LONDON METROPOLITAN UNIVERSITY

islington college
(इस्लिङ्टन कलेज)

Module Code & Module Title

CS4051NI Fundamentals of Computing

Assessment Weightage & Type

60% Individual Coursework

Year and Semester

2021-22 Spring

Student Name: Anish Lamichhane

Group:Computing c4

London Met ID:21039633

College ID: NP01CP4A210056

Assignment Due Date: 13 may 2022

Assignment Submission Date:13 may 2022

I confirm that I understand my coursework needs to be submitted online via Google Classroom under the relevant module page before the deadline in order for my 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

List of Figures

List of Tables

1. Introduction to the project

Fundamentals of Computing is one of the modules that Computing students' is compulsory to be studied in Semester 2. The first coursework of second semester was given to us in week 6 which carries 60% of total marks in modules. According to the instruction provided by the coursework, we should create a program. This coursework allows the students to develop a bike management system. it is much more easy to use a bike management system since the data stored in the program will be permanent and not easily deleted or lost. Its keep records of the sales and order for the future reference. It will be necessary to create an application to access the information stored inside this text file. It must also be able to show all of the motorcycles that are currently in stock and ready to also be sold to a new customer. when the bike was sold to the customer the bike stock should be deducted from the text file. If the customer order more than one bike the receipt of the sells bike contain the both selling details of bike. Likewise, the order has also the same scenario, when the bike was ordered from shipping company the stock should be updated automatically. The text filed generated after buying and selling of bikes should contain name, company, price, quantity date and time.

Python is a programming language for creating websites and web applications, as well as managing activities and performing data analysis. Python is a powerful programming language, which implies it is used to create a variety of applications and isn't dedicated to any given problem. Python is generally used mostly for website design & development creation, highly simplified, data science, and computer vision. Python has been used by many non-programmers, like auditors and scientists, for a variety of common activities, including such organizing resources, due to the relative ease to understanding.

IDLE is a Python integrated development environment that has been included with the language's default implementation since version 1.5.2b1. IDLE may be used to run a single argument in almost the same way as Python Shell can, as well as to build, edit, and run Python scripts. IDLE includes a packed editor with code editor, auto completion,

and smart indent for developing Python code. There was also a debugging with steps and debugging tools.

1..1Goal and objective of the project

The application must be built in a specific way with separate methods for user input, reading files, and creating selling/ordering notes and program must run in a loop displaying available books and wait for the operator to enter the necessary information. Unless the operator decides the program must stay open otherwise the program should be closed.

The objectives of the project are:

1. This project provides a reference to the company and employee to easily access information, make smooth transaction.
2. The creation of sell and order receipt provide more security in terms of data losses and frauds.
3. The sell and order receipt should contain the proper information of the name, address, contact number, bike name, company name, bike quantity, bike price.
4. The program should show the stock will increase when ordering bike and stock will decrease when selling the bikes in program.
5. If the one or more than one bike was order and sell under same person name, the receipt should be generated once.

2. Discussion and Analysis

2.1 Algorithm

Algorithm is the instruction which is used to developed the program to get the specific task. It is the one of most important factor in programming. The algorithm should be step by step to describe the flow of program.

Algorithm

Step 1: Start the program
Step 2: Display the welcome message
Step 3: Show the option
Step 4: Ask the user input 1,2,3
Step 5: If user input 1, which is to sell a bike and move forward to step 6
Step 6: Ask the user to input valid bike id and valid quantity
Step 7: Ask the user to sell another bike or not, if the user input yes move back to step 7 otherwise move forward to step 9
Step 8: Input the person name, address, contact number for the development of sell receipt.
Step 9: Create a sell receipt in a text file, and print the exact same receipt in the shell. Go back to Step 4
Step 10: If user input 2, which is to order a bike and move back to step 6 and 7
Step 11: Ask the user to buy another bike or not, if the user input yes move back to step 6 otherwise move forward to step 13
Step 12: Input the person name, address, contact number for the development of order receipt.
Step 13: Create an order receipt in a text file, and print the exact same receipt in the shell. Go back to Step 4
Step 14: If user input 4, which is to exit a program and display the “thank you for using the bike management system”.
Step 15: close the program

