LONDON METROPOLITAN UNIVERSITY

ITAHARI INTERNATIONAL COLLEGE

Module Code & Module Title:

CS4051NT Fundamentals Of Computing

Assessment Weightage & Type:

60% Individual Coursework

Year and Semester:

2024 Spring

Student Name: Adit Tamang

London Met ID: 23056499

Assignment Due Date: 18th August,2024

Word Count: 9937

Submitted to: AjayRaj Bhattarai

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.

Acknowledgement

I want to express my sincere gratitude to my module teacher, mentors, and instructors for their guidance and support throughout this coursework. I managed to remain motivated and focused thanks to their insightful advice and support. Their feedback and suggestions were really helpful to me in understanding the topic and finishing the project successfully.

Finally, I would want to express my sincere appreciation to my family for their constant support and patience during this coursework . Their encouragement and understanding helped me to stay motivated and focused, which is how I was able to finish this particular coursework. I appreciate all of your help and contributions during this journey.

Table of Contents

1)Introduction to Project 1

1.1) Introduction to CourseWork 1

1.1.1) Goals 2

1.1.2) Objectives 2

1.2) Fundamental of Computing 3

1.3) Computing 4

1.4) Python 4

1.5) IDLE 5

1.6) Tools used 6

1.6.1) Ms Word 6

1.6.2) Draw.io 6

2) Algorithm 7

3)PseudoCode 11

i) Main.py 11

ii) Operation.py 12

iii) Invoice.py 18

iv) Write.py 20

V) Read.py 21

4) FlowChart 22

5) Data Structure 23

5.1) Why Data structures is important . and what happens if it is not present? 23

5.2) Linear Data Structures 23

5.2.1) Arrays 24

5.2.2) Linked Lists 24

5.2.3) Stacks 24

5.2.4) Queues 25

5.2.5) Lists 26

5.2.6) Tuples 26
5.3) Non-Linear Data Structure 27
5.3.1) Dictionary 27
5.3.2) Sets 28
5.3.3) Strings 28
5.3.4) Integer 29
6) Program 29
6.1) Main.py 29
6.2) Operation.py 30
6.3) Write.py 35
6.4) Read.py 36
6) Testing 37
6.1) Testing 1 : Exception Handling and Input Validation 37
6.2) Testing 2: User input validation and menu navigation 38
6.3) Testing 3: File Handling to update the inventory after purchasing 40
6.4) Testing 4: File Handling to update the inventory after selling 41
6.5) Testing 5: Trying to sell more quantity than the available items. 43
6.6) Testing 6: Providing free delivery within the Itahari area. 44
6.7) Testing 7: Control Program Flow 46
6.8) Testing 8: Invoice generation after buying the multiple items 47
6.9) Testing 9: Invoice generation after selling the multiple items 49
6.10) Testing 10: Generating new files and record of transactions 51
6.11) Test 11: Updating the inventory: File handling 52
7) Conclusion 54
8) Appendix 56
i) Main.py 56

ii) Operation.py 59
iii) Invoice.py 70
iv) Write.py 75
v) Read.py 75
9) References 77
References 77
10) Plagiarism Report 81
Student passage FLAGGED 82
Student passage CITED 82
Student passage FLAGGED 82
Student passage QUOTED 83
Student passage CITED 83
Student passage FLAGGED 84
Student passage FLAGGED 84
Student passage QUOTED 84
Student passage FLAGGED 85
Student passage QUOTED 85
Student passage CITED 85
Student passage CITED 86

Table Of Figures

Figure 1:Python 4
Figure 2: Python IDLe 5
Figure 3: Ms Word 6
Figure 4: Draw.io 6
Figure 5: Flowchart 22
Figure 6: Data Structures 23
Figure 7: Stack Data Structure 25
Figure 8 : Queue Data Structure 26
Figure 9: Proof of using the Lists data structures 26
Figure 10 : Non-Linear Data Structure 27
Figure 11: Proof of using data dictionary 28
Figure 12: Proof of using the String data structure 29
Figure 13: Proof of using the integer data structures 29
Figure 14: Main function 30
Figure 15: Function to find-items 31
Figure 16: Function to define display_stock 31
Figure 17: Function to define purchase_furniture 33
Figure 18: Function for selling products 35
Figure 19: Function for write_to_file 35
Figure 20: Function define for read_furniture 36
Figure 21: Displaying appropriate messages 38
Figure 22: Displaying error messages while entering the invalid format while viewing the inventory 39
Figure 23: Displaying all the Furniture Inventory 40
Figure 24: Items before purchasing 41
Figure 25: Items after purchasing 41
Figure 26: Items before Selling 41
Figure 27: Items after selling the products 42
Figure 28: Trying to sell the more items 44
Figure 29: Selling the items present in the inventory 44

Figure 30: Checking by entering the Itahari address 45
Figure 31: Checking by different address except Itahari 45
Figure 32: Ending the program 46
Figure 33: Receipt of Purchase 48
Figure 34: Receipt of Selling the items 50
Figure 35: Highlighting the invoice that has been generated 51
Figure 36: Item before adding in the inventory 52
Figure 37: Item added in the inventory 53

Table Of Tables

Table 1: Implementing Try Except 37
Table 2: Test to display the inventory 39
Table 3: Updating the Availability 40
Table 4: Updating the availability 41
Table 5: Trying to sell more than it's available 43
Table 6: Free delivery charge 44
Table 7: Choosing 5 to exit the program 46
Table 8: Purchasing Multiple Products receipt 47
Table 9: Selling multiple Products 49
Table 10: Proof that new Invoice File is created after buying or selling the items 51
Table 11:Adding the items in the inventory 52

1)Introduction to Project

This is the 1st coursework of the Fundamental Of Computing (FOC). This coursework consists of 60% of our total module grades. This is our individual coursework. Firstly, I want to thank our module teacher for providing the coursework because this coursework help me to learn a lot of new things.

1.1) Introduction to CourseWork

In this coursework, we have to create a BRJ Furniture Store using the Python Languages. The primary objective of this project was to design a Furniture Management System that would enable a user to purchase the products. Also, we will be doing the file handling as well. Every transaction in this project at BRJ Furniture Store requires an invoice. When we buy or sell furniture from a manufacturer, the invoice should have the following information: the furniture ID, the name of the manufacturer, the product name, the quantity, the users' names who purchase or sell the products, the date and time of the transaction, and the total amount that has to be paid. When ordering more than one item, the invoice should include the total cost of all the products excluding VAT and also with including the VAT.

The invoice which was generated after buying or selling the products should have the following information: the client's name, furniture ID, brand, product name, quantity, price, purchase date, and time. The final amount including the shipping cost, and the total before shipping should also be included. The invoice should include the total cost of all the furniture a customer purchases if they buy several pieces. Same as in the case of selling the products to the customers. To handle the exception, we will be using the try except case. We also imported the date time for present date, time and year. We will be writing our code in 5 different files: i) Main.py ii) Operation.py

iii) Read.py iv) Write.py v) Invoice.py

1.1.1) Goals

The goal of this project is to create an effective and user-friendly environment for inventory management and transactions handling. This program should keep the track of the available furniture items in the store, update stocks when the items are bought or sold, and producing the invoice for each transaction. There will be free shipping within in the “Itahari” area. Our main theme is to make the customers happy by selling the products in affordable price with the best quality products.

1.1.2) Objectives

- Displaying the current furniture inventory from a text file, along with ID, product name, manufacturer, quantity and price.
- When an item is purchased or sold, the text file's availability is updated to reflect the updated inventory.
- Each transaction is added with the detailed data includes the Furniture ID, manufacturer name, product name, quantity, employee name, current datetime and total cost should all be included in orders placed with manufacturers.
- For selling, the invoice includes the student’s name, furniture ID, company name, product name, quantity of items, total cost including shipping, and the total cost after applying the VAT.
- Multiple buying and selling products are displayed in single receipt.
- Now, ensuring that the inventory item has been updated to current inventory or not.

1.2) Fundamental of Computing

> Using object-oriented design and programming, this course introduces the creative process of programming. These cover a wide range of subjects, including classes, data structures, algorithms, polymorphism, encapsulation. Concepts have been reinforced through hands-on activities that are modeled after real-world situations. (sps.nyu, 2023)

To succeed in the course, we must possess curiosity, self-discipline, and the ability to collaborate with others. We'll begin with an introduction to information processing basics and provide an overview of the several ways digital information is handled by computers and other communication devices. Furthermore, we will gain knowledge about certain fundamentals like quality, security, privacy, usability, and complexity of built solutions. In addition, by employing suitable programming languages like Python, we will learn the fundamentals of modeling, creating, implementing, and testing software systems targeted for practical applications. Math and algebra proficiency is required. (NorthEasternLondon, 2024)

Along with developing data structures and algorithms related to digital information processing, the courses also cover the use of sequential, iterative, and recursive algorithms to solve typical problems in text and numeric fields.

This is a semester-long module with 15 credit hours points. So, after the completion of this course. We will be obtaining the 15 hours credit point.

1.3) Computing

The process of utilizing computer technology to carry out a certain goal-oriented work is known as computing. Computing can refer to the design and development of hardware and software systems for a variety of uses, including the organization, processing, and management of any type of information. It can also include the creation of intelligent systems, the use of various media for communication and entertainment, and the advancement of scientific research. (Techopedia, 2012)

1.4) Python

Python is a very popular languages among the other programming languages because it's syntax is simple and also it is very easy to learn for the beginners.. It is used for creating software, websites, data analysis and many more.

Python is widely used in task automation, web and software development, data visualization, and data analysis. Because Python is so easy to learn, even non-programmers like scientists and accountants use it for a variety of everyday tasks including site creation and money management. (Coursera, 2024)

Figure 1:Python

1.5) IDLE

IDLE (Integrated Development and Learning Environment) is an integrated development environment (IDE) for Python. By default, the IDLE module is included in the Windows Python installer. Code writing is made simpler by these sets of programs. While there are various IDEs available on the website, Python IDLE is the greatest tool for a beginner programmer because it contains very basic languages.

Figure 2: Python IDLe

Like Python Shell, IDLE can be used to develop, modify, and run Python scripts in addition to execute a single statement. For writing Python scripts, IDLE offers a full of features text editor with syntax highlighting, autocompletion, and wise indentation. Additionally, it also provides a debugger with tools for breakpoints and stepping. (Tpoint Tech, 2011-2021)

1.6) Tools used

1.6.1) Ms Word

Microsoft created Microsoft Word; a popular word processor sold under license. Accompanying the Microsoft Office productivity package, Microsoft Word is also available for individual purchase. (Techopedia, 2022)

Figure 3: Ms Word

1.6.2) Draw.io

A web-based diagramming application called draw.io offers a number of capabilities, such as network diagrams, UML diagrams, and flowcharting. Diagram sharing and access is made simple by its integration with well-known cloud services like OneDrive, Dropbox, and Google Drive. Using open-source, proprietary, and commercial threat intelligence sources. The foundation of analysis is factual data that can be independently verified. (UpGuard, 2024)

Figure 4: Draw.io

2) Algorithm

Algorithms are organized collections of instructions created to carry out certain operations or solve particular issues. They work by following a set of clearly defined phases, each of which serves the main objective. (DataCamp, 2024)

BRJ Furniture Store Program Algorithm

1. Start.
2. Show the welcome message and store details.
3. Open the furniture.txt file to get the list of available furniture.
4. Show the main menu:
   - 1. View all items.
   - 2. Buy from Manufacturer.
   - 3. Selling to Customer.
   - 4. Add new items.
   - 5. Exit.
5. Step 1: Ask the user to choose an option from the menu:
   - If they choose 1, go to Step 2.
   - If they choose 2, go to Step 3.
   - If they choose 3, go to Step 10.
   - If they choose 4, go to Step 17.
   - If they choose 5, go to Step 22.
6. Step 2: Show all the furniture items available:

- Read the data from the file.
- Display the ID, Manufacturer, Name, Quantity, and Price for each item.
- Go back to the menu (Step 1).

7. Step 3: If the user wants to buy from the manufacturer:
8. Step 4: Ask for the user's name.
9. Step 5: Ask for the item ID and quantity they want to buy.
10. Step 6: Check if the item is in stock:
    - If it is, add it to their purchase list.
    - If not, tell them it’s not available and ask for a different ID.
11. Step 7: Ask if they want to buy anything else:
    - If yes, go back to Step 5.
    - If no, go to Step 8.
12. Step 8: Ask for the delivery address:
    - If the address is "Itahari," shipping is free.
    - If not, ask if they want to add a shipping cost.
13. Step 9: Calculate the total cost, including VAT and any shipping fees. Generate an invoice:
    - Include details like the items, the user’s name, shipping cost, VAT, and total price.
    - Save the invoice as a .txt file.
    - Update the inventory file.
    - Show a success message and go back to the menu (Step 1).
14. Step 10: If the user wants to sell to a customer:
15. Step 11: Ask for the customer’s name.

