



Islington college
(इसलिंग्टन कलेज)

Module Code & Module Title

CS4059NI Fundamentals Of Computing

Assessment Weightage & Type

60% Individual Coursework - 2

Year and Semester

2023-24 Spring

Student Name: Ashim Poudel

London Met ID: 22085505

College ID: NP01CP4S230145

Group: C17

Assignment Due Date: Monday, July 3, 2023

Assignment Submission Date: Friday, August 25, 2023

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

1. Introduction	4
1.1 About Python.....	4
1.2 Introduction to project.....	5
1.3 Aims & objective.....	6
a. Aims	6
b. Objective	6
2. Algorithm	7
3. Flowchart.....	10
3.1 Flowchart of Program	10
4. Pseudocode	11
a. Pseudocode of main.py	11
b. Pseudocode of opration.py.....	12
c. rentequip.py	14
d.pseudocode of returnequip.py	18
e.. Pseudocode of message.py	21
f. Pseudocode of billing.py	22
5. Data Structures	25
a. Int	25
b. String.....	26
c. boolean.....	26
d. Float	27
e. List.....	27
f. Dictionary	28
g. Tuples	28
h. sets.....	29
6. Program	30

6.1 Implementation of program for renting equipment.....	30
a Program running on IDLE.....	30
b. Termination of program	32
6.2 Implementation of the program for Returning equipment	33
a. Program run in IDLE.....	33
c. creating text file	35
7. Testing	37
7.1 Test 1: Implementation of try and except	37
.....	37
7.2 Test 2: choosing the equipment ID to rent and return	38
7.3. Test 3: File Generation by project	39
7.4 File Generation in returning equipment	42
7.5 Testing :5 Update in stock of equipment	45
8. Conclusion	46
9. Appendix	47
a. main.py.....	47
b. opration.py	48
c. message.py	50
d. rentequip.py	53
e. returnequip.py	57
g. billing.py	60

1. Introduction

1.1 About Python

A well-known programming language is Python. It was produced by Guido Van Rossum and released in 1991. Python is a well-liked choice for both novices and experts due to its readability and simplicity. There is a sizable standard library for Python. Comprises operating system interface, web service, and string processing components. In addition to. Additionally, Python provides a substantial ecosystem of outside libraries. And instruments that are easily loaded through the use of package managers like pip. The libraries here, such as Django for the web, pandas for data analysis, and NumPy for scientific computing construction, giving talents for particular sectors.

Python is an interpreted language, hence there is no need for compilation because code may be performed immediately. This facilitates quick development as well as testing and debugging of programs. Given its straightforward design, one advantage of python is how easy it is to use both dynamic typing and syntax. Python code is frequently shorter, simpler to read, and code written in other programming languages is easier to maintain. Python also provides for procedural, object-oriented, and functional programming paradigms are only a few examples.

Overall, Python is a robust and adaptable programming language that is widely used in several industries and fields. It is the best option for both beginners and experts because to its simplicity, readability, and large community. (w3schools, 2023)

1.2 Introduction to project

The project is based on Python. By using Python a system of equipment rental system has to be developed. This project aim is to provide users with the ability to rent heavy equipment which are used rarely used at home. Additionally, the system will generate a bill once the transaction is completed. The implementation of this system will utilize the python IDLE integrated development environment (IDE) for development.

The equipment rent store has been relying on manual processes to record rent equipment and returning, as well as generate bills. However, using this manual method has led to various challenges. Mistakes can be made in the entries, and there's a risk of losing physical copies of the records. Additionally, the process of generating data is slow. These issues can be effectively addressed by transitioning to an electronic system. Implementing this system in the store will significantly simplify the task of managing customer information for store managers. Furthermore, it will streamline the work for the store's accountant and facilitate the completion of record-keeping tasks.

If the user opts to rent an equipment, they are presented with a list of available equipment and are required to provide a valid ID. The system then prompts the user to enter a valid quantity of equipment they want to rent. If the specified quantity is available in stock, the system will ask if the user intends to rent more equipment. If the response is positive, the process will repeat; otherwise, the necessary steps for generating a bill will be executed. Throughout this process, the quantity of available equipment will decrease as they are sold. Ultimately, the system generates a bill and generates a file containing all the relevant bill information.

1.3 Aims & objective

a. Aims

The goal of the project is to create an application that allows renting an equipment's in an equipment rental store. Also, create an invoice at the same time.

b. Objective

- A python program will be created to allow costumer to rent equipment's.
- The program will incorporate techniques for handling exceptions, ensuring that the program runs without any disrupcion.
- The program will validate both the customer's ID and the quantity of equipment's involved in the transaction.
- After the transaction is finalized, the program will generate a text file containing the transaction details, which will serve as a bill for the customer.
- This project will also function as a chance to grasp the basics of the Python programming language.

2. Algorithm

Step 1: Start

Step 2: Display the first screen message.

Step 3: Display options to rent, return or exit.

Step 4: Provide options: 1 to rent, 2 to return, or 3 to exit.

Step 5: If option 1 is selected, proceed to step 10. Otherwise, move to step 6.

Step 6: If option 2 is chosen, proceed to step 36. Otherwise, proceed to step 9.

Step 7: If option 3 is entered, display an exit message. Otherwise, move to step 9.

Step 8: If an invalid value is entered, show an error message and return to step 3.

Step 9: Initiate the creation of a 2D list named "equipments."

Step 10: Present a table displaying the list of equipments.

Step 11: Input the equipment ID for the returning process.

Step 12: If valid ID is provided then move to step 15. If not proceed to step 23.

Step 13: If invalid ID entered then show an error message.

Step 14: Go to step 10.

Step 15: Insert the quantity of equipments to return.

Step 16: If equipment quantity is greater than 0, move to step 19.

Step 17: Show error message for invalid input and go to step 15.

Step 18: If the quantity is equal to 0, show message, and move to step 15.

Step 19: Display the message.

Step 20: Update the values in the dictionary.

Step 21: Entered the value from the list of equipments and values are stored.

Step 22: Compute the price and total using values from the “equipments”2D list.

Step 23: Inquire if the user desires to return the equipment.

Step 24: If the user entered yes, go to step 10.

Step 25: Costumer name will be entered and stored in name variable.

Step 26: If entered value is valid then, go to step 28.

Step 27: Display an error message. For incorrect input and go to Step 25

Step 28: The gross total, grand total, and delivery charge are all stated.

Step 29: Display the Date&time and name.

Step 30: Display the list Of equipment with the total line.

Step 31: Display the the gross total, grand total, and delivery charge

Step 32: In each transaction, a text file is produced with the user's information, the number of equipment rented, the gross total, the delivery charge, and the grand total.

Step 33: Display a Thank You message.

Step 34: Go to step 3.

Step 35: Create 2D list of Laptops.

Step 36: Display list of equipment.

Step 37: Entered the equipment ID that users want to rent.

Step 38: If the users entered valid id then, go to Step 41.

Step 39: Display an error message if user entered invalid and go to Step 37.

Step 40: Go to step 37.

Step 41: Entered the equipment quantity to be return.

Step 42: If the number of equipment is greater than zero then ,go to Step 45.

Step 43: Display an error message for incorrect input and go to Step 41.

Step 44: Display the available message.

Step 45: Dictionary values should be updated.

Step 46: Entered the values from the list of equipment and values are stored.

Step 47: The Price and the total is calculated and the values from the equipment List in the 2D list of "equipments".

Step 48: Ask the users to continue to rent equipment.

Step 49: If the user enter yes then, go to Step 37.

Step 50: Name of the customer will be entered and will stored in the name variable.

Step 51: If a user entered correctly then, go to Step 53.

Step 52: Display an error message if the users entered invalid number, and go to Step 51.

Step 53: The gross total, delivery chargeand grand total, are calculated.

Step 54: Display Date&Time and Name.

Step 55: Display the list Of equipment with the total line.

Step 56: Display the gross total, delivery chargeand and grand total.

Step 57: In each transaction, a text file is produced with the user's information, the number of equipment rented, the gross total, and the grand total.

