



CS4051NI\CC4059NI Fundamental of Computing

60% Individual Coursework

2023-24 Summer

Student Name: Abhigyan Shrestha

London Met ID: 23056139

College ID: np01nt4s240015

Assignment Due Date: October 18, 2024

Assignment Submission Date: October 12, 2024

Word Count: 4196

I confirm that I understand my coursework needs to be submitted online via MySecondTeacher under the relevant module page before the deadline in order for my assignment to be accepted and marked. I am fully aware that late submissions will be treated as non-submission and a marks of zero will be awarded.

Table of Contents

Introduction	3
1.1 Introduction of the project.....	3
1.2 Aims and Objectives	3
1.3 Tools used in the assignment	3
Algorithm.....	5
Flowchart	7
Pseudocode:.....	10
Pseudocode for main.py.....	10
Pseudocode for operaton.py	12
Pseudocode for write.py	14
Pseudocode for read.py	15
Data Structures	16
Program	19
Testing	22
Test 1: Show implementation of try, except.....	22
Test 2: Selection buy and sell of furnitures	23
Test 3: File generation of buying of furniture(s).....	24
Test 4: File generation of selling process of furniture(s).....	26
Test 5: Show the update in stock of furniture(s).....	27
Conclusion	30
References.....	31
Appendix.....	32
Code of main.py.....	32
Code of operation.py	35
Code of read.py	37
Code of write.py.....	38

Table of Figure:

Figure 1-Python logo	4
Figure 2-IDLE LOGO	4
Figure 3-Main FLOWchart.....	7
Figure 4-Order Flowchart	8
Figure 5-Sell Flowchart.....	9
Figure 6-Use of Integer Datatype.	16
Figure 7-Use of Float datatype	17
Figure 8-Use of string data type	17
Figure 9-Use of Dictionary.	18
Figure 10-Welcome message and Main menu of the program	19
Figure 11-Display of furniture details.	19
Figure 12-Process of ordering furniture.	20
Figure 13-Order Bill	20
Figure 14-Process of selling furniture.	20
Figure 15-Sell Bill	21
Figure 16-Exiting the system	21
Figure 17-Try-Except code	22
Figure 18-Error message try-except.....	23
Figure 19-Giving negative value	23
Figure 20-Giving non-existed value	24
Figure 21-Buying process.....	24
Figure 22-Invoice of Bought Furniture in txt file	25
Figure 23-Selling process.....	26
Figure 24-Invoice of sold furniture in txt file.	27
Figure 25-Inventory before buying from manufacturers(Pre stock)	27
Figure 26-Buying Furniture '5'	28
Figure 27-Automatic update on the txt file	28
Figure 28-Inventory before selling	28
Figure 29-Selling furniture '5'	29
Figure 30-Stock of furniture after selling	29

Introduction

1.1 Introduction of the project

This report aims to outline the development of a program for managing the buying and selling of furniture in a private company called BRJ Furniture Store. The project utilizes Python programming language to read a text file containing details about available furniture and update it after the furniture is either sell or ordered. It is designed to generate invoices with unique names for each transaction. Also, it updates the status of each furniture as transactions occur and handles error validation. This project showcases the versatility of Python programming language to solve practical problems efficiently and effectively.

1.2 Aims and Objectives

The aim of the coursework is to develop a Furniture management system that helps to e-commerce furniture related businesses more efficiently. The objectives of the coursework are:

1. To develop a system that can read raw data and converts of meaningful information about the furniture available and display to clients.
2. To update the information after buying and selling of the furniture in the text file.
3. To generate the invoices of both order and sell of furniture.
4. To test the program that check the requirements of furniture management system.

1.3 Tools used in the assignment

(1) Python:

Python is high-level programming language used by the programmers for web development, scientific computing, and artificial intelligence which is widely used all over the world. It was initially developed by Guido van Rossum in 1991 and developed by Python Software Foundation. Programmers may express concepts in fewer lines of code because of its syntax, which was primarily built with code readability in mind. The extension of python is .py.

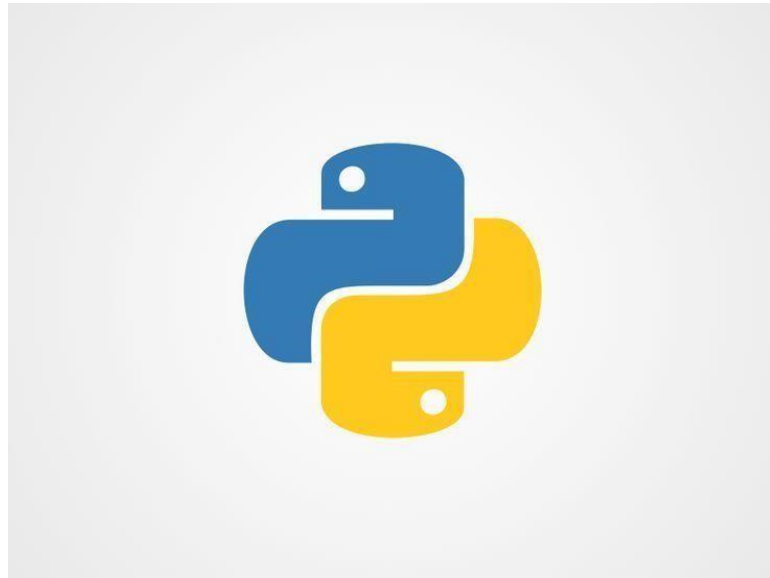


Figure 1-Python logo

(2) IDLE (Integrated Development and Learning Environment):

IDLE is Python's Integrated Development and Learning Environment. It allows programmers to easily write Python code. Just like Python Shell, IDLE can be used to execute a single statement and create, modify, and execute Python scripts. IDLE provides a fully featured text editor to create Python scripts that include features like syntax highlighting, autocompletion, and smart indent. It also has a debugger with stepping and breakpoints features. This makes debugging easier. (IDLE, 2020)



Figure 2-IDLE LOGO

(3) MS-Word:

MS-Word stands for Microsoft Word. MS-Word is the most popular word processor that allows user to create and edit the document. It is developed by Microsoft. It was first released of October 25, 1983, under the name of Multi-Tool Word for Xenix system. MSWord supports in both Windows and Mac. The file extension of MS-Word is .doc or .docx. It is popular due to its some valuable features like in-built dictionary, design, insert, layout, references, mailing, review, and help.