16. Step 12: Ask for the item ID and quantity they want to buy.
17. Step 13: Check if the item is available and if there's enough stock:
    - If yes, deduct the quantity and add it to their sale list.
    - If not, tell them there's not enough stock and ask for a different ID.
18. Step 14: Ask if they want to buy anything else:
    - If yes, go back to Step 12.
    - If no, go to Step 15.
19. Step 15: Ask for the delivery address:
    - If the address is "Itahari," shipping is free.
    - If not, ask if they want to add a shipping cost.

20. Step 16: Calculate the total cost, including VAT and any shipping fees. Generate a sales invoice:
    - Include details like the items, the customer's name, shipping cost, VAT, and total price.
    - Save the invoice as a .txt file.
    - Update the inventory file.
    - Show a success message and go back to the menu (Step 1).
21. Step 17: If the user wants to add new items to the inventory:
22. Step 18: Ask for the new item's details: ID, Manufacturer, Name, Quantity, and Price.
23. Step 19: Check that the quantity and price are valid (positive numbers).
24. Step 20: Add the new item to the inventory and save it to the file.
25. Step 21: Show a success message and go back to the menu (Step 1).
26. Step 22: If the user chooses to exit, close the program.

27. End.

3)PseudoCode

A step-by-step explanation of an algorithm is called a pseudocode. Since pseudocode is intended for human understanding rather than machine interpretation, it is written in simple English and is not represented by any programming language. Pseudocode is the stage in a high-level language that occurs between an idea and its implementation (code). (GeeksforGeeks, 2023)

i) Main.py

IMPORT read_furniture_data from read.py

IMPORT purchase_furniture, sell_furniture, add_new_item, display_stock from operation.py

Print welcome messages and menu options

DEFINE main () function :

File_name is “Furniture.txt”

WHILE True**:**

GET user choice’s from input

IF user’s choice is ‘1’:

Call the read_furniture_data from the read file

ELIF CHOICE is ‘2’:

Call the purchase_furniture from the operation file

ELIF CHOICE IS ‘3’:

Call the sell_furniture from the operation file

ELIF CHOICE is ‘4’:

Call the add_new_item from the operation file

ELIF CHOICE is ‘5’:

PRINT a bye messages

BREAK the loop

ELSE:

PRINT invalid messages

ii) Operation.py

IMPORT write_to_file from write.py

IMPORT read_furniture_data from read.py

IMPORT generate_invoice from invoice.py

DEFINE find_items(furniture_list, item_id):

FOR each item in furniture_list:

IF item['id'] equals item_id:

RETURN item

RETURN None

DEFINE display_stock(furniture_list):

PRINT "Available Furniture:"

FOR each item in furniture_list:

PRINT item details in a proper formatted way

PRINT new line(“/n”)

DEFINE purchase_furniture(file_name):

READ furniture list from file

INITIALIZE empty cart list [ ]

GET employee_name from input

WHILE True:

TRY:

GET item_id and quantity from input

CHECK if quantity is positive

EXCEPT ValueError:

PRINT error message

CONTINUE to the next iteration

CALL find_items(furniture_list, item_id) to get item

IF item exists:

ADD item and quantity to cart

PRINT confirmation message

ELSE:

PRINT error message

WHILE True:

TRY :

GET continue_choice from input

CHECK if input is 'yes' or 'no'

IF continue_choice is 'no':

BREAK loop

IF cart is not empty:

GET address from input

INITIALIZE shipping_cost to 0

IF address is not "itahari":

WHILE True:

GET add_shipping from input

CHECK if input is 'yes' or 'no'

IF add_shipping is 'yes':

WHILE True:

TRY :

GET shipping_cost from input

CHECK if shipping_cost is negative or not

BREAK

EXCEPT ValueError:

PRINT error message

ELSE:

PRINT free shipping message

CALCULATE VAT Rate, VAT Amount and total amount

FOR entry_of_item in cart

SET shipping_display based on shipping cost

CALL generate_invoice with purchase details

CALL write_to_file to update the furniture list

PRINT success message

ELSE:

PRINT no products chosen message

DEFINE sell_furniture(file_name):

READ furniture list from file

INITIALIZE empty cart list

GET customer_name from input

WHILE True:

TRY:

GET item_id and quantity from input

IF quantity is > then 0:

RAISE ValueError messages

BREAK

EXCEPT ValueError:

PRINT error message

CONTINUE to the next iteration

CALL find_items(furniture_list, item_id) to get item

IF item exists and has sufficient quantity:

ADD item and quantity to cart

UPDATE item quantity

PRINT success message

ELSE:

PRINT error message

WHILE True:

GET continue_choice from input

CHECK if input is 'yes' or 'no'

IF continue_choice is 'no':

BREAK loop

IF cart is not empty:

GET address from input

INITIALIZE shipping_cost to 0

IF address is not "itahari":

WHILE True:

GET add_shipping from input

CHECK if input is 'yes' or 'no'

IF add_shipping is 'yes':

WHILE True:

GET shipping_cost from input

CHECK if shipping_cost is non-negative

ELSE:

PRINT free shipping message

CALCULATE VAT and total amount

SET shipping_display based on shipping cost

CALL generate_invoice with sale details

CALL write_to_file to update the furniture list

PRINT success message

ELSE:

PRINT no products chosen message

DEFINE add_new_item(file_name):

READ furniture list from file

TRY:

GET item_id, manufacturer, name, quantity, and price from input

CHECK if quantity and price are positive

EXCEPT ValueError:

PRINT error message

RETURN

CREATE new item dictionary

ADD new item to furniture_list

CALL write_to_file to update the furniture list

PRINT success message

iii) Invoice.py

IMPORT datetime

GENERATE a unique invoice number using current date and time

CREATE a new receipt filename with transaction type and invoice number

OPEN the receipt file in write mode

WRITE header information to the file:

Store name, address, phone number, and email

WRITE invoice details:

Invoice title

Person's name

Invoice number

Transaction type

Current date and time

WRITE header for creating table:

ID, Manufacturer, Name, Quantity, Price

INITIALIZE total_amount to 0

FOR each item in the cart:

CALCULATE total_price (item price * quantity)

ADD total_price to total_amount

WRITE item details (ID, Manufacturer, Name, Quantity, Total Price) to the file

WRITE total amount excluding VAT

WRITE VAT amount

IF include_shipping is True:

WRITE shipping_display to the file

ADD shipping_cost to total_amount

CALCULATE total amount including VAT (total_amount + vat_amount)

WRITE total amount including VAT to the file

WRITE footer information:

Thank you message

Certification statement

CLOSE the file

PRINT message indicating that the invoice has been generated

iv) Write.py

IMPORT datetime

DEFINE write_to_file(file_name, furniture_list):

OPEN file with filename in **WRITE** mode

For each item in furniture_list:

Write the item details to the file:

ID

Manufacturer

Name

Quantity

Price (formatted as currency with two decimal places)

CLOSE the file.

V) Read.py

DEFINE read_furniture_data(filename) function:

Initialize an empty list called furniture_list.

TRY to open the file specified by filename in read mode.

FOR each line in the file:

IF the line is not empty:

Split the line by ', ' to get the elements.

CREATE a dictionary for the furniture item with the following fields:

'id': convert elements[0] to an integer.

'manufacturer': use elements[1].

'name': use elements[2].

'quantity': convert elements[3] to an integer.

'price': convert elements[4] to a float (remove the '$' sign).

Append the dictionary to furniture_list.

EXCEPT FileNotFoundError:

PRINT an error message indicating that the file was not found.

EXCEPT other exceptions:

PRINT a general error message indicating an issue occurred.

RETURN furniture_list.

4) FlowChart

An instruction set is represented by a diagram called a flowchart. Typically, the various sorts of instructions in flowcharts are represented by standard symbols. The flowchart and the problem's step-by-step solution are created using these symbols. (BBC, 2024)

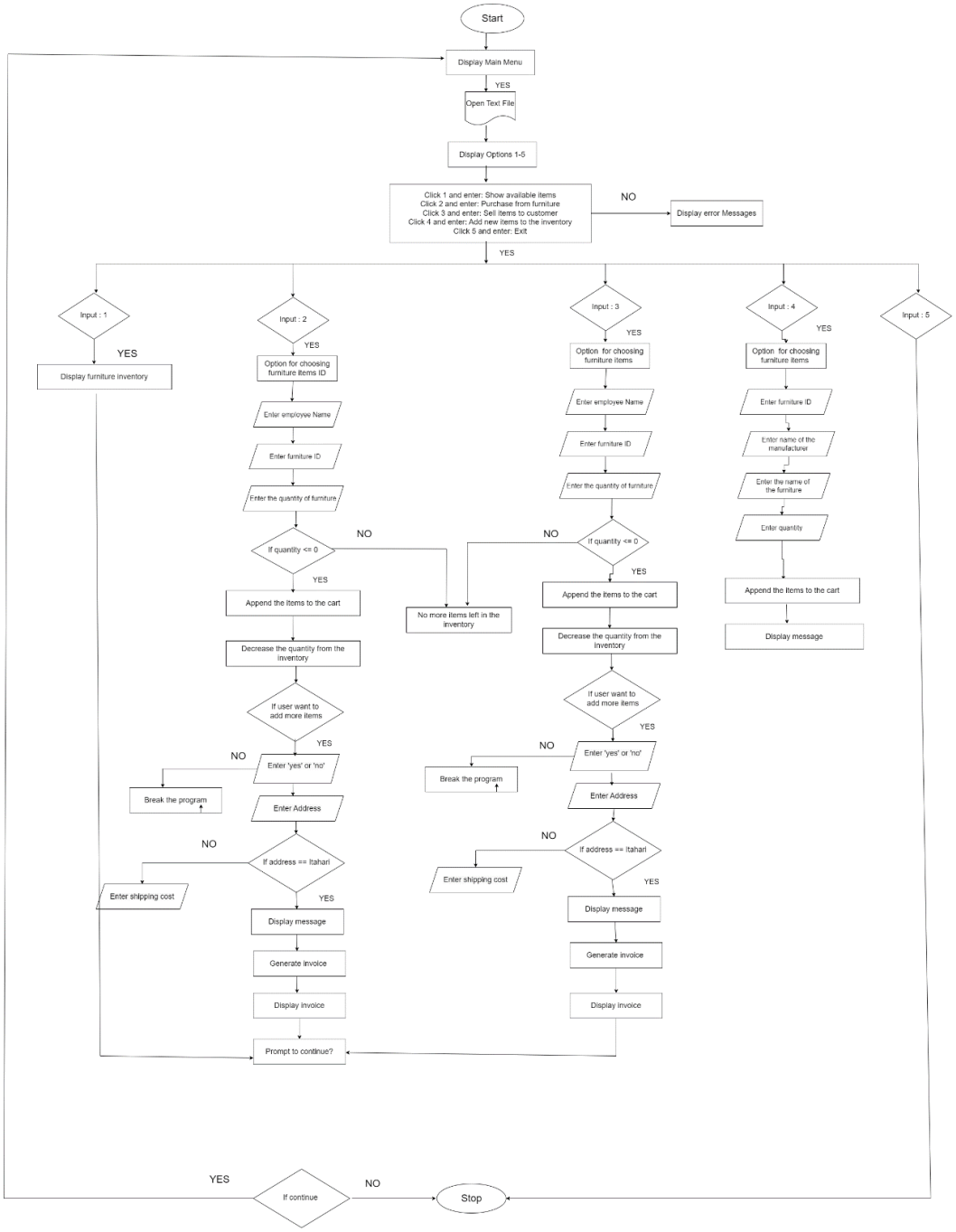

Figure 5: Flowchart

5) Data Structure

A data structure is a type of specialized format used to store, analyze, retrieve, and organize data. Data can be organized for a specific purpose using a wide range of basic and complicated data structures that are available. Data structures allow customers' access to and utilization of the required data. Generally, there are 2 types of data Structures: i) Linear & ii) Non-Linear . (altexsoft, 2024)

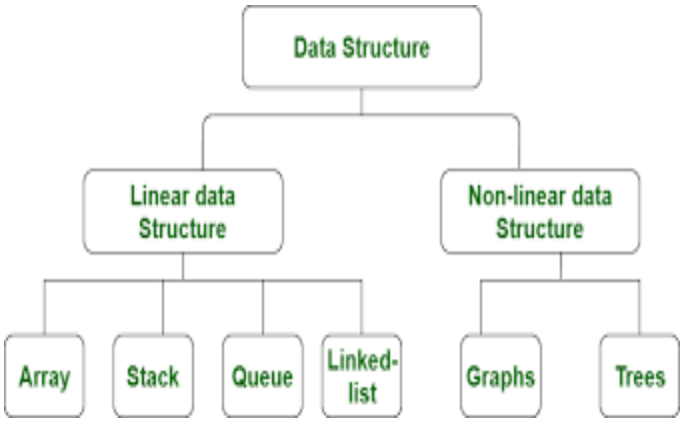

Figure 6: Data Structures

5.1) Why Data structures is important . and what happens if it is not present?

Ans) The main purpose of data structure is it is efficient and data persistence. They provide quick data retrieval, dynamic data updates, and structured data storage, all of which improve code efficiency. Understanding and using the right data structure, whether it is a set, dictionary, tuple, or list, may greatly enhance the readability and efficiency of our Python code. (Medium, 2023)

5.2) Linear Data Structures

Sequentially, the data is kept in linear data structures. Since no mathematical operations are used and the elements are stored one after the other, these are simple structures.