2.2 Flowchart

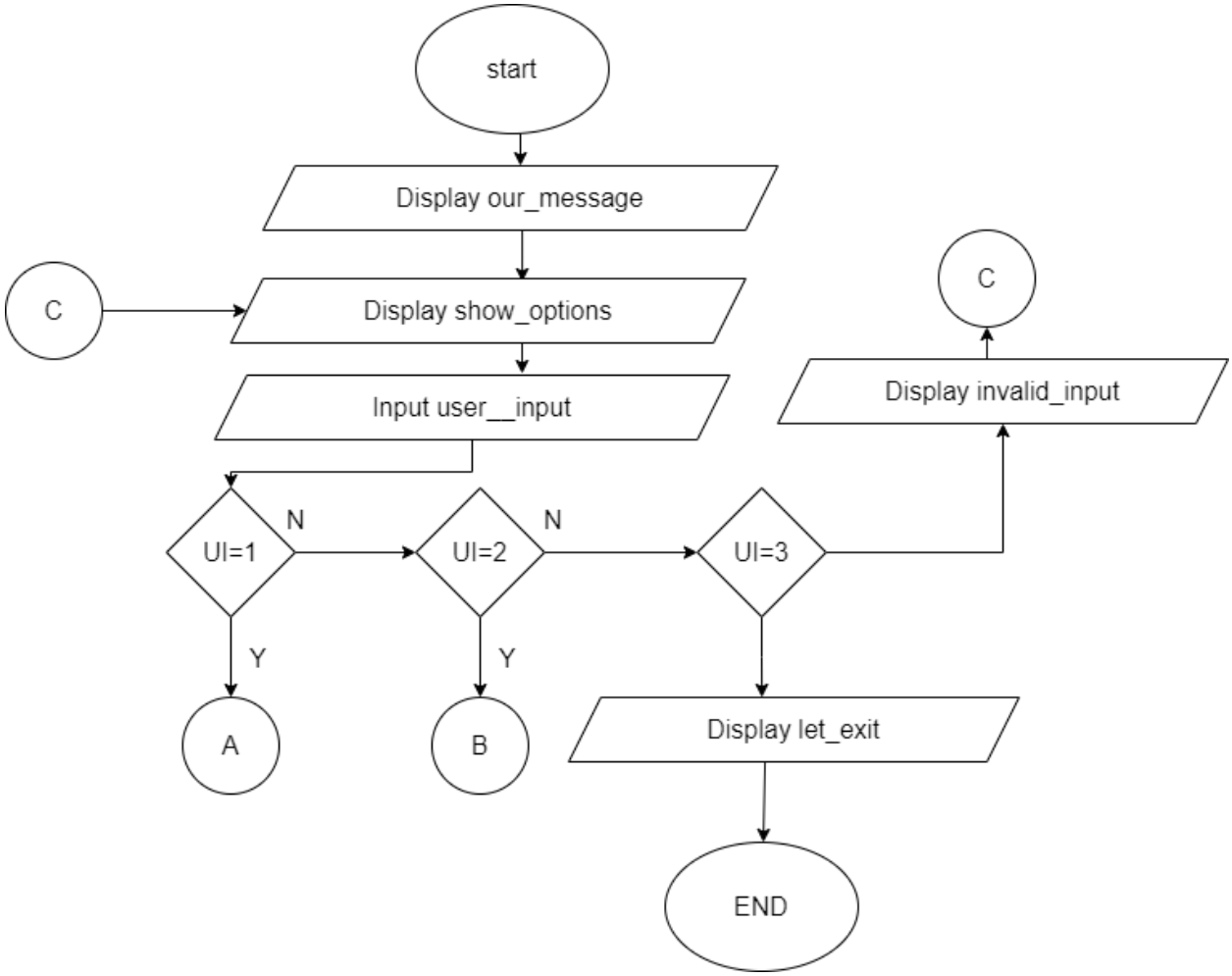


Figure 1. flowchart of main.py

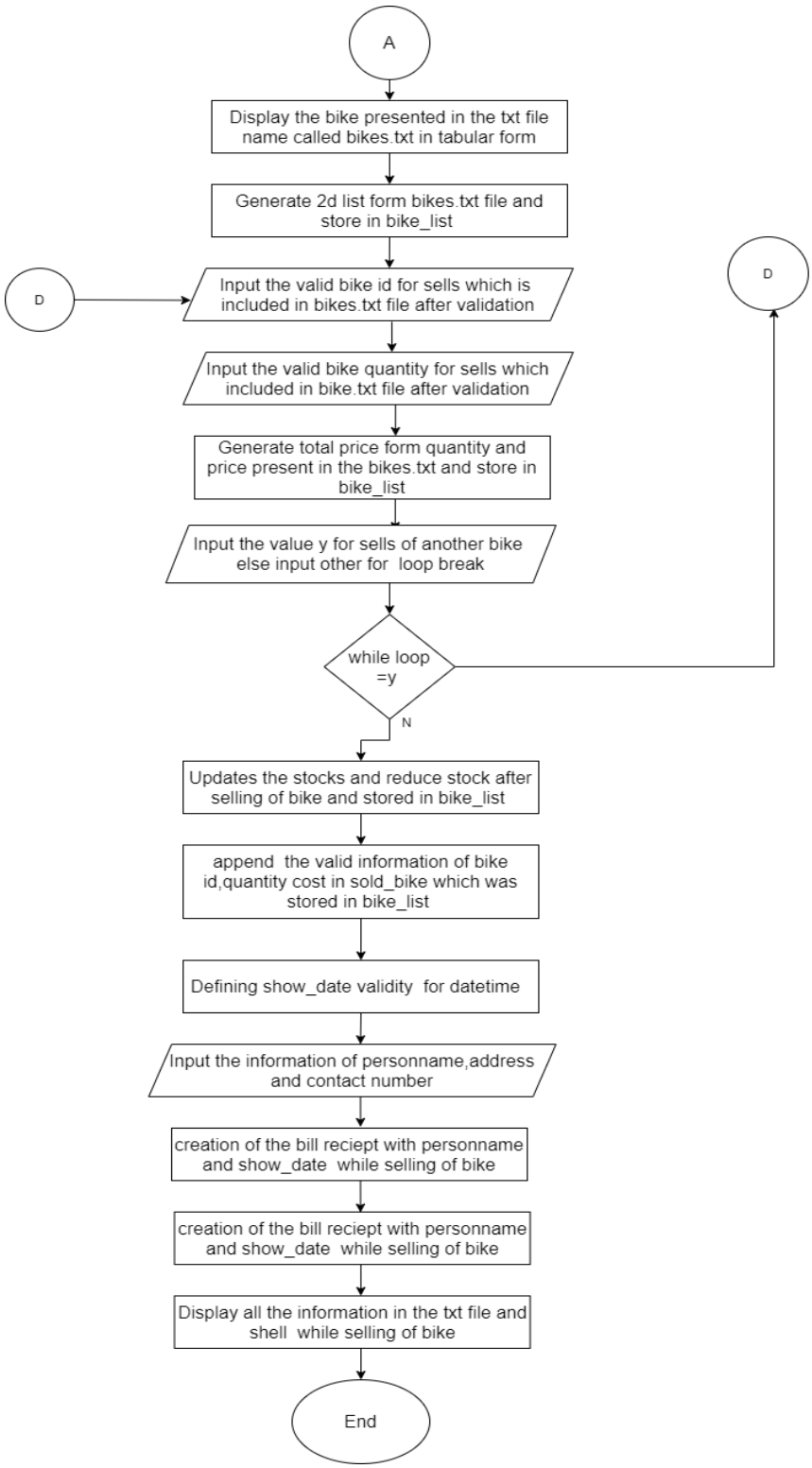


Figure 2 flowchart of sellfunction.py

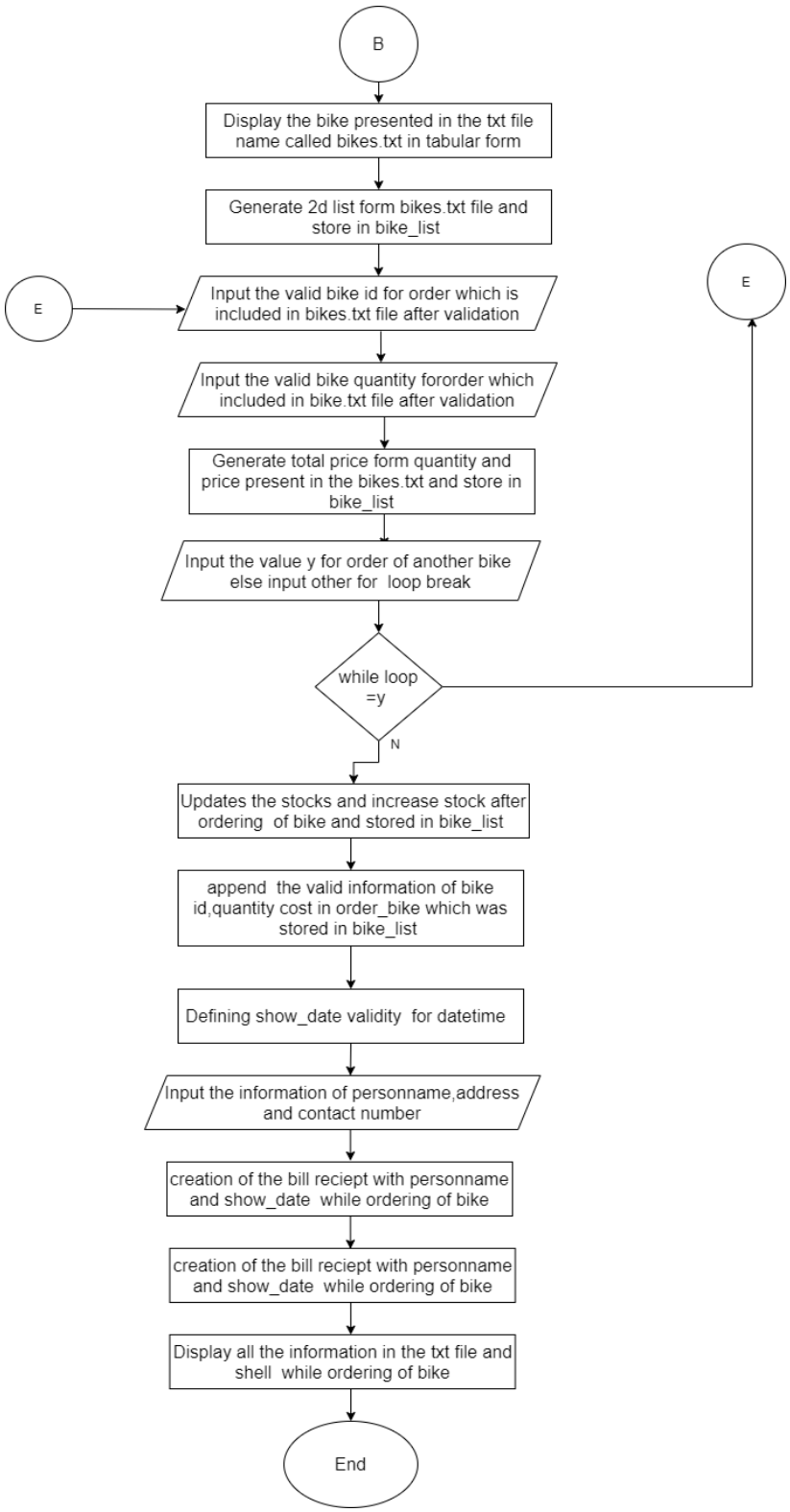


Figure 3 flowchart of orderfunction.py

2.3 PSEUDOCODE

Pseudo code is a non language specific description of what code should do. It allows the reader to understand what code they should implement with out tying it down to an implementation in one language. it's good for explanation. (w3schools, n.d.)

PSEUDOCODE OF Main.py

```
import datetime
import sellfunction
import orderfunction
Define our_message
DO
   PRINT ("-----------------------------------------------------------------------")
   PRINT ("  --------- Welcome to Bike management System------------ ")
   PRINT ("-----------------------------------------------------------------------")
END DO
DEFINE show_options
DO
   PRINT ("1. Sell Bikes")
   PRINT ("2. Order Bikes")
   PRINT ("3. Exit")
END DO
DEFINE sells_bikes
DO
   PRINT("Let's sell  Bikes")
   PRINT("-------------------------------------------------------")
END DO

DEFINE order_bikes
DO
```

```
PRINT("Let's order Bikes")
PRINT("-----------------------------------------------------")
END DO
DEFINE let_exit
DO
    PRINT ("-----------------------------------------------------")
    PRINT ("Thank you for using Bike management System")
    PRINT ("-----------------------------------------------------")
END DO
DEFINE invalid_input
DO
  PRINT ("-----------------------------------------------------")
  PRINT ("INVALID Input. Please enter the number of the options provided in the screen")
  PRINT ("-----------------------------------------------------")
END DO
SET THE counter to be True
CALL our_message
WHILE  counter
   PRINT ("-----------------------------------------------------------------")
 CALL show_options

TRY
    SET THE user_input EQUALS TO  int(INPUT("Choose an option: "))
EXCEPT
      PRINT ("-----------------------------------------------------------------")
      PRINT ("please,Enter a valid information")
      PRINT("-----------------------------------------------------------------")
      CONTINUE
     SET THE  user_input EQUALS TO 1
     SET THE  counter EQUALS TO False
```

```
        SET THE Sold_bike EQUALS TO  [ ]
        WHILE  Counter  EQUALS TO  False
           CALL sellfunction.ShowBikes
           CALL entered_bike_id IS EQUALS TO sellfunction.valid_bike_id
           CALL customer_quantity  IS EQUALS TO sellfunction.valid_bike_quantity
            (entered_ bike_id)
           CALL  bike_list IS  EQUAL TO sellfunction.return_2d_list
           CALL total_cost IS EQUAL TO sellfunction.total_bike_cost(entered_ bike_id,
           valid_bike_id)
           CALL sellfunction.update_stock(customer_quantity,entered_bike_id)
           CALL Sold_bike IS EQUALS TO sellfunction.new_bill(bike_list,entered_bike_id
           ,Sold_bike,customer_quantity,total_cost)
           CALL total_cost   IS EQUALS TO  int(bike_list[entered_bike_id-1][5]
           .replace("$",""))* customer_quantity
           SET THE counter IS EQUALS TO sellfunction.loop()
CALL sellfunction.print_bill_sell(bike_list,Sold_bike)
ELIF user_input  IS EQUALS TO  2
   CALL  order_bikes
   SET THE  counter IS EQUALS TO  False
    SET THE  order_bike IS EQUALS TO   []
    WHILE counter IS EQUALS TO   False
           CALL orderfunction.ShowBikes
           CALL entered_bike_id IS EQUALS TO orderfunction.valid_bike_id
           CALLcustomer_quantity  IS EQUALS TO orderfunction.valid_bike_
           quantity_order ( entered_bike_id)
            CALL  bike_list IS  EQUAL TO orderfunction.return_2d_list
           CALL total_cost IS EQUAL TO orderfunction.total_bike_cost(entered_ bike_id,
           valid_bike_id)
           CALL orderfunction.update_stock(customer_quantity,entered_bike_id)
           CALL order_bike IS EQUAL orderfunction.new_bill(bike_list,entered_bike_id
           order_bike,customer_quantity,total_cost)
```

```
        CALL total_cost   IS EQUALS TO  int(bike_list[entered_bike_id-1][5]
        .replace("$",""))* customer_quantity
        SET THE counter IS EQUALS TO orderfunction.loops()
CALL orderfunction.print_bill_sell(bike_list,Order_bike)



ELIF user_input IS EQUALS TO  3
        CALL   let_exit
         SET THE counter  IS EQUALS TO False
         PRINT("-------------------Have a nice day,bye-----------------------------------")
         PRINT("------------------------------------------------------------------------")

   ELSE
   invalid_input
```

3.2 PSEUDOCODE of Sellfunction.py

```
DEFINE  ShowBikes
DO
  PRINT ("\n")
   PRINT("-------------------------------------------------------------------------------------------------------
---------------------------------")
    PRINT("Bike ID\tBike-Name\tCompany Name\tColour\t  Quantity\tPrice")
     PRINT("-------------------------------------------------------------------------------------------------------
--------------")
     FILE = open("bikes.txt", "r")

FOR line in file
      PRINT( line.replace(",","\t"))
      PRINT ("-------------------------------------------------------------------------------------------------")
    PRINT("\n")
```

```
   FILE.close()
END FOR
END DO
DEFINE  return_2d_list
    DO
     READ_FILE = open("bikes.txt", "r")
      bike_list = []
      FOR  bike in read_file:
          bike = bike.replace("\n", "")
          bike_list.append(bike.split(","))
          END FOR
          END DO
          RETURN bike_list
   DEFINE valid_bike_id
          DO
          WHILE true
            TRY
             validBikeId = int(INPUT("Enter ID of the bike to sell: "))
             WHILE validBikeId <= 0 or validBikeId > len(return_2d_list()):
                    PRINT("Please provide a valid bike ID !!!")
                     CALL ShowBikes
              RETURN validBikeId
            BREAK
       EXCEPT ValueError:
            PRINT("-----------------------------------------------------------------")
            PRINT("please,Enter a valid information")
            PRINT("-----------------------------------------------------------------")
      END DO
DEFINE valid_bike_quantity(entered_bike_id)
  DO
  bike_id IS ASSIQN AS entered_bike_id
```

```
WHILE True
      TRY
         bike_quantity = int(INPUT("\nEnter quantity of bike to sell: "))
         WHILE bike_quantity <= 0 or bike_quantity > int(return_2d_list()[bike_id - 1][4]):
           PRINT ("\nPlease provide a valid bike quantity!!!\n")
            CALL ShowBikes
         RETURN bike_quantity
         BREAK
      EXCEPT
         PRINT("-----------------------------------------------------------------")
         PRINT("please,Enter a valid information")
         PRINT("-----------------------------------------------------------------")
     END DO
DEFINE  total_bike_cost(entered_bike_id,customer_quantity)
     DO
     bike_list IS EQUALS TO  return_2d_list()
      total_cost = int(bike_list[entered_bike_id-1][5].replace("$",""))* customer_quantity
      RETURN  total_cost
       END DO

DEFINE  loop
  DO
   inputdata = INPUT ("do you want to sell another bike,y/n= ").upper
   IF(inputdata IS EQUALS TO "Y"):
       counter IS ASSQIN AS False
   END IF
   ELSE
       counter IS ASSQIN AS True
     RETURN  counter
END ELSE
END DO
```

```
DEFINE update_stock(customer_quantity,entered_bike_id)
   DO
   bike_list  IS EQUALS TO  return_2d_list()
   bike_list[entered_bike_id-1][4]=int(bike_list[entered_bike_id-1][4])                    -
customer_quantity
   FILE = open("bikes.txt","w")
   FOR  bike in bike_list

FILE.WRITE(str(bike[0])+","+str(bike[1])+","+str(bike[2])+","+str(bike[3])+","+str(bike[4]
+","+ str(bike[5])+","+"/N")
   FILE.CLOSE
   END DO
   CALL ShowBikes()

DEFINE new_bill(bike_list,entered_bike_id,Sold_bike,customer_quantity,total_cost)
  DO
bike_list IS EQUALS TO return_2d_list()
   FOR bike in bike_list:
      IF (int(bike[0]) IS ASSQIN AS entered_bike_id):
            bike[4] IS ASSQIN AS customer_quantity
            bike[5] IS ASSQIN AS "$"+str(total_cost)
            Sold_bike.append(bike)
         END IF
         END FOR
    RETURN Sold_bike
         END DO
DEFINE  show_date
   IMPORT datetime
   DO
   year IS ASSIQN AS str(datetime.datetime.now() .year)
   month IS ASSIQN AS str(datetime.datetime.now().month)
```

```
    day IS ASSIQN AS str(datetime.datetime.now().day)
    hour IS ASSIQN AS str(datetime.datetime.now().hour)
    minute IS ASSIQN AS str(datetime.datetime.now().minute)
    second IS ASSIQN AS str(datetime.datetime.now().second)
    RETURN year + month + day + hour + minute + second
     END DO

CALL show_date

DEFINE  print_bill_sell(bike_list,Sold_bike)
    IMPORT datetime
    DO
    PRINT("--------------------------------Customer details--------------------------------------")
    PRINT("-------------------------------------------------------------------------------------------\n")
    personName IS ASSIQN AS  INPUT ("Enter customer's name: ")
    customer_address IS ASSIQN AS  INPUT ("Enter your address: ")
    customer_contactNumber IS ASSIQN AS  INPUT ("Enter your contact number: ")
    bike_list IS EQUALS TO return_2d_list
    PRINT ("-------------------------------------------------------------------------------------------")
    FILE= open(personName +""+show_date()+".txt", "a")
    FILE.WRITE( ==================================================\n")
    PRINT(“ ==================================================\n")
     FILE.WRITE(“=================SELL_BIKE_RECIEPT===============\n”)
    PRINT(“=================SELL_BIKE_RECIEPT===============”)
    FILE.WRITE( ==================================================\n")
    PRINT(“ ==================================================\n")
   FILE.WRITE(“=================bike management system
===============\n”)
   PRINT(“=================bike management system ===============\n”)
   FILE.WRITE ("Customer name: " + personName + "\n")
    FILE.WRITE ("Customer address: " + customer_address + "\n")
    FILE.WRITE ("Customer contact number: " +customer_contactNumber + "\n")
```

```
   FILE.WRITE ("the bike sold in this Date & Time : " + str(datetime.datetime.now()) +
"\n")
   PRINT("Customer name: " + personName + "\n")
   PRINT( "Customer address: " + customer_address + "\n")
   PRINT( "Customer contact number: " +customer_contactNumber + "\n")
   PRINT( "the bike sold in this Date & Time : " + str(datetime.datetime.now()) + "\n")
    FILE.WRITE( ==================================================\n")
   PRINT(“ ==================================================\n")
   FILE.WRITE ("S.N\t Bike Name\t Company \t   Quantity\tprice\ttotal_amount\n")
  FILE.WRITE( ==================================================\n")
   PRINT(“ ==================================================\n")
  PRINT ("S.N\t Bike Name\t Company\t    Quantity\tprice\ttotal_amount")
  SNo  IS ASSQIN AS 0
   total_amount IS ASSQIN AS 0
   FOR bike in Sold_bike:
       SNo  IS EQUALS TO  SNo + 1
       bike[5] IS EQUALS TO   bike[5].replace("$","")
       total_amount IS EQUALS TO   total_amount + int(bike[5])
        END FOR
        FILEWRITE.(str(SNo)+"\t"+(str(bike[1])+"\t"+str(bike[2])+"\t\str(bike[3])+"\
str(bike[4])+"\ str(bike[5])+"\ str(bike[ amount+"\t\t"))

FILE.WRITE( ==================================================\n")
PRINT.(str(SNo)+"\t"+(str(bike[1])+"\t"+str(bike[2])+"\t\str(bike[3])+"\str(bike[4])+"\
str(bike[5])+"\ str(bike[ amount+"\t\t"))
 PRINT( ==================================================\n")
FILE.CLOSE
END DO
```

