is equal to 2 then call function sell_from_store

Step 28: Go to step 6 to 21

Step 29: If buy more is equal to no, go to step 29

Step 30: Ask If the user want laptop to get shipped

Step 31: If yes, display shipping charge details

Step 32: If no go to step 23

Step 33: Ask for shipping location

Step 34: Display invoice

Step 35: Go to step 3

Step 36: Go to step 4

Step 37: If user input is equals to 3

Step 38: Call function exit_message

Step 39: End

2.2. Flowchart

A Flowchart is a graphical representation of a algorithm that illustrates the steps, sequences and decisions of a process or workflow (Asana, 2023). It aids in visualizing the structure of program and understanding the logic behind it. A flowchart contains several forms, each of which has a unique symbol.


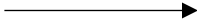


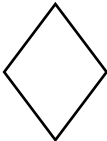
Symbol	Name	Function
	Start/End	An oval represents a start or end point.
	Arrows	A line is a connector that shows a relationship between the representative shapes.
	Input/Output	A parallelogram represents input or output.
	Process	A rectangle represents a process.
	Decision	A diamond indicates a decision.

Table 1: Symbols of flowchart

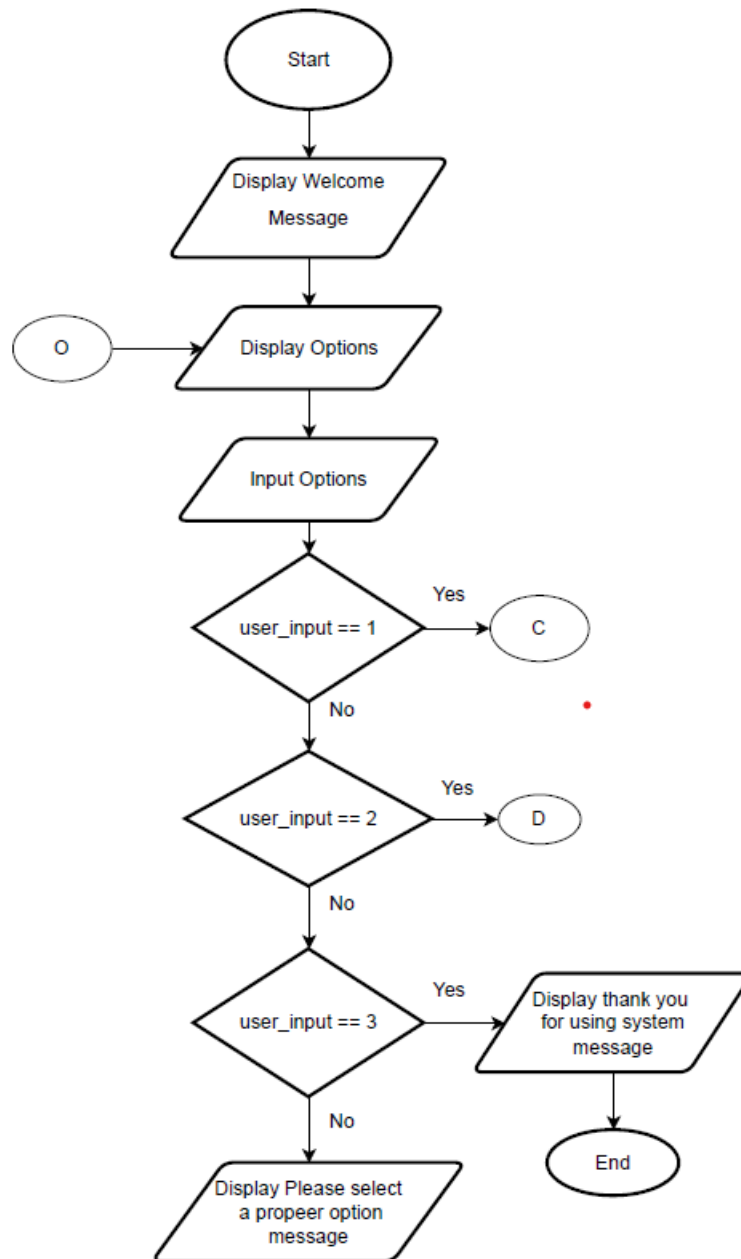


Figure 4: Main Flowchart

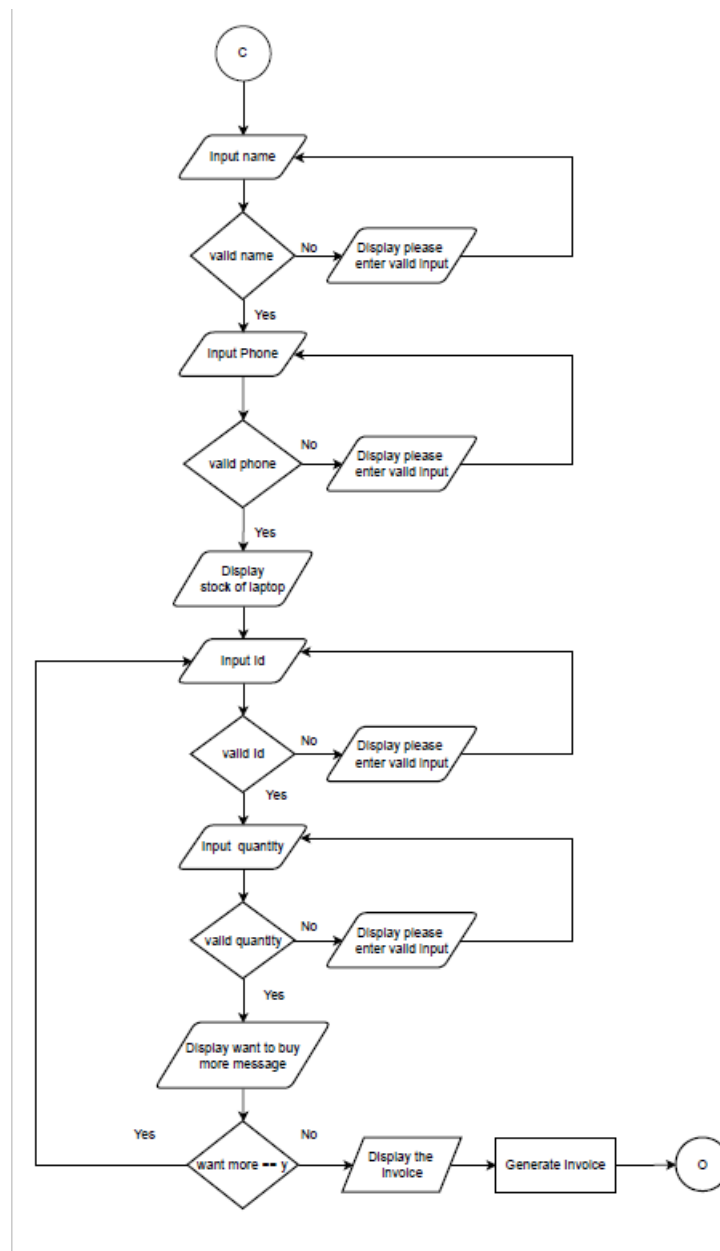


Figure 5:flowchart of purchase of laptop

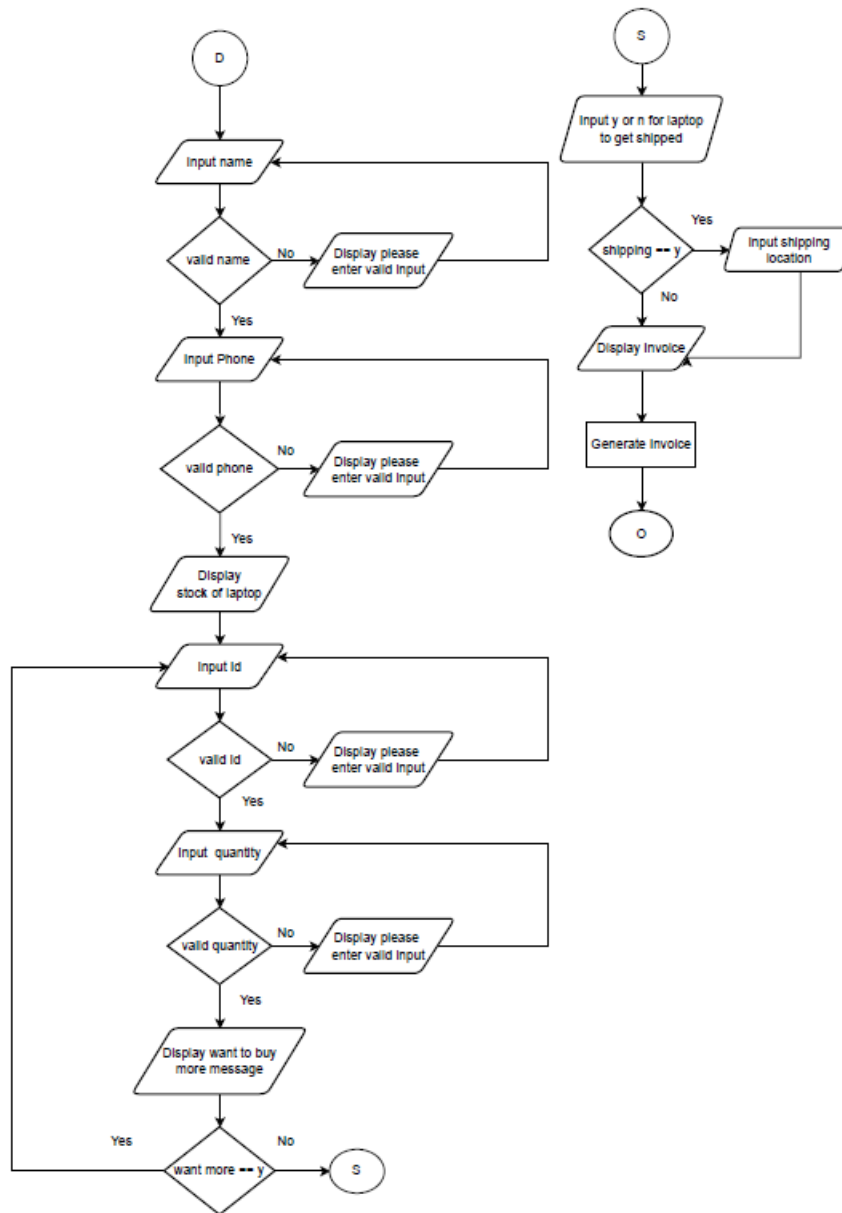


Figure 6:flowchart for Selling of laptop

2.3. Pseudo Code

Pseudo code are the false code which doesn't have any syntax like programming language and can be easily understood by everyone (the programmed words). Pseudo codes are implemented to improve the readability of a program. As this coursework was of module approach, below is the pseudo code of the program we created as according to module.

2.3.1. main module

algorithm mainModule

FROM operations **IMPORT** display_welcome, buy_FROM_manufacturer,
sell_FROM_store, exit_message

FROM write **IMPORT** invoiceBuy, invoiceSell

CALL display_welcome()

DEFINE userOptions():

 continueLoop = True

WHILE continueLoop == True:

DISPLAY("\\t \\t 1. Press 1 to buy FROM manufacturer.")