Although linear data structures are typically simple to create, time and space complexity rises when memory allocation becomes complex. (tutorialspoint, 2024)

5.2.1) Arrays

A data structure with a fixed capacity to hold several components of the same kind is called an array. Programmers can organize and manage data collections more effectively by using arrays, which store several values in a single variable rather of creating distinct variables for each item. By default in ArrayList its capacity to hold the value is 10. We create a arraylist to store a numerous values. (Simplilearn, 2024)

5.2.2) Linked Lists

Python offers an abstract data structure called linked lists, which link one list node to another, as a means of organizing data. Data addition and deletion are made simpler because the index of other items in the list doesn't change. Some of the key Methods are:

Insert() , find(), remove(), isempty(). (builtIn, 2024) .

5.2.3) Stacks

Stack is a linear data structure that operates according to the Last In First Out (LIFO) principle. This implies that the element that enters the stack initially emerges last. "Push" is the term we use to add elements to a stack, and "Pop" is the term we use to remove components from a stack. Thus, we may state that pushing and popping cannot occur at the same time in a stack because it only has one open end. Below is a visual depiction of the PUSH and POP operations in the stack:

Some of the examples of methods of Stacks are:

i) Push () :An element can be added to the stack using the user-defined stack method push(n). Its argument passes the element to be pushed.

ii) Pop() : To remove the top element from the stack, we need to use the pop() function.

iii) Isempty() : To determine if the stack is empty or not, we require the isempty() function.

iv) Size() : To determine the size of the stack, we require the size() method.

(GreatLearning, 2022)

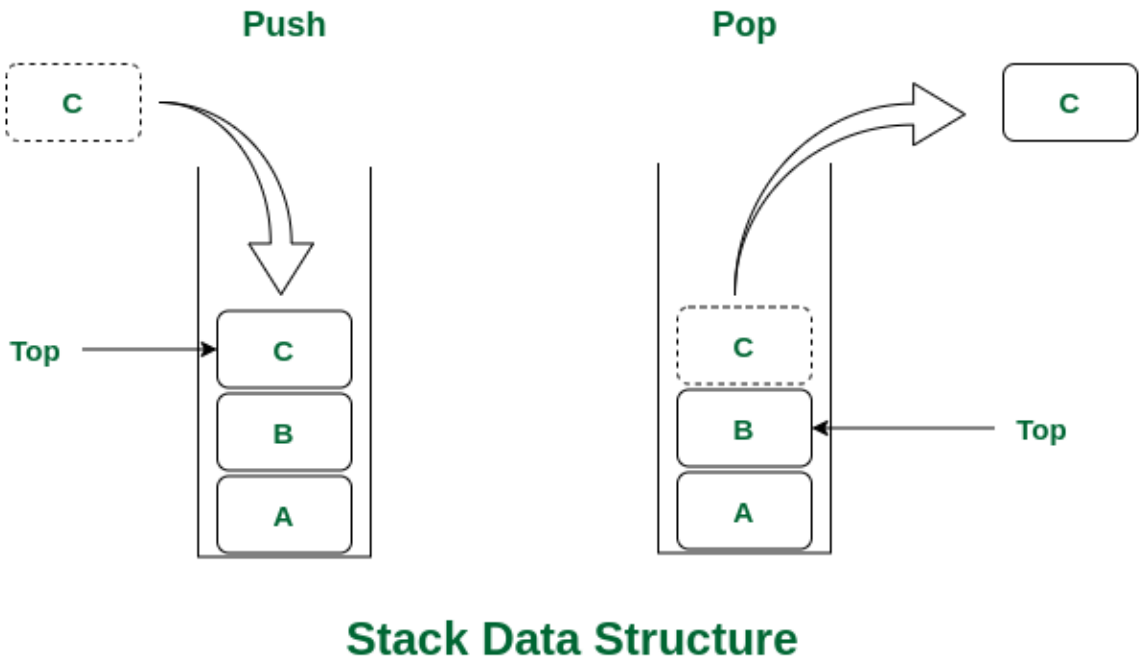

Figure 7: Stack Data Structure

5.2.4) Queues

It is kind of similar to Stack, but when adding and removing elements in Python a queue is a type of data structure that permits first-in, first-out (FIFO) ordering, where the item added to the queue will be the item removed first. (shikshalearn, 2024)

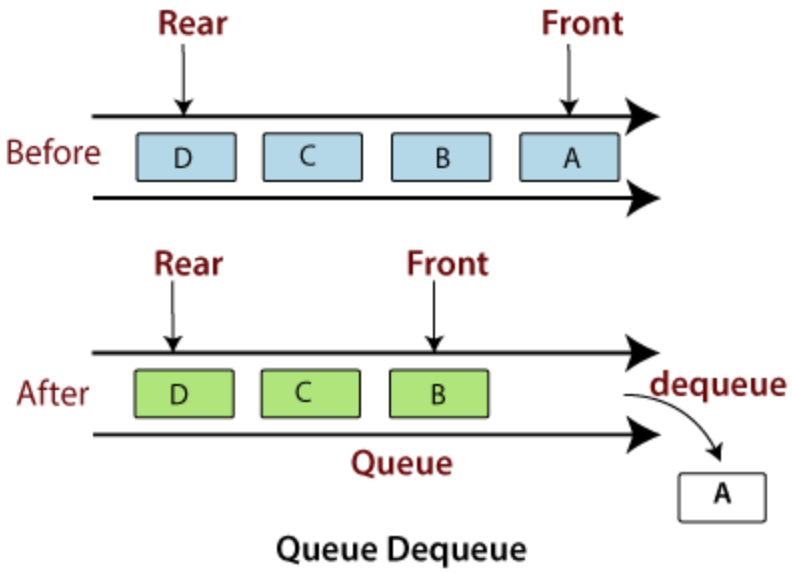

Figure 8 : Queue Data Structure

5.2.5) Lists

In Python, an ordered series of items with changing elements is called a list. An item is any element or value that is contained within a list. Lists are defined as values enclosed in square brackets, much as strings are defined as characters enclosed in quotations [ ]. (DigitalOcean, 2021) Here, I have used the empty list to store the entire information to add in the list. It can hold the multiple data of same data types.

```
def display_stock(furniture_list):
    '''Function to display the available furniture stock from the inventory'''
    print("Available Furniture:")
    print(f"{'ID':<5} {'Manufacturer':<30} {'Name':<20} {'Quantity':<10} {'Price':<10}")
    print("="*80)
    for item in furniture_list:
        print(f"{item['id']:<5} {item['manufacturer']:<30} {item['name']:<20} {item['quantity']:<10} ${item['price']:<10.2f}")
    print("="*80)
    print("\n")
```

Figure 9: Proof of using the Lists data structures

5.2.6) Tuples

Tuples are immutable ordered collections of items that are similar to Python lists. This suggests that once a tuple is formed, it cannot be changed.Commas are used to divide the items, while parentheses() are used to define tuples. For example, they assist in

storing collections of linked data or constants that shouldn't be changed. (ScholarHat, 2024)

5.3) Non-Linear Data Structure

The structure of the data elements of non-linear data structures is different from that of linear data structures in that they are not consecutive. These data structures store objects in a hierarchical or network-based form that isn't sequentially organized. These data structures enable the addition, removal, and searching of elements from the structure.

Examples : i) Trees ii) Graphs (PrepBytes, 2023)

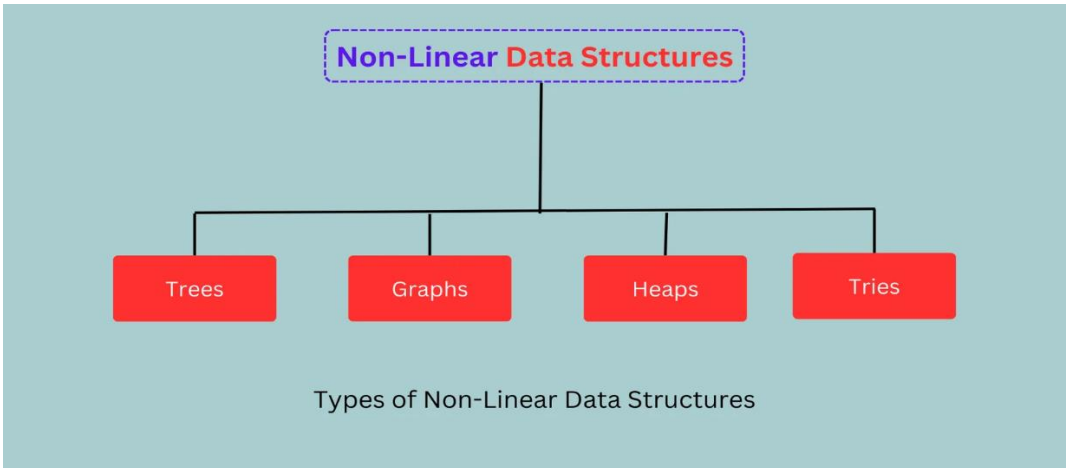

Figure 10 : Non-Linear Data Structure

5.3.1) Dictionary

Using a Python dictionary, the data is kept in a key-value pair format. A dictionary is a data type in Python that may resemble an actual data arrangement where a given value exists for a given key. A dictionary is defined by its values and keys. A key can only consist of one component. It is immutable. (Scaler, 2022)

It is represented by the {} curly bracket.

Why use a dictionary? We use dictionary because it allows us to store the data like itemId, manufacturer, name, quantity , price, etc. Because of its unique key we can easily

access and manage the data in the proper way. It will be more readable and easy to understand what each of the data represents.

What happens if we don't use a Dictionary? Without using dictionary the main problem is that it will be less readable, and there will be high chance of facing errors.

```
# Creating new item dictionary
new_item = {
    'id': item_id,
    'manufacturer': manufacturer,
    'name': name,
    'quantity': quantity,
    'price': price
}
```

Figure 11: Proof of using data dictionary

5.3.2) Sets

Set is a Python data type that allows several things to be stored in a single variable. We may add or delete elements from a set, making it changeable. Mathematical sets and sets in Python are comparable, and operations such as union, symmetric difference, intersection, and more may be performed on them. Curly brackets { } are used to write it.

(SimpliLearn, 2024)

5.3.3) Strings

Strings is represented in the textual form. A series of characters surrounded in double quotes ("") or single quotes (') is called a string. To work with strings effectively, the Python language comes with a number of built-in functions and methods. (Medium, 2023) It supports various operations like concatenation, formatting and slicing. It is mostly used for the printing the messages. It is lessly used in the program.

```
print("\n")
print("""
      ********************************************************************************************************************************************************
#     *                                                BRJ Furniture Store                                                                                   *
#     *                                                  Tarahara, Itahari, Sunsari                                                                         *
#     ********************************************************************************************************************************************************
#     *                                                                                                                                                      *
#     *                                  Welcome to the BRJ Furniture Store..!!!                                                                             *
#     *                                                                                                                                                      *
#     *            We have a wide range of furniture items available, perfect for making your choice.                    *                                     *
#     *                                                                                                                                                      *
#     ********************************************************************************************************************************************************
# """)
print("\n")
```

Figure 12: Proof of using the String data structure

5.3.4) Integer

It can either be positive, negative or zero. It is immutable in case of python. It is mostly used while converting the data type of any integer value to actual integer value.

```
    while True:
        try:
            '''Enter the input for item ID and quantity to purchase'''
            item_id = int(input("Enter furniture ID to purchase: "))
            quantity = int(input("Enter the amount of quantity to purchase: "))
            if quantity <= 0:
                raise ValueError("Quantity must be a positive number.")
        except ValueError as e:
            print(f"Invalid input. {e}")
            continue
```

Figure 13: Proof of using the integer data structures

6) Program

The provided code represents the programs that allows users to view, buy, sell from the “BRJ Furniture Store”. Here is the proper description of how the program works. There are total 5 different files: i)Main.py ii)Operation.py iii)Read.py iv) Write.py v)invoice.py

Here, if the user buy sell the products from the inventory it will update the current txt file.