3.3 PSEUDOCODE of Orderfunction.py

```
DEFINE  ShowBikes
DO
  PRINT ("\n")
   PRINT("-------------------------------------------------------------------------------------------------------
---------------------------------")
    PRINT("Bike ID\tBike-Name\tCompany Name\tColour\t  Quantity\tPrice")
     PRINT("-------------------------------------------------------------------------------------------------------
--------------")
     FILE = open("bikes.txt", "r")

FOR line in file
      PRINT( line.replace(","\t"))
      PRINT ("-------------------------------------------------------------------------------------------------")
    PRINT("\n")
    FILE.close()
END FOR
END DO
DEFINE  return_2d_list
     DO
      READ_FILE = open("bikes.txt", "r")
       bike_list = []
       FOR  bike in read_file:
           bike = bike.replace("\n", "")
           bike_list.append(bike.split(","))
           END FOR
           END DO
           RETURN bike_list
    DEFINE valid_bike_id_order
           DO
```

```
        WHILE true
         TRY
          validBikeId = int(INPUT("Enter ID of the bike to order: "))
          WHILE validBikeId <= 0 or validBikeId > len(return_2d_list()):
                PRINT("Please provide a valid bike ID !!!")
                 CALL ShowBikes
            RETURN validBikeId
        BREAK
      EXCEPT ValueError:
        PRINT("-----------------------------------------------------------------")
        PRINT("please,Enter a valid information")
        PRINT("-----------------------------------------------------------------")
     END DO
DEFINE valid_bike_quantity_order(entered_bike_id)
  DO
  bike_id IS ASSIQN AS entered_bike_id
WHILE True
      TRY
        bike_quantity = int(INPUT("\nEnter quantity of bike to order: "))
        WHILE bike_quantity <= 0 or bike_quantity > int(return_2d_list()[bike_id - 1][4]):
          PRINT ("\nPlease provide a valid bike quantity!!!\n")
           CALL ShowBikes
        RETURN bike_quantity
        BREAK
      EXCEPT
        PRINT("-----------------------------------------------------------------")
        PRINT("please,Enter a valid information")
        PRINT("-----------------------------------------------------------------")
     END DO
DEFINE  total_bike_cost(entered_bike_id,customer_quantity)
     DO
```

```
      bike_list IS EQUALS TO  return_2d_list()
       total_cost = int(bike_list[entered_bike_id-1][5].replace("$",""))* customer_quantity
       RETURN  total_cost
        END DO
```

```
DEFINE  loops
  DO
   inputdata = INPUT ("do you want to order another bike,y/n= ").upper
   IF(inputdata IS EQUALS TO "Y"):
       counter IS ASSQIN AS False
   END IF
   ELSE
       counter IS ASSQIN AS True
      RETURN  counter
END ELSE
END DO
DEFINE update_stock_order(customer_quantity,entered_bike_id)
    DO
    bike_list  IS EQUALS TO  return_2d_list()
    bike_list[entered_bike_id-1][4]=int(bike_list[entered_bike_id-1][4])                    -
customer_quantity
    FILE = open("bikes.txt","w")
    FOR  bike in bike_list

FILE.WRITE(str(bike[0])+","+str(bike[1])+","+str(bike[2])+","+str(bike[3])+","+str(bike[4]
+","+ str(bike[5)+","+"/N")
    FILE.CLOSE
    END DO
```

```
   CALL ShowBikes()

DEFINE new_bill(bike_list,entered_bike_id,Order_bike,customer_quantity,total_cost)
  DO
bike_list IS EQUALS TO return_2d_list()
   FOR bike in bike_list:
      IF (int(bike[0]) IS ASSQIN AS entered_bike_id):
            bike[4] IS ASSQIN AS customer_quantity
            bike[5] IS ASSQIN AS "$"+str(total_cost)
            Order_bike.append(bike)
         END IF
         END FOR
    RETURN Order_bike
         END DO
DEFINE  show_date
   IMPORT datetime
   DO
   year IS ASSIQN AS str(datetime.datetime.now() .year)
   month IS ASSIQN AS str(datetime.datetime.now().month)
   day IS ASSIQN AS str(datetime.datetime.now().day)
   hour IS ASSIQN AS str(datetime.datetime.now().hour)
   minute IS ASSIQN AS str(datetime.datetime.now().minute)
   second IS ASSIQN AS str(datetime.datetime.now().second)
   RETURN year + month + day + hour + minute + second
    END DO

CALL show_date

DEFINE  print_bill_order(bike_list,Order_bike)
   IMPORT datetime
   DO
   PRINT("--------------------------------Customer details--------------------------------------")
```

```
    PRINT("-----------------------------------------------------------------------------------------\n")
    personName IS ASSIQN AS  INPUT ("Enter customer's name: ")
    customer_address IS ASSIQN AS  INPUT ("Enter your address: ")
    customer_contactNumber IS ASSIQN AS  INPUT ("Enter your contact number: ")
    bike_list IS EQUALS TO return_2d_list
    PRINT ("-----------------------------------------------------------------------------------------")
    FILE= open(personName +""+show_date()+".txt", "a")
    FILE.WRITE( ==================================================\n")
    PRINT(“ ==================================================\n")

FILE.WRITE(“=================ORDER_BIKE_RECIEPT===============\n”)
PRINT(“=================ORDER_BIKE_RECIEPT===============”)
    FILE.WRITE( ==================================================\n")
    PRINT(“ ==================================================\n")
   FILE.WRITE(“=================bike management system ===============\n”)
   PRINT(“=================bike management system ===============\n”)
   FILE.WRITE ("Customer name: " + personName + "\n")
    FILE.WRITE ("Customer address: " + customer_address + "\n")
    FILE.WRITE ("Customer contact number: " +customer_contactNumber + "\n")
    FILE.WRITE ("the bike order in this Date & Time : " + str(datetime.datetime.now()) + "\n")
    PRINT("Customer name: " + personName + "\n")
    PRINT( "Customer address: " + customer_address + "\n")
    PRINT( "Customer contact number: " +customer_contactNumber + "\n")
    PRINT( "the bike order in this Date & Time : " + str(datetime.datetime.now()) + "\n")
     FILE.WRITE( ==================================================\n")
    PRINT(“ ==================================================\n")
    FILE.WRITE ("S.N\t Bike Name\t Company \t   Quantity\tprice\ttotal_amount\n")
   FILE.WRITE( ==================================================\n")
    PRINT(“ ==================================================\n")
```

```
   PRINT ("S.N\t Bike Name\t Company\t    Quantity\tprice\ttotal_amount")
   SNo  IS ASSQIN AS 0
    total_amount IS ASSQIN AS 0
    FOR bike in Order_bike:
        SNo  IS EQUALS TO  SNo + 1
        bike[5] IS EQUALS TO   bike[5].replace("$","")
        total_amount IS EQUALS TO   total_amount + int(bike[5])
         END FOR
         FILEWRITE.(str(SNo)+"\t"+(str(bike[1])+"\t"+str(bike[2])+"\t\str(bike[3])+"\
str(bike[4])+"\ str(bike[5])+"\ str(bike[ amount+"\t\t"))

FILE.WRITE( ==================================================\n")
PRINT.(str(SNo)+"\t"+(str(bike[1])+"\t"+str(bike[2])+"\t\str(bike[3])+"\str(bike[4])+"\
str(bike[5])+"\ str(bike[ amount+"\t\t"))
 PRINT( ==================================================\n")
FILE.CLOSE
END DO
```

2.4 Data structure

A data structure is a collection of data objects that provides efficient way for organizing and storing data in the system so that it may be used effectively. Data structure can have defined the logical operation to execute in program. It is most important concept in programming. (luiz, 2000)

Data structures is generally categorized into two:

1.Primitive Data Type:

These are all the metadata structures which contain basic, simple data values and offer a framework for data processing. Boolean, byte, int, float, and double are examples of primitive data types.

- Integer: integer is the data type which represent the numeric value. It includes whole, number and negative number.

Integer used in the program:

```
counter = True
our_message()
while counter :
    print("-------------------------------------------------------------
    show_options()
    #Exception handling is used for preventation form unwanted error.
    try:
        user_input = int(input("Choose an option: "))
    except:
        print("-------------------------------------------------------
        print("please,Enter a valid information")
        print("-------------------------------------------------------
        continue
```

Figure 4 Example of integer

The main module which is main.py have contain int datatype. the user_input assign the int input value option which is presented in program. The input must be 1,2,3. Int(input) assure the input value must be numeric value otherwise it will have displayed error

```
def valid_bike_id():
    ''' Validates bikeid  for sells form user input.'''
    #Exception handling is used for preventation form unwanted error.
    while True:
        try:
            validBikeId = int(input("Enter ID of the bike to sell: "))
            while validBikeId <= 0 or validBikeId > len(return_2d_list()):
                 print("Please provide a valid bike ID !!!")
                 ShowBikes()
            return validBikeId
            break
        except ValueError:
            print("-------------------------------------------------------------
            print("please,Enter a valid information")
            print("-------------------------------------------------------------
```

Figure 5 Example of integer 2

The sell function modules also contain the int datatypes. the user_input assign the int input valid bike id which is presented in bike.txt file. The input must be the numeric value which is assign as bike id. If the input value is not numeric value error will occur in program.

- String: string is datatype which is represented by alphabet value. A string is a collection of characters. Anything within quotations is a string in Python. You have the option that use single or double quotations. It can be combined by using + methods on one or more strings, but it can also be looped with *. Multiple techniques can be used to divide, identify, and obtain strings.

String present in the program:

```
def new_bill(bike_list,entered_bike_id,order_bike,customer_quantity,total_cost):
    bike_list = return_2d_list()
    '''It is used for billing receipt of orders bike'''
    for bike in bike_list:
        if(int(bike[0]) == entered_bike_id):
                bike[4] = customer_quantity
                bike[5] = "$"+str(total_cost)
                order_bike.append(bike)
    return order_bike
```

Figure 6 Example of string

The main module: order.py includes str data type. The str is used for converting any datatype to string. The str represent the total cost in the form of string. It will help to reduce error. “$” is also represent as string because the value is under double quotations.

Boolean: Boolean is the datatype which represent the condition in form of True or False. It is use for logical and arithmetic operation. It is use to check the truth value of the statement. For example, 1>0 is True and 1<o is False

```
def loop():# using loop for the proper functioning of program
    inputdata = input("do you want to sell another bike,y/n= ").upper()
    if(inputdata == "Y"):
        counter = False
    else:
        counter = True
    return counter
```

Figure 7 Example of Boolean

The Boolean datatype contain in sell function modules. If the input data is assign as the y. the counter will be false and you can sell another bike. When the user input assign value N the counter will be true and you cannot sell another bike.