(4) Draw.io:

Draw.io is a free, online diagramming tool that allows you to create flowcharts, diagrams, mind maps, organisation charts, and much more. A web-based application, Draw.io is fully integrated with Google Drive. This means that you can automatically save the results of your work in your Google Workspace or Gmail account. (Paraschiv, 2023)

Algorithm

An algorithm is simply a set of steps used to complete a specific task. These are the fundamental units of the programming that enables the operation and decision-making of devices like computers, mobile, smartphones and so on. To make the program successful, there's a certain set of steps one's need to follow in a particular order. Any application or programs work in algorithm to get the desired outcome.

The following is the algorithm of BRJ Furniture Store.

Step 1: Start the program.

Step 2: Display welcome message.

Step 3: Show the option to the user:

- Press 1: To display available furniture
- Press 2: To buy form the manufacturer
- Press 3: To sell furniture to a customer
- Press 4: To exit

Step 4: Take input from the user.

Step 5: If the input is 1 then show the available furniture.

Step 6: If the input is 2 (To buy from the manufacturer).

Step 7: Ask the user for the furniture ID and validate if the input ID is correct or not if not then ask them for the correct ID.

Step 8: Ask the Quantity to order.

Step 9: Ask the user for their name.

Step 10: Once the product is confirmed, calculate the total amount to pay.

Step 11: The order bill is generated in the txt file.

Step 12: Return to the main menu.

Step 13: If the input is 3(To sell furniture to customer).

Step 14: Ask the user for the furniture ID to sell and validate if it's correct or not if not ask for correct ID.

Step 15: Ask the user the Quantity to sell.

Step 16: Ask the user for their name.

Step 17: Ask the user for shipping cost.

Step 18: The sell bill is generated in txt file.

Step 19: Return to the main menu.

Step 20: If the input is 4(To exit the system)

Step 21: Display the exit message.

Step 22: Close the program.

Flowchart

A flowchart is a visual representation of the steps or processes involved in solving a problem or completing a task. It consists of various shapes and symbols connected by arrows to indicate the flow of control or data within a program or system. Flowcharts are widely used in software development, business process modelling, system design, and other fields where complex processes need to be understood and communicated effectively.