6.1) Main.py

This is the main entry module for the program. At first, it will display the proper welcome messages. Then, after it will display the menu with five options. After that, it asks the user to choose the number between 1- 5 according to their choices. Now, based on the the

user choices the program works. For instance, if user click 1 and enter , it will display all the items present in the inventory. If choose is 2 , it will ask the employee details to buy the furniture items from the manufacturer. If choose is 3, it will ask the customer's details to place the order of the furniture item's to their respective locations. If user click 5 then it will exit the program by displaying some thank you or goodbye messages. Or by any chance if we click any invalid number format then it will display some error messages.

```
def main(): #The main function that starts the program
    file_name = 'furniture.txt'
    while True: #Until the user decides to exit, the loop will keep displaying menu options.
        print("BRJ Furniture Management Store")
        print("-" * 60)
        print("Option             |         Description")
        print("-" * 60)
        print("Click 1 and Enter: | Show all available items in the store")
        print("Click 2 and Enter: | Purchase from Manufacturer")
        print("Click 3 and Enter: | Sell items to Customer")
        print("Click 4 and Enter: | Add new items to the inventory")
        print("Click 5 and Enter: | Exit")
        print("-" * 60)
        print("\n")

        #Choosing options from the menu
        choice = input("Enter your choice (1-5):\n ")

        if choice == '1':
            furniture_list = read_furniture_data(file_name) # Read and show all the available items afrom the inventory
            display_stock(furniture_list)
        elif choice == '2':
            purchase_furniture(file_name)   #Purchasing items from the manufacturer
        elif choice == '3':
            sell_furniture(file_name)   #Selling items to the customer's
        elif choice == '4':
            add_new_item(file_name) #Adding new items in the inventory
        elif choice == '5':
            print("Thank you for visiting the shop..Hope to see you soon..!!!")
            break   #Closing the program and breaking the loop
        else:
            print("Error: Invalid number. Please try again with valid numbers.")

if __name__ == "__main__":
    main()  #Calling the main function
```

Figure 14: Main function

6.2) Operation.py

This "find_items" function main purpose is to find the items of the furniture's unique ID from the list. It takes two inputs from the list of furniture items. Using the for in loop and it goes through each items in the list by one by one, by checking the item's ID matches the item_ID or not that the user has provided. If it finds the entered input then it will return the item. If it didn't find the item's ID then it will return None value.

```python
def find_items(furniture_list, item_id):
    '''Function define to find an item in the furniture list by its unique ID'''
    for item in furniture_list: #Loop through each items
        if item['id'] == item_id:
            return item
    return None
```

Figure 15: Function to find-items

The “display_stock” function is used to show a list of all the available furniture in the inventory. Now, the function goes through each item in the inventory and prints out its details, including name of the furniture, and the quantity available. For proper view I have added some space below the output.

```python
def display_stock(furniture_list):
    '''Function to display the available furniture stock from the inventory'''
    print("Available Furniture:")
    print(f"{'ID':<5} {'Manufacturer':<30} {'Name':<20} {'Quantity':<10} {'Price':<10}")
    print("="*80)
    for item in furniture_list:
        print(f"{item['id']:<5} {item['manufacturer']:<30} {item['name']:<20} {item['quantity']:<10} ${item['price']:<10.2f}")
    print("="*80)
    print("\n")
```

Figure 16: Function to define display_stock

The “Purchase_furniture” function is used to handle the process of buying the items from the furniture inventory. At first, the current inventory is first read from the file, and creating a empty lists which name is cart. The program first asks the employee to enter their name before entering into the loop and prompts them to enter the furnitureID and quantity. It ensures that the quantity entered by the user is positive or not. Once the ID and quantity are entered, the find_items functions start to search the inventory for the matching item. If the item is located, then the user will be notified that it has been added and the item and the quantity are added to the cart list. When items are added to the cart, the function asks the delivery address. The user is prompted with a shipping cost and if the user enter the Itahari input then the shipping cost will be free. Now, the function calculates the VAT and add the total cost of the items. After entering all the inputs fields. Finally, the updated inventory is written back to the file.

```
def purchase_furniture(file_name):
    '''Function creating to manage furniture purchases from the manufacturers'''
    furniture_list = read_furniture_data(file_name)
    cart = []   #Creating empty List
    employee_name = input("Enter your Name: ")

    while True:
        try:
            '''Enter the input for item ID and quantity to purchase'''
            item_id = int(input("Enter furniture ID to purchase: "))
            quantity = int(input("Enter the amount of quantity to purchase: "))
            if quantity <= 0:
                raise ValueError("Quantity must be a positive number.")
        except ValueError as e:
            print(f"Invalid input. {e}")
            continue

        # Finding the item by it's unique ID
        item = find_items(furniture_list, item_id)

        if item:
            cart.append({'item': item, 'quantity': quantity})
            print(f"The {quantity} amount of item has been added from the {item['manufacturer']} to this Product ID {item_id}.")
        else:
            print(f"The entered {item_id} item ID is not found.")

        while True:
            try:
                # Asking the user whether they want to add more items or not
                continue_choice = input("Do you want to add more items to your Furniture store? (yes/no): ").strip().lower()
                if continue_choice not in ['yes', 'no']:
                    raise ValueError("Please enter 'yes' or 'no'.")
                break
            except ValueError as e:
                print(f"Invalid input. {e}")

        if continue_choice == 'no':
            break   #Exiting the loop

    if cart:
        '''Input to check if user wants to add shipping cost to their items or not'''
        address = input("Enter the address for delivery: ").strip()
        shipping_cost = 0
```

```
    if address.lower() != "itahari":
        #If the address does not match then it will ask the user about shipping cost
        while True:
            add_shipping = input("Do you want to add shipping cost to your receipt? (yes/no): ").strip().lower()
            if add_shipping in ['yes', 'no']:
                break
            else:
                print("Error: Please enter 'yes' or 'no'.")

        if add_shipping == 'yes':
            while True:
                try:
                    # Input the shipping cost for the items
                    shipping_cost = float(input("Enter the shipping cost: "))
                    if shipping_cost < 0:
                        raise ValueError("Shipping cost must be a non-negative number.")
                    break
                except ValueError as e:
                    print(f"Invalid input. {e}")
    else:
        print("Free shipping within Itahari.")  #Free shipping within the Itahari Area

    # Calculating VAT and total amount while purchasing the items
    vat_rate = 0.13
    vat_amount = 0
    total_amount = 0

    for entry_of_item in cart:
        item = entry_of_item['item']
        quantity = entry_of_item['quantity']
        total_amount += item['price'] * quantity
        vat_amount += (item['price'] * quantity) * vat_rate

    shipping_display = f"Shipping Cost: ${shipping_cost:.2f}" if address.lower() != "itahari" else "Free Shipping"

    # Generating the invoice for the purchase
    generate_invoice(cart, employee_name, shipping_display, vat_amount, "Manufacture Order Bill", shipping_cost, address.lower() != "itahari", "purchase")

    # Writing the updated furniture list to the 'txt' file
    write_to_file(file_name, furniture_list)
    print("Order has been placed successfully.")
else:
    print("No products were chosen.")
```

Figure 17: Function to define purchase_furniture

The “sell_furniture” function is used to handle the process of selling the items from the furniture inventory. At first, the current inventory is first read from the file, and creating a empty lists which name is cart. The program first asks the employee to enter their name before entering into the loop and prompts them to enter the furnitureID and quantity. It ensures that the quantity entered by the user is positive or not. Once the ID and quantity are entered, the find_items functions start to search the inventory for the matching item.

If the item is located, then the user will be notified that it has been added and the item and the quantity are added to the cart list. When items are added to the cart, the function asks the delivery address. The user is prompted with a shipping cost and if the user enter the Itahari input then the shipping cost will be free. Now, the function calculates the VAT and add the total cost of the items. After entering all the inputs fields. Finally, the updated inventory is written back to the file.

```
def sell_furniture(file_name):
    '''Function creating to sell the furniture items from the inventory'''
    furniture_list = read_furniture_data(file_name)
    cart = []   #Creating empty List
    customer_name = input("Enter customer name: ")

    while True:
        try:
            '''Enter the input for item ID and quantity to sell'''
            item_id = int(input("Enter furniture ID to sell: "))
            quantity = int(input("Enter the amount of quantity to sell: "))
            if quantity <= 0:
                raise ValueError("Quantity must be a positive number.")
        except ValueError as e:
            print(f"Invalid input. {e}")
            continue

        # Find the item by it's unique ID
        item = find_items(furniture_list, item_id)
        if item and item['quantity'] >= quantity:
            cart.append({'item': item, 'quantity': quantity})
            item['quantity'] -= quantity
            print(f"The {item['manufacturer']} has sold {quantity} amount of quantity to {customer_name}.")
        else:
            print(f"The entered ID {item_id} is not found or no more items left.")

        while True:
            try:
                # Asking the user if they want to add more items or not
                continue_choice = input("Do you want to add more items? (yes/no): ").strip().lower()
                if continue_choice not in ['yes', 'no']:
                    raise ValueError("Please enter 'yes' or 'no'.")
                break
            except ValueError as e:
                print(f"Invalid input. {e}")

        if continue_choice == 'no':
            break   #Exiting the loop

    if cart:
        '''Input to check if user wants to add shipping cost to their items or not'''
        address = input("Enter the address for delivery: ").strip()
        shipping_cost = 0
```

```
        if address.lower() != "itahari":
            #If the address does not match then it will ask the user about shipping cost
            while True:
                add_shipping = input("Do you want to add shipping cost to the total amount? (yes/no): ").strip().lower()
                if add_shipping in ['yes', 'no']:
                    break
                else:
                    print("Error: Please enter 'yes' or 'no'.")

            if add_shipping == 'yes':
                while True:
                    try:
                        # Input the shipping cost for the items
                        shipping_cost = float(input("Enter shipping cost: "))
                        if shipping_cost < 0:
                            raise ValueError("Shipping cost must be a non-negative number.")
                        break
                    except ValueError as e:
                        print(f"Invalid input. {e}")
        else:
            print("Free shipping within Itahari.")   #Free shipping within the Itahari Area

        # Calculating VAT and total amount while selling the items
        vat_rate = 0.13
        vat_amount = 0
        total_amount = 0

        for entry_of_item in cart:
            item = entry_of_item['item']
            quantity = entry_of_item['quantity']
            total_amount += item['price'] * quantity
            vat_amount += (item['price'] * quantity) * vat_rate

        shipping_display = f"Shipping Cost: ${shipping_cost:.2f}" if address.lower() != "itahari" else "Free Shipping"

        # Generating the invoice for the sell
        generate_invoice(cart, customer_name, shipping_display, vat_amount, "Sales Bill", shipping_cost, address.lower() != "itahari", "Sale")

        # Writing the updated furniture list to the 'txt' file
        write_to_file(file_name, furniture_list)
        print("Item has been sold successfully.")
    else:
        print("There were no products chosen for selling.")
```

Figure 18: Function for selling products

The add_new_item function is designed to add a new item of furniture in the inventory. The inventory data is first read from a file into a list. After that, the function asks the user to provide information about the new item of furniture, such as it's ID, name, manufacturer, quantity, and price.

6.3) Write.py

```
import datetime


def write_to_file(file_name, furniture_list):
    # The updated furniture list is written by this function to the appropriate file.
     with open(file_name, 'w') as file:  #Loop through each items
          for item in furniture_list:
              file.write(f"{item['id']}, {item['manufacturer']}, {item['name']}, {item['quantity']}, ${item['price']:.2f}\n")
```

Figure 19: Function for write_to_file

A write_to_file function saves the furniture inventory to a designated file. The program opens the file inwrite mode, iterates through the list of each furniture items. Any previous content in the file is replaced with the new inventory.

6.4) Read.py

```
def read_furniture_data(filename):
     # It will read the furniture_list
    furniture_list = [] # Creating List to store data
    try:
        # File will open in read mode 'r'
        with open(filename, 'r') as file:
            for line in file:   #Loop through each line
                if line.strip():
                    elements = line.strip().split(', ')

                    #Creating a dictionary for each furniture items
                    furniture = {
                    'id': int(elements[0]),
                    'manufacturer': elements[1],
                    'name': elements[2],
                    'quantity': int(elements[3]),
                    'price': float(elements[4].replace('$', ''))
                    }
                    furniture_list.append(furniture)    #Adding the dictionary items to the List
    except FileNotFoundError:
        print(f"Error: File '{filename}' not found.")
    except Exception as e:
        print(f"An error occurred while loading data: {e}")
    return furniture_list
```

Figure 20: Function define for read_furniture

The read_furniture_data function reads furniture data from a designated file. It opens the file in the read mode and then stored in a list. It also displays some error messages if error is detected while reading the program.

6) Testing

=> This is one of the most crucial part of the program because we will be facing a lot of problems during writing the program. So, in this part we will be discussing about the test which I faced the error and how I handled it.

6.1) Testing 1 : Exception Handling and Input Validation

Objective	Implementing Try Except
Action	For instance, if I enter the any alphabetical word “a”, “b” or any negative value ..
Expected Result	It will display the appropriate error messages. “Error: Invalid number. Please try again with valid numbers” & Quantity must be a positive number.
Actual Result	The appropriate messages will display in our terminal.
Conclusion	Test has been successful.

Table 1: Implementing Try Except

```
BRJ Furniture Management Store
------------------------------------------------------------
Option              |         Description
------------------------------------------------------------
Click 1 and Enter:  | Show all available items in the store
Click 2 and Enter:  | Purchase from Manufacturer
Click 3 and Enter:  | Sell items to Customer
Click 4 and Enter:  | Add new items to the inventory
Click 5 and Enter:  | Exit
------------------------------------------------------------


Enter your choice (1-5):
 2
Enter your Name: Adit
Enter furniture ID to purchase: 3
Enter the amount of quantity to purchase: -40
Invalid input. Quantity must be a positive number.
Enter furniture ID to purchase: tmg
Invalid input. invalid literal for int() with base 10: 'tmg'
Enter furniture ID to purchase: -3
Enter the amount of quantity to purchase: 2
The entered -3 item ID is not found.
Do you want to add more items to your Furniture store? (yes/no): 2
Invalid input. Please enter 'yes' or 'no'.
Do you want to add more items to your Furniture store? (yes/no): yes
Enter furniture ID to purchase: 50
Enter the amount of quantity to purchase: 2
The entered 50 item ID is not found.
```

Figure 21: Displaying appropriate messages

6.2) Testing 2: User input validation and menu navigation

Objective	Display all the inventory of the Furniture.
Action	While we click the -1 and any alphabetical value. It displays some messages. But after we enter the correct value, it displays all the inventory of the furniture.
Expected Result	If we enter any value except the positive numeric value till number 5 it should display some error messages. But after entering the proper value it should display all the furniture inventory.