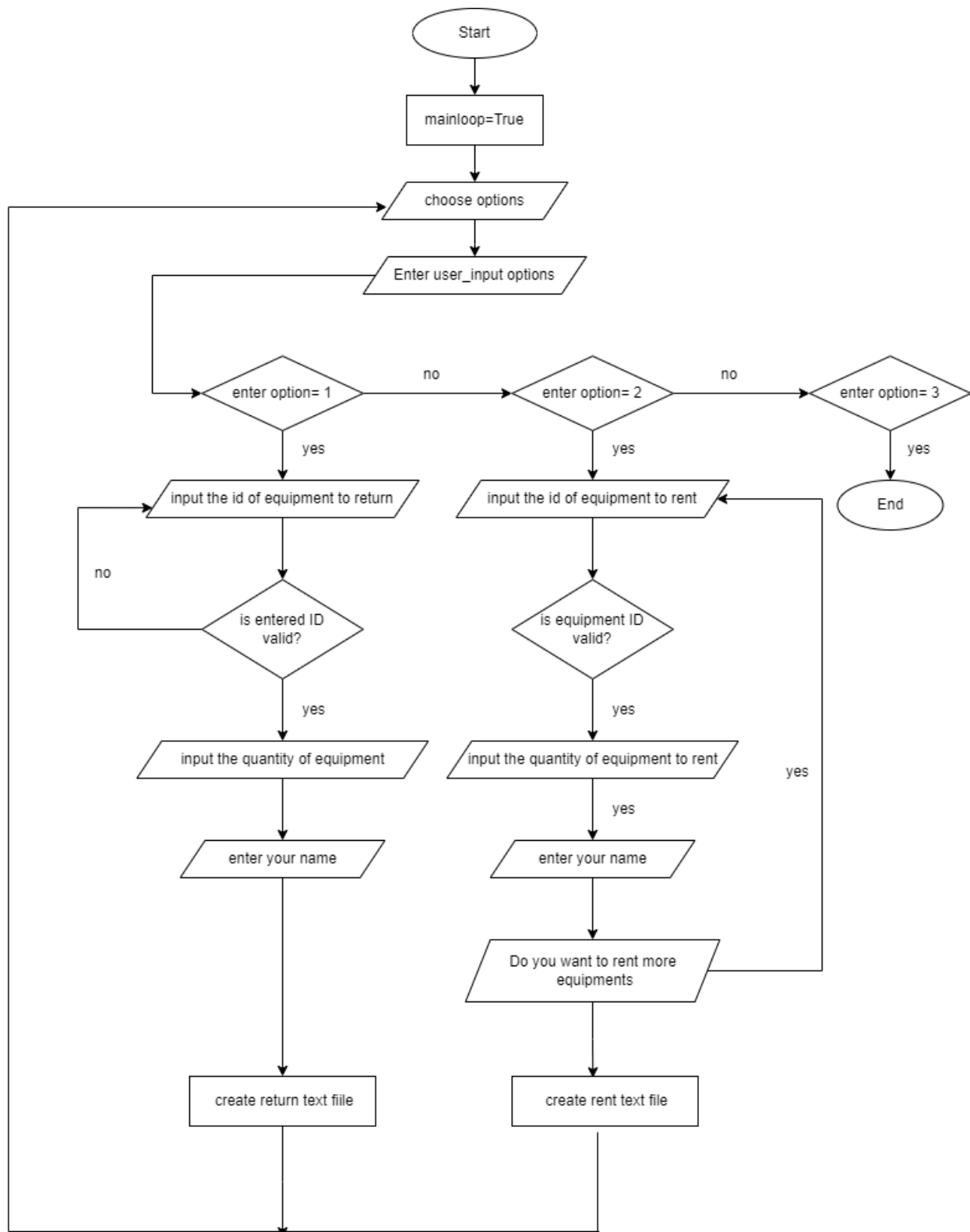
Step 58: Display Thank You message.

Step 59: Go to Step 3.

Step 60: End.

3. Flowchart

3.1 Flowchart of Program



4. Pseudocode

a. Pseudocode of main.py

Import the necessary modules: rentequip, message, and returnequip.

Display the initial screen using message.firstscreen().

Initialize the run variable to True for the main loop.

Enter a loop that continues as long as run is True.

Display the options for selection using message.select().

Attempt to get the user's choice of option.

Inside a loop, handle the case where the user's input is not a valid integer by displaying an error message using message.notvalid() and showing the initial screen again with message.firstscreen().

If the user input is a valid integer:

If the input option is 1:

Call the rentequip.rent() function to initiate the equipment rental process.

If the input option is 2:

Call the returnequip.returning() function to initiate the equipment return process.

If the input option is 3:

Display a greeting message using `message.greeting()`.

Set the run variable to False to exit the main loop.

If the input option is none of the above (invalid option):

Display an error message using `message.notvalid()`.

End the loop.

End the main loop

b. Pseudocode of operation.py

import datetime

function read_file():

data = []

open the file "equip.txt" in read mode as file

read all lines from file and store them in data

close file

return data

function date_time():

return current date and time formatted as "%y-%m-%d %H:%M"

function date():

return current date formatted as "%Y-%m-%d"

function display_items(container, main_data):

display formatted table header

for each key, value in main_data:

display formatted table row using values from the dictionary

display formatted table footer

function write_file(maindata):

open the file "equip.txt" in write mode as file

for each value in maindata:

initialize write_data as formatted string using values from the dictionary

write write_data to file

close file

function dictionary(container):

data = {}

for each index and value in container:

split the value using ',' as delimiter and initialize result in data

return data

```
data_list = call read_file()
```

```
main_data = call dictionary(data_list)
```

```
call display_items(data_list, main_data)
```

```
call write_file(main_data)
```

```
c. rentequip.py
```

```
import operation
```

```
import message
```

```
import billing
```

```
function valid_id_rent(maindata):
```

```
    valid_data = False
```

```
    while not valid_data:
```

```
        try:
```

```
            id = int(input("Enter equipment id, you want to rent"))
```

```
            if 1 <= id <= len(maindata):
```

```
                if int(maindata[id][3]) > 0:
```

```
                    valid_data = True
```

```
                    return id
```

```
            else:
```

```
        message.not_in_stock()

    else:

        message.notvalid()

    except ValueError:

        message.notvalid()


function valid_quantity_rent(maindata, id):

    valid_quantity = False

    while not valid_quantity:

        try:

            quantity = int(input("What quantity do you want to rent?"))

            if quantity > 0 and quantity <= int(maindata[id][3]):

                valid_quantity = True

                return quantity

            else:

                message.variety()

        except ValueError:

            message.notvalid()


function generate_single_item_invoice(item, customer_name, customer_phone,
delevery_charge):

    single_item_data = {
```

```
    item[0]: [item[0], item[1], item[2], '1'] # Create a temporary dictionary for the
selected item
```

```
}
```

```
    billing.generate_invoice(single_item_data, customer_name, customer_phone,
delevery_charge)
```

```
function rent():
```

```
    display "Rent equipment."
```

```
    file_content = operation.read_file()
```

```
    maindata = operation.dictionary(file_content)
```

```
    add_to_cart = []
```

```
    operation.display_items(file_content, maindata)
```

```
    secondloop = True
```

```
    while secondloop:
```

```
        id = valid_id_rent(maindata)
```

```
        message.availability()
```

```
        fn = int(valid_quantity_rent(maindata, id))
```

```
        maindata[id][3] = str(int(maindata[id][3]) - fn)
```

```
        add_to_cart.append([id, fn])
```

```
    operation.write_file(maindata)
```



```
operation.display_items(file_content, maindata)
```

```
next_action = True
```

```
while next_action:
```

```
    user_input = input("Wanna rent more?(yes/no)")
```

```
    if user_input.lower() == "yes":
```

```
        secondloop = True
```

```
        next_action = False
```

```
    else if user_input.lower() == "no":
```

```
        selected_item = maindata[id]
```

```
        customer_name = input("Enter the customer's name: ")
```

```
        customer_phone = input("Enter the customer's phone number: ")
```

```
        generate_single_item_invoice(selected_item, customer_name,  
customer_phone, 100)
```

```
        secondloop = False
```

```
        next_action = False
```

```
    else:
```

```
        message.notvalid()
```

```
        next_action = True
```

```
display
```

d.pseudocode of returnequip.py

import operation

import message

import billing

function valid_id_return(maindata):

repeat until valid_id:

try:

id = input("Enter equipment id for return")

if id is within valid range:

return id

else:

display "Invalid id"

except ValueError:

display "Invalid input"

function valid_quantity_return(maindata, id):

repeat until valid_quantity:

try:

quantity = input("Enter quantity for return")

if quantity is valid:

return quantity

else:

display "Invalid quantity"

except ValueError:

display "Invalid input"

function generate_single_item_invoice(item, customer_name, customer_phone, rental_days):

temporary_item_data = {item info}

billing.generate_return_invoice(temporary_item_data, customer_name, customer_phone, rental_days)

function returning():

display "Confirm returning equipment."