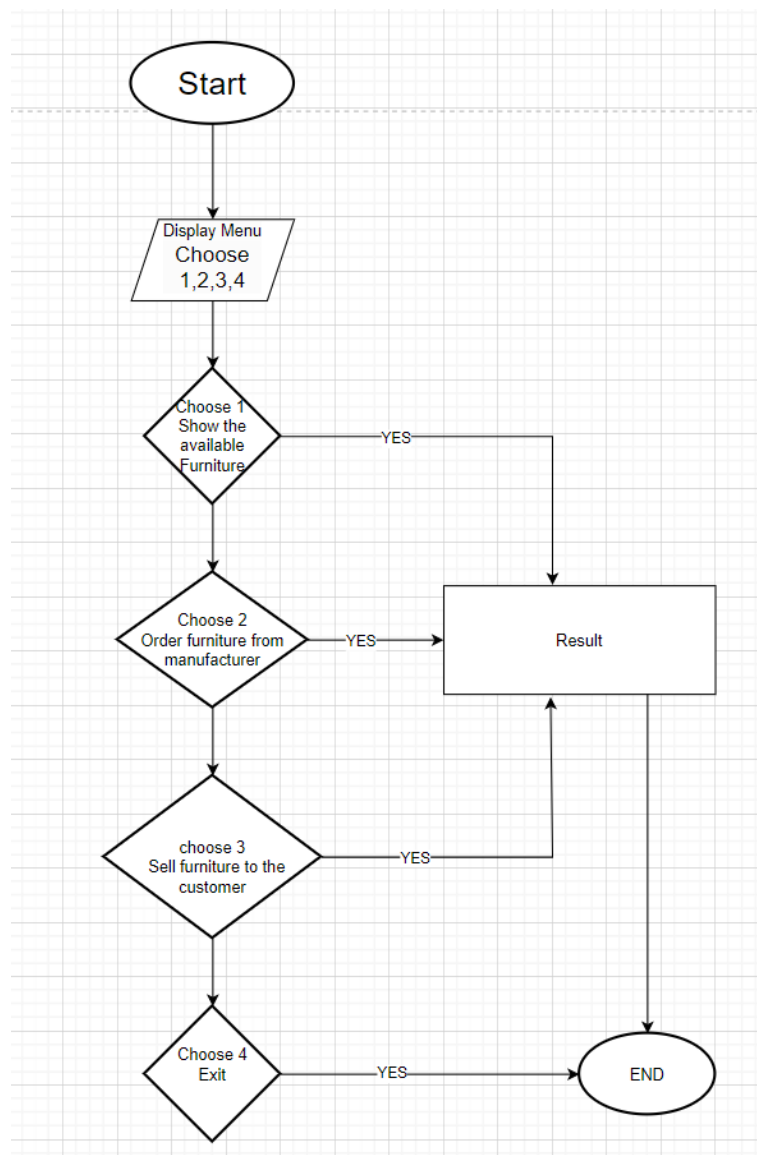


Figure 3-Main FLOWchart

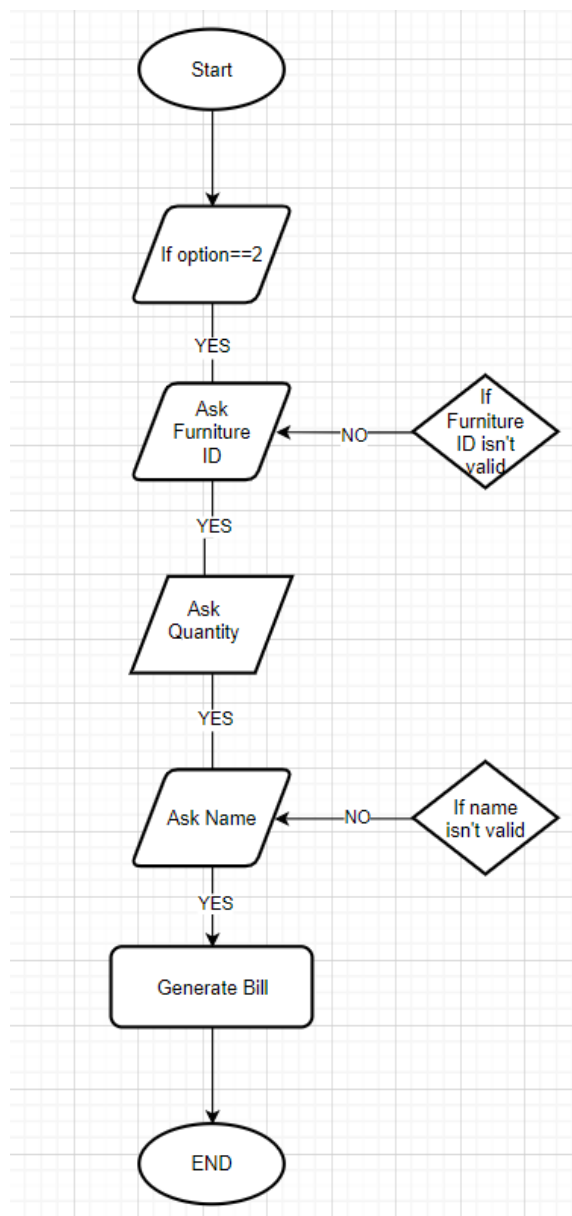


Figure 4-Order Flowchart

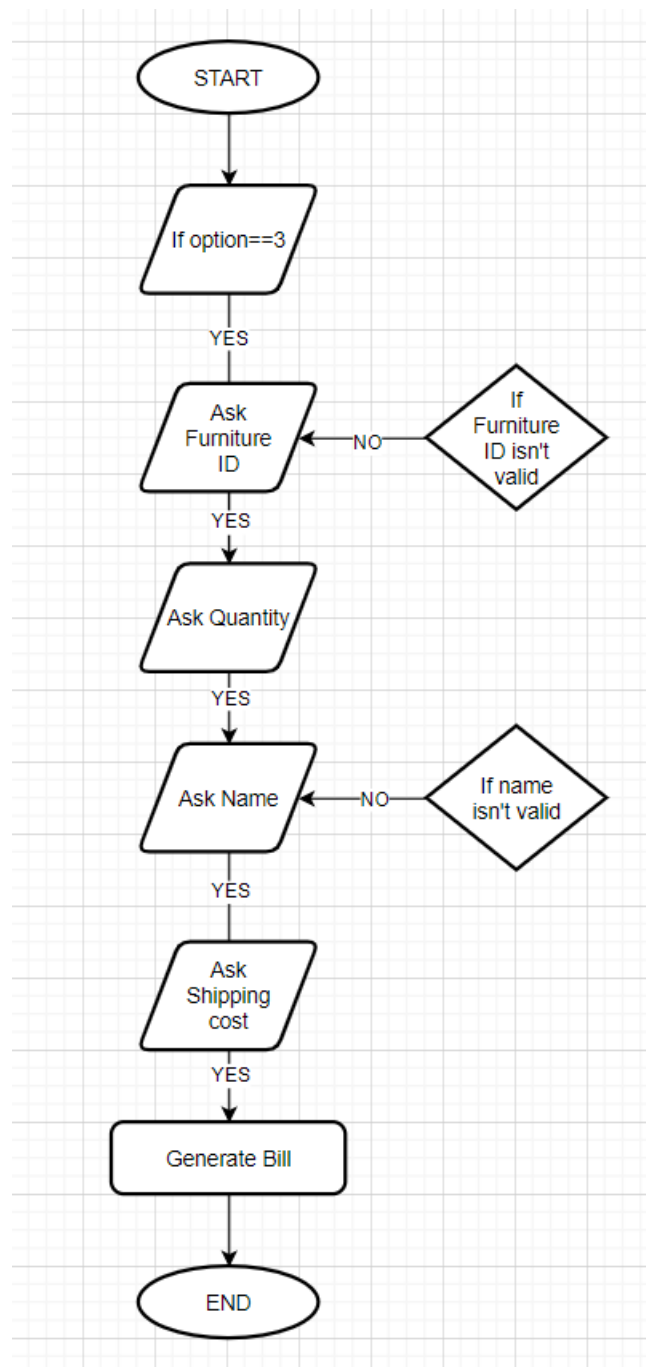


Figure 5-Sell Flowchart

Pseudocode:

Pseudo code is a term which is often used in programming and algorithm-based fields. It is a methodology that allows the programmer to represent the implementation of an algorithm. Simply, we can say that it's the cooked-up representation of an algorithm. Often at times, algorithms are represented with the help of pseudo codes as they can be interpreted by programmers no matter what their programming background or knowledge is. Pseudo code, as the name suggests, is a false code or a representation of code which can be understood by even a layman with some school level programming knowledge. It's an organized logical sequence of the actions or the approach towards a particular problem. A programmer implements an algorithm to solve a problem. Algorithms are expressed using natural verbal but somewhat technical annotations. It's simply an implementation of an algorithm in the form of annotations and informative text written in plain English. It has no syntax like any of the programming language and thus can't be compiled or interpreted by the computer. (Geeks for Geeks, 2023)

Pseudocode for main.py

```
Import order_furniture_process, sell_furniture_process
Import read_furniture_process
Function validate_Int_input(prompt)
    WHILE TRUE
        TRY
            READ integer input from user
            RETURN input
        EXCEPT ValueError
            PRINT "Invalid input. Please enter a valid number"
    END WHILE
END Function
FUNCTION validate_float_input(prompt)
    WHILE TRUE
```

```
    TRY
        READ float input from user
        RETURN input
    EXCEPT ValueError
        PRINT "Invalid input. Please enter a valid number"
END WHILE
END Function
FUNCTION welcome
    DISPLAY welcome message and store information
END FUNCTION
FUNCTION main
    WHILE TRUE
        PRINT menu options
        READ user choice

        IF choice == 1
            CALL show_furniture_list
        ELSE IF choice == 2
            READ furniture ID and quantity from user
            READ employee name from user
            CALL order_furniture_process with ID, quantity, and employee name
            PRINT "Order process completed successfully!"
        ELSE IF choice == 3
            READ furniture ID and quantity from user
            READ customer name from user
            READ shipping cost from user
            CALL sell_furniture_process with ID, quantity, customer name, and shipping cost
```

```
    PRINT "Sell process completed successfully!"
ELSE IF choice == 4
    PRINT "Exiting the program. Thank you for visiting BRJ Furniture Store!"
    BREAK
ELSE
    PRINT "Invalid choice, please enter the number between 1 to 4"
END FUNCTION
IF this script is run directly:
CALL Main()
END Script
```