Actual Result	It shows a appropriate messages while entering the incorrect value and also displays a furniture inventory after entering the proper numeric values.
Conclusion	Test has been successful.

Table 2: Test to display the inventory

```
Enter your choice (1-5):
 -1
Error: Invalid number. Please try again with valid numbers.
BRJ Furniture Management Store
------------------------------------------------------------
Option              |         Description
------------------------------------------------------------
Click 1 and Enter:  | Show all available items in the store
Click 2 and Enter:  | Purchase from Manufacturer
Click 3 and Enter:  | Sell items to Customer
Click 4 and Enter:  | Add new items to the inventory
Click 5 and Enter:  | Exit
------------------------------------------------------------


Enter your choice (1-5):
 adit
Error: Invalid number. Please try again with valid numbers.
BRJ Furniture Management Store
------------------------------------------------------------
Option              |         Description
------------------------------------------------------------
Click 1 and Enter:  | Show all available items in the store
Click 2 and Enter:  | Purchase from Manufacturer
Click 3 and Enter:  | Sell items to Customer
Click 4 and Enter:  | Add new items to the inventory
Click 5 and Enter:  | Exit
------------------------------------------------------------


Enter your choice (1-5):
 40
Error: Invalid number. Please try again with valid numbers.
```

Figure 22: Displaying error messages while entering the invalid format while viewing the inventory

```
#     ************************************************************************************************************************************************************
#     *                                                BRJ Furniture Store                                                                                       *
#     *                                                   Tarahara, Itahari, Sunsari                                                                             *
#     ************************************************************************************************************************************************************
#     *                                                                                                                                                        *
#     *                                    Welcome to the BRJ Furniture Store..!!!                                                                             *
#     *                                                                                                                                                        *
#     *            We have a wide range of furniture items available, perfect for making your choice.                    *                                          *
#     *                                                                                                                                                        *
#     ************************************************************************************************************************************************************


BRJ Furniture Management Store
------------------------------------------------------------
Option             |         Description
------------------------------------------------------------
Click 1 and Enter: | Show all available items in the store
Click 2 and Enter: | Purchase from Manufacturer
Click 3 and Enter: | Sell items to Customer
Click 4 and Enter: | Add new items to the inventory
Click 5 and Enter: | Exit
------------------------------------------------------------


Enter your choice (1-5):
 1
+----+-----------------------------------------+------------------------+----------+------------+
| ID |  Manufacturer                           |          Name          | Quantity |   Price    |
+----+-----------------------------------------+------------------------+----------+------------+
| 1  | HNI Corporation                         | Bunk Bed               | 115      | $400.00    |
| 2  | HNI CorporationHaworth Inc.             | Twin Bed               | 478      | $600.00    |
| 3  | Achham furniture                        | Sleeper Sofa           | 1100     | $200.00    |
| 4  | Kimball International Inc.              | Corner sofa            | 139      | $350.00    |
| 5  | Kohler Co.                              | Armchair               | 130      | $150.00    |
| 6  | Masco Corporation                       | Desk chair             | 289      | $100.00    |
| 7  | Gorkha                                  | bed                    | 112      | $675.00    |
| 8  | Itahari Furniture                       | CHair                  | 901      | $65.00     |
| 9  | Khotange Inc                            | table                  | 950      | $40.00     |
| 10 | Furni World                             | Cupboard               | 99       | $450.00    |
+----+-----------------------------------------+------------------------+----------+------------+
```

Figure 23: Displaying all the Furniture Inventory

6.3) Testing 3: File Handling to update the inventory after purchasing

Objective	Check the available of furniture after buying the items from the inventory.
Action	The availability of the items should change after buying the items.
Expected Result	The items should be updated after each operation.
Actual Result	The program executed smoothly.
Conclusion	Test has been successful.

Table 3: Updating the Availability

```
5     Kohler Co.                    Armchair            40          $150.00
```

Figure 24: Items before purchasing

```
Enter your choice (1-5):
 2
Enter your Name: Adit
Enter furniture ID to purchase: 5
Enter the amount of quantity to purchase: 62
The 62 amount of item has been added from the Kohler Co. to this Product ID 5.

5     Kohler Co.                    Armchair            102         $150.00
```

Figure 25: Items after purchasing

6.4) Testing 4: File Handling to update the inventory after selling

Objective	Check the available of furniture after selling the items from the inventory.
Action	The availability of the items should change after selling the items.
Expected Result	The items should be updated after each operation.
Actual Result	The program executed smoothly.
Conclusion	Test has been successful.

Table 4: Updating the availability

```
9     Khotange Inc                  table               991         $40.00
```

Figure 26: Items before Selling

```
Enter your choice (1-5):
 3
Enter customer name: adit
Enter furniture ID to sell: 9
Enter the amount of quantity to sell: 91
The Khotange Inc has sold 91 amount of quantity to adit.
```

```
9     Khotange Inc                  table                 900         $40.00
```

Figure 27: Items after selling the products

6.5) Testing 5: Trying to sell more quantity than the available items.

Objective	Entering more quantity than available in the inventory.
Action	Entering the items quantity that is not in in the inventory and it should display error messages.
Expected Result	It should display some error messages. That the items are no more left to sell.
Actual Result	It display some messages.
Conclusion	Test has been successful.

Table 5: Trying to sell more than it's available

```
BRJ Furniture Management Store
------------------------------------------------------------
Option              |         Description
------------------------------------------------------------
Click 1 and Enter:  | Show all available items in the store
Click 2 and Enter:  | Purchase from Manufacturer
Click 3 and Enter:  | Sell items to Customer
Click 4 and Enter:  | Add new items to the inventory
Click 5 and Enter:  | Exit
------------------------------------------------------------


Enter your choice (1-5):
 1
+----+-----------------------------------------+-------------------------+----------+------------+
| ID |  Manufacturer                           |          Name           | Quantity |   Price    |
+----+-----------------------------------------+-------------------------+----------+------------+
| 1  | HNI Corporation                         | Bunk Bed                | 116      | $400.00    |
| 2  | HNI CorporationHaworth Inc.             | Twin Bed                | 478      | $600.00    |
| 3  | Achham furniture                        | Sleeper Sofa            | 1098     | $200.00    |
| 4  | Kimball International Inc.              | Corner sofa             | 140      | $350.00    |
| 5  | Kohler Co.                              | Armchair                | 135      | $150.00    |
| 6  | Masco Corporation                       | Desk chair              | 292      | $100.00    |
| 7  | Gorkha                                  | bed                     | 116      | $675.00    |
| 8  | Itahari Furniture                       | CHair                   | 906      | $65.00     |
| 9  | Khotange Inc                            | table                   | 952      | $40.00     |
| 10 | Furni World                             | Cupboard                | 106      | $450.00    |
| 11 | Hare Krishna Furniture Store            | Study Table             | 21       | $50.00     |
+----+-----------------------------------------+-------------------------+----------+------------+
```

```
Enter your choice (1-5):
 3
Enter customer name: pakhrin
Enter furniture ID to sell: 11
Enter the amount of quantity to sell: 25
The entered ID 11 is not found or no more items left.
```

Figure 28: Trying to sell the more items

```
Enter furniture ID to sell: 11
Enter the amount of quantity to sell: 10
The Hare Krishna Furniture Store has sold 10 amount of quantity to pakhrin.
Do you want to add more items? (yes/no): no
Enter the address for delivery: Lalitpur
Do you want to add shipping cost to the total amount? (yes/no): yes
Enter shipping cost: 54
 Thank you.!!! Your Invoice has been generated: Sale_invoice_20240816215807.txt
Item has been sold successfully.
```

Figure 29: Selling the items present in the inventory

6.6) Testing 6: Providing free delivery within the Itahari area.

Objective	Checking the shipping cost for different address and free shipping to Itahari..
Action	Entering the Itahari for free shipping and different address for the shipping cost.
Expected Result	The invoice will be generated without the shipping cost but if we enter the any address except the Itahari then it will ask the user to enter the shipping cost.
Actual Result	It works properly.
Conclusion	Test has been successful.

Table 6: Free delivery charge

```
BRJ Furniture Management Store
------------------------------------------------------------
Option             |         Description
------------------------------------------------------------
Click 1 and Enter: | Show all available items in the store
Click 2 and Enter: | Purchase from Manufacturer
Click 3 and Enter: | Sell items to Customer
Click 4 and Enter: | Add new items to the inventory
Click 5 and Enter: | Exit
------------------------------------------------------------


Enter your choice (1-5):
 2
Enter your Name:
Error: Name cannot be empty and must contain only alphabets without any special characters or numbers.
Enter your Name: adit
Enter furniture ID to purchase: 7
Enter the amount of quantity to purchase: 1
The 1 amount of item has been added from the Gorkha to this Product ID 7.
Do you want to add more items to your Furniture store? (yes/no): no
Enter the address for delivery: Itahari
Free shipping within Itahari.
 Thank you.!!! Your Invoice has been generated: purchase_invoice_20240816172905.txt
Order has been placed successfully.
```

Figure 30: Checking by entering the Itahari address

```
BRJ Furniture Management Store
------------------------------------------------------------
Option             |         Description
------------------------------------------------------------
Click 1 and Enter: | Show all available items in the store
Click 2 and Enter: | Purchase from Manufacturer
Click 3 and Enter: | Sell items to Customer
Click 4 and Enter: | Add new items to the inventory
Click 5 and Enter: | Exit
------------------------------------------------------------


Enter your choice (1-5):
 3
Enter customer name: Tanka
Enter furniture ID to sell: 4
Enter the amount of quantity to sell: 2
The Kimball International Inc. has sold 2 amount of quantity to Tanka.
Do you want to add more items? (yes/no): no
Enter the address for delivery: Tarahara
Do you want to add shipping cost to the total amount? (yes/no): yes
Enter shipping cost: 88
 Thank you.!!! Your Invoice has been generated: Sale_invoice_20240816173023.txt
Item has been sold successfully.
```

Figure 31: Checking by different address except Itahari

6.7) Testing 7: Control Program Flow

Objective	Checking whether the program will exit or not if the user clicks the exit button.
Action	Pressing the exit button.
Expected Result	The program should end if user clicks the exit button.
Actual Result	The program ended.
Conclusion	Test has been successful.

Table 7: Choosing 5 to exit the program

```
BRJ Furniture Management Store
------------------------------------------------------------
Option              |         Description
------------------------------------------------------------
Click 1 and Enter:  | Show all available items in the store
Click 2 and Enter:  | Purchase from Manufacturer
Click 3 and Enter:  | Sell items to Customer
Click 4 and Enter:  | Add new items to the inventory
Click 5 and Enter:  | Exit
------------------------------------------------------------


Enter your choice (1-5):
 5
Thank you for visiting the shop..Hope to see you soon..!!!
```

Figure 32: Ending the program

6.8) Testing 8: Invoice generation after buying the multiple items

Objective	Buying multiple products from the manufacturer. The program generate only single receipt.
Action	Employees fill the entire details to buy the multiple items and it will generate in the single receipt with the entire details and total amount of the items.
Expected Result	The employee must be able to buy the multiple products from the manufacturers and should only generate the single receipt.
Actual Result	It generates the single invoice with employee entire details.
Conclusion	Test has been successful.