- Float = The float datatype is known as “Floating Point Number”. It includes the rational number having decimal points. For example: 1.21, 4.13, or 9.6.

```
SNo = 0
total_amount = 0
for bike in order_bike:
    SNo = SNo + 1
    bike[5] = bike[5].replace("$","")
    total_amount = total_amount + float(bike[5])
    file.write(str(SNo)+"\t"+(str(bike[1])+"\t"+str(bike[2])+"\t\t"+str(bik
    file.write("==================================================================
    print(str(SNo)+"\t"+(str(bike[1])+"\t"+str(bike[2])+"\t\t"+str(bike[4])
    print("==================================================================
file.close
```

Figure 8 Example of Float

The float datatype is used in orderfunction.py module. it is used to get the numeric value in the form of decimal which is not supported by int datatype. if price of the bike is in decimal number it will add them in float and stored in total amount. When the customer orders a more bike float value of total price is added to total amount.

2. Non-Primitive Data Type: It is a datatype that are developed from primitive data types and have more function. It is the collection of the value in different ways rather than primitive datatype

- List = In Python, a list is a structure it is a mutable, and dynamic, ordered series of items. An item is any element or value contained within a list. List are generated by having values between brackets [ ]. Python has a lot of techniques for manipulating and working with lists. It is possible to add new items, remove some, sort out, extend, append, and perform many other things. (w3schools, n.d.)

```
def return_2d_list():
    '''generate 2d list form bikes.txt file and store in bike_list'''
    read_file = open("bikes.txt", "r")
    bike_list = []
    for bike in read_file:
        bike = bike.replace("\n", "")
        bike_list.append(bike.split(","))

    return bike_list
```

Figure 9 Example of list

The sellFunction module: sellFunction.py includes many lists. One of them being bike_List which is an empty list and another being columns which includes help to append a bike in the required for a two-dimensional list. After the bike.txt has been opened and split it where is bike. Split and append the split into the bike_list. bike.split is used to split a string into a list and. append is used to add bikes to the an existing list.

- Dictionaries: A dictionary is a group of objects, each of which is a Key combination. A unique key is provided to each value or amount of items. A key is used to store the data in the dictionaries and splits by using comma. Curly braces contain all of the Key-Value pairs. It is not mutable and store all type of values. (luiz, 2000)

```
========================= RESTART: D:/python cw/1333.py ================
>>>
... dic_1 = {"first Name":"presa" ,"last Name":"koirala","age":"18"}
>>> print(dic_1)
{'first Name': 'presa', 'last Name': 'koirala', 'age': '18'}
>>>
```

Figure 10 Example of Dictionary

- Set =A set is an unsorted and unindexed collection of items. Curly brackets are used to write sets in Python.it is mutable but the value cannot be repeated (luiz, 2000)

```
=======================
>>> suii = {1,2,3,4}
>>> print(suii)
{1, 2, 3, 4}
>>>
```

Figure 11 Example of Set

- Tuple = Several things can be contained in a single variable using tuples. A tuple is a set that is both sorted and immutable. You cannot append remove delete any item inside tuple. (w3schools, n.d.)

```
>>> a = {10,"barcelona",6.69}
>>> print(a)
{10, 6.69, 'barcelona'}
>>>
```

Figure 12 Example of tuple

3. Program

3.1 Implementation of program with short description

The program consists of three modules main.py, sellfunction.py and order function, each function contains different responsible for the execute the program. the bike details contained in bikes.txt

Main.py

This module is responsible for the running program smoothly.it is major aspect to run the program with functionality. if the main.py module doesnot run properly the program will be crash

This module have our_message() for welcoming to the program show_options() is used to show the program. its shows the input 1,2 and 3. The option 1 is used to sell the bikes and 2 use to order bike and 3 is used to exit program.

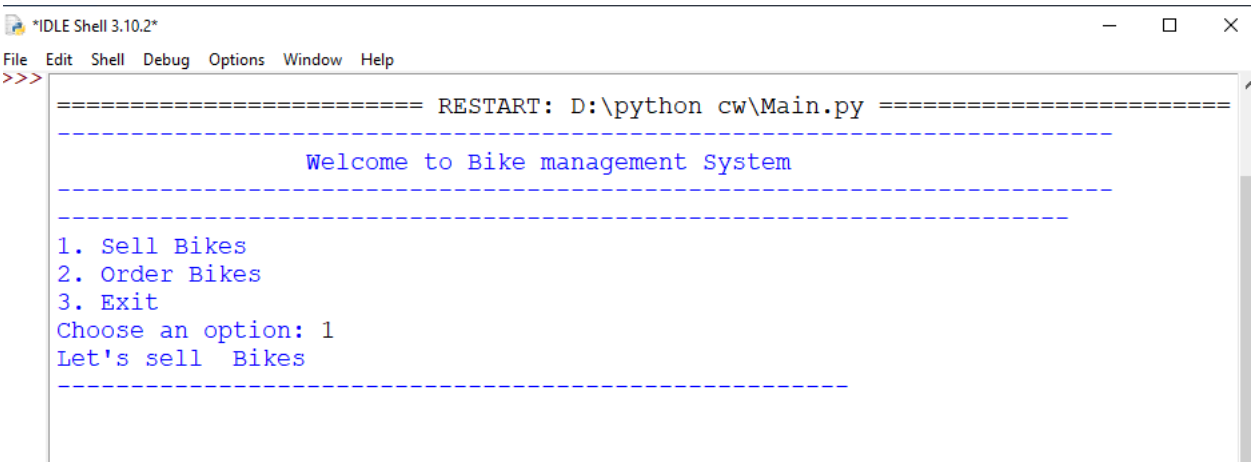


Figure 13 Implementation of program

If the user chooses 1 then, the bikes.txt file is read and displayed using split, and append to the existing empty list named bike_List. The headings list which is columns is also printed before the bike_List to separate bike ID, Name, Company Name, Stock available and price . . Once a bike is sell with valid quantity , 'Do you want to sell another bike?' option is displayed in which if you type y, the process is repeated but if the input is n, then a sell receipt is created in shell and txt file

.

```
*IDLE Shell 3.10.2*
File  Edit  Shell  Debug  Options  Window  Help
>>>
========================= RESTART: D:\python cw\Main.py =========================
-----------------------------------------------------------------------
                 Welcome to Bike management System
-----------------------------------------------------------------------
-----------------------------------------------------------------------
1. Sell Bikes
2. Order Bikes
3. Exit
Choose an option: 1
Let's sell  Bikes
-----------------------------------------------------


-------------------------------------------------------------------------------
-----------------------------------------------
Bike ID Bike-Name       Company Name    Colour     Quantity     Price
-------------------------------------------------------------------------------
-----------------------------------------------
1       Classic 350     Royal Enfield   Gun Metal Grey  49      $4500

2       KTM RC 2000     KTM Bike        Orange Black    52      $5300

3       Brutale 800     Mv Agusta       White Red       33      $21000

4       KTMduke 390     Bajaj auto      Grey black      26      $20000

5       Yamaha FZFI     yamaha company  dark blue       48      $10000

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

Figure 14 Implementation of program 2

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


Enter ID of the bike to sell: 1

Enter quantity of bike to sell: 2


-------------------------------------------------------------------------------
-----------
Bike ID Bike-Name       Company Name    Colour      Quantity     Price
-------------------------------------------------------------------------------
-----------
1       Classic 350     Royal Enfield   Gun Metal Grey  47      $4500

2       KTM RC 2000     KTM Bike        Orange Black    52      $5300

3       Brutale 800     Mv Agusta       White Red       33      $21000

4       KTMduke 390     Bajaj auto      Grey black      26      $20000

5       Yamaha FZFI     yamaha company  dark blue       48      $10000

-------------------------------------------------------------------------------
-----------


The total price of the bike is:  $ 9000
do you want to sell another bike,y/n=
```

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


The total price of the bike is:  $ 9000
do you want to sell another bike,y/n= y
Let's sell  Bikes
-------------------------------------------------------


-------------------------------------------------------------------------------
-----------
Bike ID Bike-Name       Company Name    Colour      Quantity     Price
-------------------------------------------------------------------------------
-----------
1       Classic 350     Royal Enfield   Gun Metal Grey  47      $4500

2       KTM RC 2000     KTM Bike        Orange Black    52      $5300

3       Brutale 800     Mv Agusta       White Red       33      $21000

4       KTMduke 390     Bajaj auto      Grey black      26      $20000

5       Yamaha FZFI     yamaha company  dark blue       48      $10000

-------------------------------------------------------------------------------
-----------


Enter ID of the bike to sell:
```

Figure 15 Implementation of program 3

```
Enter ID of the bike to sell: 2

Enter quantity of bike to sell: 3


-------------------------------------------------------------------------------------------
-----------
Bike ID Bike-Name       Company Name    Colour      Quantity     Price
-------------------------------------------------------------------------------------------
-----------
1       Classic 350     Royal Enfield   Gun Metal Grey  47      $4500

2       KTM RC 2000     KTM Bike        Orange Black    49      $5300

3       Brutale 800     Mv Agusta       White Red       33      $21000

4       KTMduke 390     Bajaj auto      Grey black      26      $20000

5       Yamaha FZFI     yamaha company  dark blue       48      $10000

-------------------------------------------------------------------------------------------
-----------


The total price of the bike is:  $ 15900
do you want to sell another bike,y/n= n
```

Figure 16 Implementation of program 4

```
*IDLE Shell 3.10.2*
File  Edit  Shell  Debug  Options  Window  Help
do you want to sell another bike,y/n= n
--------------------------------Customer details-------------------------------------
-----------------------------------------------------------------------------------------

Enter customer's name: anish lamichhane
Enter your address: kamalbinayak bkt 7
Enter your contact number: 09999
-------------------------------------------------------------------------------------------
=========================================================================================

===================================Sell Bike Receipt=====================================

=========================================================================================

==========================Bike Management System=================================

Customer name: anish lamichhane

Customer address: kamalbinayak bkt 7

Customer contact number: 09999

the bike sold in this Date & Time : 2022-05-12 22:34:11.105988

=============================================================================================

S.N      Bike Name       Company            Quantity    price   total_amount
=============================================================================================

1       Classic 350     Royal Enfield          2       9000    9000
=============================================================================================
Activate Windows
Go to Settings to activate Windows.
Ln: 106  Col: 0
Type here to search   71°F Partly cloudy   10:34 PM 5/12/2022
```

Figure 17 Implementation of program 5

```
*IDLE Shell 3.10.2*
File  Edit  Shell  Debug  Options  Window  Help
==================================================================================

==================================Sell Bike Receipt==================================

==================================================================================

===========================Bike Management System=================================

Customer name: anish lamichhane

Customer address: kamalbinayak bkt 7

Customer contact number: 09999

the bike sold in this Date & Time : 2022-05-12 22:34:11.105988

=================================================================================================================

S.N      Bike Name       Company           Quantity    price   total_amount
=================================================================================================================

1       Classic 350     Royal Enfield          2       9000    9000
=================================================================================================================

2       KTM RC 2000     KTM Bike               3       15900   24900
=================================================================================================================

-------------------------------------------------------------------
1. Sell Bikes
2. Order Bikes
3. Exit
Choose an option:
```

Figure 18 Implementation of program 7

```
anish lamichhane2022512223410 - Notepad
File  Edit  Format  View  Help
==================================================================================
==============================Sell Bike Receipt===================================
==================================================================================
============================Bike Management System=================================
Customer name: anish lamichhane
Customer address: kamalbinayak bkt 7
Customer contact number: 09999
the bike sold in this Date & Time : 2022-05-12 22:34:11.043546
=================================================================================================================
S.N      Bike Name       Company           Quantity     price   total_amount
=================================================================================================================
1       Classic 350     Royal Enfield          2       9000    9000
=================================================================================================================
2       KTM RC 2000     KTM Bike               3       15900   24900
=================================================================================================================
```

Figure 19 Implementation of program 8

If the user chooses 2 then, the bikes.txt file is read and displayed using split, and append to the existing empty list named bike_List. The headings list which is columns is also printed before the bike_List to separate bike ID, Name, Company Name, Stock available and price . . Once a bike is order with valid quantity , ‘Do you want to order

another bike?' option is displayed in which if you type y, the process is repeated but if the input is n, then a order receipt is created in shell and txt file

```
*IDLE Shell 3.10.2*
File  Edit  Shell  Debug  Options  Window  Help
                 Welcome to Bike management System
----------------------------------------------------------------------
-------------------------------------------------------------------
1. Sell Bikes
2. Order Bikes
3. Exit
Choose an option: 2
Let's order Bikes
-------------------------------------------------------
---------------------------------------------------------------------------------------------------------------
-----------
Bike ID Bike-Name       Company Name    Colour     Quantity     Price
---------------------------------------------------------------------------------------------------------------
-----------
1       Classic 350     Royal Enfield   Gun Metal Grey  47      $4500