Pseudocode for operaton.py

```
IMPORT read_furniture_Process
IMPORT Write_furniture_Process, generate_invoice_process
FUNCTION order_furniture_process
    IF quantity <= 0
        PRINT "Quantity must be greater than zero."
        RETURN
    READ furniture data from file
    IF id exists in furniture data
        CALCULATE amount = price of furniture id * quantity
        GENERATE invoice for order with details (id, manufacturer, product name, quantity,
employee, amount)
        WRITE updated furniture data to file
        PRINT "Order processed successfully."
    ELSE
```

```
PRINT "Error: Furniture ID not found."

END FUNCTION

DEFINE FUNCTION sell_furniture_process

  IF quantity <= 0

    PRINT "Quantity must be greater than zero."

    RETURN

  IF shipping_cost < 0

    PRINT "Shipping cost cannot be negative."

    RETURN

  READ furniture data from file

  IF id exists in furniture data

    IF quantity is available in stock

      CALCULATE total amount = price of furniture id * quantity

      CALCULATE total amount with shipping = total amount + shipping cost

      GENERATE invoice for sale with details (id, manufacturer, product name, quantity,
customer name, price, total amount, shipping cost, total amount with shipping)

      WRITE updated furniture data to file

      PRINT "Sale processed successfully."

    ELSE

      PRINT "Error: Insufficient stock."

    ELSE

      PRINT "Error: Furniture ID not found."

  END FUNCTION
```

Pseudocode for write.py

FUNCTION write_furniture_process(furniture_data)**OPEN** file for writing**FOR EACH** item in furniture_data**CREATE** line with item details (ID, manufacturer, product name, quantity, price)**WRITE** line to file**CLOSE** file**END FUNCTION****FUNCTION** generate_invoice_process(transaction_type, **kwargs)**GET** current date and time**INITIALIZE** invoice text with header and date**IF** transaction_type == 'Order'**ADD** employee, ID, manufacturer, product name, quantity, and amount to invoice text**ELSE IF** transaction_type == 'Sale'**ADD** customer name, ID, manufacturer, product name, quantity, price, total amount, shipping cost, and total amount with shipping to invoice text**WRITE** invoice text to file**PRINT** invoice text to terminal**END FUNCTION**

Pseudocode for read.py

FUNCTION read_furniture_process

OPEN file FURNITURE_FILE for reading

INITIALIZE empty dictionary furniture_data

FOR EACH line in file

SPLIT line into parts using comma and space as separator

IF number of parts is equal to 5

CONVERT parts[0] to integer and store in id_

STORE parts[1] in manufacturer

STORE parts[2] in product_name

CONVERT parts[3] to integer and store in quantity

REMOVE dollar sign from parts[4] and convert to float, then store in price

ADD furniture data to dictionary furniture_data with id_ as key

SET manufacturer, product_name, quantity, and price for id_ in furniture_data

CLOSE file

RETURN furniture_data

END FUNCTION

Data Structures

Data Structures are a way of organizing data so that it can be accessed more efficiently depending upon the situation. Data Structures are fundamentals of any programming language around which a program is built. Python helps to learn the fundamental of these data structures in a simpler way as compared to other programming languages. (Geeks for Geeks, 2024)

In development of this system, a text file is read and organized in a structured format using a file handling approach. The data, represented as a list of dictionaries, allows for a efficient storage, retrieval and presentation of the text file. By encapsulating file reading and data formatting functionalities, the program promotes code modularity and abstraction, contributing to the overall clarity and maintainability of the Furniture management system. This program also updates and edits the text file after a furniture is either ordered or sold.

There are two types of data structure in python. They are;

- Primitive Data type
- Non-primitive Data type

Primitive Data type:

Primitive data types are the most basic data structure. They are also known as the building blocks of the data manipulation and contain pure, simple values of data.

Python consists of four primitive data types:

- Integers:

Integers are those data types which store numeric values in Python and represented by “int” in python.

```
if choice == '1':  
    show_furniture_list()  
elif choice == '2':  
    id_ = validate_int_input("Enter Furniture ID to order: ")  
    quantity = validate_int_input("Enter quantity to order: ")  
    employee_name = input("Enter your name (Employee name): ")  
    order_furniture_manager(id_, quantity, employee_name)
```

Figure 6-Use of Integer Datatype.

- Float:

Float data type are used to convert decimal number to point number and denoted by “float”. We can use it for rational numbers ending with decimal figures such as 3.14.

```
if choice == '1':
    show_furniture_list()
elif choice == '2':
    id_ = validate_int_input("Enter Furniture ID to order: ")
    quantity = validate_int_input("Enter quantity to order: ")
    employee_name = input("Enter your name (Employee name): ")

    order_furniture(id_, quantity, employee_name)
```

Figure 7-Use of Float datatype

- Strings:

Strings are the collections of alphabets, words, and other characters and denoted by “str” in python.

```
furniture_data[id_] = {
    'manufacturer': manufacturer,
    'product_name': product_name,
    'quantity': quantity,
    'price': price
}
```

Figure 8-Use of string data type

- Booleans:

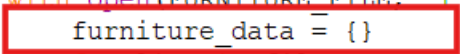
This is built-in data type that checks True and False after taking the values and often make them interchangeable with in the integers 1 and 0. Booleans are useful in conditional and comparison expressions.

Non-Primitive Data type:

Non-primitive data types are the refined members of the data structure family. They not only store the value but also a collection of values in different formats.

- Dictionary:

Dictionaries are mutable data structures that allow you to store key-values pairs.

Dictionary are created using curly braces '{}'.


```
# read the furniture data
def read_furniture_process():

    with open(FURNITURE_FILE, 'r') as file:
        furniture_data = {}
        for line in file:
```

Figure 9-Use of Dictionary.

- List:

Lists in python are also used to store the collections of heterogeneous data. These are mutable. One can recognize list by their square brackets “[]” and elements are separated by comma “,”.