DISPLAY("\\t \\t 2. Press 2 to sell to customers.")

DISPLAY("\\t \\t 3. Press 3 to exit FROM system.")

DISPLAY("\\n")

 options = True

WHILE options == True:

TRY:

 optionsInput = int(input("\\t Enter FROM option:"))

DISPLAY("\\n")

 options = False

EXCEPT:

DISPLAY("\\n")

DISPLAY("\\t \\t Uh-Oh!The option you provided doesn't exist.Please TRY again.")

DISPLAY("\\n")

IF optionsInput == 1:

CALL validLaptopId, laptopQuantity, userName, customersDictionary = buy_FROM_manufacturer()

CALL invoiceBuy(validLaptopId, laptopQuantity,userName, customersDictionary)

ELIF optionsInput == 2:

CALL validLaptopId, laptopQuantity, userName, userPhone, location, shippingCharge, customersDictionary = sell_FROM_store()

CALL invoiceSell(validLaptopId, laptopQuantity,userName, userPhone, location, shippingCharge, customersDictionary)

ELIF optionsInput == 3:

continueLoop = False

CALL exit_message()

ELSE:

DISPLAY("\\t \\t To continue, Please select a proper option.")

DISPLAY("\\n")

END IF

END WHILE

END WHILE

CALL userOptions()

END main module

2.3.2. operations module

algorithm operations module

FROM datetime **IMPORT** datetime

FROM write **IMPORT** writeTextFile

FROM read **IMPORT** readFROMTextFile, openFile, valid_name, valid_phone, valid_id,
quantity

laptopDetailsDictionary = {}

DEFINE display_welcome():

DISPLAY("\\n")

DISPLAY("\\t\\t", " " "*"50)

DISPLAY("\\t \\t \\t \\t \\t \\t Welcome to Bit & Byte Store ")

DISPLAY("\\n")

DISPLAY("\\t\\t", " " "*"50)

DISPLAY("\\t \\t \\t \\t \\t \\t London, England")

DISPLAY("\\n")

DISPLAY("\\t\\t", " " "*"50)

DISPLAY("\\n")

DEFINE buy_FROM_manufacturer():

 customersDictionary = []

 loop = True

CALL laptopDetailsDictionary = openFile()

DISPLAY("-*130)

DISPLAY("\\t\\t\\t\\t\\t\\t Let's shop.")

DISPLAY("-*130)

DISPLAY("\\n")

DISPLAY("\\t\\t\\t\\t We will need your name and contact number FOR the invoice.")

```
DISPLAY("\\n")
CALL userName = valid_name()
CALL userPhone = valid_phone()
DISPLAY("\\n")
DISPLAY("-*130)
DISPLAY("S.N \\t Name \\t\\t\\t Brand \\t\\t\\t Price \\t  Quantity \\t\\t Processor \\t\\t Graphic
Card")
DISPLAY("-*130)
CALL readFROMTextFile()
DISPLAY("\\n")

WHILE loop == True:
    CALL validLaptopId = valid_id()
    CALL laptopQuantity = quantity()
    requiredQuantity = laptopDetailsDictionary[validLaptopId][3]

    WHILE laptopQuantity <= 0:
        DISPLAY("\\t\\t\\t " + "That doesn't seem a valid quantity. Please provide valid
quantity!")
        DISPLAY("\\n")
        laptopQuantity = int(input("Please provide the number of quantity of laptop: "))
        DISPLAY("\\n")
    DISPLAY("\\n")
END WHILE

laptopDetailsDictionary[validLaptopId][3] =
int(laptopDetailsDictionary[validLaptopId][3]) + int(laptopQuantity)
writeTextFile(laptopDetailsDictionary)
nameOfProduct = laptopDetailsDictionary[validLaptopId][0]
```

```
totalQuantity = laptopQuantity  
unitPrice = (laptopDetailsDictionary[validLaptopId][2])  
price = laptopDetailsDictionary[validLaptopId][2].replace("$", "")  
totalPrice = int(price) * int(totalQuantity)  
dateandtime = datetime.now()  
datetime_ms = int(datetime.timestamp(datetime.now())) * 1000  
brandName = laptopDetailsDictionary[validLaptopId][1]  
  
customersDictionary.append([nameOfProduct, brandName, totalQuantity,  
unitPrice, price, totalPrice])  
  
buy = input("Do you want to buy more? Enter y to buy more:".lower())  
DISPLAY("\\n")  
DISPLAY("\\n")  
IF buy == "y":  
    loop = True  
ELSE:  
    loop = False  
END IF  
  
END WHILE  
  
DISPLAY("\\n")  
DISPLAY("-*130")  
DISPLAY("Bit & Byte Store", "\\t\\t\\t\\t\\t\\t\\t\\t\\t\\t \" Date: ",dateandtime.strftime("%d\" \" "  
"%b\"", "%Y"))  
DISPLAY("London,England", "\\t\\t\\t\\t\\t\\t\\t\\t\\t\\t \" \" Time:",dateandtime.strftime("%I"  
":\" \"%M\" \"%p"))  
DISPLAY("bitandbyte@gmail.com")  
DISPLAY("073599487")
```

```

DISPLAY("\\n")
DISPLAY("-*130)
DISPLAY("Bill Number:" + str(datetime_ms))
DISPLAY("-*130)
DISPLAY("S.N" + "\\t\\t" + "Product's Name" + "\\t\\t" + "Brand" + "\\t\\t " + "Quantity" + "\\t\\t
" + "Unit Price" + "\\t\\t\\t" + "Amount")
DISPLAY("-*130)
counter = 1
total = 0
FOR key in customersDictionary:
    DISPLAY(str(counter) + "\\t\\t" + str(key[0]) + "\\t\\t " + str(key[1]) + "\\t\\t" + str(key[2])
+ "\\t\\t\\t" + str(key[3]) + "\\t\\t\\t\\t" + "$" + str(key[5]))
    counter = counter+1
    total = total + int(key[5])
DISPLAY("\\n")
DISPLAY("\\n")
DISPLAY("-*130)
DISPLAY("Net Amount:" + "\\t\\t\\t\\t\\t\\t\\t\\t\\t\\t\\t\\t\\t\\t\\t\\t", "$" + str(total))
vatAmount = 0.13 * total
DISPLAY("Vat Amount:" + "\\t\\t\\t\\t\\t\\t\\t\\t\\t\\t\\t\\t\\t " , "$" + str(vatAmount))
DISPLAY("-*130)
DISPLAY("Gross Amount:" + "\\t\\t\\t\\t\\t\\t\\t\\t\\t\\t\\t\\t\\t " , "$" + str(total + vatAmount))
DISPLAY("-*130)
DISPLAY("\\n")
DISPLAY("Please note that 13% VAT is added on your net amount.")
DISPLAY("\\n")
DISPLAY("Terms & Conditions" "\\t\\t\\t\\t\\t\\t\\t\\t\\t\\t" + "Bank Details")
DISPLAY("Payment is due within 15 days." "\\t\\t\\t\\t\\t\\t\\t\\t " " " "Bank of United
Kingdom")

```



```
DISPLAY("\t\t\t\t\t\t\t\t\t\t" + "Account Number:0530 1178 7908")
```

DISPLAY("\n")

```
DISPLAY("\t\t\t\t\t" + "Thank You FOR your order.")
```

DISPLAY("-"*130)**DISPLAY("\n")**

RETURN validLaptopId, laptopQuantity, userName, customersDictionary

DEFINE sell_FROM_store():

```
customersDictionary = []
```

```
loop = True
```

```
laptopDetailsDictionary = openFile()
```

DISPLAY("-"*130)

```
DISPLAY("\t \t \t \t \t Let's shop!")
```

DISPLAY("-"*130)**DISPLAY("\n")**

```
DISPLAY("\\t \\t \\t \\t We will need your name and contact number FOR the invoice.")
```

DISPLAY("\n")

CALL userName = valid_name()

CALL userPhone = valid_phone()

DISPLAY("\n")**DISPLAY("-"*130)**

```
DISPLAY("S.N \t Name \t\t\t Brand \t\t\t Price \t Quantity \t\t Processor \t\t Graphic  
Card")
```

DISPLAY("-"*130)

CALL readFROMTextFile()

DISPLAY("\n")

WHILE loop == True:

CALL validLaptopId = valid_id()

```
CALL laptopQuantity = quantity()

DISPLAY("\\n")

requiredQuantity = laptopDetailsDictionary[validLaptopId][3]

WHILE laptopQuantity <= 0 or laptopQuantity > int(requiredQuantity):

    DISPLAY("\\t\\t Dear" + " " + str(userName) + " the quantity you are looking
FOR is not on stock!!!")

    DISPLAY("\\n")

    laptopQuantity = int(input("Please provide the number of quantity of laptop: "))

DISPLAY("\\n")

END WHILE

laptopDetailsDictionary[validLaptopId][3] =
int(laptopDetailsDictionary[validLaptopId][3]) - int(laptopQuantity)

writeTextFile(laptopDetailsDictionary)

nameOfProduct = laptopDetailsDictionary[validLaptopId][0]
totalQuantity = laptopQuantity
unitPrice = laptopDetailsDictionary[validLaptopId][2]
price = laptopDetailsDictionary[validLaptopId][2].replace("$", "")
totalPrice = int(price) * int(totalQuantity)
brandName = laptopDetailsDictionary[validLaptopId][1]
dateandtime = datetime.now()
datetime_ms = int(datetime.timestamp(datetime.now())) * 1000
customersDictionary.append([nameOfProduct, brandName, totalQuantity,
unitPrice, price, totalPrice])

buy = input("Do you want to buy more?Enter y to buy more:").lower()

IF buy == "y":
```

```
        loop = True
    ELSE:
        loop = False
    END IF
DISPLAY("\n")
END WHILE
shipping = (input("Do you want your laptop to be shipped?(y/n):"))

IF shipping == "y".lower():
    DISPLAY("\n")
    DISPLAY("\t\t\t Shipping charge varies inside and outside city.")
    DISPLAY("\n")
    shippingLocation = (input("Where do want your laptop to be delivered??(i/o):"))

    IF shippingLocation == "i".lower():
        shippingCharge = 20
    ELIF shippingLocation == "o".lower():
        shippingCharge = 40
        DISPLAY("\n")
    DISPLAY("\n")
    END IF
    location = input("Please provide us your full address FOR the delivery:")

ELSE:
    DISPLAY("Thank You FOR your order.")
    shippingCharge = 0
    location = None
END IF
```

DISPLAY("\n")**DISPLAY("-"*130)**

```
DISPLAY("Bit & Byte Store" "\t\t\t\t" " + "Bill To" "\t\t\t\t\t\t\t\t" "Date: "
+str(dateandtime.strftime("%d" " " "%b", "%Y")))
```

```
DISPLAY("London,England" + "\t\t\t\t " + "Name:" + str(userName) + "\t\t\t\t\t " +  
"Time: " +str(dateandtime.strftime("%I" ":" "%M" "%p")))
```

```
DISPLAY("bitandbyte@gmail.com" + "\t\t\t\t " + "Location:" + str(location))
```

```
DISPLAY("073599487" + "\t\t\t\t\t " + "Contact:" + str(userPhone))
```