load file_content from operation.read_file()

create maindata using operation.dictionary(file_content)

repeat until user finishes returning:

display items from maindata

id = valid_id_return(maindata)

quantity = valid_quantity_return(maindata, id)

increase stock in maindata for the returned items

operation.write_file(maindata)

display updated items from maindata

repeat until user finishes returning more:

customer_input = input("Return more equipments? (yes/no)")

if customer_input is "no":

display "Bill, please return"

selected_item = maindata[id]

customer_name = input("Enter customer's name: ")

customer_phone = input("Enter customer's phone number: ")

rental_days = input("Enter rental days: ")

generate_single_item_invoice(selected_item, customer_name,
customer_phone, rental_days)

exit inner loop

else if customer_input is "yes":

exit inner loop

else:

display "Invalid input"

e.. Pseudocode of message.py

call ****firstscreen****()

repeat forever:

call select()

 user_choice = input("Enter your choice: ")

if user_choice is "1":

call ****availability****()

Call the function to handle renting

else if user_choice is "2":

call availability()

 # Call the function to handle returning

else if user_choice is "3":

exit loop

else:

call ****notvalid****()

call greeting()

f. Pseudocode of billing.py

import datetime

import message

import billing

function read_file():

Read and return the first line from "equip.txt"

function read_updated_file(list_d):

Open "equip.txt" in write mode

Write each item from list_d to the file

function billing_address(maindata, Gross_total, shippingPrice, grandTotal):

Generate an invoice file name using current date and time

Open a file with the generated name in write mode

Write header, customer info, and product details to the file

Close the file

Display invoice header, purchase details, and totals

function generate_invoice(maindata, customer_name, customer_phone, delivery_charge):

Calculate total and grand_total

Generate an invoice file name using current date and customer name

Open a file with the generated name in write mode

Write header, customer info, and product details to the file

Close the file

Display "Invoice generated successfully."

function generate_return_invoice(maindata, customer_name, customer_phone, rental_days):

Calculate total, additional_charge, and grand_total

Generate an invoice file name using current date and customer name

Open a file with the generated name in write mode

Write header, customer info, and product details to the file

Close the file

Display "Invoice generated successfully."

function write_file(maindata):

 Open "equip.txt" in write mode

 Write each item from maindata to the file

function date_time():

 Get and format the current date and time

 Return the formatted value

function date():

 Get and format the current date

 Return the formatted value

End

5. Data Structures

An efficient way to organize and store data in a computer so that it may be accessed and updated is through the use of data structures. It gives data a structure so that it may be arranged and kept in memory or on storage. Arrays, linked lists, stacks, and queues are a few examples of data structures. There are several ways to represent and store data, including using trees and graphs.

The data structure to be utilized is determined by the specific problem that needs to be addressed and the actions that need to be made with the data. A linked list, for instance, is great for adding or removing members at any point in the list, but an array is ideal for maintaining a certain number of members of the same type of components made up of similar elements.

Data types used in Python include Integer, String, Boolean, Float, List, Dictionary, Tuples, and Sets. There are many more data types, however these are a few that were utilized in the creation of the application. 2023) (Geeks for Geeks)

a. Int

In Python, integers are represented by the int data type. It can be used to represent whole numbers that have no decimal or fractional portions and are either positive, negative, or zero. In Python, integers can be any size and have an unlimited precision as long as there is enough memory to store them. 20223 (W3schools)

Example Of int():

```
messages.firstscreen()
mainloop = True
while mainloop == True:
    messages.choose()
    firstloop = False
    while firstloop == False:
        try:
            user_input = int(input("Please enter the option you want to select: "))
            firstloop = True
        except ValueError:
            print("Invalid input. Please select one of the options provided.")
            messages.choose()
```

b. String

In Python, a string is a collection of characters enclosed in single or double quotes ('...'). The most fundamental data type in Python is a string, which is used to represent textual data like names, addresses, messages, and so on. Python strings are immutable, which means that once they are created, their contents cannot be changed. But you may make new strings by joining together already-existing strings or by using string formatting methods.

Example of string

```
def write_file(maindata):  
    with open("equip.txt", "w") as file:  
        for value in maindata.values():  
            write_data = str(value[0]) + "," + str(value[1]) + "," + str(value[2])  
            file.write(write_data)
```

c. boolean

A boolean value is represented by the Python data type bool. Boolean values are logical values with the True or False alternatives. In Python, boolean values are represented by the keywords True and False. In conditional statements and loops, for example, boolean values are frequently used to control how the program proceeds. Additionally, they may be utilized in boolean expressions, which are expressions that, when evaluated, provide True or False results. 20223 (W3schools)

Example of boolean

```
if user_input.lower() == "no":  
    secondloop = False  
    next_action = False  
  
elif user_input.lower() == "yes":  
    secondloop = True  
    next_action = False
```

d. Float

A floating-point value is represented by the built-in data type float in Python. A decimal number having a fractional element that can be positive or negative is referred to as a floating-point number. Numerals like 3.14, 0.5, -2.75, and other non-integer quantities are represented by floats.

Python uses the IEEE 754 standard to implement floating-point integers, which provides a specified way of describing and executing arithmetic operations on floating-point values. constructed in 2023

```
A= 4.98
```

```
Print(type(A))
```

e. List

A list is a grouping of objects in a certain order that can be of any data type (such as texts, numbers, or other lists). One of the most adaptable and widely used data structures in programming is this one. The order or content of the items in a list can be changed after it has been generated because lists are changeable. A list's elements can be accessed by their index as well as added, deleted, or modified. Lists are made in Python by enclosing the elements in square brackets and separating them with commas. 2023 (Digital Ocean)

For example:

```
Animal = ['lion', 'tiger', 'elephant' ]
```

In this example, list of animals have showne: 'lion','tiger','elephent' there are three elemens in list. We can access individual elements of the list using their index, like this:

```
print(Animal[0]) # output: 'lion'
```

```
print(Animal[2]) # output: 'elephant'
```

f. Dictionary

In Python, a dictionary is a group of key-value pairs. You may use a key to retrieve a corresponding value by mapping each key in the dictionary to a corresponding value. Dictionaries can be added to, removed from, and modified after they are formed since they are unordered and malleable.

Dictionaries may be swiftly changed, allowing for the addition and deletion of words. It is an unordered collection, hence there is no concept of order or position. They provide a logical approach for indexing things using index/keys. They are represented by curly braces. (2023, Real Python)

Example of dictionary used in project

```
def dictionary(container):  
    data = {}  
    for index in range(len(container)):  
        data[index + 1] = container[index].replace("\n", "").split(",")  
    return data
```

g. Tuples

A tuple is the Python counterpart of a list. The difference between the two is that list items may be modified, but once a tuple is allocated, its elements cannot.