- Set:

Set is also a data type which store the data in a single variable. Though it can be identified as it also start from curly braces but different in context of unindexed and unordered. Set in python is as like the set-in mathematics and perform operations like intersection, union, etc.

- Tuples:

Tuples are the data structure which is very similar to the list, but tuples are immutable, and they are rigid once created. This makes them ideal for storing data that should not be changed.

In summary, proper usages of data structure is essential in building a reliable and effective program.

Program

This program consists of instructions that enable the manipulation of data by taking the input from the clients. The program allows users to interact with Furniture management system. This program is about the Furniture management system used for ordering and selling furniture.

The program begins with by displaying Welcome message for the user and ask users to choose an option to either show the available furniture, order furniture from manufacturer, sell furniture to customers or to exit the system. If user wish to choose to exit the system the program will simply end by displaying a Thankyou message.

If the user chooses to order furniture from manufacturer, then user firstly asked to enter the furniture id ,then asked for quantity of furniture to order , and lastly asked the name of the employee. After the user inputs all the thing valid the system will display the order processed successfully and print the bill.

If the user choose to sell furniture to customers, then it asks for the furniture id to sell, then asked for quantity of furniture to sell, and then asked to enter the customer name, after that it will ask for the shipping cost, if all the input were valid it will display the sale processed successfully and print the bill.

Evidence:

```
*****
                        Welcome To BRJ Furniture Store
                        Thankot, Kathmandu | 9808190819
                        Embrace the art of living beautifully-your dream space awaits!
*****

Please select an option:

-> Press 1: *To display the available furniture*
-> Press 2: *To buy from the manufacturer*
-> Press 3: *To sell furniture to customer*
-> Press 4: *To Exit from the system*
*****

Enter your choice: |
```

Figure 10-Welcome message and Main menu of the program

```
Enter your choice: 1
ID: 1, Manufacturer: HNI Corporation, Product: Bunk Bed, Quantity: 126, Price: $400.0
ID: 2, Manufacturer: HNI Corporation Haworth Inc., Product: Twin Bed, Quantity: 252, Price: $600.0
ID: 3, Manufacturer: Achham furniture, Product: Sleeper Sofa, Quantity: 307, Price: $200.0
ID: 4, Manufacturer: Kimball International Inc., Product: Corner sofa, Quantity: 55, Price: $350.0
ID: 5, Manufacturer: Kohler Co., Product: Armchair, Quantity: 13, Price: $150.0
ID: 6, Manufacturer: Masco Corporation, Product: Desk chair, Quantity: 96, Price: $100.0
```

Figure 11-Display of furniture details.

Please select an option:

-> Press 1: *To display the available furniture*
 -> Press 2: *To buy from the manufacturer*
 -> Press 3: *To sell furniture to customer*
 -> Press 4: *To Exit from the system*

Enter your choice: 2
 Enter Furniture ID to order: 1
 Enter quantity to order: 4
 Enter your name (Employee name): Abhigyan

Figure 12-Process of ordering furniture.

 BRJ FURNITURE STORE
 ---- -Thankot, Kathmandu-----

Date: 2024-10-12
Transaction Type: Order

Employee: Abhigyan
 ID: 1
 Manufacturer: HNI Corporation
 Product: Bunk Bed
 Quantity: 4

Amount to Pay: \$1600.00

-----Thank You for choosing us. Please visit again.-----

Figure 13-Order Bill

-> Press 1: *To display the available furniture*
 -> Press 2: *To buy from the manufacturer*
 -> Press 3: *To sell furniture to customer*
 -> Press 4: *To Exit from the system*

Enter your choice: 3
 Enter Furniture ID to sell: 4
 Enter quantity to sell: 5
 Enter customer name: Swasti
 Enter shipping cost: \$500

Figure 14-Process of selling furniture.

```

-----
***BRJ FURNITURE STORE***
---- -Thankot, Kathmandu-----
-----
Date: 2024-10-12
Transaction Type: Sale
-----
Customer: Swasti
ID: 4
Manufacturer: Kimball International Inc.|
Product: Corner sofa
Quantity: 5
Price per Unit: $350.00
Total Amount: $1750.00
Shipping Cost: $500.00
-----
Total Amount to Pay: $2250.00
-----
Goods once sold will not be returned.
Exchange within 7 days with the receipt except Saturday.
-----
-----Thank You for choosing us. Please visit again.-----

```

Figure 15-Sell Bill

```

Please select an option:

-> Press 1: *To display the available furniture*
-> Press 2: *To buy from the manufacturer*
-> Press 3: *To sell furniture to customer*
-> Press 4: *To Exit from the system*
*****
Enter your choice: 4
Exiting the program.
Thank you for visiting BRJ Furniture Store!
|

```

Figure 16-Exiting the system

Testing

Test 1: Show implementation of try, except

Objective	To show implementation of try-except block
Action	The input for Quantity is given in unsuitable datatype. i.e. String.
Expected Result	Error message should be shown.
Actual Result	Error message is show when we input unsuitable datatype
Conclusion	The test is successful

Table 1: Test to show the implementation of try-except block.

Evidences:

```
# Get integer input for validation
def validate_int_input(prompt):
    while True:
        try:
            return int(input(prompt))
        except ValueError:
            print("Invalid input. Please enter a valid number eg. 1")

# Get float input for validation
def validate_float_input(prompt):
    while True:
        try:
            return float(input(prompt))
        except ValueError:
            print("Invalid input. Please enter a valid number eg. 500.56")
```

Figure 17-Try-Except code

```

*****
                        Welcome To BRJ Furniture Store
                        Thankot, Kathmandu | 9808190819
                        Embrace the art of living beautifully-your dream space awaits!
*****

Please select an option:

-> Press 1: *To display the available furniture*
-> Press 2: *To buy from the manufacturer*
-> Press 3: *To sell furniture to customer*
-> Press 4: *To Exit from the system*
*****

Enter your choice: 2
Enter Furniture ID to order: 1
Enter quantity to order: Ace
Invalid input. Please enter a valid number eg. 1
Enter quantity to order:

```

Figure 18-Error message try-except

Test 2: Selection buy and sell of furnitures

Objective	To show selection buy and sell of furniture
Action	<ul style="list-style-type: none"> ○ Negative value is given as the input. ○ Non-existing value is given as the input.
Expected Result	Error message should be shown
Actual Result	Error message was shown
Conclusion	The test is successful.