DISPLAY("\n")**DISPLAY("-"*130)**

```
DISPLAY("Bill Number: " + str(datetime_ms))
```

DISPLAY("-"*130)

```
DISPLAY("S.N" + "\t\t" + "Product's Name" + "\t\t" + "Brand" + "\t\t" + "Quantity" +
"\t\t" + "Unit Price" + "\t\t" + "Amount")
```

DISPLAY("-"*130)

```
counter = 1
```

total = 0

FOR key in customersDictionary:

```
DISPLAY(str(counter) + "\t\t" + str(key[0]) + "\t\t" + str(key[1]) + "\t\t" + str(key[2])
+ "\t\t" + str(key[3]) + "\t\t" + "$" + str(key[5]))
```

```
counter = counter+1
```

```
total = total+int(key[5])
```

DISPLAY("\n")**DISPLAY("\n")****DISPLAY("-"*130)**[illegible][illegible]**DISPLAY("-"*130)**[illegible]

DISPLAY("-"*130)**DISPLAY("\n")**[illegible][illegible]

```
DISPLAY("\t\t\t\t\t\t\t\t\t\t" + "Account Number:0530 1178 7908")
```

DISPLAY("\n")

```
DISPLAY("\t\t\t\t\t" + "Thank You FOR your order.")
```

DISPLAY("-"*130)**DISPLAY("\n")**

RETURN validLaptopId, laptopQuantity, userName, userPhone, location, shippingCharge, customersDictionary

DEFINE exit_message():

```
DISPLAY("Thank You FOR using our system.Have a good day!.")
```

DISPLAY("\n")**DISPLAY("\n")**

END operations module

2.3.3. read module**algorithm** read module**DEFINE** openFile():

```
laptopDetailsDictionary = {}  
file = open("laptopDetails.txt", "r")  
laptopId = 1  
FOR line in file:  
    line = line.replace("\n", "")  
    laptopDetailsDictionary[laptopId] = line.split(",")  
    laptopId += 1  
RETURN laptopDetailsDictionary
```

DEFINE readFROMTextFile():

```
a = 1  
file = open("laptopDetails.txt", "r")  
FOR line in file:  
    DISPLAY(a, "\t "+line.replace(",", "\t\t"))  
    a += 1  
file.close()
```

DEFINE valid_name():

```
WHILE True:  
    userName = input("Please provide us your name:")  
    TRY:  
        int(userName)  
    EXCEPT ValueError:
```

```
    RETURN userName
    DISPLAY("\n")
    DISPLAY("\t\t\t Please provide valid name.")
    DISPLAY("\n")
END WHILE
```

```
DEFINE valid_phone():
    WHILE True:
        DISPLAY("\n")
        userPhone = input("Your phone number here:")
        TRY:
            int(userPhone)
            RETURN userPhone
        EXCEPT ValueError:
            DISPLAY("\n")
            DISPLAY("\t\t\t Invalid Phone Number. Please provide valid phone numbers
only.")
            DISPLAY("\n")
    END WHILE
```

```
DEFINE valid_id():
    WHILE True:
        TRY:
            DISPLAY("\n")
            validLaptopId = int(input("Please Provide the ID of the laptop you want to buy:"))
            IF validLaptopId <= 0 or validLaptopId > len(openFile()):
                DISPLAY("\n")
```

```
        DISPLAY("\t\t\t Please provide a valid laptop id!!!")
        DISPLAY("\n")
    ELSE:
        RETURN validLaptopId
    END IF
    DISPLAY("\n")
EXCEPT ValueError:
    DISPLAY("\n")
    DISPLAY("\t\t\t Uh-Oh! It doesn't seem a valid ID. Please check the details
above and continue.")
    DISPLAY("\n")
END WHILE

DEFINE quantity():
    WHILE True:
        TRY:
            DISPLAY("\n")
            laptopQuantity = int(input("Please provide the quantity of a laptop you want to
buy:"))
            DISPLAY("\n")
            RETURN laptopQuantity
        EXCEPT:
            DISPLAY("\n")
            DISPLAY("\t\t\t Please provide valid quantity.")
            DISPLAY("\n")
    END WHILE
END read module
```


2.3.4. write module

algorithm write module

FROM read **IMPORT** openFile

FROM datetime **IMPORT** datetime

DEFINE writeTextFile(laptopDetailsDictionary):

 file = open("laptopDetails.txt", "w")

FOR values in laptopDetailsDictionary.values():

 file.write(str(values[0]) + "," + str(values[1]) + "," + str(values[2]) + "," + str(values[3])
+ "," + str(values[4]) + "," + str(values[5]))

 file.write("\n")

 file.close()

DEFINE invoiceBuy(validLaptopId, laptopQuantity, userName, customersDictionary):

 dateandtime = datetime.now()

CALL laptopDetailsDictionary = openFile()

 nameOfProduct = laptopDetailsDictionary[validLaptopId][0]

 totalQuantity = laptopQuantity

 unitPrice = (laptopDetailsDictionary[validLaptopId][2])

 price = laptopDetailsDictionary[validLaptopId][2].replace("\$", "")

 totalPrice = int(price) * int(totalQuantity)

 dateandtime = datetime.now()

 datetime_ms = int(datetime.timestamp(datetime.now()) * 1000)

 brandName = laptopDetailsDictionary[validLaptopId][1]

 with open(str(userName) + str(datetime_ms) + ".txt", 'w') as buy:

[illegible]

[illegible]

DEFINE invoiceSell(validLaptopId, laptopQuantity, userName, userPhone, location, shippingCharge, customersDictionary):

```
dateandtime = datetime.now()
```

CALL laptopDetailsDictionary = openFile()

```
nameOfProduct = laptopDetailsDictionary[validLaptopId][0]
```

```
totalQuantity = laptopQuantity
```

```
unitPrice = laptopDetailsDictionary[validLaptopId][2]
```

```
price = laptopDetailsDictionary[validLaptopId][2].replace("$", "")
```

```
totalPrice = int(price) * int(totalQuantity)
```

```
brandName = laptopDetailsDictionary[validLaptopId][1]
```

```
datetime = datetime.now()
```



```
sell.write("Terms & Conditions" "\t\t\t\t\t" + "Bank Details")  
sell.write("\n")  
sell.write("Payment is due within 15 days." "\t\t\t\t\t" + "Bank of United Kingdom")  
sell.write("\n")  
sell.write("\t\t\t\t\t\t\t\t\t\t\t" + "Account Number:0530 1178 7908")  
sell.write("\n")  
sell.write("\n")  
sell.write("\t\t\t\t\t\t\t\t\t\t\t" + "Thank You for your order.")  
sell.write("\n")  
sell.write("\n")  
sell.write("-"*98)  
sell.write("\n")
```

END write module

3. Data Structures

Data Structures are a way of organizing and storing data so that they can be accessed efficiently which can further be divided into two types: Primitive and non-primitive data structures (Jaishwal, 2023). For the effective use of data and for the speed of processing, data structures are absolutely essential. Nearly all areas of computer science, including Artificial intelligence, graphics, operating systems, compiler designs, etc., frequently involve data structures. Data Structure handles the Information and renders the data in accordance with user requirements. It is possible to add, delete, or search for elements in each data structure using a unique set of operations and methods.

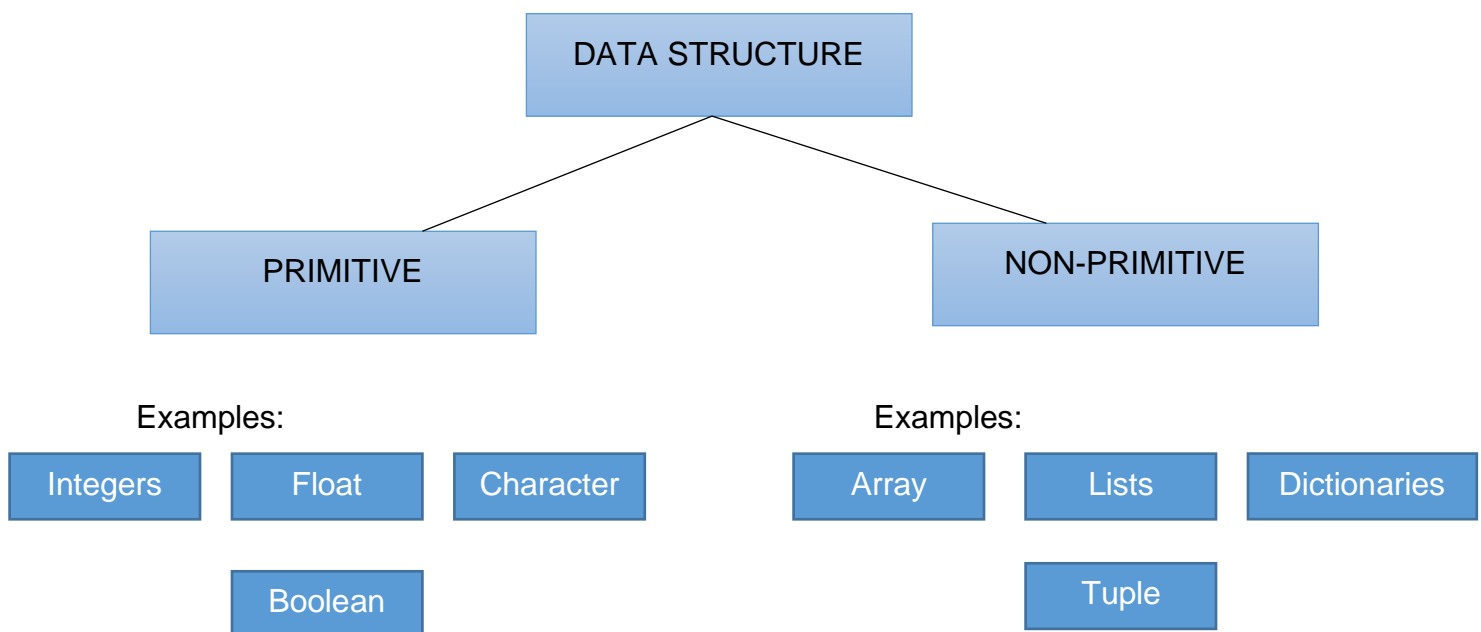


Figure 7: Data Structure

Primitive data types are most common and basic data structures which are building blocks for data manipulation and contains simple values of data (Jaishwal, 2023). The non-primitive data are composed of one or more primitive data types. These are the sophisticated members of the data structure family which not just store a value but rather a collection of values in various Formats (Jaishwal, 2023).

From non-primitive data structures, we have implemented two of them which are Lists and Dictionaries.

3.1. Lists

List is a collection of items which keeps data together that belongs together, condense the code and Perform the same methods and operations on multiple values at once (Tagliaferri, 2021). Lists are mutable, i.e., its content can be changed without changing its identity can contain items of different data types. Each element inside the list is called an item. List is denoted by square bracket [] and every elements inside lists are separated by comma.