Table 8: Purchasing Multiple Products receipt

```
BRJ Furniture Management Store
-------------------------------------------------------------
Option             |         Description
-------------------------------------------------------------
Click 1 and Enter: | Show all available items in the store
Click 2 and Enter: | Purchase from Manufacturer
Click 3 and Enter: | Sell items to Customer
Click 4 and Enter: | Add new items to the inventory
Click 5 and Enter: | Exit
-------------------------------------------------------------


Enter your choice (1-5):
 2
Enter your Name: Kashiraj Tamang
Enter furniture ID to purchase: 1
Enter the amount of quantity to purchase: 3
The 3 amount of item has been added from the HNI Corporation to this Product ID 1.
Do you want to add more items to your Furniture store? (yes/no): yes
Enter furniture ID to purchase: 4
Enter the amount of quantity to purchase: 2
The 2 amount of item has been added from the Kimball International Inc. to this Product ID 4.
Do you want to add more items to your Furniture store? (yes/no): yes
Enter furniture ID to purchase: 6
Enter the amount of quantity to purchase: 3
The 3 amount of item has been added from the Masco Corporation to this Product ID 6.
Do you want to add more items to your Furniture store? (yes/no): yes
Enter furniture ID to purchase: 8
Enter the amount of quantity to purchase: 5
The 5 amount of item has been added from the Itahari Furniture to this Product ID 8.
Do you want to add more items to your Furniture store? (yes/no): yes
Enter furniture ID to purchase: 10
Enter the amount of quantity to purchase: 7
The 7 amount of item has been added from the Furni World to this Product ID 10.
Do you want to add more items to your Furniture store? (yes/no): no
Enter the address for delivery: Kathmandu
Do you want to add shipping cost to your receipt? (yes/no): yes
Enter the shipping cost: 130
 Thank you.!!! Your Invoice has been generated: purchase_invoice_20240816181241.txt
Order has been placed successfully.
```

```
================================================================================================
       BRJ Furniture Store
Tarahara, Itahari, Sunsari
Phone no: 9807373362    Email: BRJfurniture@gmail.com
================================================================================================

Manufacture Order Bill
Name: Kashiraj Tamang
Invoice Number: 20240816181241
Transaction Type: purchase
Date: 2024-08-16 18:12:41

|----------------------------------------------------------------------------------------------|
ID       Manufacturer                    Name                  Quantity    Price
|----------------------------------------------------------------------------------------------|
1         HNI Corporation                 Bunk Bed              3           $1200.00
4         Kimball International Inc.      Corner sofa           2           $700.00
6         Masco Corporation               Desk chair            3           $300.00
8         Itahari Furniture               CHair                 5           $325.00
10        Furni World                     Cupboard              7           $3150.00

================================================================================================
Total Amount (excluding VAT) : $5675.00
VAT Amount: $737.75
Shipping Cost: $130.00
Total Amount (including VAT) : $6542.75
================================================================================================

        Thank you for trusting us. Please visit us again
================================================================================================
            Certified Company, Nepal Government
================================================================================================
```

Figure 33: Receipt of Purchase

6.9) Testing 9: Invoice generation after selling the multiple items

Objective	Selling multiple products from the BRJ Furniture inventory. The program generate only single receipt.
Action	Customers fill the entire details to buy the multiple items and it will generate in the single receipt with the entire details and total amount of the items.
Expected Result	The customer must be able to buy the multiple products from the manufacturers and should only generate the single receipt.
Actual Result	It generates the single invoice with customer entire details.
Conclusion	Test has been successful.

Table 9: Selling multiple Products

```
BRJ Furniture Management Store
------------------------------------------------------------
Option              |         Description
------------------------------------------------------------
Click 1 and Enter:  | Show all available items in the store
Click 2 and Enter:  | Purchase from Manufacturer
Click 3 and Enter:  | Sell items to Customer
Click 4 and Enter:  | Add new items to the inventory
Click 5 and Enter:  | Exit
------------------------------------------------------------


Enter your choice (1-5):
 3
Enter customer name: Ashmika Tamang
Enter furniture ID to sell: 1
Enter the amount of quantity to sell: 2
The HNI Corporation has sold 2 amount of quantity to Ashmika Tamang.
Do you want to add more items? (yes/no): yes
Enter furniture ID to sell: 3
Enter the amount of quantity to sell: 2
The Achham furniture has sold 2 amount of quantity to Ashmika Tamang.
Do you want to add more items? (yes/no): yes
Enter furniture ID to sell: 5
Enter the amount of quantity to sell: 3
The Kohler Co. has sold 3 amount of quantity to Ashmika Tamang.
Do you want to add more items? (yes/no): yes
Enter furniture ID to sell: 9
Enter the amount of quantity to sell: 4
The Khotange Inc has sold 4 amount of quantity to Ashmika Tamang.
Do you want to add more items? (yes/no): yes
Enter furniture ID to sell: 11
Enter the amount of quantity to sell: 9
The Hare Krishna Furniture Store has sold 9 amount of quantity to Ashmika Tamang.
Do you want to add more items? (yes/no): no
Enter the address for delivery: Chitwan
Do you want to add shipping cost to the total amount? (yes/no): yes
Enter shipping cost: 110
 Thank you.!!! Your Invoice has been generated: Sale_invoice_20240816182043.txt
Item has been sold successfully.
```

File Edit Format Run Options Window Help

```
================================================================================================
       BRJ Furniture Store
Tarahara, Itahari, Sunsari
Phone no: 9807373362    Email: BRJfurniture@gmail.com
================================================================================================

Sales Bill
Name: Ashmika Tamang
Invoice Number: 20240816182043
Transaction Type: Sale
Date: 2024-08-16 18:20:43

|----------------------------------------------------------------------------------------------|
ID       Manufacturer                    Name                  Quantity    Price
|----------------------------------------------------------------------------------------------|
1         HNI Corporation                 Bunk Bed              2           $800.00
3         Achham furniture                Sleeper Sofa          2           $400.00
5         Kohler Co.                      Armchair              3           $450.00
9         Khotange Inc                    table                 4           $160.00
11        Hare Krishna Furniture Store    Study Table           9           $450.00

================================================================================================
Total Amount (excluding VAT) : $2260.00
VAT Amount: $293.80
Shipping Cost: $110.00
Total Amount (including VAT) : $2663.80
================================================================================================

        Thank you for trusting us. Please visit us again
================================================================================================
            Certified Company, Nepal Government
================================================================================================
```

Figure 34: Receipt of Selling the items

6.10) Testing 10: Generating new files and record of transactions

Objective	The program should generate the new file after buying or selling the items.
Action	Selling and buying the items to generate the receipt.
Expected Result	The program should generate the invoice after the program ends.
Actual Result	The new invoice file was created for each transaction.
Conclusion	Test has been successful.

Table 10: Proof that new Invoice File is created after buying or selling the items

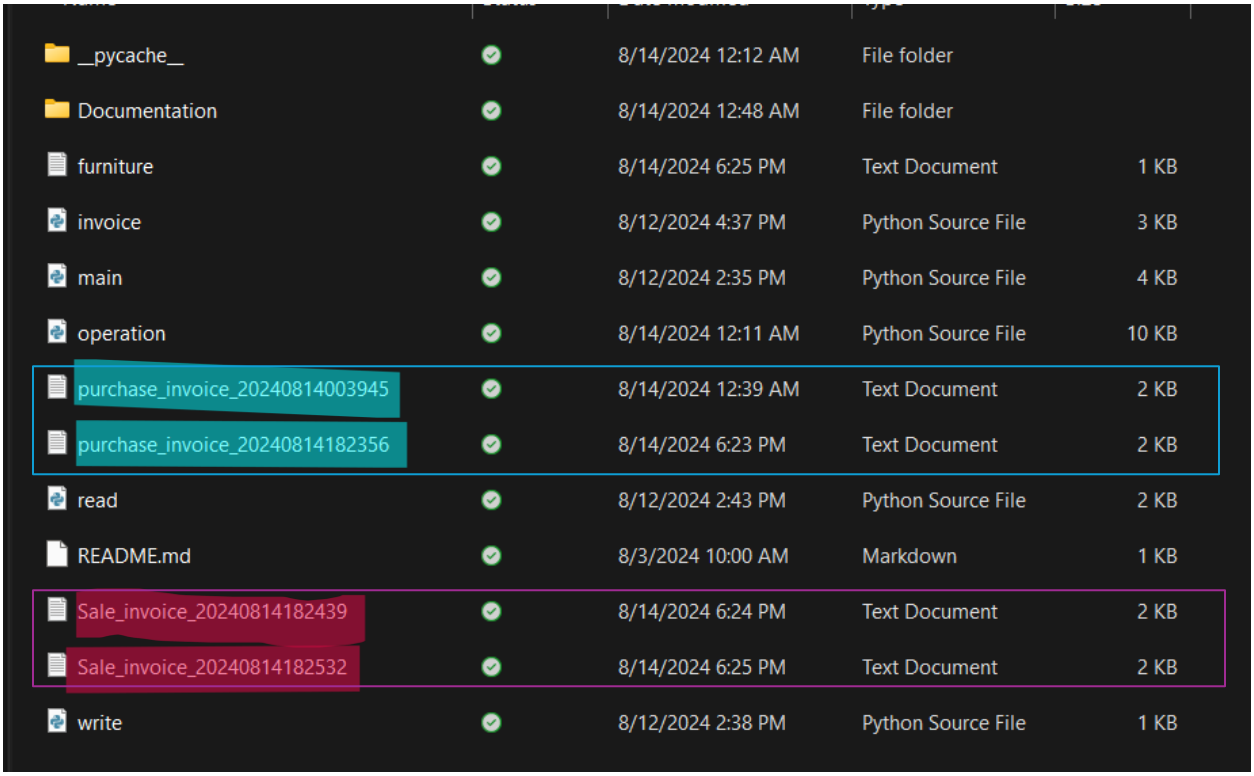

Figure 35: Highlighting the invoice that has been generated

6.11) Test 11: Updating the inventory: File handling

Objective	Adding the items in the inventory.
Action	Items adding in the inventory for more items in the store.
Expected Result	The items should be added in the 'furniture.txt' file and the inventory must be updated with the new items.
Actual Result	The new items added successfully.
Conclusion	Test has been successful.

Table 11:Adding the items in the inventory

```
BRJ Furniture Management Store
------------------------------------------------------------
Option              |         Description
------------------------------------------------------------
Click 1 and Enter:  | Show all available items in the store
Click 2 and Enter:  | Purchase from Manufacturer
Click 3 and Enter:  | Sell items to Customer
Click 4 and Enter:  | Add new items to the inventory
Click 5 and Enter:  | Exit
------------------------------------------------------------


Enter your choice (1-5):
 4
Enter the new furniture ID: 11
Enter the manufacturer: Hare Krishna Furniture Store
Enter the name of the furniture: Study Table
Enter the quantity: 30
Enter the price: 50
New item has been added successfully.
```

Figure 36: Item before adding in the inventory

```
BRJ Furniture Management Store
-------------------------------------------------------------
Option              |         Description
-------------------------------------------------------------
Click 1 and Enter: | Show all available items in the store
Click 2 and Enter: | Purchase from Manufacturer
Click 3 and Enter: | Sell items to Customer
Click 4 and Enter: | Add new items to the inventory
Click 5 and Enter: | Exit
-------------------------------------------------------------


Enter your choice (1-5):
 1
+----+-----------------------------------------+-------------------------+----------+------------+
| ID |  Manufacturer                           |          Name           | Quantity |   Price    |
+----+-----------------------------------------+-------------------------+----------+------------+
| 1  | HNI Corporation                         | Bunk Bed                | 115      | $400.00    |
| 2  | HNI CorporationHaworth Inc.             | Twin Bed                | 478      | $600.00    |
| 3  | Achham furniture                        | Sleeper Sofa            | 1100     | $200.00    |
| 4  | Kimball International Inc.              | Corner sofa             | 137      | $350.00    |
| 5  | Kohler Co.                              | Armchair                | 130      | $150.00    |
| 6  | Masco Corporation                       | Desk chair              | 289      | $100.00    |
| 7  | Gorkha                                  | bed                     | 113      | $675.00    |
| 8  | Itahari Furniture                       | CHair                   | 901      | $65.00     |
| 9  | Khotange Inc                            | table                   | 950      | $40.00     |
| 10 | Furni World                             | Cupboard                | 99       | $450.00    |
| 11 | Hare Krishna Furniture Store            | Study Table             | 30       | $50.00     |
+----+-----------------------------------------+-------------------------+----------+------------+
```

Figure 37: Item added in the inventory

7) Conclusion

Throughout the completion of this coursework, I have extensively involved myself in the coursework of Fundamental Of Computing (FOC), to create a BRJ Furniture by using the python languages. Allowing me to deepen my understanding of Python languages. My goal with this project was to develop a system that could manage different functions like processing sales transactions, adding and removing products, and producing invoices.

I successfully completed my 1st python programming languages, which taught me a lot about handling the files. By participating in the course's curriculum, I was able to identify and tackle difficulties in the application, which enhanced my problem-solving skills. My coding abilities have been greatly improved by the practical use of Python programming languages, especially in managing file I/O operations, putting logic for managing the inventory, arranging the code into modules. In addition, I also gained a experience in resolving a real world challenges like handling the different exceptions. It helped me to broaden my creativity, ideas, and researched based skills.

Now, I've become an expert in using Python's many flexible data structures, such as lists and dictionaries, which I believe will be essential for my next projects. This practical understanding of programming's real-world applications has deepened my view on the role that programming plays across many industries and companies. My desire to learn advanced programming techniques has grown, motivating me to work toward gaining greater understanding of computers.

This coursework not only enhanced my understanding in Python, but also provided me the insightful information about the real world applications. Throughout this coursework, I also faced a lot of problems. After doing this coursework I learned a lot about the

Python languages, Algorithm, FlowChart , File Handling. Moreover, I also learned to managed the time.

To describe it shortly, this coursework was an best opportunity for me to apply the knowledge what I have learned in classes. It forced me to reflect carefully and value the modulation of Python languages.

I sincerely want to thank my module teacher, instructors for all the proper guidance and encouragement, which really helped me to get through this challenging project's. I also want to express my gratitude to my family members for their continuous support and guidance during this journey. Their support has played a crucial role in completing this project without any worries. These fundamental skills have given me the ability to navigate the ever-evolving technical world and contribute significantly to the programming community.