Table 2: Test to show selection buy and sell of furniture

Evidence:

```

Please select an option:

-> Press 1: *To display the available furniture*
-> Press 2: *To buy from the manufacturer*
-> Press 3: *To sell furniture to customer*
-> Press 4: *To Exit from the system*
*****

Enter your choice: 2
Enter Furniture ID to order: -5
Enter quantity to order: 2
Enter your name (Employee name): 2
Error: Furniture ID -5 not found.

```

Figure 19-Giving negative value

Please select an option:

-> Press 1: *To display the available furniture*
 -> Press 2: *To buy from the manufacturer*
 -> Press 3: *To sell furniture to customer*
 -> Press 4: *To Exit from the system*

Enter your choice: 6
 Invalid choice, please enter the number between 1 to 4.

Figure 20-Giving non-existed value

Test 3: File generation of buying of furniture(s)

Objective	To show file generation of buying of furniture(s)
Action	Input is given to print invoice in the terminal
Expected Result	The invoice for order would be generated.
Actual Result	The invoice for order was generated in a text file with a unique name
Conclusion	The test is successful.

Table 3: Test to show file generation of buying of furniture(s)

Evidence:

```
*****
                                Welcome To BRJ Furniture Store
                                Thankot, Kathmandu | 9808190819
                                Embrace the art of living beautifully-your dream space awaits!
*****

Please select an option:

-> Press 1: *To display the available furniture*
-> Press 2: *To buy from the manufacturer*
-> Press 3: *To sell furniture to customer*
-> Press 4: *To Exit from the system*
*****

Enter your choice: 2
Enter Furniture ID to order: 3
Enter quantity to order: 25
Enter your name (Employee name): Abhigyan

-----
***BRJ FURNITURE STORE***
---- -Thankot, Kathmandu----
-----
Date: 2024-10-12
Transaction Type: Order
-----
Employee: Abhigyan
ID: 3
Manufacturer: Achham furniture
Product: Sleeper Sofa
Quantity: 25
-----
Amount to Pay: $5000.00
-----
-----Thank You for choosing us. Please visit again.-----

Order processed successfully. 25 units of furniture ID 3 ordered.
Order process completed successfully!
```

Figure 21-Buying process

```
-----  
***BRJ FURNITURE STORE***  
---- -Thankot, Kathmandu-----  
-----  
Date: 2024-10-12  
Transaction Type: Order  
-----  
Employee: Abhigyan  
ID: 3  
Manufacturer: Achham furniture  
Product: Sleeper Sofa  
Quantity: 25  
-----  
Amount to Pay: $5000.00  
-----  
-----Thank You for choosing us. Please visit again.-----
```

Figure 22-Invoice of Bought Furniture in txt file

Test 4: File generation of selling process of furniture(s)

Objective	To show file generation of selling process of furniture(s)
Action	Input is given to print invoice in the terminal
Expected Result	The invoice for sell would be generated.
Actual Result	The invoice for sell was generated in a text file with a unique name
Conclusion	The test is successful

Table 4: Test to show file generation of selling process of furniture(s)

Evidence:

```

Please select an option:
-> Press 1: *To display the available furniture*
-> Press 2: *To buy from the manufacturer*
-> Press 3: *To sell furniture to customer*
-> Press 4: *To Exit from the system*
*****

Enter your choice: 3
Enter Furniture ID to sell: 3
Enter quantity to sell: 25
Enter customer name: Swasti
Enter shipping cost: $50

-----
***BRJ FURNITURE STORE***
---- -Thankot, Kathmandu-----
-----
Date: 2024-10-12
Transaction Type: Sale
-----
Customer: Swasti
ID: 3
Manufacturer: Achham furniture
Product: Sleeper Sofa
Quantity: 25
Price per Unit: $200.00
Total Amount: $5000.00
Shipping Cost: $50.00
-----
Total Amount to Pay: $5050.00
-----
Goods once sold will not be returned.
Exchange within 7 days with the receipt except Saturday.
-----
-----Thank You for choosing us. Please visit again.-----

Sale processed successfully. 25 units of furniture ID 3 sold to Swasti.
Sell process completed successfully!

```

Figure 23-Selling process

```

***BRJ FURNITURE STORE***
---- -Thankot, Kathmandu-----
-----
Date: 2024-10-12
Transaction Type: Sale
-----
Customer: Swasti
ID: 3
Manufacturer: Achham furniture
Product: Sleeper Sofa
Quantity: 25
Price per Unit: $200.00
Total Amount: $5000.00
Shipping Cost: $50.00
-----
Total Amount to Pay: $5050.00
-----
Goods once sold will not be returned.
Exchange within 7 days with the receipt except Saturday.
-----
-----Thank You for choosing us. Please visit again.-----

```

Figure 24-Invoice of sold furniture in txt file.

Test 5: Show the update in stock of furniture(s)

Objective	To show the update in stock of furniture(s)
Action	Furniture Id: 5, Armchair is Bought from manufacturer. Furniture Id: 5, Armchair is sold to customer.
Expected Result	The status of Furniture should be updated in txt file.
Actual Result	The status of Furniture was updated in txt file.
Conclusion	The test is successful.

Table 5: Test to show the update in stock of furniture(s)

Evidence:

```

1, HNI Corporation, Bunk Bed, 135, $400.0
2, HNI Corporation Haworth Inc., Twin Bed, 252, $600.0
3, Achham furniture, Sleeper Sofa, 307, $200.0
4, Kimball International Inc., Corner sofa, 50, $350.0
5, Kohler Co., Armchair, 13, $150.0
6, Masco Corporation, Desk chair, 96, $100.0

```

Figure 25-Inventory before buying from manufacturers(Pre stock)

```

Please select an option:
-> Press 1: *To display the available furniture*
-> Press 2: *To buy from the manufacturer*
-> Press 3: *To sell furniture to customer*
-> Press 4: *To Exit from the system*
*****
Enter your choice: 2
Enter Furniture ID to order: 5
Enter quantity to order: 7
Enter your name (Employee name): Abhigyan

-----
***BRJ FURNITURE STORE***
---- -Thankot, Kathmandu-----
-----
Date: 2024-10-12
Transaction Type: Order
-----
Employee: Abhigyan
ID: 5
Manufacturer: Kohler Co.
Product: Armchair
Quantity: 7
-----
Amount to Pay: $1050.00
-----
-----Thank You for choosing us. Please visit again.-----

Order processed successfully. 7 units of furniture ID 5 ordered.
Order process completed successfully!