Example: Details = ["Beck", 25, True, 1]

In this coursework, data structure list is implemented. A empty customerDictionary List is created and later all the details of product that the customer bought is added on it.

```
# when user wants to buy from manufacturer
def buy_from_manufacturer():
    customersDictionary = []
    loop = True
    laptopDetailsDictionary = openFile()
    print("-"*130)
    print("\t\t\t\t\t Let's shop.")
    print("-"*130)
```

Figure 8: Implementation of List

```
brandName = laptopDetailsDictionary[variablesLaptop][1]

customersDictionary.append([nameOfProduct, brandName, totalQuantity, unitPrice, price, totalPrice])
buy = input("Do you want to buy more? Enter y to buy more:").lower()
print("\n")
print("\n")
if buy == "y":
```

Figure 9: Screenshot Implementation of List 1.1

3.2. Dictionary

Dictionary is a mutable data structure that stores items in key-value pairs where key is the unique identifier FOR an item and a value is the data associated with that key (Simplilearn, 2023). Dictionary is denoted by curly braces { } and each item inside dictionary is separated by comma.

Example:

➔ defining a dictionary

```
my_dictionary = {"name": "Ram",  
                "age" : 15,  
                "grade" : 5  
                }
```

➔ Accessing values in dictionary

```
print(my_dictionary["name"])
```

➔ Adding a new key-value pair

```
my_dictionary["height"] = 4.5
```

In this coursework, dictionary is implemented. An empty dictionary is created and later is used to store the details of laptops read from a file. The key here is validLaptopId and values are the list of details of each laptop. By using this dictionary, system can easily access details of any laptop by its valid ID.

```
from time import time
from read import readFromTextFile, openFile,
from datetime import datetime

laptopDetailsDictionary = {}

# function, welcome message
def display_welcome():
    print("\n")
    print("\t\t", " " * 50)
    print("\t \t \t \t \t \t \t Welcome to I")
    print("\n")
    print("\t\t", " " * 50)
```

Figure 10: Screenshot of Implementation of dictionary

```
def openFile():
    laptopDetailsDictionary = {}
    file = open("laptopDetails.txt", "r")
    laptopId = 1
    for line in file:
        line = line.replace("\n", "")
        laptopDetailsDictionary[laptopId] = line.split(",")
        laptopId += 1
    return laptopDetailsDictionary

def readFromTextFile():
```

Figure 11: Screenshot of Implementation of Dictionary 1.1

4. PROGRAM

4.1 Implementation of Program

The program created for this project is designed to run on loop until user decides to end it. This program is used to buy laptop from manufacturer and sell laptop to the customers. When program is launched, it first displays a welcome message and gives user three options. The option includes buying from manufacturer, selling to the customers and terminating the system. The program asks user to input their choice from the options given. When a user wants to buy from manufacturer, it asks user all the details for the transaction and to generate the bill which is generated as text file and updates the stock according to the purchase. If the user wants to sell laptop to the customer, it asks user to input all the details required for transaction and to generate the invoice. Every time user purchase laptop, it generates a unique bill with updated details. If user urge to buy more. To handle any errors during program execution, a Try-Except block is included which makes program function well.

```
*****
                          Welcome to Bit & Byte Store
*****

*****
                          London, England
*****

1. Press 1 to buy from manufacturer.
2. Press 2 to sell to customers.
3. Press 3 to exit from system.

Enter from option:
```

Figure 12: Screenshot of Implementation of Program; Welcome Message

4.2. Purchase And Sale of Laptop

4.2.1. Purchase of Laptop

Whenever user inputs option 1, the system directs user to purchase laptops from manufacturer. The system works like this:

After providing 1 as an input to the system, the system lets user know that, it needs their name and phone number for printing bill. Hence, the system asks user to provide their name and phone number and displays all the details of laptop.

```

Enter from option:1

-----
Let's shop.
-----

We will need your name and contact number for the invoice.

Please provide us your name:Missy

Your phone number here:071577633

-----
S.N    Name           Brand           Price    Quantity    Processor      Graphic Card
-----
1      Razer Blade    Razer           $2000    20          i7 7th Gen     GTX 3060
2      XPS            Dell            $1976    15          i5 9th Gen     GTX 3070
3      Alienware     Alienware      $1978    24          i5 9th Gen     GTX 3070
4      Swift 7       Acer            $900     12          i5 9th Gen     GTX 3070
5      Macbook Pro 16 Apple          $3500    10          i5 9th Gen     GTX 3070
-----

```

Figure 13:Screenshot of when user inputs option 1, and inputs name and number

Now, the system asks user to input the ID and quantity of a laptop they want to buy.

```

-----
S.N    Name           Brand           Price    Quantity    Processor      Graphic Card
-----
1      Razer Blade    Razer           $2000    20          i7 7th Gen     GTX 3060
2      XPS            Dell            $1976    15          i5 9th Gen     GTX 3070
3      Alienware     Alienware      $1978    24          i5 9th Gen     GTX 3070
4      Swift 7       Acer            $900     12          i5 9th Gen     GTX 3070
5      Macbook Pro 16 Apple          $3500    10          i5 9th Gen     GTX 3070
-----

Please Provide the ID of the laptop you want to buy:3

Please provide the quantity of a laptop you want to buy:5

```

Figure 14:Screenshot of id and quantity input

The system now asks user If they want to buy more. If user says yes the system loops from asking laptops ID to asking if they want to buy more.

```
Do you want to buy more? Enter y to buy more:y
```

```
Please Provide the ID of the laptop you want to buy:2
```

```
Please provide the quantity of a laptop you want to buy:1
```

```
Do you want to buy more? Enter y to buy more:|
```

Figure 15:screenshot of program if user wants to buy more

```
Do you want to buy more? Enter y to buy more:n
```

```
Bit & Byte Store                                     Date: 11 May,2023
London,England                                       Time: 12:53AM
bitandbyte@gmail.com
073599487
```

```
Bill Number:1683745730506
```

S.N	Product's Name	Brand	Quantity	Unit Price	Amount
1	Alienware	Alienware	5	\$1978	\$9890
2	XPS	Dell	1	\$1976	\$1976

```
Net Amount:                                     $11866
Vat Amount:                                     $1542.58000000000002
Gross Amount:                                   $13408.58
```

```
Please note that 13% VAT is added on your net amount.
```

```
Terms & Conditions                               Bank Details
Payment is due within 15 days.                    Bank of United Kingdom
                                                    Account Number:0530 1178 7908
```

```
Thank You for your order.
```

Figure 16:Screenshot of Printed Invoice when user buys more



Figure 17: screenshot of Generated invoice when user buys more

If user does not want to buy more laptop, the system prints the bill with all necessary details. After successful printing the invoice, the system then generates the bill as a text file with unique bill number for every purchase which contains every details of laptop bought by a user.

```

Do you want to buy more? Enter y to buy more:n

-----
Bit & Byte Store                                     Date: 11 May,2023
London,England                                       Time: 03:16PM
bitandbyte@gmail.com
073599487
-----
Bill Number:1683797484242
-----
S.N          Product's Name      Brand      Quantity      Unit Price      Amount
-----
1            Razer Blade         Razer      30            $2000           $60000
-----

Net Amount:                                         $60000
Vat Amount:                                         $7800.0
-----
Gross Amount:                                       $67800.0
-----

Please note that 13% VAT is added on your net amount.

Terms & Conditions                                Bank Details
Payment is due within 15 days.                    Bank of United Kingdom
Account Number:0530 1178 7908

```

Figure 18: Screenshot of Printed Invoice when user only buys one product at a time

```

*max1683797487553.txt - Notepad
File Edit Format View Help
-----
Bit & Byte Store                                     Date: 11 May,2023
London,England                                       Time:03:16PM
bitandbyte@gmail.com
073599487
-----
Bill Number:1683797487553
-----
S.N          Product's Name      Brand      Quantity      Unit Price      Amount
-----
1            Razer Blade         Razer      30            $2000           $60000
-----

Net Amount:                                         $60000
Vat Amount:                                         $7800.0
-----
Gross Amount:                                       $67800.0
-----

Please note that 13% VAT is added on your net amount.

Terms & Conditions                                Bank Details
Payment is due within 15 days.                    Bank of United Kingdom
Account Number:0530 1178 7908

Thank You for your order.
-----

```

Figure 19: Screenshot of Generated invoice when user buys only one product at a time

The system now again iterates and displays the option .

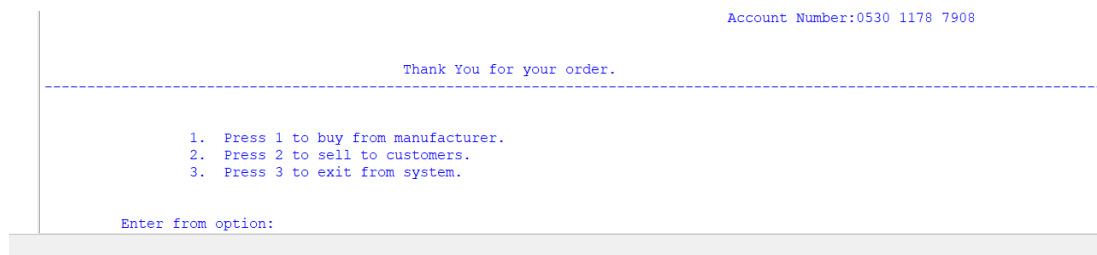


Figure 20:Screenshot of Program iteration

4.2.2 Sell to Customers

Whenever user inputs option 2, the system directs user to sell laptops to customers. The system works like this:

After providing 2 as an input to the system, the system lets user know that it needs their name and phone number for printing bill. Hence, the system asks user to provide their name and phone number and displays all the details of laptop.

```

Enter from option:2

-----
Let's shop!
-----

We will need your name and contact number for the invoice.

Please provide us your name:Georgei

Your phone number here:075674333

-----
S.N      Name      Brand      Price  Quantity      Processor      Graphic Card
-----
1      Razer Blade      Razer      $2000      17      i7 7th Gen      GTX 3060
2      XPS      Dell      $1976      12      i5 9th Gen      GTX 3070
3      Alienware      Alienware      $1978      29      i5 9th Gen      GTX 3070
4      Swift 7      Acer      $900      12      i5 9th Gen      GTX 3070
5      Macbook Pro 16      Apple      $3500      10      i5 9th Gen      GTX 3070
-----

```

Figure 21: Screenshot of program When user inputs 2 from option and inputs name and number

Now, the system asks user to input the ID and quantity of a laptop the customers want to buy.

```

-----
S.N      Name      Brand      Price  Quantity      Processor      Graphic Card
-----
1      Razer Blade      Razer      $2000      17      i7 7th Gen      GTX 3060
2      XPS      Dell      $1976      12      i5 9th Gen      GTX 3070
3      Alienware      Alienware      $1978      29      i5 9th Gen      GTX 3070
4      Swift 7      Acer      $900      12      i5 9th Gen      GTX 3070
5      Macbook Pro 16      Apple      $3500      10      i5 9th Gen      GTX 3070
-----

Please Provide the ID of the laptop you want to buy:2

Please provide the quantity of a laptop you want to buy:3

```

Figure 22: screenshot of valid ID and quantity input

The system now asks user If the customer want to buy more. If customer says yes the system loops from asking laptops ID to asking If they want to buy more.

```
Please Provide the ID of the laptop you want to buy:2

Please provide the quantity of a laptop you want to buy:3

Do you want to buy more?Enter y to buy more:y

Please Provide the ID of the laptop you want to buy:1

Please provide the quantity of a laptop you want to buy:4

Do you want to buy more?Enter y to buy more:|
```

Figure 23: screenshot of program when customer wants to buy more

If the customer doesn't want to buy more, the system asks them If they want their laptop to be shipped. If the customer wants their laptop to get shipped, the system displays the information about shipping charge and ask the customer If they want their laptop get shipped inside or outside city. And then, system asks customer to input the full location for their laptop to get shipped.

```
Do you want to buy more?Enter y to buy more:n

Do you want your laptop to be shipped?(y/n):y

Shipping charge varies inside and outside city.

Where do want your laptop to be delivered??(i/o):i

Please provide us your full address for the delivery:Street 504
```

Figure 24: Shipping Details Input

System now prints the bill with all necessary details followed by generating bill as a text file which is unique for every purchase and contains all the details of laptop bought by that one particular customer.