The components (elements) are enclosed in parentheses () and separated by commas to create a tuple. Although they are not required, parentheses are advised to be used. (2023, Programiz)

#Empty tuple

```
my_tuple = ()
```

```
Print(my_tuple)
```

#Tuple having integers

```
My_tuple = (6,7,8,9)
```

```
Print(my_tuple)
```

#Tuple with mixed datatypes

```
My_tuple = (1,"Hello",3.4)
```

```
Print(my_tuple)
```

h. sets

To store several items in a single variable, use sets. The four built-in data types in Python for storing collections of data are list, tuple, dictionary, and set. Each has a unique set of attributes and uses. Curly brackets are used for writing sets. An unsorted, immutable*, and unindexed collection is referred to as a set. A set of things is said to be unordered if they are not in any specific order. Set pieces cannot be accessible using an index or key and may appear in a different sequence each time they are utilized. The components of a set cannot be altered after the set has been created since they are immutable. You cannot modify the items in a set once it has been formed, but you may delete items and add new ones. (W3schools, 2023)

For example:

```
hisset = {"lion", "tiger", "elephant"} #this is set
```

```
print(len(hisset))
```

#set can have different data types

6. Program

6.1 Implementation of program for renting equipment

a Program running on IDLE

```

*****
*****
                Welcome To The equipment rental store
                kathmandu, Banepa
                Ph.no: 9867564324 , 071-87654                Pan no:0014921
*****
*****

Enter 1 to rent.
Enter 2 to returned.
Enter 3 to Exit.

chhose an option:|

```

The user is first given the option to choose whether he or she wants to rent, return, or Exit the Program. The visitor will be sent to the rent product services page if they choose one of the choices. The user is sent to the return product services if they choose two selections.

```

chhose an option:1

Rent equipment.

-----
|id   |Equipments Name          |Brand           |Prise    |Quantity |
-----
|1    |Velvet Table Cloth       |Saathi          |$8        |20       |
|2    |Microphone Set           |Audio Technica  |$189      |15       |
|3    |Disco Light Set          |Sonoff          |$322      |24       |
|4    |7.1 Surround Sound Speaker Set |Dolby          |$489      |4        |
|5    |Dinner Table 8x5         |Panda Furnitures|$344      |8        |
-----

Enter equipment id, you want to rent

```

After entering 1 user can now rent there choice of equipment. Equipment details are displayed clearly.

```
Enter equipment id, you want to rent 3
```

```
-----
Equipment available:
-----
```

```
What quantity do you want to rent?20
```

```
-----
```

id	Equipments Name	Brand	Prise	Quantity	
1	Velvet Table Cloth	Saathi	\$8	20	
2	Microphone Set	Audio Technica	\$189	15	
3	Disco Light Set	Sonoff	\$322	4	
4	7.1 Surround Sound Speaker Set	Dolby	\$489	4	
5	Dinner Table 8x5	Panda Furnitures	\$344	8	

```
-----
```

```
-----
Wanna rent more?(yes/no)|
```

After selecting equipment ID if equipment is available then user have to enter required quantity of equipment. And the table of equipment appears with update of quantity.

```
Wanna rent more?(yes/no)yes
```

```
Enter equipment id, you want to rent5
```

```
-----
Equipment available:
-----
```

```
What quantity do you want to rent?4
```

```
-----
```

id	Equipments Name	Brand	Prise	Quantity	
1	Velvet Table Cloth	Saathi	\$8	20	
2	Microphone Set	Audio Technica	\$189	15	
3	Disco Light Set	Sonoff	\$322	17	
4	7.1 Surround Sound Speaker Set	Dolby	\$489	4	
5	Dinner Table 8x5	Panda Furnitures	\$344	4	

```
-----
```

```
Wanna rent more?(yes/no)no
```

```
Enter the customer's name: Ashim Poudel
```

```
Enter the customer's phone number: 9867876543
```

```
Invoice generated successfully.
```

```
*****
                Thank You for Visiting our store! We hope to see you soon.
                Have a Great Day!!
*****
```

When the user enter no , then billing process will start. User have to provide name and phone number for bill.

```
*****
                        Date: 2023-08-25-08-22-28
                        Invoice for Ashim Poudel
                        Phone Number: 9867876543
*****

Details of the purchase:
=====
Product name           Brand           Quantity   Unit Price   Total
=====
7.1 Surround Sound Speaker Set   Dolby           1           $489        $489.00
=====

Gross Total: $489.00
Delevery Charge: $100.00
Grand Total: $589.00
```

A bill is generated with customer details. The list of equipment rented is displayed with their total cost according to the quantity they have rented. Bill contain purchasing date and time.

b. Termination of program

```
*****
Welcome To The equipment rental store
kathmandu, Banepa
Ph.no: 9867564324 , 071-87654           Pan no:0014921
*****
*****
Enter 1 to rent.
Enter 2 to returned.
Enter 3 to Exit.

chhose an option:3

*****
Thank You for Visiting our store! We hope to see you soon.
Have a Great Day!!
*****
```


6.2 Implementation of the program for Returning equipment

a. Program run in IDLE

```
*****
*****
                        Welcome To The equipment rental store
                        kathmandu, Banepa
                        Ph.no: 9867564324 , 071-87654                      Pan no:0014921
*****
*****

Enter 1 to rent.
Enter 2 to returned.
Enter 3 to Exit.
```

First, the user is prompted to select whether he or she wishes to rent or return a equipment or Exit the program. If the user selects 2 of the options, he or she will be routed to the return equipment services or. If the user selects 3 options, he or she is routed to exit the application.

```
Enter 1 to rent.
Enter 2 to returned.
Enter 3 to Exit.

chhose an option:2

  Conferm returning equipment.

-----
|id  |Equipments Name          |Brand          |Prise          |Quantity  |
-----
|1   |Velvet Table Cloth       |Saathi         |$8             |20        |
|2   |Microphone Set           |Audio Technica |$189           |15        |
|3   |Disco Light Set          |Sonoff         |$322           |24        |
|4   |7.1 Surround Sound Speaker Set |Dolby         |$489           |4         |
|5   |Dinner Table 8x5         |Panda Furnitures |$344           |8         |
-----

Enter equipment id you wanna return:|
```

After selecting option 2 user is now in return process. Application ask user details about equipment and generate bills.

id	Equipments Name	Brand	Prise	Quantity
1	Velvet Table Cloth	Saathi	\$8	20
2	Microphone Set	Audio Technica	\$189	15
3	Disco Light Set	Sonoff	\$322	24
4	7.1 Surround Sound Speaker Set	Dolby	\$489	4
5	Dinner Table 8x5	Panda Furnitures	\$344	8

Enter equipment id you wanna return:5
Enter a number of equipment you wanna return:4

id	Equipments Name	Brand	Prise	Quantity
1	Velvet Table Cloth	Saathi	\$8	20
2	Microphone Set	Audio Technica	\$189	15
3	Disco Light Set	Sonoff	\$322	24
4	7.1 Surround Sound Speaker Set	Dolby	\$489	4
5	Dinner Table 8x5	Panda Furnitures	\$344	12

Want to return more equipments?(yes/no)

After returning the product the quantity of product increase in table and it ask user want to return more equipment if they have. If user have more equipment user have to select yes and if no then enter no.

```
Want to return more equipments?(yes/no)no
Bill please return
```

After entering no it will generate bill.

```
Enter the customer's name: Ashim poudel
Enter the customer's phone number: 9876543212
Enter the number of rental days: 6
Invoice generated successfully.
```

If customer return product after five days they have to pay \$50 in daily basis what days they keep the equipment after crushing day five

```

*****
Date: 2023-08-25-09-01-54
Invoice for Ashim poudel
Phone Number: 9876543212
*****

Details of the purchase:
=====
Product name          Brand          Quantity  Unit Price  Total
=====
Dinner Table 8x5      Panda Furnitures  1          $344        $344.00
=====

Gross Total: $344.00

Additional Charge (if applicable): $50.00

Grand Total: $394.00

```

A bill is generated which contains the user details i.e., his/her name and date&time of the purchase is also present in the bill which is updated from the date time() function. The list of the equipment that were returned is displayed with their updated price according to the quantity ordered and grand total of the product is also displayed with penalty of \$50.