8) Appendix

i) Main.py

```
from read import read_furniture_data

from operation import purchase_furniture, sell_furniture, add_new_item, display_stock


print("\n")

print("""

**********************************************************************************************************
********************************

#    *                                BRJ Furniture Store                                          *

#    *                                 Tarahara, Itahari, Sunsari                                  *

#
**********************************************************************************************************
********************************

#    *                                                                                             *

#    *                                Welcome to the BRJ Furniture Store..!!!

#    *                                                                                             *

#    *            We have a wide range of furniture items available, perfect for making your
choice.                  *

#    *                                                                                             *
```

```
#
**********************************************************************************************************
*********************************
"""")
print("\n")


def main(): #The main function that starts the program
    file_name = 'furniture.txt'
    while True: #Until the user decides to exit, the loop will keep displaying menu options.
        print("BRJ Furniture Management Store")
        print("-" * 60)
        print("Option              |         Description")
        print("-" * 60)
        print("Click 1 and Enter: | Show all available items in the store")
        print("Click 2 and Enter: | Purchase from Manufacturer")
        print("Click 3 and Enter: | Sell items to Customer")
        print("Click 4 and Enter: | Add new items to the inventory")
        print("Click 5 and Enter: | Exit")
        print("-" * 60)
        print("\n")


        #Choosing options from the menu
        choice = input("Enter your choice (1-5):\n ")
```

```
        if choice == '1':
            furniture_list = read_furniture_data(file_name) # Read and show all the available items afrom the inventory
            display_stock(furniture_list)
        elif choice == '2':
            purchase_furniture(file_name)   #Purchasing items from the manufacturer
        elif choice == '3':
            sell_furniture(file_name)   #Selling items to the customer's
        elif choice == '4':
            add_new_item(file_name) #Adding new items in the inventory
        elif choice == '5':
            print("Thank you for visiting the shop..Hope to see you soon..!!!")
            break   #Closing the program and breaking the loop
        else:
            print("Error: Invalid number. Please try again with valid numbers.")


if __name__ == "__main__":
    main()  #Calling the main function
```

ii) Operation.py

```
import re
from read import read_furniture_data
from write import write_to_file
from invoice import generate_invoice


def validate_name(name):
    """Ensure the name contains only alphabets and no special characters or numbers."""
if re.match("^[A-Za-z]+(?: [A-Za-z]+)*$", name):
 #It allows for more spaces in between the names and also ensure that it ends with valid characters
            return True
    return False


def find_items(furniture_list, item_id):
    # Function to find an item in the furniture list by its unique ID
    for item in furniture_list:  # Loop through each item
        if item['id'] == item_id:
            return item
    return None
```

```
def display_stock(furniture_list):
    # Function to display the available furniture stock from the inventory
    print("Available Furniture:")
    print(f"{'ID':<5} {'Manufacturer':<30} {'Name':<20} {'Quantity':<10} {'Price':<10}")
    print("="*80)
    for item in furniture_list:
                        print(f"{item['id']:<5}      {item['manufacturer']:<30}      {item['name']:<20}
{item['quantity']:<10} ${item['price']:<10.2f}")
    print("="*80)
    print("\n")


def purchase_furniture(file_name):
    # Function to manage furniture purchases from the manufacturers
    furniture_list = read_furniture_data(file_name)
    cart = []  # Create empty list


    while True:
        employee_name = input("Enter your Name: ").strip()
        if employee_name and validate_name(employee_name):
            break
         print("Error: Name cannot be empty and must contain only alphabets without any
special characters or numbers.")
```

```
    while True:
        try:
            item_id = int(input("Enter furniture ID to purchase: "))
            quantity = int(input("Enter the amount of quantity to purchase: "))
            if quantity <= 0:
                raise ValueError("Quantity must be a positive number.")
        except ValueError as e:
            print(f"Invalid input. {e}")
            continue

        item = find_items(furniture_list, item_id)

        if item:
            cart.append({'item': item, 'quantity': quantity})
            item['quantity'] += quantity
                    print(f"The {quantity} amount of item has been added from the {item['manufacturer']} to this Product ID {item_id}.")
        else:
            print(f"The entered {item_id} item ID is not found.")

        while True:
            try:
```

```
            continue_choice = input("Do you want to add more items to your Furniture store? (yes/no): ").strip().lower()
            if continue_choice not in ['yes', 'no']:
                raise ValueError("Please enter 'yes' or 'no'.")
            break
        except ValueError as e:
            print(f"Invalid input. {e}")

    if continue_choice == 'no':
        break  # Exit the loop

if cart:
    while True:
        address = input("Enter the address for delivery: ").strip()
        if address:
            break
        print("Error: Address cannot be empty.")

    shipping_cost = 0

    if address.lower() != "itahari":
        while True:
```

```
            add_shipping = input("Do you want to add shipping cost to your receipt? (yes/no): ").strip().lower()
            if add_shipping in ['yes', 'no']:
                break
            else:
                print("Error: Please enter 'yes' or 'no'.")

        if add_shipping == 'yes':
            while True:
                try:
                    shipping_cost = float(input("Enter the shipping cost: "))
                    if shipping_cost < 0:
                        raise ValueError("Shipping cost must be a non-negative number.")
                    break
                except ValueError as e:
                    print(f"Invalid input. {e}")
    else:
        print("Free shipping within Itahari.")  # Free shipping within the Itahari Area

    vat_rate = 0.13
    vat_amount = 0
    total_amount = 0
```

```
        for entry_of_item in cart:
            item = entry_of_item['item']
            quantity = entry_of_item['quantity']
            total_amount += item['price'] * quantity
            vat_amount += (item['price'] * quantity) * vat_rate

        shipping_display = f"Shipping Cost: ${shipping_cost:.2f}" if address.lower() != "itahari" else "Free Shipping"

                generate_invoice(cart, employee_name, shipping_display, vat_amount, "Manufacture Order Bill", shipping_cost, address.lower() != "itahari", "purchase")

        write_to_file(file_name, furniture_list)
        print("Order has been placed successfully.")
    else:
        print("No products were chosen.")

def sell_furniture(file_name):
    # Function to sell the furniture items from the inventory
    furniture_list = read_furniture_data(file_name)
    cart = []  # Create empty list

    while True:
```

```
        customer_name = input("Enter customer name: ").strip()
        if customer_name and validate_name(customer_name):
            break
         print("Error: Name cannot be empty and must contain only alphabets without any special characters or numbers.")


    while True:
        try:
            item_id = int(input("Enter furniture ID to sell: "))
            quantity = int(input("Enter the amount of quantity to sell: "))
            if quantity <= 0:
                raise ValueError("Quantity must be a positive number.")
        except ValueError as e:
            print(f"Invalid input. {e}")
            continue


        item = find_items(furniture_list, item_id)
        if item and item['quantity'] >= quantity:
            cart.append({'item': item, 'quantity': quantity})
            item['quantity'] -= quantity
                print(f"The {item['manufacturer']} has sold {quantity} amount of quantity to {customer_name}.")
        else:
```

```
            print(f"The entered ID {item_id} is not found or no more items left.")


        while True:
            try:
                    continue_choice = input("Do you want to add more items? (yes/no): ").strip().lower()
                if continue_choice not in ['yes', 'no']:
                    raise ValueError("Please enter 'yes' or 'no'.")
                break
            except ValueError as e:
                print(f"Invalid input. {e}")


        if continue_choice == 'no':
            break  # Exit the loop


    if cart:
        while True:
            address = input("Enter the address for delivery: ").strip()
            if address:
                break
            print("Error: Address cannot be empty.")


        shipping_cost = 0
```

```
    if address.lower() != "itahari":
        while True:
            add_shipping = input("Do you want to add shipping cost to the total amount? (yes/no): ").strip().lower()
            if add_shipping in ['yes', 'no']:
                break
            else:
                print("Error: Please enter 'yes' or 'no'.")

        if add_shipping == 'yes':
            while True:
                try:
                    shipping_cost = float(input("Enter shipping cost: "))
                    if shipping_cost < 0:
                        raise ValueError("Shipping cost must be a non-negative number.")
                    break
                except ValueError as e:
                    print(f"Invalid input. {e}")
    else:
        print("Free shipping within Itahari.")  # Free shipping within the Itahari Area

    vat_rate = 0.13
```

```
        vat_amount = 0
        total_amount = 0

        for entry_of_item in cart:
            item = entry_of_item['item']
            quantity = entry_of_item['quantity']
            total_amount += item['price'] * quantity
            vat_amount += (item['price'] * quantity) * vat_rate

            shipping_display = f"Shipping Cost: ${shipping_cost:.2f}" if address.lower() != "itahari" else "Free Shipping"

        generate_invoice(cart, customer_name, shipping_display, vat_amount, "Sales Bill", shipping_cost, address.lower() != "itahari", "Sale")

        write_to_file(file_name, furniture_list)
        print("Item has been sold successfully.")
    else:
        print("There were no products chosen for selling.")

def add_new_item(file_name):
    furniture_list = read_furniture_data(file_name)
```

```
    try:
        item_id = int(input("Enter the new furniture ID: "))
        manufacturer = input("Enter the manufacturer: ")
        name = input("Enter the name of the furniture: ")
        quantity = int(input("Enter the quantity: "))
        if quantity <= 0:
            raise ValueError("Quantity must be a positive number.")
        price = float(input("Enter the price: "))
        if price <= 0:
            raise ValueError("Price must be a positive number.")
    except ValueError as e:
        print(f"Invalid input. {e}")
        return

    new_item = {
        'id': item_id,
        'manufacturer': manufacturer,
        'name': name,
        'quantity': quantity,
        'price': price
    }
```

```
    furniture_list.append(new_item)
    write_to_file(file_name, furniture_list)
    print("New item has been added successfully.")
```

iii) Invoice.py

```
import datetime

def generate_invoice(cart, person_name, shipping_display, vat_amount, invoice_title, shipping_cost, include_shipping,transaction_type):
    '''Creating a transaction invoice and saves it to a text file.'''

    #Generate a unique invoice number using the current moment time
    invoice_number = f"{datetime.datetime.now().strftime('%Y%m%d%H%M%S')}"
    #It create a new receipt filename with the transactions type and invoice number
    receipt = f"{transaction_type}_invoice_{invoice_number}.txt"

    #File will open in write mode 'w'
    with open(receipt, 'w') as file:
        file.write("="*90 + "\n")
        file.write(f"       BRJ Furniture Store\n")
        file.write(f"Tarahara, Itahari, Sunsari\n")
```

```
        file.write(f"Phone no: 9807373362    Email: BRJfurniture@gmail.com\n")
        file.write("="*90 + "\n\n")

        #Receipt Title and entire details of the user's
        file.write(f"{invoice_title}\n")
        file.write(f"Name: {person_name}\n")
        file.write(f"Invoice Number: {invoice_number}\n")
        file.write(f"Transaction Type: {transaction_type}\n")
        file.write(f"Date: {datetime.datetime.now().strftime('%Y-%m-%d %H:%M:%S')}\n")
        file.write("\n")
        file.write("|-----------------------------------------------------------------------------------------|\n")
        file.write(f"{'ID':<8}    {'Manufacturer':<30}        {'Name':<20}        {'Quantity':<10}
{'Price':<10}\n")
        file.write("|-----------------------------------------------------------------------------------------|\n")

        total_amount = 0

        #Loop through each item in the cart and record its information on the invoice.
        for entry_of_item in cart:
            item = entry_of_item['item']
            quantity = entry_of_item['quantity']
            total_price = item['price'] * quantity
            total_amount += total_price
```

```
        file.write(f"{item['id']:<8}        {item['manufacturer']:<30}        {item['name']:<20}
{quantity:<10}  ${total_price:<10.2f}\n")


    file.write("\n")
    file.write("="*90 + "\n")
    file.write(f"Total Amount (excluding VAT) : ${total_amount:.2f}\n")
    file.write(f"VAT Amount: ${vat_amount:.2f}\n")

    #If shipping cost is included then show the shipping cost and add it to the total
amount
    if include_shipping:
        file.write(f"{shipping_display}\n")
        total_amount += shipping_cost

    #Calculate the total amount including with VAT
    total_with_vat = total_amount + vat_amount
    file.write(f"Total Amount (including VAT) : ${total_with_vat:.2f}\n")
    file.write("="*90 + "\n")

    file.write("\n")
    file.write("        Thank you for trusting us. Please visit us again\n")
    file.write("="*90 + "\n")
    file.write("          Certified Company, Nepal Government\n")
```

```
        file.write("="*90 + "\n")



    print(f" Thank you.!!! Your Invoice has been generated: {receipt}")      file.write(f"      BRJ Furniture Store\n")
        file.write(f"Tarahara, Itahari, Sunsari\n")
        file.write(f"Phone no: 9807373362     Email: BRJfurniture@gmail.com\n")
        file.write("="*90 + "\n\n")
        file.write(f"{invoice_title}\n")
        file.write(f"Name: {person_name}\n")
        file.write(f"Invoice Number: {invoice_number}\n")
        file.write(f"Transaction Type: {transaction_type}\n")
        file.write(f"Date: {datetime.datetime.now().strftime('%Y-%m-%d %H:%M:%S')}\n")
        file.write("\n")
        file.write("|-------------------------------------------------------------------------------------|\n")
        file.write(f"{'ID':<8}     {'Manufacturer':<30}          {'Name':<20}          {'Quantity':<10}     {'Price':<10}\n")
        file.write("|-------------------------------------------------------------------------------------|\n")


        total_amount = 0


        for entry_of_item in cart:
            item = entry_of_item['item']
```

```
            quantity = entry_of_item['quantity']
            total_price = item['price'] * quantity
            total_amount += total_price
            file.write(f"{item['id']:<8}          {item['manufacturer']:<30}          {item['name']:<20}
{quantity:<10}  ${total_price:<10.2f}\n")

        file.write("\n")
        file.write("="*90 + "\n")
        file.write(f"Total Amount (excluding VAT) : ${total_amount:.2f}\n")
        file.write(f"VAT Amount: ${vat_amount:.2f}\n")

        if include_shipping:
            file.write(f"{shipping_display}\n")
            total_amount += shipping_cost

        total_with_vat = total_amount + vat_amount
        file.write(f"Total Amount (including VAT) : ${total_with_vat:.2f}\n")
        file.write("="*90 + "\n")

        file.write("\n")
        file.write("        Thank you for trusting us. Please visit us again\n")
        file.write("="*90 + "\n")
        file.write("           Certified Company, Nepal Government\n")
```

```
        file.write("="*90 + "\n")


    print(f" Thank you.!!! Your Invoice has been generated: {receipt}")
```