```

Figure 26-Buying Furniture '5'

```

1, HNI Corporation, Bunk Bed, 135, $400.0
2, HNI Corporation Haworth Inc., Twin Bed, 252, $600.0
3, Achham furniture, Sleeper Sofa, 307, $200.0
4, Kimball International Inc., Corner sofa, 50, $350.0
5, Kohler Co., Armchair, 20, $150.0
6, Masco Corporation, Desk chair, 96, $100.0

```

Figure 27-Automatic update on the txt file

```

1, HNI Corporation, Bunk Bed, 135, $400.0
2, HNI Corporation Haworth Inc., Twin Bed, 252, $600.0
3, Achham furniture, Sleeper Sofa, 307, $200.0
4, Kimball International Inc., Corner sofa, 50, $350.0
5, Kohler Co., Armchair, 20, $150.0
6, Masco Corporation, Desk chair, 96, $100.0

```

Figure 28-Inventory before selling

```

Please select an option:

-> Press 1: *To display the available furniture*
-> Press 2: *To buy from the manufacturer*
-> Press 3: *To sell furniture to customer*
-> Press 4: *To Exit from the system*
*****

Enter your choice: 3
Enter Furniture ID to sell: 5
Enter quantity to sell: 10
Enter customer name: Swasti
Enter shipping cost: $50

-----
***BRJ FURNITURE STORE***
---- -Thankot, Kathmandu-----
-----
Date: 2024-10-12
Transaction Type: Sale
-----
Customer: Swasti
ID: 5
Manufacturer: Kohler Co.
Product: Armchair
Quantity: 10
Price per Unit: $150.00
Total Amount: $1500.00
Shipping Cost: $50.00
-----
Total Amount to Pay: $1550.00
-----
Goods once sold will not be returned.
Exchange within 7 days with the receipt except Saturday.
-----
-----Thank You for choosing us. Please visit again.-----

Sale processed successfully. 10 units of furniture ID 5 sold to Swasti.
Sell process completed successfully!

```

Figure 29-Selling furniture '5'

```

1, HNI Corporation, Bunk Bed, 135, $400.0
2, HNI Corporation Haworth Inc., Twin Bed, 252, $600.0
3, Achham furniture, Sleeper Sofa, 307, $200.0
4, Kimball International Inc., Corner sofa, 50, $350.0
5, Kohler Co., Armchair, 10, $150.0
6, Masco Corporation, Desk chair, 96, $100.0

```

Figure 30-Stock of furniture after selling

Conclusion

In conclusion, this project was all about working with data using Python. Python is comparatively easier to use than other programming languages when it comes to data manipulation and syntax. I tried to implement everything that was taught to us in college. I believe my project is now successful and functions as expected. I used procedural programming and created many functions in appropriate modules for different uses.

I feel more confident in Python after completing this project. Despite the frustrations of code not functioning as expected sometimes, solving those problems has improved my debugging skills(somewhat). I faced unexpected outcomes due to some indentation mistakes, which was very very very infuriating. I remember making a single invoice for a single return. Thanks to our tutor's coursework guideline video, I was able to modify my program on time. Despite all the difficulties, I look forward to facing new challenges and am excited about moving towards data science.

Although I doubt my coding was quite complex, I am quite proud of the logic I applied. I tried to make the program more user-friendly, making sure that the user would not get stuck in any part of the code due to rare situations. Finally, I am thankful to those who helped me finish this coursework. I am grateful to my teachers who guided me through every doubt, my family for providing me with resources, my friends who encouraged me, and everyone who helped me.

References

Geeks for Geeks. (2023, november 23). Retrieved from How to write pseudocode:
<https://www.geeksforgeeks.org/how-to-write-a-pseudo-code/>

Geeks for Geeks. (2024, 08 16). Retrieved from Python data structures:
<https://www.geeksforgeeks.org/python-data-structures/>

IDLE. (2020, april 10). Retrieved from Educative:
<https://www.educative.io/answers/definition-idle>

Paraschiv, L. (2023, 01 10). *FOTC*. Retrieved from Draw.io online – a step-by-step guide for users: <https://fotc.com/blog/draw-io-online-guide/>

Appendix

Code of main.py

```
from operation import order_furniture_process, sell_furniture_process
from read import read_furniture_process

# Display the available furniture
def show_furniture_list():
    furniture_data = read_furniture_process()
    for id_, details in furniture_data.items():
        print(f"ID: {id_}, Manufacturer: {details['manufacturer']}, Product:
{details['product_name']}, Quantity: {details['quantity']}, Price: ${details['price']}")

# Get integer input for validation
def validate_int_input(prompt):
    while True:
        try:
            return int(input(prompt))
        except ValueError:
            print("Invalid input. Please enter a valid number eg. 1")

# Get float input for validation
def validate_float_input(prompt):
    while True:
        try:
            return float(input(prompt))
        except ValueError:
            print("Invalid input. Please enter a valid number eg. 500.56")
```

```
def welcome():
```

```
    print("*****
*****")
```

```
        print("                Welcome To BRJ Furniture Store")
```

```
        print("                Thankot, Kathmandu | 9808190819")
```

```
        print("                Embrace the art of living beautifully—your dream
space awaits!")
```

```
    print("*****
*****")
```

```
welcome()
```

```
# Interact with the user
```

```
def main():
```

```
    while True:
```

```
        print()
```

```
        print("\nPlease select an option:")
```

```
        print()
```

```
        print("-> Press 1: *To display the available furniture*")
```

```
        print("-> Press 2: *To buy from the manufacturer*")
```

```
        print("-> Press 3: *To sell furniture to customer*")
```

```
        print("-> Press 4: *To Exit from the system*")
```

```
    print("*****
*****")
```

```
print()

choice = input("Enter your choice: ")

if choice == '1':
    show_furniture_list()
elif choice == '2':
    id_ = validate_int_input("Enter Furniture ID to order: ")
    quantity = validate_int_input("Enter quantity to order: ")
    employee_name = input("Enter your name (Employee name): ")

    order_furniture_process(id_, quantity, employee_name)
    print("Order process completed successfully!")
elif choice == '3':
    id_ = validate_int_input("Enter Furniture ID to sell: ")
    quantity = validate_int_input("Enter quantity to sell: ")
    customer_name = input("Enter customer name: ")
    shipping_cost = validate_float_input("Enter shipping cost: $")
    sell_furniture_process(id_, quantity, customer_name, shipping_cost)
    print("Sell process completed successfully!")
elif choice == '4':
    print("Exiting the program." "\n" "Thank you for visiting BRJ Furniture Store!")
    break
else:
    print("Invalid choice, please enter the number between 1 to 4.")

if __name__ == "__main__":
    main()
```