c. creating text file

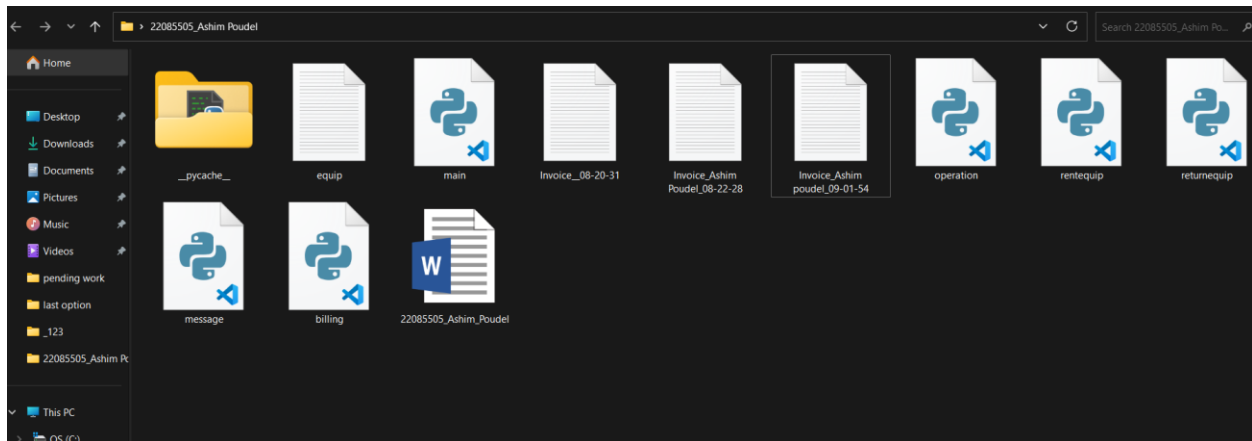
```

def billing_address( maindata , Gross_total ,shippingPrice, grandTotal):
    current = datetime.datetime.now()
    dateTimeForInv = current.strftime("%H-%M-%S")
    dateTimeForBill = current.strftime("%Y-%m-%d-%H-%M-%S")

    fileName = "This is name" + dateTimeForInv + ".txt"
    file = open(fileName,"w")
    file.write("*****\n")
    file.write("\t\t\t\t\t Invoice" + "\n")
    file.write("*****\n")
    file.write("\n")
    file.write("Laptop Details are displayed below : ")
    file.write("\n")
    file.write("\n")
    file.write("Date and Time of the purchase : " + dateTimeForBill + "\n")
    file.write("\n")
    file.write("Details of the purchase : ")
    file.write("\n")
    file.write("Laptop name \t\t Laptop brand \t\t Quantity \t\t Unit Price \t\t Total" + "\n")
    file.write("*****\n")
    for item in maindata:
        file.write(str(item[0]) + " " + str(item[1]) + "\t\t\t\t\t " + str(item[2]) + "\t\t\t\t\t " + str(item[3]) + "\t\t\t\t\t " + "\n")
    file.write("\n")
    file.write("Gross Total is : " + str(Gross_total) + "\n")
    file.write("\n")
    file.write("Shipping cost is : " + str(shippingPrice) + "\n")
    file.write("\n")
    file.write("Grand Total is : " + str(grandTotal) + "\n")
    file.write("\n")

```

In the picture above, a section of the code is shown where a text file is generated and the bill data are saved. The text file contains the user's information as well as the overall details of equipment rent.



The user who sold the goods during the previous activities is represented by the invoice that is displayed above. The text file, which was created after the bill creation was displayed in the IDLE, contains the given invoice. The text file is given the user's name and the purchase date as names.

```
File Edit View

*****
Date: 2023-08-25-08-22-28
Invoice for Ashim Poudel
Phone Number: 9867876543
*****

Details of the purchase:
=====
Product name          Brand          Quantity    Unit Price    Total
=====
7.1 Surround Sound Speaker Set    Dolby          1           $489          $489.00
=====

Gross Total: $489.00
Delevery Charge: $100.00
Grand Total: $589.00
```

7. Testing

7.1 Test 1: Implementation of try and except

Objective: To demonstrate the applicability of the try and except

Action	Not valid information entered.
Expected Result	The exception should be handled by the program.
Actual result	The exception was handled by the program
Conclusion	The test was successful.

```
Enter 1 to rent.  
Enter 2 to returned.  
Enter 3 to Exit.
```

```
chhose an option:7
```

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

```
Invalid input, please choose given options:
```

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

```
Enter 1 to rent.  
Enter 2 to returned.  
Enter 3 to Exit.
```

```
chhose an option:|
```

7.2 Test 2: choosing the equipment ID to rent and return

Objective: To enter an incorrect id and show invalid message.

Action	The invalid ID of equipment is entered. Negative int wad entered.
Excepted Result	The application should display not valid message
Actual Result	The application showed invalid message.
conclusion	Successful!

```

Enter 1 to rent.
Enter 2 to returned.
Enter 3 to Exit.

chhose an option:-2

-----
-----
                Invalid input, please choose given options:
-----
-----

Enter 1 to rent.
Enter 2 to returned.
Enter 3 to Exit.

```

```

Enter 1 to rent.
Enter 2 to returned.
Enter 3 to Exit.

chhose an option:6

-----
-----
                Invalid input, please choose given options:

```

7.3. Test 3: File Generation by project

Objective	To show file generating of buy product.
Action	The rent option was chosen. Several equipment were purchased. In the shell, bill was displayed. The bill for the refilled equipment was saved as a text file.
Expected Result	A text file of the bought laptops should be generated.
Actual Result	A text file of the rent equipment was generated.
Conclusion	Successful!

```

-----
|id   |Equipments Name          |Brand          |Prise   |Quantity |
-----
|1    |Velvet Table Cloth       |Saathi         |$8      |20       |
|2    |Microphone Set           |Audio Technica |$189    |15       |
|3    |Disco Light Set          |Sonoff         |$322    |24       |
|4    |7.1 Surround Sound Speaker Set |Dolby         |$489    |4        |
|5    |Dinner Table 8x5         |Panda Furnitures|$344    |8        |
-----

Enter equipment id, you want to rent2

-----
Equipment available:
-----

What quantity do you want to rent5

```

```
Wanna rent more?(yes/no)yes
Enter equipment id, you want to rent5
```

```
-----
Equipment available:
-----
```

```
What quantity do you want to rent?3
-----
```

id	Equipments Name	Brand	Prise	Quantity	
1	Velvet Table Cloth	Saathi	\$8	20	
2	Microphone Set	Audio Technica	\$189	10	
3	Disco Light Set	Sonoff	\$322	24	
4	7.1 Surround Sound Speaker Set	Dolby	\$489	4	
5	Dinner Table 8x5	Panda Furnitures	\$344	5	

```
Wanna rent more?(yes/no)yes
Enter equipment id, you want to rent3
```

```
-----
Equipment available:
-----
```

```
What quantity do you want to rent?20
-----
```

id	Equipments Name	Brand	Prise	Quantity	
1	Velvet Table Cloth	Saathi	\$8	20	
2	Microphone Set	Audio Technica	\$189	10	
3	Disco Light Set	Sonoff	\$322	4	
4	7.1 Surround Sound Speaker Set	Dolby	\$489	4	
5	Dinner Table 8x5	Panda Furnitures	\$344	5	

```
-----
Wanna rent more?(yes/no)no
Enter the customer's name: JOHN HANRY
Enter the customer's phone number: 87878787
Invoice generated successfully.
```



```

*****
Date: 2023-08-25-10-00-56
Invoice for JOHN HANRY
Phone Number: 876576444
*****

```

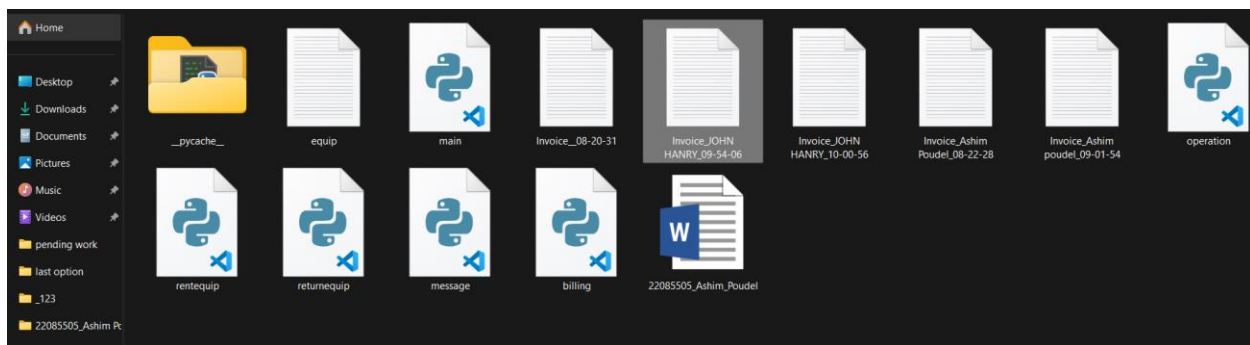
Details of the purchase:

Product name	Brand	Quantity	Unit Price	Total
7.1 Surround Sound Speaker Set	Dolby	1	\$489	\$489.00
Microphone Set	Audio Technica	4	\$189	\$756.00
Dinner Table 8x5	Panda Furnitures	3	\$344	\$1002.00

```

=====
Gross Total: $2247
Delevery Charge: $100.00
Grand Total: $2347.00

```



7.4 File Generation in returning equipment

Objective	To show file generating of Selling product.
Action	The return option was chosen
Excepted result	A text file of returning equipments.
Actual result	A text file of returning equipment generated.
Conclusion	Successful!!

```
Enter 1 to rent.
Enter 2 to returned.
Enter 3 to Exit.
```

```
chhose an option:2
```

```
Conferm returning equipment.
```

```
-----
|id  |Equipments Name          |Brand          |Prise          |Quantity |
-----
|1   |Velvet Table Cloth       |Saathi         |$8             |20       |
|2   |Microphone Set           |Audio Technica |$189           |15       |
|3   |Disco Light Set          |Sonoff         |$322           |24       |
|4   |7.1 Surround Sound Speaker Set |Dolby         |$489           |4        |
|5   |Dinner Table 8x5         |Panda Furnitures |$344           |8        |
-----
```

```
Enter equipment id you wanna return:
```

Option 2 is selected.

```

-----
Enter equipment id you wanna return:4
Enter a number of equipment you wanna return5
-----
|id  |Equipments Name          |Brand          |Prise          |Quantity |
-----
|1   |Velvet Table Cloth       |Saathi         |$8             |20       |
|2   |Microphone Set           |Audio Technica |$189           |15       |
|3   |Disco Light Set          |Sonoff         |$322           |24       |
|4   |7.1 Surround Sound Speaker Set |Dolby         |$489           |9        |
|5   |Dinner Table 8x5         |Panda Furnitures |$344          |8        |
-----
Want to return more equipments?(yes/no)|

```

After returning five equipment the number of equipment in table was increased

So further others can rent the equipment.

```

-----
Want to return more equipments?(yes/no)no
Bill please return
Enter the customer's name: Ashim Poudel
Enter the customer's phone number: 989898989898
Enter the number of rental days: 5
Invoice generated successfully.

```

In this I have entered days of rental is five for five days there is no penalty while returning.

```

*****
Date: 2023-08-25-10-25-36
Invoice for Ashim Poudel
Phone Number: 989898989898
*****

Details of the purchase:
=====
Product name          Brand          Quantity    Unit Price    Total
=====
7.1 Surround Sound Speaker Set    Dolby         5           $489          $2445.00
=====

Gross Total: $2445.00
Additional Charge (if applicable): $0.00
Grand Total: $2445.00

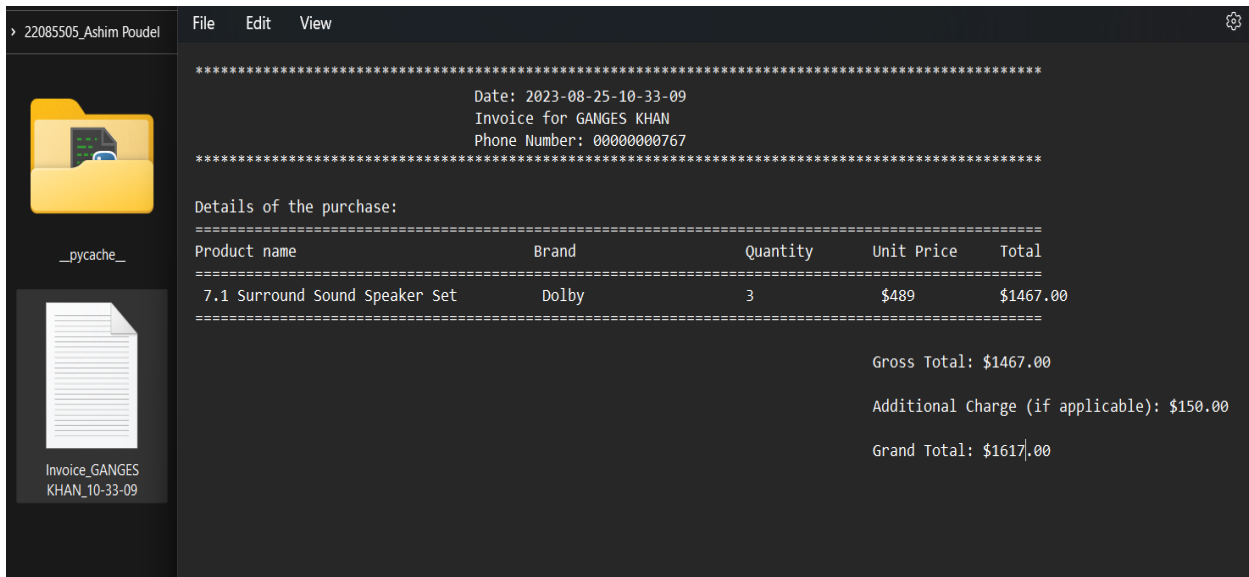
```

```

Want to return more equipments?(yes/no)no
Bill please return
Enter the customer's name: GANGES KHAN
Enter the customer's phone number: 00000000767
Enter the number of rental days: 8
Invoice generated successfully.

```

Returning days has cross five days so penalty for three days should include in bill.



The screenshot shows a file explorer window with a sidebar containing a folder icon and a file icon labeled 'Invoice_GANGES KHAN_10-33-09'. The main pane displays the contents of the invoice file, which is a text-based document with a dark background and light-colored text.

Date: 2023-08-25-10-33-09
Invoice for GANGES KHAN
Phone Number: 00000000767

Details of the purchase:

Product name	Brand	Quantity	Unit Price	Total
7.1 Surround Sound Speaker Set	Dolby	3	\$489	\$1467.00

Gross Total: \$1467.00
Additional Charge (if applicable): \$150.00
Grand Total: \$1617.00

Penalty of three days has included in bill.

7.5 Testing :5 Update in stock of equipment

Objective	To show update in stock of equipment.
Action	Equipment is re stocked.
Expected result	While restocking the equipment, the quantity of equipment should increase.
Actual result	Quantity is increased.
Conclusion	Successful!!

```

Enter 1 to rent.
Enter 2 to returned.
Enter 3 to Exit.

chhose an option:2

Conferm returning equipment.

-----
|id  |Equipments Name          |Brand          |Prise  |Quantity |
-----
|1   |Velvet Table Cloth       |Saathi         |$8     |20       |
|2   |Microphone Set           |Audio Technica |$189   |15       |
|3   |Disco Light Set          |Sonoff         |$322   |24       |
|4   |7.1 Surround Sound Speaker Set |Dolby         |$489   |4        |
|5   |Dinner Table 8x5         |Panda Furnitures|$344   |8        |
-----

Enter equipment id you wanna return:

```

As you can see the stock of id 4 there are four quantity. After returning equipment ID of four stock is replace.

```

-----
Enter equipment id you wanna return:4
Enter a number of equipment you wanna return5

-----
|id  |Equipments Name          |Brand          |Prise  |Quantity |
-----
|1   |Velvet Table Cloth       |Saathi         |$8     |20       |
|2   |Microphone Set           |Audio Technica |$189   |15       |
|3   |Disco Light Set          |Sonoff         |$322   |24       |
|4   |7.1 Surround Sound Speaker Set |Dolby         |$489   |9        |
|5   |Dinner Table 8x5         |Panda Furnitures|$344   |8        |
-----

Want to return more equipments?(yes/no)

```

You can clearly see the stock of ID no 4.