Please provide us your full address for the delivery: Street 504

Bit & Byte Store
London, England
bitandbyte@gmail.com
073599487

Bill To
Name: Georgei
Location: Street 504
Contact: 075674333

Date: 11 May, 2023
Time: 01:15AM

Bill Number: 1683747003136

S.N	Product's Name	Brand	Quantity	Unit Price	Amount
1	XPS	Dell	3	\$1976	\$5928
2	Razer Blade	Razer	4	\$2000	\$8000

Net Amount:	\$13928
Shipping Charge:	\$20
Gross Amount:	\$13948

Terms & Conditions
Payment is due within 15 days.

Bank Details
Bank of United Kingdom
Account Number: 0530 1178 7908

Thank You for your order.

Figure 25: screenshot of Printed invoice with shipping details

*Georgei1683747073291.txt - Notepad
File Edit Format View Help

Bit & Byte Store
London, England
bitandbyte@gmail.com
073599487

Bill To
Name: Georgei
Location: Street 504
Contact: 075674333

Date: 11 May, 2023
Time: 01:16AM

Bill Number: 1683747073291

S.N	Product's Name	Brand	Quantity	Unit Price	Amount
1	XPS	Dell	3	\$1976	\$5928
2	Razer Blade	Razer	4	\$2000	\$8000

Net Amount:	\$13928
Shipping Charge:	\$20
Gross Amount:	\$13948

Terms & Conditions
Payment is due within 15 days.

Bank Details
Bank of United Kingdom
Account Number: 0530 1178 7908

Thank You for your order.

Figure 26: screenshot of Generated Invoice with shipping details

If the customer doesn't want their laptop to get shipped, system then prints the invoice followed by generating invoice on text file.

```

Do you want your laptop to be shipped?(y/n):n
Thank You for your order.

-----
Bit & Byte Store                               Bill To                               Date: 11 May,2023
London,England                                Name:billy                               Time: 01:21AM
bitandbyte@gmail.com                          Location:None
073599487                                     Contact:432123456

-----
Bill Number: 1683747399331
-----
S.N      Product's Name      Brand      Quantity      Unit Price      Amount
-----
1         Alienware         Alienware      4             $1978          $7912
2         Macbook Pro 16      Apple         1             $3500          $3500
-----
Net Amount:                                     $11412
Shipping Charge:                               $0
Gross Amount:                                     $11412
-----
Terms & Conditions                               Bank Details
Payment is due within 15 days.                    Bank of United Kingdom
                                                Account Number:0530 1178 7908

Thank You for your order.
-----

```

Figure 27:screenshot of Printed invoice when no shipping required

```

billy1683747402812.txt - Notepad
File Edit Format View Help

-----
Bit & Byte Store                               Bill To                               Date: 11 May,2023
London,England                                Name:billy                               Time: 01:21AM
bitandbyte@gmail.com                          Location:None
073599487                                     Contact:432123456

-----
Bill Number: 1683747402812
-----
S.N      Product's Name      Brand      Quantity      Unit Price      Amount
-----
1         Alienware         Alienware      4             $1978          $7912
2         Macbook Pro 16      Apple         1             $3500          $3500
-----
Net Amount:                                     $11412
Shipping Charge:                               $0
Gross Amount:                                     $11412
-----
Terms & Conditions                               Bank Details
Payment is due within 15 days.                    Bank of United Kingdom
                                                Account Number:0530 1178 7908

Thank You for your order.
-----

```

Figure 28:Screenshot of Generated invoice when no shipping required

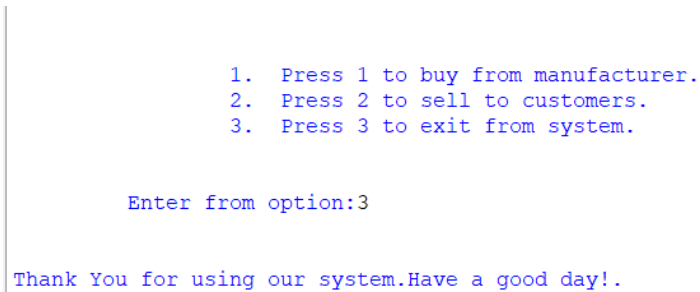
The system now again iterates and displays the option .

```
-----  
Thank You for your order.  
  
1. Press 1 to buy from manufacturer.  
2. Press 2 to sell to customers.  
3. Press 3 to exit from system.  
  
Enter from option:
```

Figure 29: Screenshot of Iteration of a program

4.3. Termination of a Program

From the three options system provides, If user inputs option 3, the system gets terminated with Thank You Message.

A screenshot of a program's termination sequence. It shows a list of three options: '1. Press 1 to buy from manufacturer.', '2. Press 2 to sell to customers.', and '3. Press 3 to exit from system.'. Below this, the prompt 'Enter from option:' is followed by the user input '3'. Finally, a message 'Thank You for using our system.Have a good day!.' is displayed.

```
1. Press 1 to buy from manufacturer.  
2. Press 2 to sell to customers.  
3. Press 3 to exit from system.  
  
Enter from option:3  
  
Thank You for using our system.Have a good day!.
```

Figure 30:Screenshot of Termination of Program

5. TESTING

Following are the testing for given project. Test 1 includes implementation of Try-except. Test 2 includes selection purchase and sale of laptops , Test 3 includes file generation of purchase of laptop , Test 4 includes File generation of sales process of laptop and Test 5 includes updated stocks.

5.1. TEST 1: Implementation of Try Except

TEST	1
OBJECTIVE:	To show implementation TRY-EXCEPT
ACTION PERFORMED:	<ul style="list-style-type: none"> ➔ When a system asked for a input to select options, invalid input was provided ➔ When a system asked for a phone number of a user/customer, an invalid input was provided.
EXPECTED RESULT:	<ul style="list-style-type: none"> ➔ System should display "Uh-Oh! The option you provided doesn't exist. Please try again" message ➔ System should display "Invalid Phone Number. Please provide valid phone numbers only" message
ACTUAL RESULT:	<ul style="list-style-type: none"> ➔ System displayed "Uh-Oh! The option you provided doesn't exist. Please Try again" message ➔ System displayed "Invalid Phone Number. Please provide valid phone numbers only" message
CONCLUSION:	Hence, the test was successful.

Table 2: Test to check if system runs on invalid input


```

*****
                        Welcome to Bit & Byte Store
*****

                        London, England
*****

1. Press 1 to buy from manufacturer.
2. Press 2 to sell to customers.
3. Press 3 to exit from system.

Enter from option:i

```

Figure 31: Screenshot of Welcome Message

```

1. Press 1 to buy from manufacturer.
2. Press 2 to sell to customers.
3. Press 3 to exit from system.

Enter from option:i

Uh-Oh!The option you provided doesn't exist.Please try again.

Enter from option:|

```

Figure 32: Screenshot of nvalid option input

```

Enter from option:1

-----
                        Let's shop.
-----

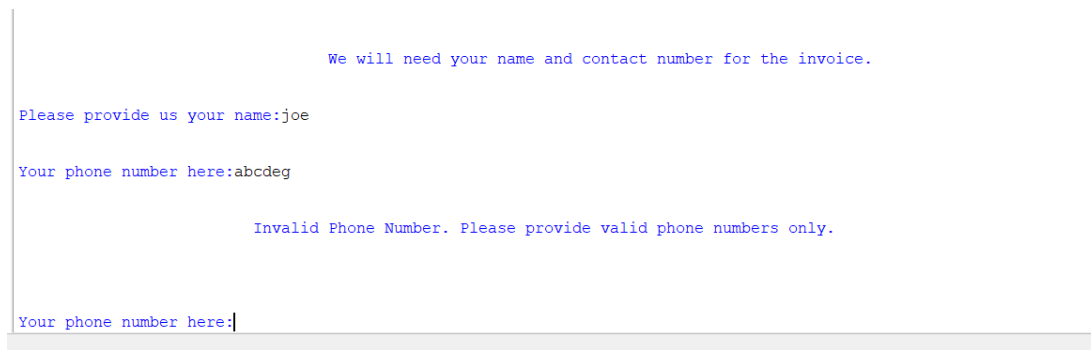
We will need your name and contact number for the invoice.

Please provide us your name:joe

Your phone number here:abcdeg

```

Figure 33: Screenshot of valid option input



We will need your name and contact number for the invoice.

Please provide us your name:joe

Your phone number here:abcdeg

Invalid Phone Number. Please provide valid phone numbers only.

Your phone number here:|

Figure 34: Screenshot of :Invalid Phone Number Input

5.2. TEST 2: Selection Purchase and Sale of Laptop

TEST	2
OBJECTIVE:	To Try to Buy and Sell Laptop with negative value and non-existence value.
ACTION PERFORMED:	<ul style="list-style-type: none"> ➔ When a system asked for the quantity of laptop user want to buy, negative input was provided. ➔ When a system asked for valid id to sell laptop to the customers, non-existence value was provided
EXPECTED RESULT:	<ul style="list-style-type: none"> ➔ System should display "That doesn't seem a valid quantity. Please provide valid quantity!" message ➔ System should display "Please provide a valid laptop id!!!" message
ACTUAL RESULT:	<ul style="list-style-type: none"> ➔ System displayed "That doesn't seem a valid quantity. Please provide valid quantity!" message ➔ System displayed "Please provide a valid laptop id!!!" message.
CONCLUSION:	Hence, the test was successful.