2       KTM RC 2000     KTM Bike        Orange Black    55      $5300

3       Brutale 800     Mv Agusta       White Red       33      $21000

4       KTMduke 390     Bajaj auto      Grey black      26      $20000

5       Yamaha FZFI     yamaha company  dark blue       48      $10000

---------------------------------------------------------------------------------------------------------------
-----------
                                                                                                Activate Windo
```

```
Enter ID of the bike to order: 3

Enter quantity of bike to order: 3


---------------------------------------------------------------------------------------------------------------
-----------
Bike ID Bike-Name       Company Name    Colour     Quantity     Price
---------------------------------------------------------------------------------------------------------------
-----------
1       Classic 350     Royal Enfield   Gun Metal Grey  47      $4500

2       KTM RC 2000     KTM Bike        Orange Black    55      $5300

3       Brutale 800     Mv Agusta       White Red       36      $21000

4       KTMduke 390     Bajaj auto      Grey black      26      $20000

5       Yamaha FZFI     yamaha company  dark blue       48      $10000

---------------------------------------------------------------------------------------------------------------
-----------


The total price of the bike is:  $ 63000
do you want to order another bike,y/n= |
```

Figure 20 Implementation of program 9

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


Enter ID of the bike to order: 4

Enter quantity of bike to order: 4


-------------------------------------------------------------------------------------------------
-----------
Bike ID Bike-Name       Company Name    Colour     Quantity     Price
-------------------------------------------------------------------------------------------------
-----------
1       Classic 350     Royal Enfield   Gun Metal Grey  47      $4500

2       KTM RC 2000     KTM Bike        Orange Black    55      $5300

3       Brutale 800     Mv Agusta       White Red       36      $21000

4       KTMduke 390     Bajaj auto      Grey black      30      $20000

5       Yamaha FZFI     yamaha company  dark blue       48      $10000

-------------------------------------------------------------------------------------------------
-----------


The total price of the bike is:  $ 80000
do you want to order another bike,y/n=
```

```
The total price of the bike is:  $ 80000
do you want to order another bike,y/n= n
=================================================================================================

Enter shiping company name: xys
Enter shiping company address: kathmandu
Enter your contact number: 009
=================================================================================================

==================================Order Bike Receipt=============================================

=================================================================================================

==================================Bike Management System=========================================

Shiping company  name: xys

Shiping company address: kathmandu

contact Number: 009

the bike is order in this Date & Time : 2022-05-12 23:26:54.951929

=================================================================================================

S.N      Bike Name       Company            Quantity    price   total_amount
=================================================================================================
```

Figure 21 Implementation of program 10

```
==================================Order Bike Receipt==========================================

=============================================================================================

==================================Bike Management System======================================

Shiping company  name: xys

Shiping company address: kathmandu

contact Number: 009

the bike is order in this Date & Time : 2022-05-12 23:26:54.951929

=============================================================================================

S.N      Bike Name       Company            Quantity    price   total_amount
=============================================================================================

1       Brutale 800     Mv Agusta               3       63000   63000.0
=============================================================================================

2       KTMduke 390     Bajaj auto              4       80000   143000.0
=============================================================================================

-------------------------------------------------------------------
1. Sell Bikes
2. Order Bikes
3. Exit
Choose an option:
```


xys2022512233830 - Notepad

File Edit Format View Help

```
=============================================================================================
==================================Order Bike Receipt==========================================
=============================================================================================
==================================Bike Management System======================================
shiping company  name: xys
shiping company address: kathmandu
contact Number 009
the bike is order in this Date & Time : 2022-05-12 23:38:31.134591
=============================================================================================
S.N      Bike Name       Company            Quantity    price   total_amount
=============================================================================================
1       Brutale 800     Mv Agusta               3       63000   63000.0
=============================================================================================
2       KTMduke 390     Bajaj auto              4       80000   143000.0
=============================================================================================
```

Figure 22 Implementation of program 11

If the user input 3 the program will be exit and Thank you for using Bike Management System" will be displayed

```
*IDLE Shell 3.10.2*
File  Edit  Shell  Debug  Options  Window  Help
---------------------------------------------------------------------
                 Welcome to Bike management System
---------------------------------------------------------------------
-------------------------------------------------------------------
1. Sell Bikes
2. Order Bikes
3. Exit
Choose an option: 3
-------------------------------------------------------
Thank you for using Bike management System
-------------------------------------------------------
-------------------Have a nice day,bye-----------------------------------
-------------------------------------------------------------------------
```

If the user input any other number than 1,2 and 3 the INVALID Input. Please enter the number of the options provided in the screen will be displayed

```
*IDLE Shell 3.10.2*
File  Edit  Shell  Debug  Options  Window  Help
                 Welcome to Bike management System
---------------------------------------------------------------------
-------------------------------------------------------------------
1. Sell Bikes
2. Order Bikes
3. Exit
Choose an option: 4
-----------------------------------------------------
INVALID Input. Please enter the number of the options provided in the screen
-----------------------------------------------------
-------------------------------------------------------------------
```

If the user input any invalid data, plese,enter a valid information will be occur

```
---------------------------------------------------------------------
1. Sell Bikes
2. Order Bikes
3. Exit
Choose an option: s
-------------------------------------------------------------------
please,Enter a valid information
-------------------------------------------------------------------
---------------------------------------------------------------------
1. Sell Bikes
```

Figure 23 Implementation of program 12

3.2 Showing selling of bikes and adding the bikes

When the input value of bikeid and quantity is valid the stock will be decrease while sell of the bike

```
*IDLE Shell 3.10.2*
File  Edit  Shell  Debug  Options  Window  Help
>>>
========================= RESTART: D:\python cw\Main.py =========================
-----------------------------------------------------------------------
                 Welcome to Bike management System
-----------------------------------------------------------------------
-------------------------------------------------------------------
1. Sell Bikes
2. Order Bikes
3. Exit
Choose an option: 1
Let's sell  Bikes
-----------------------------------------------------


-------------------------------------------------------------------------------
-----------------------------------------------
Bike ID Bike-Name       Company Name    Colour     Quantity     Price
-------------------------------------------------------------------------------
-----------------------------------------------
1       Classic 350     Royal Enfield   Gun Metal Grey  49      $4500

2       KTM RC 2000     KTM Bike        Orange Black    52      $5300

3       Brutale 800     Mv Agusta       White Red       33      $21000

4       KTMduke 390     Bajaj auto      Grey black      26      $20000

5       Yamaha FZFI     yamaha company  dark blue       48      $10000

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

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


Enter ID of the bike to sell: 1

Enter quantity of bike to sell: 2


-------------------------------------------------------------------------------
-----------
Bike ID Bike-Name       Company Name    Colour     Quantity     Price
-------------------------------------------------------------------------------
-----------
1       Classic 350     Royal Enfield   Gun Metal Grey  47      $4500

2       KTM RC 2000     KTM Bike        Orange Black    52      $5300

3       Brutale 800     Mv Agusta       White Red       33      $21000

4       KTMduke 390     Bajaj auto      Grey black      26      $20000

5       Yamaha FZFI     yamaha company  dark blue       48      $10000

-------------------------------------------------------------------------------
-----------


The total price of the bike is:  $ 9000
do you want to sell another bike,y/n=
```

Figure 24 Deduction of stock

When the input value of bikeid and quantity is valid the stock will be increase while ordering bike

```
*IDLE Shell 3.10.2*
File  Edit  Shell  Debug  Options  Window  Help
                 Welcome to Bike management System
-------------------------------------------------------------------
-------------------------------------------------------------------
1. Sell Bikes
2. Order Bikes
3. Exit
Choose an option: 2
Let's order Bikes
-----------------------------------------------------


-------------------------------------------------------------------------------------------------------------
-----------
Bike ID Bike-Name       Company Name    Colour     Quantity     Price
-------------------------------------------------------------------------------------------------------------
-----------
1       Classic 350     Royal Enfield   Gun Metal Grey  47      $4500

2       KTM RC 2000     KTM Bike        Orange Black    55      $5300

3       Brutale 800     Mv Agusta       White Red       33      $21000

4       KTMduke 390     Bajaj auto      Grey black      26      $20000

5       Yamaha FZFI     yamaha company  dark blue       48      $10000

-------------------------------------------------------------------------------------------------------------
-----------
                                                                                                Activate Windo
```

```
Enter ID of the bike to order: 3

Enter quantity of bike to order: 3


-------------------------------------------------------------------------------------------------------------
-----------
Bike ID Bike-Name       Company Name    Colour     Quantity     Price
-------------------------------------------------------------------------------------------------------------
-----------
1       Classic 350     Royal Enfield   Gun Metal Grey  47      $4500

2       KTM RC 2000     KTM Bike        Orange Black    55      $5300

3       Brutale 800     Mv Agusta       White Red       36      $21000

4       KTMduke 390     Bajaj auto      Grey black      26      $20000

5       Yamaha FZFI     yamaha company  dark blue       48      $10000

-------------------------------------------------------------------------------------------------------------
-----------


The total price of the bike is:  $ 63000
do you want to order another bike,y/n=
```

Figure 25 Increase in stock

3.3 Creation of the txt file

'Do you want to sell another bike?' option is displayed in which if you type y, the process is repeated but if the input is n, then a order receipt is created after input name,address and contact number in shell and txt file

```
The total price of the bike is:  $ 80000
do you want to order another bike,y/n= n
==================================================================================================

Enter shiping company name: xys
Enter shiping company address: kathmandu
Enter your contact number: 009
==================================================================================================

==================================Order Bike Receipt==============================================

==================================================================================================

==================================Bike Management System==========================================

Shiping company  name: xys

Shiping company address: kathmandu

contact Number: 009

the bike is order in this Date & Time : 2022-05-12 23:26:54.951929

==================================================================================================

S.N      Bike Name       Company            Quantity    price   total_amount
==================================================================================================
```

Figure 26 Receipt

3.4 Open txt and show the bill

After input the name,address and contact number in shell and txt file

```
anish lamichhane2022512223410 - Notepad
File  Edit  Format  View  Help
==============================================================================
==============================Sell Bike Receipt==================================
==============================================================================
===========================Bike Management System=================================
Customer name: anish lamichhane
Customer address: kamalbinayak bkt 7
Customer contact number: 09999
the bike sold in this Date & Time : 2022-05-12 22:34:11.043546
=================================================================================================
S.N      Bike Name       Company           Quantity     price   total_amount
=================================================================================================
1       Classic 350     Royal Enfield           2       9000    9000
=================================================================================================
2       KTM RC 2000     KTM Bike                3       15900   24900
=================================================================================================
```

Figure 27 Bike.txt

3.5 Close the program

If the user input 3 the program will be exit and Thank you for using Bike Management System" will be displayed

```
*IDLE Shell 3.10.2*
File  Edit  Shell  Debug  Options  Window  Help
---------------------------------------------------------------------
                Welcome to Bike management System
---------------------------------------------------------------------
--------------------------------------------------------------------
1. Sell Bikes
2. Order Bikes
3. Exit
Choose an option: 3
-------------------------------------------------------
Thank you for using Bike management System
-------------------------------------------------------
-------------------Have a nice day,bye-----------------------------------
-----------------------------------------------------------------------
```

Figure 28 Exit

4. Testing

4.1 Test 1

- Show implementation of try, except

= Provide invalid input and show the message

Table 1 Test 1

Test no	1
Objective	Show implementation of try, except Provide invalid input and show the message
Action	Test 1.1 • Open the main,py file or open the terminal to run the module. • When the program asks you to select your choice input anything except 1,2, and 3 • To provide invalid input and input wrong numeric value Test 1.2 • To test the implementation of try and except, “invalid input error” which is a string is entered.
Expected Result:	Display provide invalid input will apear
Actual	provide invalid input which value doesnot exixt will display please,Enter a valid information will display
Conclusion:	The test was successful.

```
                  Welcome to Bike management System
-----------------------------------------------------------------------
-------------------------------------------------------------------
1. Sell Bikes
2. Order Bikes
3. Exit
Choose an option: 4
-------------------------------------------------------
INVALID Input. Please enter the number of the options provided in the screen
-------------------------------------------------------
-------------------------------------------------------------------
1. Sell Bikes
2. Order Bikes
3. Exit
Choose an option: s
-----------------------------------------------------------------
please,Enter a valid information
-----------------------------------------------------------------
-------------------------------------------------------------------
1. Sell Bikes
2. Order Bikes
3. Exit
Choose an option:
```

Figure 29 Test 1

4.2 Test 2

- Selection adding bikes in stock and selling of bikes
  = Provide negative value as input
  = Provide non existed value as input