Code of operation.py

```
from read import read_furniture_process
from write import write_furniture_process, generate_invoice_process

# Order furniture from manufacturer
def order_furniture_process(id_, quantity, employee_name):
    if quantity <= 0:
        print("Quantity must be greater than zero.")
        return

    furniture_data = read_furniture_process()

    if id_ in furniture_data:
        furniture_data[id_]['quantity'] += quantity
        amount = furniture_data[id_]['price'] * quantity

    generate_invoice_process(
        'Order',
        id=id_,
        manufacturer=furniture_data[id_]['manufacturer'],
        product_name=furniture_data[id_]['product_name'],
        quantity=quantity,
        employee=employee_name,
        amount=amount
    )
```

```
    write_furniture_process(furniture_data)

    print(f"Order processed successfully. {quantity} units of furniture ID {id_} ordered.")
else:
    print(f"Error: Furniture ID {id_} not found.")

# Sell furniture to customers
def sell_furniture_process(id_, quantity, customer_name, shipping_cost):
    if quantity <= 0:
        print("Quantity must be greater than zero.")
        return

    if shipping_cost < 0:
        print("Shipping cost cannot be negative.")
        return

    furniture_data = read_furniture_process()

    if id_ in furniture_data:
        if furniture_data[id_]['quantity'] >= quantity:
            total_amount = furniture_data[id_]['price'] * quantity
            total_amount_with_shipping = total_amount + shipping_cost

            furniture_data[id_]['quantity'] -= quantity

            generate_invoice_process(
                'Sale',
                id=id_,
```

```
        manufacturer=furniture_data[id_]['manufacturer'],
        product_name=furniture_data[id_]['product_name'],
        quantity=quantity,
        customer_name=customer_name,
        price=furniture_data[id_]['price'],
        total_amount=total_amount,
        shipping_cost=shipping_cost,
        total_amount_with_shipping=total_amount_with_shipping
    )

    write_furniture_process(furniture_data)

    print(f"Sale processed successfully. {quantity} units of furniture ID {id_} sold to
{customer_name}.")

    else:

        print(f"Error: Insufficient stock. Only {furniture_data[id_]['quantity']} units
available.")

    else:

        print(f"Error: Furniture ID {id_} not found.")
```

Code of read.py

```
FURNITURE_FILE = 'furniture.txt'

# read the furniture data
def read_furniture_process():

    with open(FURNITURE_FILE, 'r') as file:
        furniture_data = {}
        for line in file:
            parts = line.strip().split(',')
            if len(parts) == 5:
                id_ = int(parts[0])
                manufacturer = parts[1]
```

```

        product_name = parts[2]
        quantity = int(parts[3])
        price = float(parts[4].replace('$', ''))
        furniture_data[id_] = {
            'manufacturer': manufacturer,
            'product_name': product_name,
            'quantity': quantity,
            'price': price
        }
    return furniture_data

```

Code of write.py

```
FURNITURE_FILE = 'furniture.txt'
```

```
INVOICE_FILE = 'invoice.txt'
```

```

def write_furniture_process(furniture_data):
    with open(FURNITURE_FILE, 'w') as file:
        for id_, details in furniture_data.items():
            line = f'{id_}, {details["manufacturer"]}, {details["product_name"]},
{details["quantity"]}, ${details["price"]}\n'
            file.write(line)

```

Generate an invoice and print the receipt

```
def generate_invoice_process(transaction_type, **kwargs):
```

```
    import datetime
```

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

Build the invoice text

```
invoice_text = (
```

```
    f'\n{"-"*100}\n'
```

```
    "***BRJ FURNITURE STORE***\n"
```

```

" ---- -Thankot, Kathmandu-----\n"

"-----\n"

f"Date: {now.strftime('%Y-%m-%d')}\n"

f"Transaction Type: {transaction_type}\n"

)

if transaction_type == 'Order':

    invoice_text += (

        "-----\n"

        f"Employee: {kwargs.get('employee')}\n"

        f"ID: {kwargs.get('id')}\n"

        f"Manufacturer: {kwargs.get('manufacturer')}\n"

        f"Product: {kwargs.get('product_name')}\n"

        f"Quantity: {kwargs.get('quantity')}\n"

        "-----\n"

        f"Amount to Pay: ${kwargs.get('amount'):.2f}\n"

        "-----\n"

        "-----Thank You for choosing us. Please visit again.-----\n"

    )

elif transaction_type == 'Sale':

    invoice_text += (

        "-----\n"

        f"Customer: {kwargs.get('customer_name')}\n"

        f"ID: {kwargs.get('id')}\n"

        f"Manufacturer: {kwargs.get('manufacturer')}\n"

        f"Product: {kwargs.get('product_name')}\n"

```



```

    f"Quantity: {kwargs.get('quantity')}\n"

    f"Price per Unit: ${kwargs.get('price'):.2f}\n"

    f"Total Amount: ${kwargs.get('total_amount'):.2f}\n"

    f"Shipping Cost: ${kwargs.get('shipping_cost'):.2f}\n"

    "-----\n"

    f"Total Amount to Pay: ${kwargs.get('total_amount_with_shipping'):.2f}\n"

    "-----\n"

    "Goods once sold will not be returned.\n"

    "Exchange within 7 days with the receipt except Saturday.\n"

    "-----\n"

    "-----Thank You for choosing us. Please visit again.-----\n"

)

# Write the invoice to the file
with open(INVOICE_FILE, 'a') as file:
    file.write(invoice_text)

# Print the invoice to the terminal
print(invoice_text)

```