iv) Write.py

```
import datetime


def write_to_file(file_name, furniture_list):
   # The updated furniture list is written by this function to the appropriate file.
    with open(file_name, 'w') as file:  #Loop through each items
        for item in furniture_list:
            file.write(f"{item['id']}, {item['manufacturer']}, {item['name']}, {item['quantity']}, ${item['price']:.2f}\n")
```

v) Read.py

```
def read_furniture_data(filename):
     # It will read the furniture_list
    furniture_list = [] # Creating List to store data
    try:
        # File will open in read mode 'r'
        with open(filename, 'r') as file:
            for line in file:   #Loop through each line
                if line.strip():
```

```
            elements = line.strip().split(', ')


            #Creating a dictionary for each furniture items
            furniture = {
            'id': int(elements[0]),
            'manufacturer': elements[1],
            'name': elements[2],
            'quantity': int(elements[3]),
            'price': float(elements[4].replace('$', ''))
            }
            furniture_list.append(furniture)    #Adding the dictionary items to the List
    except FileNotFoundError:
        print(f"Error: File '{filename}' not found.")
    except Exception as e:
        print(f"An error occurred while loading data: {e}")
    return furniture_list
```

9) References

References

altexsoft, 2024. *altexsoft.* [Online]
Available at: https://www.altexsoft.com/blog/data-structure/
[Accessed 26 07 2024].

BBC, 2024. *BBC.* [Online]
Available at: https://www.bbc.co.uk/bitesize/guides/z3bq7ty/revision/3
[Accessed 22 07 2024].

builtIn, 2024. *builtIn.* [Online]
Available at: https://builtin.com/data-science/python-linked-list#:~:text=A%20Python%20linked%20list%20is%20an%20abstract%20data%20type%20in,other%20items%20in%20the%20list.
[Accessed 26 07 2024].

Coursera, 2024. *Coursera.* [Online]
Available at: https://www.coursera.org/articles/what-is-python-used-for-a-beginners-guide-to-using-python
[Accessed 21 07 2024].

DataCamp, 2024. *DataCamp.* [Online]
Available at: https://www.datacamp.com/blog/what-is-an-algorithm
[Accessed 10 08 2024].

DigitalOcean, 2021. *DigitalOcean.* [Online]
Available at: https://www.digitalocean.com/community/tutorials/understanding-lists-in-python-3
[Accessed 09 08 2024].

GeeksforGeeks, 2023. *GeeksforGeeks.* [Online]
Available at: https://www.geeksforgeeks.org/what-is-pseudocode-a-complete-tutorial/
[Accessed 10 08 2024].

GreatLearning, 2022. *GreatLearning.* [Online] Available at: https://www.mygreatlearning.com/blog/python-stack/ [Accessed 30 07 2024].

Medium, 2023. *Medium.* [Online] Available at: https://medium.com/@riverferguson/the-importance-of-data-structures-in-python-enhancing-efficiency-and-organization-f4537f37e50 [Accessed 06 08 2024].

Medium, 2023. *Medium.* [Online] Available at: https://ranyel.medium.com/introduction-to-strings-in-python-18caf6e02152#:~:text=In%20Python%2C%20strings%20are%20used,to%20work%20with%20strings%20efficiently. [Accessed 10 08 2024].

NorthEasternLondon, 2024. *NorthEasternLondon.* [Online] Available at: https://www.nulondon.ac.uk/academic-handbook/programme-specifications-and-handbooks/undergraduate-programmes/university-course-list-year-one/computing/lcsci4207/ [Accessed 22 07 2024].

PrepBytes, 2023. *PrepBytes.* [Online] Available at: https://www.prepbytes.com/blog/data-structure/non-linear-data-structure/ [Accessed 30 07 2024].

Scaler, 2022. *Scaler.* [Online] Available at: https://www.scaler.com/topics/python/dictionary-in-python/ [Accessed 09 08 2024].

ScholarHat, 2024. *ScholarHat.* [Online] Available at: https://www.scholarhat.com/tutorial/python/tuples-in-python [Accessed 09 08 2024].

shikshalearn, 2024. *shikshalearn.* [Online] Available at: https://www.shiksha.com/online-courses/articles/queue-implementation-in-

python/
[Accessed 30 07 2024].

Simplilearn, 2024. *Simplilearn.* [Online] Available at: https://www.simplilearn.com/tutorials/python-tutorial/python-arrays [Accessed 26 07 2024].

SimpliLearn, 2024. *SimpliLearn.* [Online] Available at: https://www.simplilearn.com/tutorials/python-tutorial/set-in-python#:~:text=A%20set%20is%20mutable%2C%20i.e.,and%20more%20can%20be%20applied.
[Accessed 10 08 2024].

sps.nyu, 2023. *sps.nyu.* [Online] Available at: https://www.sps.nyu.edu/homepage/academics/courses/ISMM1-UC0746-fundamentals-of-computing.html
[Accessed 22 07 2024].

Techopedia, 2012. *Techopedia.* [Online] Available at: https://www.techopedia.com/definition/6597/computing [Accessed 22 07 2024].

Techopedia, 2022. *Techopedia.* [Online] Available at: https://www.techopedia.com/definition/3840/microsoft-word [Accessed 10 08 2024].

Tpoint Tech, 2011-2021. *Javatpoint.* [Online] Available at: https://www.javatpoint.com/idle-software-in-python [Accessed 21 07 2024].

tutorialspoint, 2024. *tutorialspoint.* [Online] Available at: https://www.tutorialspoint.com/data_structures_algorithms/data_structures_and_types.htm
[Accessed 26 07 2024].

UpGuard, 2024. *UpGuard.* [Online] Available at: https://www.upguard.com/security-report/drawio [Accessed 22 07 2024].

10) Plagiarism Report

Originality report

COURSE NAME CS4051NT_FOC

STUDENT NAME Adit Tamang

FILE NAME FOC CourseWork

REPORT CREATED Aug 16, 2024

Summary

Flagged passages	5	0.9%
Cited/quoted passages	7	0.7%

Web matches

cliffsnotes.com	1	0.5%
stackoverflow.com	4	0.3%
shiksha.com	1	0.2%
scaler.com	1	0.2%
lumifywork.com	1	0.1%
geeksforgeeks.org	1	0.1%
questionai.com	1	0.1%
hix.ai	1	0.1%

brainly.ph	1	0.1%

1 of 12 passages

Student passage FLAGGED

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**…

Top web match

Sugat Man Shakya Word Count: 2562 **I confirm that I understand my coursework needs to be submitted online via** Google Classroom **under the relevant module page before the deadline in order for my**…

ReportSample2 (pdf) - CliffsNotes https://www.cliffsnotes.com/study-notes/14766206

2 of 12 passages

Student passage CITED

Because Python **is** so **easy to learn**, even **non-programmers** like **scientists** and accountants use it **for a variety of everyday tasks** including site creation **and** money management. (Coursera, 2024)

Top web match

Because it **is** relatively **easy to learn**, it has also been adopted by **non-programmers** such as accountants and data analysts and **scientists, for a variety of everyday tasks** like organising finances **and**…

Python Programming Short & Training Courses - Lumify Work https://www.lumifywork.com/enau/courses/python-courses/

3 of 12 passages

Student passage FLAGGED

…explanation of an algorithm is called a pseudocode. Since **pseudocode is intended for human understanding rather than machine** interpretation, it is written in simple English and is…

Top web match

Pseudocode does not use any programming language in its representation instead it uses the simple English language text as it **is intended for human understanding rather than machine** reading.…

What is PseudoCode: A Complete Tutorial - GeeksforGeeks https://www.geeksforgeeks.org/what-ispseudocode-a-complete-tutorial/

---

4 of 12 passages

Student passage QUOTED

PRINT an error message indicating that the file was not found.

Top web match

If the file cannot be opened, **print an error message indicating that the file was not found**. Step9: Handle ValueError. If a ValueError occurs, print the ...

Try: File_name = "path/to/your/file.txt" Replace with the Actual File ... https://www.questionai.com/questionstrYmA51kaQ/try-filename-pathtoyourfiletxt-replace-actual-file-path

---

5 of 12 passages

Student passage CITED

in Python a queue is a type of **data structure that** permits **first-in, first-out (FIFO**) ordering, where **the** item **added to the queue will be the** item removed **first**. (shikshalearn, 2024)

Top web match

In Python, a queue is a data structure that allows adding and removing elements in a **first-in, first-out (FIFO**) order, i.e., **the** first element **added to the queue will be the first** one to be removed.

Queue Implementation in Python: Basic Operations and Examples https://www.shiksha.com/onlinecourses/articles/queue-implementation-in-python/

---

6 of 12 passages

Student passage FLAGGED

…the data is kept in a key-value pair format. **A dictionary is a data type** in Python **that may** resemble an actual **data arrangement** where **a given value exists for a** given **key**

Top web match

In Python, **a dictionary is a data type that may** replicate a real-world **data arrangement** in which **a given value exists for a** specified **key**. It's a data structure that can be changed. The keys and…

Dictionary in Python (With Examples) - Scaler Topics https://www.scaler.com/topics/python/dictionary-inpython/

---

7 of 12 passages

Student passage FLAGGED

It can either be positive, negative or zero. It is immutable in case of python. It is…

Top web match

It is a whole number (not a fractional number)that **can either be positive, negative, or zero**. A. counting number C. fraction D. decimal B.

It is a whole number (not a fractional number)that can either be ... https://brainly.ph/question/6935632

---

8 of 12 passages

Student passage QUOTED

…will display the appropriate error messages. “Error: Invalid number. **Please try again with valid numbers**” & Quantity must be a positive number.

Top web match

except: text = "An error accured, **please try again with valid numbers**." await ctx.send(text) ```. python · list · discord.py · append · Share. Share a link to this question. Copy link

Log command list .append() doesn't work in discord.py https://stackoverflow.com/questions/69702434/logcommand-list-append-doesnt-work-in-discord-py

---

9 of 12 passages

Student passage FLAGGED

…really helped me to get through this challenging project's. **I** also **want to express my gratitude** to my family members **for** their **continuous support and guidance**

Top web match

I want to express my gratitude for your **continuous support and guidance**. Your mentorship has had a significant impact on my professional growth. Thank you for being an amazing co-worker.

100+ Appreciation Thank You Messages: Express Your Gratitude https://resource.hix.ai/messages/appreciation-thank-you-messages

---

10 of 12 passages

Student passage QUOTED

…print("Error: Invalid number. **Please try again with valid numbers**.")

Top web match

except: text = "An error accured, **please try again with valid numbers**." await ctx.send(text) ```. python · list · discord.py · append · Share. Share a link to this question. Copy link

Log command list .append() doesn't work in discord.py https://stackoverflow.com/questions/69702434/logcommand-list-append-doesnt-work-in-discord-py

---

11 of 12 passages

Student passage CITED

…add_shipping = input("Do you **want to add shipping cost to the total amount**? (yes/no): ").strip().lower()

Top web match

I **want to add Shipping Cost to the total** checkout **amount**. I am using the stripe prebuild checkout page to complete the payment. When I enabled ...

How to add Shipping Cost in Total Amount in Stripe Checkout?

https://stackoverflow.com/questions/67248020/how-to-add-shipping-cost-in-total-amount-in-stripe-checkout

---

12 of 12 passages

Student passage CITED

print("**New item has been added successfully**.")

Top web match

if(response.status=="success"){ alertify.success("**New item has been added successfully**"); }else if(response.status=="error"){ alertify.error("Error while adding the item"); }. Even the query runs…

Retrieve response text in AJAX - Stack Overflow https://stackoverflow.com/questions/42158897/retrieveresponse-text-in-ajax

---