8. Conclusion

This project's involvement with Python has been difficult the past few weeks. Python is, in my opinion, both easier and more pleasurable to develop in. To print something, just type "print"; complex syntax is not necessary. It was difficult for me to keep up with all the functions, classes, and other components because this was my first time learning Python. As time went on, it became clear how to use every aspect of Python, and a sense of proficiency in Python programming began to take shape.

The purpose of this project was to create an application that would allow a equipment store to rent equipment. Learning about how Python functions and how to use it was shared and acquired while working on this project. The usage of functions in programming, program modularity, the use of comments and doc strings, and Python exception handling were all discussed. The most valuable lesson I learned about Python functions was from this project. Programming has shown to be easier and more convenient when using functions.

Many problems were faced while working on this project. The code would sometimes fail to operate properly, the application would crash, and there were issues with exception handling. Many difficulties came when working on flowcharts and algorithms. All of the challenges were finally handled one by one using slides, and internet tools.

9. Appendix

a. main.py

```
import rentequip
```

```
import message
```

```
import returnequip
```

```
message.firstscreen()
```

```
run = True
```

```
while run:
```

```
    message.sellelect()
```

```
    correct_option = False
```

```
    while not correct_option:
```

```
        try:
```

```
            option = int(input("chhose an option:"))
```

```
            correct_option = True
```

```
        except ValueError:
```

```
            message.notvalid()
```

```
            message.firstscreen()
```

```
    if option == 1:
```

```
        rentequip.rent()
```

```
elif option == 2:  
    return equip.returning()
```

```
elif option == 3:  
    message.greading()  
    run = False
```

```
else:  
    message.notvalid()
```

b. opration.py

```
import datetime
```

```
def read_file():  
    data = []  
    with open("equip.txt", "r") as file:  
        data = file.readlines()  
    return data
```

```
def date_time():  
    return datetime.datetime.now().strftime("%y-%m-%d %H:%M")
```



```

def date():

    return datetime.datetime.now().strftime("%Y-%m-%d")


def display_items(container, main_data):

    print("-----")
    print("-----")

    print("|{:<5}|{:<45}|{:22}|{:12}|{:<10}|".format("id", "Equipments Name", "Brand", "Prise",
"Quantity"))

    print("-----")
    print("-----")

    for key, Value in main_data.items():

        print("|{:<5}|{:<45}|{:22}|{:12}|{:<10}|".format(key, Value[0], Value[1], Value[2],
Value[3]))

    print("-----")
    print("-----")


def write_file(maindata):

    with open("equip.txt", "w") as file:

        for value in maindata.values():

            write_data = str(value[0]) + "," + str(value[1]) + "," + str(value[2]) + "," + str(value[3])

            file.write(write_data)


def dictionary(container):

    data = {}

```

```

for index in range(len(container)):

    data[index + 1] = container[index].replace("\n", "").split(",")

return data

```

c. message.py

```
def firstscreen():
```

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

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

```
    print("                Welcome To The equipment rental store
")
```

```
    print("                kathmandu, Banepa
")
```

```
    print("                Ph.no: 9867564324 , 071-87654                Pan
no:0014921                ")
```

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

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

```
def sellect():
```

```
    print("Enter 1 to rent.")
```

```
    print("Enter 2 to returned.")
```

```
    print("Enter 3 to Exit.")
```

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

```
def availability():
```

```
    print("\n-----  
-----")  
  
    print("        Equipment available:        ")  
  
    print("-----  
-----\n")
```

```
def renting():
```

```
    print("\n-----  
-----")  
  
    print("        You Have Rented Equipment Sucessfully:  
")  
  
    print("-----  
-----\n")
```

```
def returning():
```

```
    print("\n-----  
-----")  
  
    print("        You Have return Equipment Sucessfully:  
")  
  
    print("-----  
-----\n")
```

```
def not_in_stock():
```

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

```
    print("        Your choice of equipment is out of stock:  
")
```

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

```
def notvalid():
```

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

```
    print("        Invalid input, please choose given options:  
")
```

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

```
def variety():
```

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

```
    print("        Out Of Range:                ")
```

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

```
def greading():
```

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

```

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

    print("                Thank You for Visiting our store! We hope to see you soon.
")

    print("                Have a Great  Day!!                ")

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

```

d. rentequip.py

```
import operation
```

```
import message
```

```
import billing
```

```

def valid_id_rent(maindata):

    valid_data = False

    while not valid_data:

        try:

            id = int(input("Enter equipment id, you want to rent"))

            if 1 <= id <= len(maindata):

                if int(maindata[id][3])>0:

                    valid_data = True

```

```
        return id

    else:

        message.not_in_stock()

    else:

        message.notvalid()

except ValueError:

    message.notvalid()


def valid_quantity_rent(maindata, id):

    valid_quantity = False

    while not valid_quantity:

        try:

            quantity = int(input("What quantity do you want to rent?"))

            if quantity > 0 and quantity <= int(maindata[id][3]):

                valid_quantity = True

                return quantity

            else:

                message.variety()

        except ValueError:

            message.notvalid


def generate_single_item_invoice(item, customer_name,
customer_phone,delevery_charge):
```

```
single_item_data = {  
    item[0]: [item[0], item[1], item[2], '1'] # Create a temporary dictionary for the  
selected item  
}  
  
billing.generate_invoice(single_item_data, customer_name,  
customer_phone,delevery_charge)
```

```
def rent():  
  
    print("\n Rent equipment. \n")  
  
    file_content = operation.read_file()  
    maindata =operation.dictionary(file_content)  
  
    add_to_cart = []  
    operation.display_items(file_content, maindata)  
    secondloop = True  
    while secondloop:  
        id = valid_id_rent(maindata)  
        message.availability()  
        fn = int(valid_quantity_rent(maindata, id))  
        maindata[id][3] = str(int(maindata[id][3]) - fn)  
        add_to_cart.append([id, fn])
```

```
operation.write_file(maindata)

operation.display_items(file_content, maindata)

next_action = True

while next_action == True:

    user_input = input("Wanna rent more?(yes/no)")

    if user_input.lower() == "yes":

        secondloop = True

        next_action = False

    elif user_input.lower() == "no":

        selected_item = maindata[id]

        customer_name = input("Enter the customer's name: ")

        customer_phone = input("Enter the customer's phone number: ")

        generate_single_item_invoice(selected_item, customer_name,
customer_phone,100)

        secondloop=False

        next_action = False

    else:

        message.notvalid()

        next_action = True
```



```
print()
```

e. `returnequip.py`

```
import operation
```

```
import message
```

```
import billing
```

```
def valid_id_return(maindata):
```

```
    valid_id = False
```

```
    while valid_id == False:
```

```
        try:
```

```
            id = int(input("Enter equipment id you wanna return:"))
```

```
            if 1 <= id <= len(maindata):
```

```
                valid_id = True
```

```
                return id
```

```
            else:
```

```
                message.notvalid()
```

```
        except ValueError:
```

```
            message.notvalid()
```

```
def valid_quantity_return(maindata, id):
```

```
    valid_quantity = False
```

```
while valid_quantity == False:
```

```
    try:
```

```
        quantity = int(input("Enter a number of equipment you wanna return"))
```

```
        if quantity > 0:
```

```
            valid_quantity = True
```

```
            return quantity
```

```
        else:
```

```
            message.notvalid()
```

```
    except ValueError:
```

```
        message.notvalid()
```

```
def generate_single_item_invoice(item, customer_name, customer_phone,  
rental_days):
```

```
    single_item_data = {
```

```
        item[0]: [item[0], item[1], item[2], '1'] # Create a temporary dictionary for the  
selected item
```

```
    }
```

```
    billing.generate_return_invoice(single_item_data, customer_name, customer_phone,  
rental_days)
```

```
def returning():
```

```
    print("\n Confirm returning equipment. \n")
```

```
file_content = operation.read_file()

maindata = operation.dictionary(file_content)


add_to_cart = []

thirdloop = True

while thirdloop == True:

    operation.display_items(file_content, maindata)

    id = valid_id_return(maindata)

    fn = int(valid_quantity_return(maindata, id))

    maindata[id][3] = str(int(maindata[id][3]) + fn)

    add_to_cart.append([id, fn])


operation.write_file(maindata)

operation.display_items(file_content, maindata)


again = True

while again == True:

    customer_input = input("Want to return more equipments?(yes/no)")

    if customer_input.lower() == "no":

        thirdloop = False

        again = False

        print('Bill please return')
```

```
    selected_item = maindata[id]

    customer_name = input("Enter the customer's name: ")

    customer_phone = input("Enter the customer's phone number: ")

    rental_days = int(input("Enter the number of rental days: "))

    generate_single_item_invoice(selected_item, customer_name,
customer_phone, rental_days)
```

```
elif customer_input.lower() == "yes":
```

```
    thirdloop = True
```

```
    again = False
```

```
else:
```

```
    message.notvalid()
```

```
    again = True
```

```
print()
```

```
if __name__ == "__main__":
```

```
    returning()
```