Table 2 Test 2

Test no	2
Objective	Selection adding bikes in stock and selling of bikes Provide negative value as input Provide non existed value as input
Action	Test 2.1 • Open the main,py file or open the terminal to run the module. • Select 2 as input and input a entered bike_id • To provide negative value and in bike_id Test 2.2 • Select 2 as input and input a entered bike_id • Input the non existing as input
Expected Result:	Display provide invalid input will appear
Actual	provide invalid input which value does not exit will display please, Enter a valid information will display
Conclusion:	The test was successful.

```
Select C:\WINDOWS\py.exe
----------------------------------------------------------------------
                 Welcome to Bike management System
----------------------------------------------------------------------
-------------------------------------------------------------------
1. Sell Bikes
2. Order Bikes
3. Exit
Choose an option: 1
Let's sell  Bikes
-----------------------------------------------------


-------------------------------------------------------------------------------------------
Bike ID Bike-Name       Company Name    Colour     Quantity     Price
-------------------------------------------------------------------------------------------
1       Classic 350     Royal Enfield   Gun Metal Grey  47      $4500

2       KTM RC 2000     KTM Bike        Orange Black    55      $5300

3       Brutale 800     Mv Agusta       White Red       51      $21000

4       KTMduke 390     Bajaj auto      Grey black      46      $20000

5       Yamaha FZFI     yamaha company  dark blue       48      $10000

-------------------------------------------------------------------------------------------


Enter ID of the bike to sell: -1
Please provide a valid bike ID !!!
```

Figure 30 Test 2.1

```
C:\WINDOWS\py.exe
-----------------------------------------------------------------------
                 Welcome to Bike management System
-----------------------------------------------------------------------
-------------------------------------------------------------------
1. Sell Bikes
2. Order Bikes
3. Exit
Choose an option: 1
Let's sell  Bikes
-----------------------------------------------------


-------------------------------------------------------------------------------------------------------------------------------
Bike ID Bike-Name       Company Name    Colour     Quantity     Price
-------------------------------------------------------------------------------------------------------------------------------
1       Classic 350     Royal Enfield   Gun Metal Grey  45      $4500

2       KTM RC 2000     KTM Bike        Orange Black    55      $5300

3       Brutale 800     Mv Agusta       White Red       51      $21000

4       KTMduke 390     Bajaj auto      Grey black      46      $20000

5       Yamaha FZFI     yamaha company  dark blue       48      $10000

-------------------------------------------------------------------------------------------------------------------------------


Enter ID of the bike to sell: 6
Please provide a valid bike ID !!!
```

Figure 31 Test 2.2

4.3 Test 3

- File generation of adding bikes in stock
= Show complete adding bikes in stock
= Show output in the shell as well
= Finally show the ordering/adding the bikes in stock note in txt file

Table 3 Test 3

Test no	3
Objective	File generation of adding bikes in stock = Show complete adding bikes in stock = Show output in the shell as well = Finally show the ordering/adding the bikes in stock note in txt file
Action	Test 3.1 • Open the main,py file or open the terminal to run the module. • Select 2 as input and input a entered bike_id • To provide valid value and in bike_id • Input the name address and contact number
Expected Result:	Receipt and txt file will appear
Actual	Receipt and txt file will appear
Conclusion:	The test was successful.

```
C:\WINDOWS\py.exe
-----------------------------------------------------------------------
                 Welcome to Bike management System
-----------------------------------------------------------------------
-------------------------------------------------------------------
1. Sell Bikes
2. Order Bikes
3. Exit
Choose an option: 2
Let's order Bikes
-----------------------------------------------------


-----------------------------------------------------------------------------------------------------------------------------------
Bike ID Bike-Name       Company Name    Colour     Quantity     Price
-----------------------------------------------------------------------------------------------------------------------------------
1       Classic 350     Royal Enfield   Gun Metal Grey  43      $4500

2       KTM RC 2000     KTM Bike        Orange Black    57      $5300

3       Brutale 800     Mv Agusta       White Red       51      $21000

4       KTMduke 390     Bajaj auto      Grey black      46      $20000

5       Yamaha FZFI     yamaha company  dark blue       48      $10000

-----------------------------------------------------------------------------------------------------------------------------------


Enter ID of the bike to order: 1

Enter quantity of bike to order: 1


-----------------------------------------------------------------------------------------------------------------------------------
Bike ID Bike-Name       Company Name    Colour     Quantity     Price
-----------------------------------------------------------------------------------------------------------------------------------
1       Classic 350     Royal Enfield   Gun Metal Grey  44      $4500

2       KTM RC 2000     KTM Bike        Orange Black    57      $5300

3       Brutale 800     Mv Agusta       White Red       51      $21000

4       KTMduke 390     Bajaj auto      Grey black      46      $20000
```

Figure 32 Test 3.1

```
C:\WINDOWS\py.exe

The total price of the bike is:  $ 4500
do you want to order another bike,y/n= y


-------------------------------------------------------------------------------------------------------
Bike ID Bike-Name       Company Name    Colour     Quantity     Price
-------------------------------------------------------------------------------------------------------
1       Classic 350     Royal Enfield   Gun Metal Grey  44      $4500

2       KTM RC 2000     KTM Bike        Orange Black    57      $5300

3       Brutale 800     Mv Agusta       White Red       51      $21000

4       KTMduke 390     Bajaj auto      Grey black      46      $20000

5       Yamaha FZFI     yamaha company  dark blue       48      $10000

-------------------------------------------------------------------------------------------------------


Enter ID of the bike to order: 2

Enter quantity of bike to order: 2


-------------------------------------------------------------------------------------------------------
Bike ID Bike-Name       Company Name    Colour     Quantity     Price
-------------------------------------------------------------------------------------------------------
1       Classic 350     Royal Enfield   Gun Metal Grey  44      $4500

2       KTM RC 2000     KTM Bike        Orange Black    59      $5300

3       Brutale 800     Mv Agusta       White Red       51      $21000

4       KTMduke 390     Bajaj auto      Grey black      46      $20000

5       Yamaha FZFI     yamaha company  dark blue       48      $10000

-------------------------------------------------------------------------------------------------------


The total price of the bike is:  $ 10600
do you want to order another bike,y/n=
```

Figure 33 Test 3.2

```
C:\WINDOWS\py.exe
4       KTMduke 390     Bajaj auto      Grey black      46      $20000

5       Yamaha FZFI     yamaha company  dark blue       48      $10000

-------------------------------------------------------------------------------------------------------------------------------


The total price of the bike is:  $ 10600
do you want to order another bike,y/n= n
Enter distributors's name: xys
Enter shiping company address: ktm
Enter your contact number: 009
==========================================================================================================

==================================Order Bike Receipt=========================================================

==============================================================================================================

==================================Bike Management System=====================================================

shiping company  name: xys

shiping company address: ktm

contact Number 009

the bike is order in this Date & Time : 2022-05-13 03:07:36.308688

==========================================================================================================

S.N      Bike Name       Company            Quantity    price   total_amount
==========================================================================================================

1       Classic 350     Royal Enfield           1       4500    4500
==========================================================================================================

2       KTM RC 2000     KTM Bike                2       10600   15100
==========================================================================================================

-------------------------------------------------------------------
1. Sell Bikes
2. Order Bikes
3. Exit
Choose an option:
```

Figure 34 Test 3.3

```
xys20225133736 - Notepad
File  Edit  Format  View  Help
==========================================================================================================
==================================Order Bike Receipt=========================================================
==========================================================================================================
==================================Bike Management System=====================================================
shiping company  name: xys
shiping company address: ktm
contact Number 009
the bike is order in this Date & Time : 2022-05-13 03:07:36.308688
==========================================================================================================
S.N      Bike Name       Company            Quantity    price   total_amount
==========================================================================================================
1       Classic 350     Royal Enfield           1       4500    4500
==========================================================================================================
2       KTM RC 2000     KTM Bike                2       10600   15100
==========================================================================================================
```

Figure 35 Test 3.4

4.4 Test 4

- File generation of adding bikes in stock
  = Show complete adding bikes in stock
  = Show output in the shell as well
  = Finally show the ordering/adding the bikes in stock note in txt file

Table 4 Test 4

Test no	4
Objective	File generation of adding bikes in stock = Show complete adding bikes in stock = Show output in the shell as well = Finally show the ordering/adding the bikes in stock note in txt file
Action	Test 4.1 • Open the main,py file or open the terminal to run the module. • Select 2 as input and input a entered bike_id • To provide valid value and in bike_id • Input the name address and contact number
Expected Result:	Receipt and txt file will appear
Actual	Receipt and txt file will appear
Conclusion:	The test was successful.

```
C:\WINDOWS\py.exe
----------------------------------------------------------------------
                 Welcome to Bike management System
----------------------------------------------------------------------
-------------------------------------------------------------------
1. Sell Bikes
2. Order Bikes
3. Exit
Choose an option: 1
Let's sell  Bikes
-----------------------------------------------------


-----------------------------------------------------------------------------------------------------------
Bike ID Bike-Name       Company Name    Colour    Quantity      Price
-----------------------------------------------------------------------------------------------------------
1       Classic 350     Royal Enfield   Gun Metal Grey  43      $4500

2       KTM RC 2000     KTM Bike        Orange Black    53      $5300

3       Brutale 800     Mv Agusta       White Red       51      $21000

4       KTMduke 390     Bajaj auto      Grey black      46      $20000

5       Yamaha FZFI     yamaha company  dark blue       48      $10000

-----------------------------------------------------------------------------------------------------------


Enter ID of the bike to sell: 1

Enter quantity of bike to sell: 2


-----------------------------------------------------------------------------------------------------------
Bike ID Bike-Name       Company Name    Colour    Quantity      Price
-----------------------------------------------------------------------------------------------------------
1       Classic 350     Royal Enfield   Gun Metal Grey  41      $4500

2       KTM RC 2000     KTM Bike        Orange Black    53      $5300

3       Brutale 800     Mv Agusta       White Red       51      $21000

4       KTMduke 390     Bajaj auto      Grey black      46      $20000
```

Figure 36 Test 4.1

```
C:\WINDOWS\py.exe
do you want to sell another bike,y/n= y
Let's sell  Bikes
----------------------------------------------------


-----------------------------------------------------------------------------------------------------------------------------------
Bike ID Bike-Name       Company Name    Colour    Quantity      Price
-----------------------------------------------------------------------------------------------------------------------------------
1       Classic 350     Royal Enfield   Gun Metal Grey  41      $4500

2       KTM RC 2000     KTM Bike        Orange Black    53      $5300

3       Brutale 800     Mv Agusta       White Red       51      $21000

4       KTMduke 390     Bajaj auto      Grey black      46      $20000

5       Yamaha FZFI     yamaha company  dark blue       48      $10000

-----------------------------------------------------------------------------------------------------------------------------------


Enter ID of the bike to sell: 1

Enter quantity of bike to sell: 2


-----------------------------------------------------------------------------------------------------------------------------------
Bike ID Bike-Name       Company Name    Colour    Quantity      Price
-----------------------------------------------------------------------------------------------------------------------------------
1       Classic 350     Royal Enfield   Gun Metal Grey  39      $4500

2       KTM RC 2000     KTM Bike        Orange Black    53      $5300

3       Brutale 800     Mv Agusta       White Red       51      $21000

4       KTMduke 390     Bajaj auto      Grey black      46      $20000

5       Yamaha FZFI     yamaha company  dark blue       48      $10000

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

Figure 37 Test 4.2

```
C:\WINDOWS\py.exe
-----------------------------------------------------------------------------------------------------------------------------------


The total price of the bike is:  $ 9000
do you want to sell another bike,y/n= n
---------------------------------Customer details-------------------------------------
-------------------------------------------------------------------------------------------

Enter customer's name: anish
Enter your address: bkt
Enter your contact number: 09
-------------------------------------------------------------------------------------------
==============================================================================

==================================Sell Bike Receipt=================================

==============================================================================

==========================Bike Management System==================================

Customer name: anish

Customer address: bkt

Customer contact number: 09

the bike sold in this Date & Time : 2022-05-13 02:44:29.541352

=================================================================================================================

S.N      Bike Name       Company            Quantity    price   total_amount
=================================================================================================================

1       Classic 350     Royal Enfield           2       9000    9000
=================================================================================================================

2       Classic 350     Royal Enfield           2       9000    18000
=================================================================================================================

-------------------------------------------------------------------
1. Sell Bikes
2. Order Bikes
3. Exit
```

Figure 38 Test 4.3

```
anish202251324429 - Notepad
File  Edit  Format  View  Help
==============================================================================
==============================Sell Bike Receipt=================================
==============================================================================
============================Bike Management System==================================
Customer name: anish
Customer address: bkt
Customer contact number: 09
the bike sold in this Date & Time : 2022-05-13 02:44:29.525734
=================================================================================================================
S.N      Bike Name       Company            Quantity     price   total_amount
=================================================================================================================
1       Classic 350     Royal Enfield           2       9000    9000
=================================================================================================================
2       Classic 350     Royal Enfield           2       9000    18000
=================================================================================================================
```

Figure 39 Test 4.4

4.5 Test 5

- Show the update in stock of bike
  = Show the quantity being deducted while selling the bike
  = Show the quantity being added while ordering/adding the bike

Table 5 Test 5

Test no	5
Objective	Show the update in stock of bike = Show the quantity being deducted while selling the bike = Show the quantity being added while ordering/adding the bike
Action	Test 5.1 • Open the main,py file or open the terminal to run the module. • Select 2 as input and input a entered bike_id • To provide valid value and in sell bike_id Test 5.2 • Open the main,py file or open the terminal to run the module. • Select 2 as input and input a entered bike_id • To provide valid value and in order bike_id
Expected Result:	Stock will be increase and decrease
Actual	Stock is increase and decrease
Conclusion:	The test was successful.