Table 3: Test to check if system accepts negative values or not existing values

Please provide us your name:Bicky

Your phone number here:0123611

S.N	Name	Brand	Price	Quantity	Processor	Graphic Card
1	Razer Blade	Razer	\$2000	15	i7 7th Gen	GTX 3060
2	XPS	Dell	\$1976	9	i5 9th Gen	GTX 3070
3	Alienware	Alienware	\$1978	25	i5 9th Gen	GTX 3070
4	Swift 7	Acer	\$900	12	i5 9th Gen	GTX 3070
5	Macbook Pro 16	Apple	\$3500	9	i5 9th Gen	GTX 3070

Please Provide the ID of the laptop you want to buy:5

Please provide the quantity of a laptop you want to buy:-10

Figure 35:Screenshot of Negative Input

S.N	Name	Brand	Price	Quantity	Processor	Graphic Card
1	Razer Blade	Razer	\$2000	15	i7 7th Gen	GTX 3060
2	XPS	Dell	\$1976	9	i5 9th Gen	GTX 3070
3	Alienware	Alienware	\$1978	25	i5 9th Gen	GTX 3070
4	Swift 7	Acer	\$900	12	i5 9th Gen	GTX 3070
5	Macbook Pro 16	Apple	\$3500	9	i5 9th Gen	GTX 3070

Please Provide the ID of the laptop you want to buy:5

Please provide the quantity of a laptop you want to buy:-10

Dear, Bicky Please provide valid quantity!

Please provide the number of quantity of laptop: |

Figure 36:Screenshot of Message displayed on invalid input

```

-----
Please provide us your name:Ram

Your phone number here:0123998

-----
S.N      Name      Brand      Price  Quantity      Processor      Graphic Card
-----
1        Razer Blade  Razer      $2000   15             i7 7th Gen     GTX 3060
2        XPS          Dell       $1976   9              i5 9th Gen     GTX 3070
3        Alienware    Alienware  $1978   25             i5 9th Gen     GTX 3070
4        Swift 7       Acer       $900    12             i5 9th Gen     GTX 3070
5        Macbook Pro 16  Apple     $3500   11             i5 9th Gen     GTX 3070

-----
Please Provide the ID of the laptop you want to buy:9
-----

```

Figure 37:Screenshot of Non-existing input

```

-----
S.N      Name      Brand      Price  Quantity      Processor      Graphic Card
-----
1        Razer Blade  Razer      $2000   15             i7 7th Gen     GTX 3060
2        XPS          Dell       $1976   9              i5 9th Gen     GTX 3070
3        Alienware    Alienware  $1978   25             i5 9th Gen     GTX 3070
4        Swift 7       Acer       $900    12             i5 9th Gen     GTX 3070
5        Macbook Pro 16  Apple     $3500   11             i5 9th Gen     GTX 3070

-----
Please Provide the ID of the laptop you want to buy:9

Please provide a valid laptop id!!!
-----

```

Figure 38:Screenshot of Message displayed on non-existing input

5.3. Test 3 : To Generate a File for Purchase of Laptop

TEST	3
OBJECTIVE:	To Generate a File for Purchase of Laptop
ACTION PERFORMED:	<ul style="list-style-type: none"> ➔ When a system asked for an user input to select from options, 1 was provided. ➔ Valid Name and Phone Number was provided. Name: Faith Cooper Phone: 0712577899 ➔ Valid ID, and quantity was provided. ID : 2 Quantity: 20 ➔ When system asked IF the user wanted to buy more, "y" input was provided ➔ Again, valid ID and laptop quantity was provided. ID: 5 Quantity: 19 ➔ When system again asked If the user wanted to buy more, "n" input was provided.
EXPECTED RESULT:	<ul style="list-style-type: none"> ➔ System should display the invoice and generate invoice as text file containing all the updated details of purchase.
ACTUAL RESULT:	<ul style="list-style-type: none"> ➔ System successfully displayed invoice and and text file was also generated as invoice containing all the updated details of purchase.
CONCLUSION:	Hence, the test was successful.

Table 4:Test to generate a file for purchase of laptop

```

*****
                        Welcome to Bit & Byte Store
*****
                        London, England
*****

1. Press 1 to buy from manufacturer.
2. Press 2 to sell to customers.
3. Press 3 to exit from system.

Enter from option:1

-----
                        Let's shop.
-----

                        We will need your name and contact number for the invoice.

Please provide us your name:Faith Cooper

Your phone number here:012577899

```

Figure 39:Screenshot of name and phone input

```

                        We will need your name and contact number for the invoice.

Please provide us your name:Faith Cooper

Your phone number here:012577899

-----
S.N      Name      Brand      Price      Quantity      Processor      Graphic Card
-----
1        Razer Blade  Razer      $2000      15            i7 7th Gen     GTX 3060
2        XPS        Dell        $1976      9             i5 9th Gen     GTX 3070
3        Alienware  Alienware  $1978      25            i5 9th Gen     GTX 3070
4        Swift 7     Acer        $900       12            i5 9th Gen     GTX 3070
5        Macbook Pro 16 Apple      $3500      11            i5 9th Gen     GTX 3070
-----

Please Provide the ID of the laptop you want to buy:2

Please provide the quantity of a laptop you want to buy:20

Do you want to buy more? Enter y to buy more:y

Please Provide the ID of the laptop you want to buy:|

```

Figure 40:screenshot of system when user wants to buy more

```

Do you want to buy more? Enter y to buy more:y

Please Provide the ID of the laptop you want to buy:5

Please provide the quantity of a laptop you want to buy:19

Do you want to buy more? Enter y to buy more:

```

Figure 41: Screenshot of New id and quantity input

```

Do you want to buy more? Enter y to buy more:n

-----
Bit & Byte Store                                     Date: 11 May,2023
London,England                                       Time: 11:59AM
bitandbyte@gmail.com
073599487
-----

Bill Number:1683785663352
-----
S.N      Product's Name      Brand      Quantity      Unit Price      Amount
-----
1         XPS                               Dell        20             $1976           $39520
2         Macbook Pro 16             Apple       19             $3500           $66500
-----

Net Amount:                                           $106020
Vat Amount:                                           $13782.6
Gross Amount:                                         $119802.6
-----

Please note that 13% VAT is added on your net amount.

Terms & Conditions                                     Bank Details
Payment is due within 15 days.                         Bank of United Kingdom
                                                         Account Number:0530 1178 7908

Thank You for your order.
-----

```

Figure 42: Screenshot of Printed Bill of more than one product

*Faith Cooper1683785696750.txt - Notepad
File Edit Format View Help

Bit & Byte Store
London, England
bitandbyte@gmail.com
073599487

Date: 11 May, 2023
Time: 11:59AM

Bill Number: 1683785696750

S.N	Product's Name	Brand	Quantity	Unit Price	Amount
1	XPS	Dell	20	\$1976	\$39520
2	Macbook Pro 16	Apple	19	\$3500	\$66500
Net Amount:					\$106020
Vat Amount:					\$13782.6
Gross Amount:					\$119802.6

Please note that 13% VAT is added on your net amount.

Terms & Conditions
Payment is due within 15 days.

Bank Details
Bank of United Kingdom
Account Number: 0530 1178 7908

Thank You for your order.

Figure 43: Screenshot of Generated Bill of more than one product

5.4. Test 4 : To Generate a File for sale of Laptop

TEST	4
OBJECTIVE:	To Generate a File for Sale of Laptop
ACTION PERFORMED:	<ul style="list-style-type: none"> ➔ When a system asked for an user input to select FROM options, 2 was provided. ➔ Valid Name and Phone Number was provided. Name: Steve Sharma Phone: 071577386 ➔ Valid ID, and quantity was provided. ID : 2 Quantity: 3 ➔ When system asked IF the customer wanted to buy more, "y" input was provided ➔ Again, valid ID and laptop quantity was provided. ID: 3 Quantity: 15 ➔ When system again asked If the customer wanted to buy more, "n" input was provided. ➔ When a system asked If customer wanted their laptop to get shipped, "y" input was provided ➔ Again, when system asked about the delivery location, valid location was provided.
EXPECTED RESULT:	<ul style="list-style-type: none"> ➔ System should display the invoice and generate invoice as text file containing all the updated details of purchase.
ACTUAL RESULT:	<ul style="list-style-type: none"> ➔ System successfully displayed invoice and and text file was also generated as invoice containing all the updated details of purchase.
CONCLUSION:	Hence, the test was successful.

Table 5:To test to generate file for sale of laptop

```

*****
Welcome to Bit & Byte Store
*****

*****
London, England
*****

*****

1. Press 1 to buy from manufacturer.
2. Press 2 to sell to customers.
3. Press 3 to exit from system.

Enter from option:2

-----
Let's shop!
-----

We will need your name and contact number for the invoice.

```

Figure 44: Screenshot of a system when user inputs option 2

We will need your name and contact number for the invoice.

Please provide us your name:Steve Sharma

Your phone number here:071577386

S.N	Name	Brand	Price	Quantity	Processor	Graphic Card
1	Razer Blade	Razer	\$2000	15	i7 7th Gen	GTX 3060
2	XPS	Dell	\$1976	29	i5 9th Gen	GTX 3070
3	Alienware	Alienware	\$1978	25	i5 9th Gen	GTX 3070
4	Swift 7	Acer	\$900	12	i5 9th Gen	GTX 3070
5	Macbook Pro 16	Apple	\$3500	30	i5 9th Gen	GTX 3070

Please Provide the ID of the laptop you want to buy:

Figure 45: Screenshot of name and phone input

```
Please Provide the ID of the laptop you want to buy:2

Please provide the quantity of a laptop you want to buy:3


Do you want to buy more?Enter y to buy more:y

Please Provide the ID of the laptop you want to buy:1

Please provide the quantity of a laptop you want to buy:5


Do you want to buy more?Enter y to buy more:
```

Figure 46: Screenshot of program when customers wants to buy more

```
Do you want to buy more?Enter y to buy more:n

Do you want your laptop to be shipped?(y/n):y

        Shipping charge varies inside and outside city.

Where do want your laptop to be delivered??(i/o):o

Please provide us your full address for the delivery:Strret 808
```

Figure 47: Screenshot of Shipping Details

Please provide us your full address for the delivery:Strret 808

Bit & Byte Store London,England bitandbyte@gmail.com 073599487	Bill To Name:Steve Sharma Location:Strret 808 Contact:071577386	Date: 11 May,2023 Time: 12:25PM
---	--	------------------------------------

Bill Number: 1683787223520

S.N	Product's Name	Brand	Quantity	Unit Price	Amount
1	XPS	Dell	3	\$1976	\$5928
2	Razer Blade	Razer	5	\$2000	\$10000

Net Amount:	\$15928
Shipping Charge:	\$40
Gross Amount:	\$15968

Terms & Conditions Payment is due within 15 days.	Bank Details Bank of United Kingdom Account Number:0530 1178 7908
--	---

Thank You for your order.