[g. billing.py](#)

```
import datetime
```

```
import message
```

```
import billing
```

```
def read_file():
```

```
    data = []
```

```
    with open ("equip.txt","r") as file:
```

```
        date = file.readline()
```

```
    return data
```

```
def read_updated_file(list_d):
```

```
    file = open("equip.txt","w")
```

```
    for i in list_d.values():
```

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

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

```
    file.close()
```

```
def billing_address( maindata , Gross_total ,shippingPrice, grandTotal):
```

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

```
    dateTimeForInv = current.strftime("%H-%M-%S")
```

```
    dateTimeForBill = current.strftime("%Y-%m-%d-%H-%M-%S")
```

```
    fileName = "THis is name" + dateTimeForInv + ".txt"
```

```
    file = open(fileName,"w")
```

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

```
file.write("\t\t\t Invoice" + "\n")
```

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

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

```
file.write("Laptop Details are displayed below : ")
```

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

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

```
file.write("Date and Time of the purchase : " + dateTimeForBill + "\n")
```

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

```
file.write("Details of the purchase : ")
```

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

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

```
file.write("Laptop name \t\t Laptop brand \t\t Quantity \t\t Unit Price \t\t Total" + "\n")
```

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

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

```
for item in maindata:
```

[illegible]

#printing the invoice

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

```
print("\t\t\t\t Invoice" + "\n")
```

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

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

```
print("Laptop Details are displayed below : ")
```

```
print("\n")

print("Date and Time of the purchase : " + dateTimeForBill + "\n")

print("Details of the purchase : ")

print("\n")


print("=====
=====
=====")

    print("Laptop name \t\t      Laptop brand \t\t      Quantity \t\t Unit Price \t\t Total" +
"\n")


print("=====
=====
=====")

    print("\n")

    for item in maindata:

        print(str(item[0]) + "\t\t" +str(item[1]) + "\t\t"+str(item[2]) + "\t\t " +str(item[3]) + "\t\t
" + "$"+str(item[4]) + "\n")

    print("\n")


print("=====
=====
=====")

    print("\n")

    print(" \t\t\t\t\t\t\t\t\t\t Gross Total is   : " + str(Gross_total) + "\n")

    print(" \t\t\t\t\t\t\t\t\t\t Shipping cost is : " + str(shippingPrice) + "\n")

    print(" \t\t\t\t\t\t\t\t\t\t Grand Total   is : " + str(grandTotal) + "\n")

    print("\n")
```



```
def generate_invoice(maindata, customer_name, customer_phone,delevery_charge):
```

```
    total = sum(int(item[2].replace('$', '').replace(',', '')) * int(item[3]) for item in
maindata.values())
```

```
    grand_total = total + delevery_charge
```

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

```
    dateTimeForInv = current.strftime("%H-%M-%S")
```

```
    dateTimeForBill = current.strftime("%Y-%m-%d-%H-%M-%S")
```

```
    fileName = f"Invoice_{customer_name}_{dateTimeForInv}.txt"
```

```
    file = open(fileName, "w")
```

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

```
    file.write(f"\t\t\t\t Date: {dateTimeForBill}\n")
```

```
    file.write(f"\t\t\t\t Invoice for {customer_name}\n")
```

```
    file.write(f"\t\t\t\t Phone Number: {customer_phone}\n") # Include the customer's
phone number
```

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

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

```

file.write("Details of the purchase:\n")

file.write("=====  

=====\\n")

file.write("{:<40}{:<25}{:<15}{:<15}{:<15}\\n".format("Product name", "Brand",  

"Quantity", "Unit Price", "Total"))

file.write("=====  

=====\\n")

for item in maindata.values():

    product_name = item[0]

    brand = item[1]

    quantity = item[3]

    unit_price = item[2]

    total_price = "${:.2f}".format(int(item[2].replace('$', ' ').replace(',', ' ')) * int(item[3]))

    file.write("{:<40}{:<25}{:<15}{:<15}{:<15}\\n".format(product_name, brand, quantity,  

unit_price, total_price))

file.write("=====  

=====\\n")

file.write("\\n")

file.write("\\t\\t\\t\\t\\t\\t\\t\\t\\t\\tGross Total: {:.2f}\\n".format(total))

```

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

file.write("\t\t\t\t\t\t\t\t\t\tDelevery Charge: ${:.2f}\n".format(delevery_charge))

file.write("\n")

file.write("\t\t\t\t\t\t\t\t\t\tGrand Total: ${:.2f}\n".format(grand_total))

file.write("\n")

file.close()
```

```
print("Invoice generated successfully.")
```

```
message.greading()
```

```
def generate_return_invoice(maindata, customer_name, customer_phone, rental_days):
```

```
    total = sum(int(item[2].replace('$', '').replace(',', '')) * int(item[3]) for item in
maindata.values())
```

```
    additional_charge = 0
```

```
    if rental_days > 5:
```

```
        additional_charge = 50 * (rental_days - 5)
```

```
    grand_total = total + additional_charge
```

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

```
    dateTimeForInv = current.strftime("%H-%M-%S")
```

```
    dateTimeForBill = current.strftime("%Y-%m-%d-%H-%M-%S")
```

```
fileName = f"Invoice_{customer_name}_{dateTimeForInv}.txt"
```

```
file = open(fileName, "w")
```

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

```
file.write(f"\t\t\t\t Date: {dateTimeForBill}\n")
```

```
file.write(f"\t\t\t\t Invoice for {customer_name}\n")
```

```
file.write(f"\t\t\t\t Phone Number: {customer_phone}\n") # Include the customer's
phone number
```

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

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

```
file.write("Details of the purchase:\n")
```

```
file.write("=====\n")
```

```
file.write("{:<40}{:<25}{:<15}{:<15}{:<15}\n".format("Product name", "Brand",
"Quantity", "Unit Price", "Total"))
```

```
file.write("=====\n")
```

```

for item in maindata.values():

    product_name = item[0]

    brand = item[1]

    quantity = item[3]

    unit_price = item[2]

    total_price = "${:.2f}".format(int(item[2].replace('$', ' ').replace(',', '')) * int(item[3]))

    file.write("{:<40}{:<25}{:<15}{:<15}{:<15}\n".format(product_name, brand, quantity,
unit_price, total_price))

file.write("=====  

=====\\n")

file.write("\\n")

file.write("\\t\\t\\t\\t\\t\\t\\t\\t\\t\\tGross Total: {:.2f}\\n".format(total))

file.write("\\n")

file.write("\\t\\t\\t\\t\\t\\t\\t\\t\\t\\tAdditional Charge (if applicable):  

${:.2f}\\n".format(additional_charge))

file.write("\\n")

file.write("\\t\\t\\t\\t\\t\\t\\t\\t\\t\\tGrand Total: {:.2f}\\n".format(grand_total))

file.write("\\n")

file.close()

print("Invoice generated successfully.")

```

```
message.greading()

def write_file(maindata):

    with open("equip.txt","w") as file:

        for value in maindata.values():

            write_data = str(value[0]) + "," + str(value[1]) + "," + str(value[2]) + "," +
str(value[3]) + "\n"

            file.write(write_data)


def date_time():

    return datetime.datetime.now().strftime("%Y-%m-%d %H:%M")


def date():

    return datetime.datetime.now().strftime("%Y-%m-%d")
```