```
C:\WINDOWS\py.exe
-----------------------------------------------------------------
1. Sell Bikes
2. Order Bikes
3. Exit
Choose an option: 1
Let's sell  Bikes
-----------------------------------------------------


-------------------------------------------------------------------------------------------------------------------------------
Bike ID Bike-Name       Company Name    Colour    Quantity      Price
-------------------------------------------------------------------------------------------------------------------------------
1       Classic 350     Royal Enfield   Gun Metal Grey  44      $4500

2       KTM RC 2000     KTM Bike        Orange Black    59      $5300

3       Brutale 800     Mv Agusta       White Red       51      $21000

4       KTMduke 390     Bajaj auto      Grey black      46      $20000

5       Yamaha FZFI     yamaha company  dark blue       48      $10000

-------------------------------------------------------------------------------------------------------------------------------


Enter ID of the bike to sell: 1

Enter quantity of bike to sell: 4


-------------------------------------------------------------------------------------------------------------------------------
Bike ID Bike-Name       Company Name    Colour    Quantity      Price
-------------------------------------------------------------------------------------------------------------------------------
1       Classic 350     Royal Enfield   Gun Metal Grey  40      $4500

2       KTM RC 2000     KTM Bike        Orange Black    59      $5300

3       Brutale 800     Mv Agusta       White Red       51      $21000

4       KTMduke 390     Bajaj auto      Grey black      46      $20000

5       Yamaha FZFI     yamaha company  dark blue       48      $10000
```

Figure 40 Test 5.1

```
C:\WINDOWS\py.exe
1. Sell Bikes
2. Order Bikes
3. Exit
Choose an option: 2
Let's order Bikes
----------------------------------------------------


-------------------------------------------------------------------------------------------------------------------------------------------
Bike ID Bike-Name       Company Name    Colour     Quantity     Price
-------------------------------------------------------------------------------------------------------------------------------------------
1       Classic 350     Royal Enfield   Gun Metal Grey  40      $4500

2       KTM RC 2000     KTM Bike        Orange Black    59      $5300

3       Brutale 800     Mv Agusta       White Red       51      $21000

4       KTMduke 390     Bajaj auto      Grey black      46      $20000

5       Yamaha FZFI     yamaha company  dark blue       48      $10000

-------------------------------------------------------------------------------------------------------------------------------------------


Enter ID of the bike to order: 2

Enter quantity of bike to order: 2


-------------------------------------------------------------------------------------------------------------------------------------------
Bike ID Bike-Name       Company Name    Colour     Quantity     Price
-------------------------------------------------------------------------------------------------------------------------------------------
1       Classic 350     Royal Enfield   Gun Metal Grey  40      $4500

2       KTM RC 2000     KTM Bike        Orange Black    61      $5300

3       Brutale 800     Mv Agusta       White Red       51      $21000

4       KTMduke 390     Bajaj auto      Grey black      46      $20000

5       Yamaha FZFI     yamaha company  dark blue       48      $10000
```

Figure 41 Test 5.2

5.Conclusion

The report is bases on the bike management system. according to instruction, the project should be working in the loop and its will sell and buy a bike. This project is made the program for the user-friendly in order to people who have basic knowledge can use this type of program. this program consists of three modules. Main,py is the main function of the program. sell function is use for to sell of bike in program. Order function is for the order of the bike. The main.py has imported other 2 modules for the smoothness to run a program. The main,py is the brain of the program without it the program will be just a useless code. People those have a little knowledge about programing can run this program in rea life with suitable environment. I get many error while coding the programing after that suggestion to the module leader and tutot my code become ore smooth than it was before.

While doing, coursework the main aspect is the time management, even the time provided us is 5 weeks. there is different factor that was affecting to the coursework, time management. Even the coursework deadline becoming very close I never panic and performed by coursework for smooth result Every project needs data or information analysis and research to be performed and finished. This is essential to be have sufficient information and analysis on the topic. It is nearly impossible to complete coursework without appropriate study and skills. It significantly enabled the growth of knowledge on the topic, which otherwise would have been limited to lectures and tutorials. As a result, various websites were used as a research source for the correct completion of this project.

The billing part was just confusing part because most of the time my bill does not append the data of the biked which I have been input at the beginning of the project. even when I was doing billing part, the whole information should be inputted again. The problem shows the solution to solve my queries. With much effort, I was able to solve this problem and I am sure this experience will make my future attempt much smoother.

6.Appendix

6.1 Main.py

```
# Importing all the modules.
import datetime  # Importing date from datetime.
import sellfunction
import orderfunction

def our_message():
    """ Main function of the program."""
    print("-----------------------------------------------------------------------")
    print("                Welcome to Bike management System            ")
    print("-----------------------------------------------------------------------")
def show_options():#Its help for selecting task in program for better performation of
program
    print("1. Sell Bikes")
    print("2. Order Bikes")
    print("3. Exit")

def sells_bikes():#This function is used to sell of bikes
        print("Let's sell  Bikes")
        print("-------------------------------------------------------")

def order_bikes():#This function is used to order of bikes
        print("Let's order Bikes")
        print("-------------------------------------------------------")
def let_exit():#This function is used to close the program
        print("-------------------------------------------------------")
        print("Thank you for using Bike management System")
        print("-------------------------------------------------------")
def invalid_input():#This function is used to provide invalid input which value doesnot
exixt in program
        print("-------------------------------------------------------")
        print("INVALID Input. Please enter the number of the options provided in the
screen")
        print("-------------------------------------------------------")



counter = True
our_message()
while counter :
    print("-----------------------------------------------------------------------")
```

```
    show_options()
    #Exception handling is used for preventation form unwanted error.
    try:
        user_input = int(input("Choose an option: "))

        if user_input == 1:
            counter = False
            Sold_bike = []
            while counter == False:
                sells_bikes()
                sellfunction.ShowBikes()
                entered_bike_id = sellfunction.valid_bike_id()
                customer_quantity = sellfunction.valid_bike_quantity(entered_bike_id)
                bike_list = sellfunction.return_2d_list()
                total_cost = sellfunction.total_bike_cost(entered_bike_id,customer_quantity)
                sellfunction.update_stock(customer_quantity,entered_bike_id)
                Sold_bike =
sellfunction.new_bill(bike_list,entered_bike_id,Sold_bike,customer_quantity,total_cost)
                total_cost = int(bike_list[entered_bike_id-1][5].replace("$",""))*
customer_quantity
                print("The total price of the bike is: ","$",total_cost)

                counter = sellfunction.loop()
            sellfunction.print_bill_sell(bike_list,Sold_bike)

        elif user_input == 2:
            order_bikes()
            counter = False
            order_bike = []
            while counter == False:
                orderfunction.ShowBikes()
                entered_bike_id =orderfunction.valid_bike_id_order()
                customer_quantity =
orderfunction.valid_bike_quantity_order(entered_bike_id)
                bike_list = orderfunction.return_2d_list()
                total_cost= orderfunction.total_bike_cost(entered_bike_id,customer_quantity)
                order_bike =
orderfunction.new_bill(bike_list,entered_bike_id,order_bike,customer_quantity,total_cos
t)
                orderfunction.update_stock_order(customer_quantity,entered_bike_id)
                print("The total price of the bike is: ","$",total_cost)
                counter = orderfunction.loops()
            orderfunction.print_bill_order(bike_list,order_bike )
        elif user_input == 3:
                let_exit()
                # Exit logic.
```

```
            counter = False
            print("-------------------Have a nice day,bye-----------------------------------")
            print("------------------------------------------------------------------------")

        else:
            invalid_input()
    except:
        print("-----------------------------------------------------------------")
        print("please,Enter a valid information")
        print("-----------------------------------------------------------------")
```

6.2 sell function.py

```
def ShowBikes():#shows the bike in the tabular form
    print("\n")
    print("-------------------------------------------------------------------------------------------------------------
-----------------------")
    print("Bike ID\tBike-Name\tCompany Name\tColour\t  Quantity\tPrice")
    print("-------------------------------------------------------------------------------------------------------------
-----------------------")
    file = open("bikes.txt", "r")

    for line in file:
        print( line.replace(",","\t"))

    print("-------------------------------------------------------------------------------------------------------------
-----------------------")
    print("\n")
    file.close()

def return_2d_list():
    '''generate 2d list form bikes.txt file and store in bike_list'''
    read_file = open("bikes.txt", "r")
    bike_list = []
    for bike in read_file:
        bike = bike.replace("\n", "")
        bike_list.append(bike.split(","))

    return bike_list



def valid_bike_id():
    ''' Validates bikeid  for sells form user input.'''
```

```
    #Exception handling is used for preventation form unwanted error.
    id = True
    while id == True:
        try:
            validBikeId = int(input("Enter ID of the bike to sell: "))
            while validBikeId <= 0 or validBikeId >len(return_2d_list()):
                print("Please provide a valid bike ID !!!")
                print("--------------------------------------------------")
                validBikeId = int(input("Enter ID of the bike to sell: "))
                print("--------------------------------------------------")
            return validBikeId

        except :
            print("-----------------------------------------------------------------")
            print("please,Enter a valid information")
            print("-----------------------------------------------------------------")
            continue

def valid_bike_quantity(entered_bike_id):
    ''' Validates bike quantity for sells form user input.'''
    bike_id = entered_bike_id
    #Exception handling is used for preventation form unwanted error.
    qty = True
    while qty == True:
        try:
            bike_quantity = int(input("\nEnter quantity of bike to sell: "))
            while bike_quantity <= 0 or bike_quantity > int(return_2d_list()[bike_id - 1][4]):
                print("\nPlease provide a valid bike quantity!!!\n")
                print("--------------------------------------------------")
                validBikeId = int(input("Enter quantity of the bike to sell: "))
                print("--------------------------------------------------")
                ShowBikes()
            return bike_quantity
            break
        except:
            print("-----------------------------------------------------------------")
            print("please,Enter a valid information")
            print("-----------------------------------------------------------------")


def total_bike_cost(entered_bike_id,customer_quantity):
    ''' total_cost  for sells of the bike .'''
    bike_list = return_2d_list()
    total_cost = int(bike_list[entered_bike_id-1][5].replace("$",""))* customer_quantity
    return total_cost
```

```
def loop():# using loop for the proper functioning of program
    inputdata = input("do you want to sell another bike,y/n= ").upper()
    if(inputdata == "Y"):
        counter = False
    else:
        counter = True
    return counter

def update_stock(customer_quantity,entered_bike_id):
    ''' Updates the stocks and reduce stock after selling of bike.'''
    bike_list = return_2d_list()
    bike_list[entered_bike_id-1][4] = int(bike_list[entered_bike_id-1][4]) -
customer_quantity
    file = open("bikes.txt","w")
    for bike in bike_list:

file.write(str(bike[0])+","+str(bike[1])+","+str(bike[2])+","+str(bike[3])+","+str(bike[4])+","+s
tr(bike[5])+"\n")
    file.close()
    ShowBikes()


def new_bill(bike_list,entered_bike_id,Sold_bike,customer_quantity,total_cost):
    '''It is used for billing receipt of sells bike'''
    bike_list = return_2d_list()
    for bike in bike_list:
        if(int(bike[0]) == entered_bike_id):
                bike[4] = customer_quantity
                bike[5] = "$"+str(total_cost)
                Sold_bike.append(bike)
    return Sold_bike

def show_date():#defining show_date() for datetime
    import datetime
    year = str(datetime.datetime.now() .year)
    month = str(datetime.datetime.now().month)
    day = str(datetime.datetime.now().day)
    hour = str(datetime.datetime.now().hour)
    minute = str(datetime.datetime.now().minute)
    second = str(datetime.datetime.now().second)
    return year + month + day + hour + minute + second
show_date()
def print_bill_sell(bike_list,Sold_bike):#Creating sells receipt in txt and shell.
    import datetime
    print("--------------------------------Customer details--------------------------------------")
```