Figure 48:Screenshot of printed bill when customers buy more than one laptop

Steve Sharma1683787441464.txt - Notepad

File Edit Format View Help

Bit & Byte Store London,England bitandbyte@gmail.com 073599487	Bill To Name:Steve Sharma Location:Strret 808 Contact:071577386	Date: 11 May,2023 Time: 12:29PM
---	--	------------------------------------

Bill Number: 1683787441464

S.N	Product's Name	Brand	Quantity	Unit Price	Amount
1	XPS	Dell	3	\$1976	\$5928
2	Razer Blade	Razer	5	\$2000	\$10000

Net Amount:	\$15928
Shipping Charge:	\$40
Gross Amount:	\$15968

Terms & Conditions Payment is due within 15 days.	Bank Details Bank of United Kingdom Account Number:0530 1178 7908
--	---

Thank You for your order.

Figure 49:Screenshot of Generated Bill when customers buy more than one laptop

5.5. Test 3 : To Update the Stock of Laptop

TEST	5
OBJECTIVE:	To Update the Stock of Laptop
ACTION PERFORMED:	<ul style="list-style-type: none"> ➔ When a system asked for an user input to select FROM options, 1 was provided. ➔ Valid Name and Phone Number was provided. Name: Marry Phone: 071577233 ➔ Valid ID, and quantity was provided. ID : 1 Quantity: 100 ➔ When a system asked for an user input to select from options, 2 was provided. ➔ Valid Name and Phone Number was provided. Name: Constance Phone: 071577311 ➔ Valid ID, and quantity was provided. ID : 5 Quantity: 2
EXPECTED RESULT:	<ul style="list-style-type: none"> ➔ System should show the updated stock in text file as well as while on shell.
ACTUAL RESULT:	<ul style="list-style-type: none"> ➔ System successfully displayed the updated stock in text file as well as while on shell.
CONCLUSION:	Hence, the test was successful.

Table 6:Test to update stock of laptop

Enter from option:1

Let's shop.

We will need your name and contact number for the invoice.

Please provide us your name:Marry

Your phone number here:071577233

S.N	Name	Brand	Price	Quantity	Processor	Graphic Card
1	Razer Blade	Razer	\$2000	10	i7 7th Gen	GTX 3060
2	XPS	Dell	\$1976	26	i5 9th Gen	GTX 3070
3	Alienware	Alienware	\$1978	25	i5 9th Gen	GTX 3070
4	Swift 7	Acer	\$900	12	i5 9th Gen	GTX 3070
5	Macbook Pro 16	Apple	\$3500	30	i5 9th Gen	GTX 3070

Please Provide the ID of the laptop you want to buy:|

Figure 50:Screenshot of name and number input

S.N	Name	Brand	Price	Quantity	Processor	Graphic Card
1	Razer Blade	Razer	\$2000	10	i7 7th Gen	GTX 3060
2	XPS	Dell	\$1976	26	i5 9th Gen	GTX 3070
3	Alienware	Alienware	\$1978	25	i5 9th Gen	GTX 3070
4	Swift 7	Acer	\$900	12	i5 9th Gen	GTX 3070
5	Macbook Pro 16	Apple	\$3500	30	i5 9th Gen	GTX 3070

Please Provide the ID of the laptop you want to buy:1

Please provide the quantity of a laptop you want to buy:100


Figure 51:Screenshot of Id and quantity input

Bit & Byte Store London, England bitandbyte@gmail.com 073599487				Date: 11 May, 2023 Time: 12:46PM	
Bill Number:1683788466812					
S.N	Product's Name	Brand	Quantity	Unit Price	Amount
1	Razer Blade	Razer	100	\$2000	\$200000
Net Amount:					\$200000
Vat Amount:					\$26000.0
Gross Amount:					\$226000.0
Please note that 13% VAT is added on your net amount.					
Terms & Conditions Payment is due within 15 days.			Bank Details Bank of United Kingdom Account Number:0530 1178 7908		
Thank You for your order.					

Figure 52: Screenshot of Printed invoice

We will need your name and contact number for the invoice.						
Please provide us your name: Constance						
Your phone number here: 071577311						
S.N	Name	Brand	Price	Quantity	Processor	Graphic Card
1	Razer Blade	Razer	\$2000	110	i7 7th Gen	GTX 3060
2	XPS	Dell	\$1976	26	i5 9th Gen	GTX 3070
3	Alienware	Alienware	\$1978	25	i5 9th Gen	GTX 3070
4	Swift 7	Acer	\$900	12	i5 9th Gen	GTX 3070
5	Macbook Pro 16	Apple	\$3500	30	i5 9th Gen	GTX 3070

Figure 53: Screenshot of Updated stock shown on system

 *laptopDetails.txt - Notepad

File	Edit	Format	View	Help
Razer Blade, Razer, \$2000, 110, i7 7th Gen, GTX 3060				
XPS, Dell, \$1976, 26, i5 9th Gen, GTX 3070				
Alienware, Alienware, \$1978, 25, i5 9th Gen, GTX 3070				
Swift 7, Acer, \$900, 12, i5 9th Gen, GTX 3070				
Macbook Pro 16, Apple, \$3500, 30, i5 9th Gen, GTX 3070				

Figure 54: Screenshot of Updated stock on text file

S.N	Name	Brand	Price	Quantity	Processor	Graphic Card
1	Razer Blade	Razer	\$2000	110	i7 7th Gen	GTX 3060
2	XPS	Dell	\$1976	26	i5 9th Gen	GTX 3070
3	Alienware	Alienware	\$1978	25	i5 9th Gen	GTX 3070
4	Swift 7	Acer	\$900	12	i5 9th Gen	GTX 3070
5	Macbook Pro 16	Apple	\$3500	30	i5 9th Gen	GTX 3070

Please Provide the ID of the laptop you want to buy:5

Please provide the quantity of a laptop you want to buy:2

Do you want to buy more?Enter y to buy more:n

Do you want your laptop to be shipped?(y/n):y

Shipping charge varies inside and outside city.

Where do want your laptop to be delivered??(i/o):i

Please provide us your full address for the delivery:strret 300

Figure 55:Screenshot When customer buys from store

Please provide us your full address for the delivery:strret 300

Bit & Byte Store London,England bitandbyte@gmail.com 073599487	Bill To Name:Constance Location:strret 300 Contact:071577311	Date: 11 May,2023 Time: 12:51PM
---	---	------------------------------------

Bill Number: 1683788813048

S.N	Product's Name	Brand	Quantity	Unit Price	Amount
1	Macbook Pro 16	Apple	2	\$3500	\$7000

Net Amount:	\$7000
Shipping Charge:	\$20
Gross Amount:	\$7020

Terms & Conditions Payment is due within 15 days.	Bank Details Bank of United Kingdom Account Number:0530 1178 7908
--	---

Thank You for your order.

- Press 1 to buy from manufacturer.
- Press 2 to sell to customers.
- Press 3 to exit from system.

Figure 56:Screenshot of Printed Invoice

Enter from option:1

Let's shop.


We will need your name and contact number for the invoice.

Please provide us your name:ram

Your phone number here:56789

S.N	Name	Brand	Price	Quantity	Processor	Graphic Card
1	Razer Blade	Razer	\$2000	110	i7 7th Gen	GTX 3060
2	XPS	Dell	\$1976	26	i5 9th Gen	GTX 3070
3	Alienware	Alienware	\$1978	25	i5 9th Gen	GTX 3070
4	Swift 7	Acer	\$900	12	i5 9th Gen	GTX 3070
5	Macbook Pro 16	Apple	\$3500	28	i5 9th Gen	GTX 3070

Figure 57: Screenshot of Updated stock shown on system

 laptopDetails.txt - Notepad

File Edit Format View Help

```
Razer Blade, Razer,      $2000,110, i7 7th Gen, GTX 3060
XPS,      Dell,      $1976,26, i5 9th Gen, GTX 3070
Alienware, Alienware, $1978,25, i5 9th Gen, GTX 3070
Swift 7, Acer,      $900,12, i5 9th Gen, GTX 3070
Macbook Pro 16, Apple,      $3500,28, i5 9th Gen, GTX 3070
|
```

Figure 58: Screenshot of Updated stock on text file

CONCLUSION

This coursework involved developing a system for a laptop shop that allowed users to purchase laptops directly from manufacturers and then resell them to clients. This report featured a detailed description of this project, information on how it was created, instructions for use, results of testing, and algorithms and flowcharts which undoubtedly aids on better understanding of the logic. Making this project a success required a lot of devotion and effort for me and the understanding for this project was greatly enhanced by the Tutorial sessions, workshop classes, and other learning materials provided by our respected tutors, and module leaders.

The significant aspect of this project was its module based and function-based approach, which definitely challenged us to write modular and reusable code and taught to break down a complex problems into smaller which result in better readability of the program. The project also required us to work with file handling, where we had to read and write data to and from files. Another most focused area of this project was data structures, specifically lists and dictionaries. Through this project, we were able to gain a deeper understanding of these data structures and how they can used to organize data efficiently. Exception handling was only one of the significant part of the program which made program run efficiently by handling any possible exceptions that would arise.

I encountered numerous errors and issues during the development process, and it was only through careful testing and debugging that we were able to identify and resolve these problems. Moreover, I looked for a few online resources to help me with the issues that were arise while writing the program. Overall, I believe that working on this project has improved our programming skills and we were able to obtain real-world experience. The process of developing this system has been challenging at times, but it has allowed us to push our boundaries and improve our skills. To sum it all up, this project was an enriching experience for us to gain valuable insights and practical knowledge about developing a functional system using Python and I am grateful for the opportunity to have worked on it.

References

- Asana, T. (2023, jan 7). *asana*. Retrieved from asana:
<https://asana.com/resources/what-is-a-flowchart>
- gumster, J. v. (n.d.). *what is python*. Retrieved from opensource.
- hope, c. (2021, 11 06). *computer hope*. Retrieved from computer hope:
<https://www.computerhope.com/jargon/m/microsoft-word.htm>
- Jaishwal, S. (2023). *datacamp*. Retrieved from datacamp:
<https://www.datacamp.com/tutorial/data-structures-python>
- Paraschiv, L. (2023, Jan 10). *FOTC*. Retrieved from FOTC: <https://fotc.com/blog/draw-io-online-guide/>
- rishabPrabhu. (n.d.). *Geekforgeeks*. Retrieved from geekforgeeks:
<https://www.geeksforgeeks.org/introduction-to-algorithms/>
- Sharma, A. (2020, May). *datacamp*. Retrieved from datacamp:
<https://www.datacamp.com/tutorial/python-IDLE>
- Simplilearn. (2023, Feb 27). *simplilearn*. Retrieved from simplilearn:
<https://www.simplilearn.com/dictionary-in-python-article>
- Tagliaferri, L. (2021, August 20). *DigitalOcean*. Retrieved from Digital Ocean:
<https://www.digitalocean.com/community/tutorials/understanding-lists-in-python-3>
- theprogrammedwords. (n.d.). *geeksforgeeks*. Retrieved from
<https://www.geeksforgeeks.org/how-to-write-a-pseudo-code/>
- vegibit*. (n.d.). Retrieved from vegibit: <https://vegibit.com/what-are-python-syntax-basics/>

APPENDIX

Main module

```
from operations import display_welcome, buy_from_manufacturer, sell_from_store,
exit_message
```

```
from write import invoiceBuy, invoiceSell
```

```
# displays welcome message
```

```
display_welcome()
```

```
# function defining user options
```

```
def userOptions():
```

```
    continueLoop = True
```

```
    while continueLoop == True:
```

```
        print("\t\t 1. Press 1 to buy from manufacturer.")
```

```
        print("\t\t 2. Press 2 to sell to customers.")
```

```
        print("\t\t 3. Press 3 to exit from system.")
```

```
        print("\n")
```

```
        options = True
```

```
        while options == True:
```

```
            try:
```

```
                optionsInput = int(input("\t Enter from option:"))
```

```
                print("\n")
```

```
                options = False
```

```
            except:
```

```
                print("\n")
```

```
                print("\t\t Uh-Oh!The option you provided doesn't exist.Please try again.")
```

```
                print("\n")
```

```
    if optionsInput == 1:
        validLaptopId, laptopQuantity, userName, customersDictionary =
buy_from_manufacturer()
        invoiceBuy(validLaptopId, laptopQuantity,userName, customersDictionary)

    elif optionsInput == 2:
        validLaptopId, laptopQuantity, userName, userPhone, location, shippingCharge,
customersDictionary = sell_from_store()
        invoiceSell(validLaptopId, laptopQuantity,userName, userPhone, location,
shippingCharge, customersDictionary)

    elif optionsInput == 3:
        continueLoop = False
        exit_message()

    else:
        print("\t\t To continue, Please select a proper option.")
        print("\n")

userOptions()
```



```
print("\n")
userName = valid_name()
userPhone = valid_phone()
print("\n")
print("-"*130)
print("S.N \t Name \t\t\t Brand \t\t\t Price \t \t Quantity \t\t Processor \t\t Graphic Card")
print("-"*130)
readFromTextFile()
print("\n")

while loop == True:
    validLaptopId = valid_id()
    laptopQuantity = quantity()
    requiredQuantity = laptopDetailsDictionary[validLaptopId][3]

    while laptopQuantity <= 0:
        print("\t\t\t " + "That doesn't seem a valid quantity. Please provide valid quantity!")
        print("\n")
        laptopQuantity = int(input("Please provide the number of quantity of laptop: "))
        print("\n")
    print("\n")

# Update the text file
laptopDetailsDictionary[validLaptopId][3] =
int(laptopDetailsDictionary[validLaptopId][3]) + int(laptopQuantity)
writeTextFile(laptopDetailsDictionary)
nameOfProduct = laptopDetailsDictionary[validLaptopId][0]
totalQuantity = laptopQuantity
```


[illegible]

[illegible]

```
return validLaptopId, laptopQuantity, userName, customersDictionary
```

```
# when user enters option 2
```

```
def sell_from_store():
```

```
customersDictionary = []
```

```
loop = True
```

```
laptopDetailsDictionary = openFile()
```

```
print("-"*130)
```

```
print("\t \t \t \t \t \t Let's shop!")
```

```
print("-"*130)
```

```
print("\n")
```

```
print("\t\t\t\t We will need your name and contact number for the invoice.")
```

```
print("\n")
```

```
userName = valid_name()
```

```
userPhone = valid_phone()
```

```
print("\n")
```

```
print("-"*130)
```

```
print("S.N \t Name \t\t Brand \t\t Price \t Quantity \t\t Processor \t\t Graphic Card")
```

```
print("-"*130)
```

readFromTextFile()

```
print("\n")
```

```
while loop == True:
```

```
validLaptopId = valid_id()
```

```
laptopQuantity = quantity()
```

```
print("\n")
```

```
requiredQuantity = laptopDetailsDictionary[validLaptopId][3]
```

```
while laptopQuantity <= 0 or laptopQuantity > int(requiredQuantity):

    print("\t\t\t Dear" + " " + str(userName) + " the quantity you are looking for is not
on stock!!!")

    print("\n")

    laptopQuantity = int(input("Please provide the number of quantity of laptop: "))

    print("\n")


# Update the text file

laptopDetailsDictionary[validLaptopId][3] =
int(laptopDetailsDictionary[validLaptopId][3]) - int(laptopQuantity)

writeTextFile(laptopDetailsDictionary)


# getting user purchased items


nameOfProduct = laptopDetailsDictionary[validLaptopId][0]
totalQuantity = laptopQuantity
unitPrice = laptopDetailsDictionary[validLaptopId][2]
price = laptopDetailsDictionary[validLaptopId][2].replace("$", "")
totalPrice = int(price) * int(totalQuantity)
brandName = laptopDetailsDictionary[validLaptopId][1]
dateandtime = datetime.now()
datetime_ms = int(datetime.timestamp(datetime.now())) * 1000
customersDictionary.append(
    [nameOfProduct, brandName, totalQuantity, unitPrice, price, totalPrice])
buy = input("Do you want to buy more?Enter y to buy more:").lower()


if buy == "y":
    loop = True
else:
```

```
        loop = False

    print("\n")
    shippingCharge = 0
    shipping = (input("Do you want your laptop to be shipped?(y/n):"))

    shipping == "y".lower():
        print("\n")
        print("\t\t\t Shipping charge varies inside and outside city.")
        print("\n")
        while True:
            try:
                shippingLocation = (input("Where do want your laptop to be
delivered??(i/o):"))
                if shippingLocation == "i".lower():
                    shippingCharge = 20
                    break
                elif shippingLocation == "o".lower():
                    shippingCharge = 40
                    break
                else:
                    print("Invalid Input")

            except ValueError:
                print("Invalid Input")
        print("\n")
        location = input("Please provide us your full address for the delivery:")

    else:
        print("Thank You for your order.")
```

[illegible]

read module

```
def openFile():
    laptopDetailsDictionary = {}
    file = open("laptopDetails.txt", "r")
    laptopId = 1
    for line in file:
        line = line.replace("\n", "")
        laptopDetailsDictionary[laptopId] = line.split(",")
        laptopId += 1
    return laptopDetailsDictionary
```

```
def readFromTextFile():
    a = 1
    file = open("laptopDetails.txt", "r")
    for line in file:
        print(a, "\t "+line.replace(", ", "\t\t"))
        a += 1
    file.close()
```

name_validation function

```
def valid_name():
    while True:
        userName = input("Please provide us your name:")
        try:
            int(userName)
```



```
except ValueError:
    return userName
print("\n")
print("\t\t\t Please provide valid name.")
print("\n")
```

phone validation function

```
def valid_phone():
    while True:
        print("\n")
        userPhone = input("Your phone number here:")
        try:
            int(userPhone)
            return userPhone
        except ValueError:
            print("\n")
            print("\t\t\t Invalid Phone Number. Please provide valid phone numbers only.")
            print("\n")
```

id validation function

```
def valid_id():
    while True:
        try:
            print("\n")
            validLaptopId = int(input("Please Provide the ID of the laptop you want to buy:"))
            if validLaptopId <= 0 or validLaptopId > len(openFile()):
```

```
        print("\n")
        print("\t\t\t Please provide a valid laptop id!!!")
        print("\n")
    else:
        return validLaptopId
    print("\n")
except ValueError:
    print("\n")
    print("\t\t\t Uh-Oh! It doesn't seem a valid ID. Please check the details above and
continue.")
    print("\n")
```

quantity validation function

```
def quantity():
    while True:
        try:
            print("\n")
            laptopQuantity = int(input("Please provide the quantity of a laptop you want to
buy:"))
            print("\n")
            return laptopQuantity
        except:
            print("\n")
            print("\t\t\t Please provide valid quantity.")
            print("\n")
```

write module

```
from read import openFile
from datetime import datetime

# laptopDetails.txt file ma naya write garna or update garna
def writeTextFile(laptopDetailsDictionary):
    file = open("laptopDetails.txt", "w")
    for values in laptopDetailsDictionary.values():
        file.write(str(values[0]) + "," + str(values[1]) + "," + str(values[2]) + "," + str(values[3])
+ "," + str(values[4]) + "," + str(values[5]))
        file.write("\n")
    file.close()

# writes invoice on new next file when user enters option 1
def invoiceBuy(validLaptopId, laptopQuantity, userName, customersDictionary):
    dateandtime = datetime.now()
    laptopDetailsDictionary = openFile()

    nameOfProduct = laptopDetailsDictionary[validLaptopId][0]
    totalQuantity = laptopQuantity
    unitPrice = (laptopDetailsDictionary[validLaptopId][2])
    price = laptopDetailsDictionary[validLaptopId][2].replace("$", "")
    totalPrice = int(price) * int(totalQuantity)
    dateandtime = datetime.now()
    datetime_ms = int(datetime.timestamp(datetime.now())) * 1000
    brandName = laptopDetailsDictionary[validLaptopId][1]
```

```

with open(str(userName) + str(datetime_ms) + ".txt", 'w') as buy:
    buy.write("\n")
    buy.write("-"*130)
    buy.write("\n")
    buy.write("Bit & Byte Store" + "\t\t\t\t\t\t\t\t\t\t" + " Date: " +datetime.strftime("%d"
" " "%b""", ""%Y"))
    buy.write("\n")
    buy.write("London,England" + "\t\t\t\t\t\t\t\t\t\t" " " "Time:"
+datetime.strftime("%I" " ":" "%M" "%p"))
    buy.write("\n")
    buy.write("bitandbyte@gmail.com")
    buy.write("\n")
    buy.write("073599487")
    buy.write("\n")
    buy.write("\n")
    buy.write("\n")
    buy.write("\n")
    buy.write("Bill Number:" + str(datetime_ms))
    buy.write("\n")
    buy.write("\n")
    buy.write("-"*130)
    buy.write("\n")
    buy.write("S.N" + "\t\t" + "Product's Name" + "\t\t" + "Brand" + "\t\t" + "Quantity" +
"\t\t" + " Unit Price" + "\t\t\t\t" + "Amount")
    buy.write("\n")
    buy.write("-"*130)
    buy.write("\n")
    buy.write("\n")
    counter = 1

```


[illegible]

```
# writes invoice on new txt file when user enters option 2
```

```
def invoiceSell(validLaptopId, laptopQuantity, userName, userPhone, location,
shippingCharge, customersDictionary):
```

```
datetime = datetime.now()
```

```
laptopDetailsDictionary = openFile()
```

```
nameOfProduct = laptopDetailsDictionary[validLaptopId][0]
```

```
totalQuantity = laptopQuantity
```

```
unitPrice = laptopDetailsDictionary[validLaptopId][2]
```

```
price = laptopDetailsDictionary[validLaptopId][2].replace("$", "")
```

[illegible]


```
sell.write("-"*98)
sell.write("\n")
sell.write("\n")
sell.write("Terms & Conditions" "\t\t\t\t\t" + "Bank Details")
sell.write("\n")
sell.write("Payment is due within 15 days." "\t\t\t\t\t" + "Bank of United Kingdom")
sell.write("\n")
sell.write("\t\t\t\t\t\t\t\t\t\t\t" + "Account Number:0530 1178 7908")
sell.write("\n")
sell.write("\n")
sell.write("\t\t\t\t\t" " " + "Thank You for your order.")
sell.write("\n")
sell.write("\n")
sell.write("-"*98)
sell.write("\n")
```

THE END