```
    print("-----------------------------------------------------------------------------------------\n")
    personName = input("Enter customer's name: ")
    customer_address = input("Enter your address: ")
    customer_contactNumber = input("Enter your contact number: ")
    bike_list = return_2d_list()
    print("-----------------------------------------------------------------------------------------")
    file= open(personName +""+show_date()+".txt", "a")

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

print("==============================================================
======================\n")
    file.write("==============================Sell Bike
Receipt===================================\n")
    print("===================================Sell Bike
Receipt===================================\n")

file.write("==============================================================
========================\n")
    file.write("============================Bike Management
System==================================\n")

print("==============================================================
====================\n")
    print("============================Bike Management
System==================================\n")
    file.write("Customer name: " + personName + "\n")
    file.write("Customer address: " + customer_address + "\n")
    file.write("Customer contact number: " +customer_contactNumber + "\n")
    file.write("the bike sold in this Date & Time : " + str(datetime.datetime.now()) + "\n")
    print("Customer name: " + personName + "\n")
    print("Customer address: " + customer_address + "\n")
    print("Customer contact number: " +customer_contactNumber + "\n")
    print("the bike sold in this Date & Time : " + str(datetime.datetime.now()) + "\n")
    #using now() function to get current date and time and store in variable


file.write("==============================================================
=================================================\n")
    file.write("S.N\t Bike Name\t Company \t   Quantity\tprice\ttotal_amount\n")

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

```
print("==================================================================
=============================================\n")
    print("S.N\t Bike Name\t Company\t    Quantity\tprice\ttotal_amount")

print("==================================================================
=============================================\n")
    SNo = 0
    total_amount = 0
    for bike in Sold_bike:
        SNo = SNo + 1
        bike[5] = bike[5].replace("$","")
        total_amount = total_amount + int(bike[5])

file.write(str(SNo)+"\t"+(str(bike[1])+"\t"+str(bike[2])+"\t\t"+str(bike[4])+"\t"+str(bike[5])+"\t
"+str(total_amount)+"\t\t"))

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

print(str(SNo)+"\t"+(str(bike[1])+"\t"+str(bike[2])+"\t\t"+str(bike[4])+"\t"+str(bike[5])+"\t"+st
r(total_amount)+"\t\t"))

print("==================================================================
=============================================\n")
    file.close
```

6.3 Orderfunction.py

```
def return_2d_list():
    '''generate 2d list form bikes.txt file and store in bike_list'''
    read_file = open("bikes.txt", "r")
    bike_list = []
    for bike in read_file:
        bike = bike.replace("\n", "")
        bike_list.append(bike.split(","))

    return bike_list

def ShowBikes():#shows the bike in the tabular form
    print("\n")
    print("-----------------------------------------------------------------------------------------------------
----------------------")
    print("Bike ID\tBike-Name\tCompany Name\tColour\t   Quantity\tPrice")
```

```
    print("-----------------------------------------------------------------------------------------------------------
-----------------------")
    file = open("bikes.txt", "r")

    for line in file:
        print( line.replace(","","\t"))

    print("-----------------------------------------------------------------------------------------------------------
-----------------------")
    print("\n")
    file.close()
```

6.3 orderfunction .py

```
def valid_bike_id_order():
    ''' Validates bikeid  for order form user input.'''
    #Exception handling is used for preventation form unwanted error.
    id = True
    while id == True:
        try:
            validBikeId = int(input("Enter ID of the bike to order: "))
            while validBikeId <= 0 or validBikeId >len(return_2d_list()):
                print("Please provide a valid bike ID !!!")
                print("--------------------------------------------------")
                validBikeId = int(input("Enter ID of the bike to order: "))
                print("--------------------------------------------------")
            return validBikeId

        except :
            print("-----------------------------------------------------------------")
            print("please,Enter a valid information")
            print("-----------------------------------------------------------------")
            continue
def valid_bike_quantity_order(entered_bike_id):
    bike_id = entered_bike_id
    ''' Validates bike quantity for sells form user input.'''
    #Exception handling is used for preventation form unwanted error.

    qty = True
    while qty == True:
        try:
            bike_quantity = int(input("\nEnter quantity of bike to order: "))
            while bike_quantity <= 0 or bike_quantity > int(return_2d_list()[bike_id - 1][4]):
                print("\nPlease provide a valid bike quantity!!!\n")
                print("--------------------------------------------------")
```

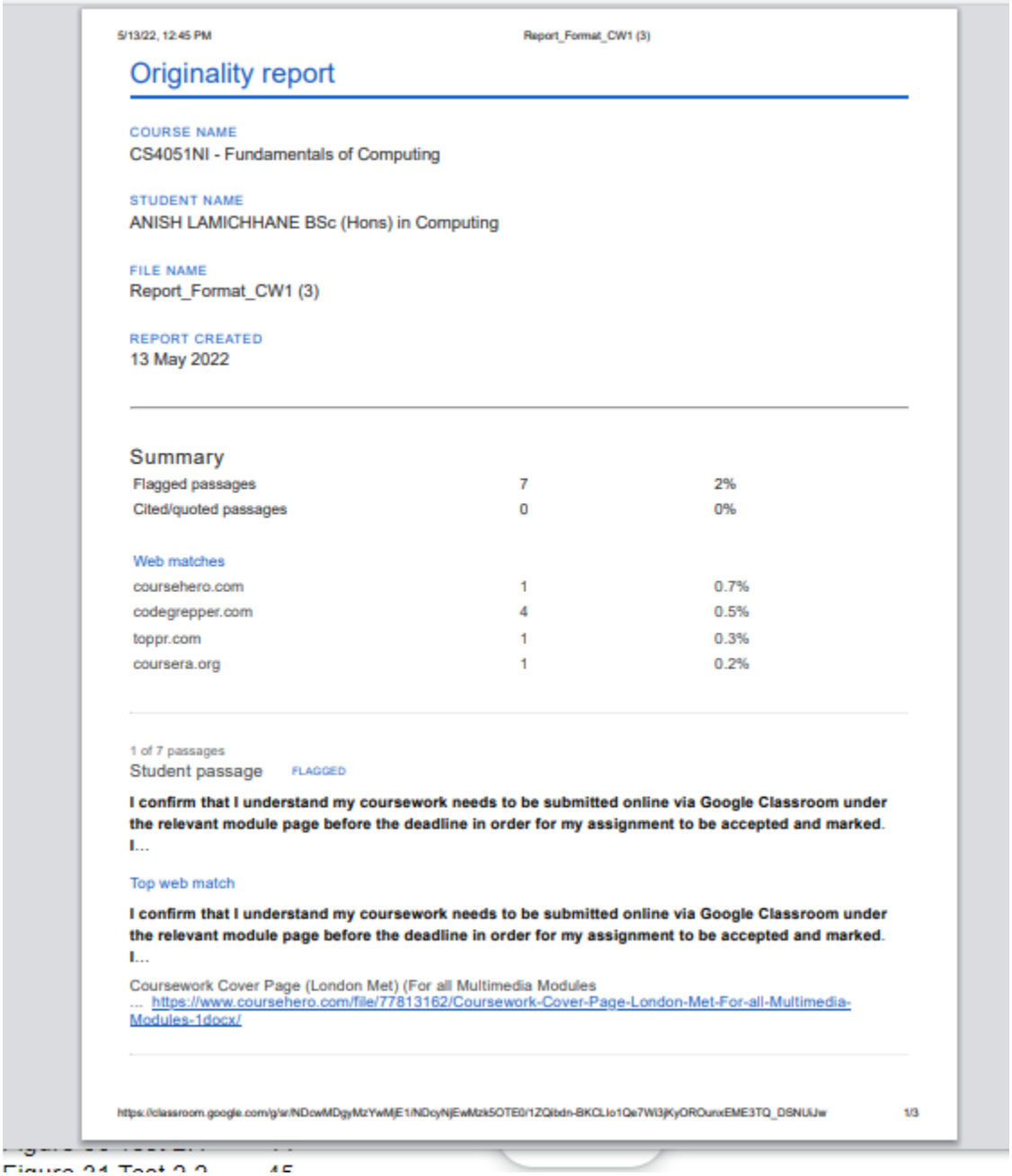
```
            validBikeId = int(input("Enter quantity of the bike to order: "))
            print("--------------------------------------------------")
            ShowBikes()
        return bike_quantity
        break
    except:
        print("-----------------------------------------------------------------")
        print("please,Enter a valid information")
        print("-----------------------------------------------------------------")


def total_bike_cost(entered_bike_id,customer_quantity):
    ''' total_cost  for orders of the bike .'''
    bike_list = return_2d_list()
    total_cost = int(bike_list[entered_bike_id-1][5].replace("$",""))* customer_quantity
    return total_cost


def loops():# using loop for the proper functioning of program
    inputdata = input("do you want to order another bike,y/n= ").upper()
    if(inputdata == "Y"):
        counter = False
    else:
        counter = True
    return counter



def update_stock_order(customer_quantity,entered_bike_id):
    bike_list = return_2d_list()
    ''' Updates the stocks and increase the  stock after ordering of bike.'''
    bike_list[entered_bike_id-1][4] = int(bike_list[entered_bike_id-1][4]) +
customer_quantity
    file = open("bikes.txt","w")
    for bike in bike_list:

file.write(str(bike[0])+","+str(bike[1])+","+str(bike[2])+","+str(bike[3])+","+str(bike[4])+","+s
tr(bike[5])+"\n")
    file.close()
    ShowBikes()

def new_bill(bike_list,entered_bike_id,order_bike,customer_quantity,total_cost):
    bike_list = return_2d_list()
    '''It is used for billing receipt of sells bike'''
    for bike in bike_list:
        if(int(bike[0]) == entered_bike_id):
```

```
            bike[4] = customer_quantity
            bike[5] = "$"+str(total_cost)
            order_bike.append(bike)
    return order_bike

def show_date():#defining show_date() for datetime
    import datetime
    year = str(datetime.datetime.now() .year)
    month = str(datetime.datetime.now().month)
    day = str(datetime.datetime.now().day)
    hour = str(datetime.datetime.now().hour)
    minute = str(datetime.datetime.now().minute)
    second = str(datetime.datetime.now().second)
    return year + month + day + hour + minute + second
show_date()

def print_bill_order (bike_list,order_bike):#Creating orders receipt in txt and shell.
    import datetime
    company_Name = input("Enter distributors's name: ")
    shiping_company_address = input("Enter shiping company address: ")
    contact_Number = input("Enter your contact number: ")
    file= open(company_Name +""+show_date()+".txt", "a")

file.write("==========================================================
=================================================\n")
    file.write("==================================0rder Bike
Receipt==========================================================\n")

file.write("==========================================================
=============================================\n")
    file.write("==================================Bike Management
System==============================================\n")

print("==============================================================
=============================================\n")
    print("==================================0rder Bike
Receipt==========================================================\n")

print("==============================================================
=============================================\n")
    print("==================================Bike Management
System========================================================= \n")
    file.write("shiping company  name: " + company_Name + "\n")
    file.write("shiping company address: " + shiping_company_address + "\n")
    file.write("contact Number " +contact_Number + "\n")
```

```
    file.write("the bike is order in this Date & Time : " + str(datetime.datetime.now()) +
"\n")
    print("shiping company  name: " + company_Name + "\n")
    print("shiping company address: " + shiping_company_address + "\n")
    print("contact Number " +contact_Number + "\n")
    print("the bike is order in this Date & Time : " + str(datetime.datetime.now()) + "\n")

file.write("==========================================================
=================================================\n")
    file.write("S.N\t Bike Name\t Company \t   Quantity\tprice\ttotal_amount\n")

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

print("==========================================================
=================================================\n")
    print("S.N\t Bike Name\t Company\t    Quantity\tprice\ttotal_amount")

print("==========================================================
=================================================\n")
    SNo = 0
    total_amount = 0
    for bike in order_bike:
        SNo = SNo + 1
        bike[5] = bike[5].replace("$","")
        total_amount = total_amount + int(bike[5])

file.write(str(SNo)+"\t"+(str(bike[1])+"\t"+str(bike[2])+"\t\t"+str(bike[4])+"\t"+str(bike[5])+"\t
"+str(total_amount)+"\t\t"))

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

print(str(SNo)+"\t"+(str(bike[1])+"\t"+str(bike[2])+"\t\t"+str(bike[4])+"\t"+str(bike[5])+"\t"+st
r(total_amount)+"\t\t"))

print("==========================================================
=================================================\n")
    file.close
```

7.Bibliography

luiz, m., 2000. Learning python. In: *Learning python.* s.l.:s.n.
w3schools, n.d. *w3Schools.* [Online]
Available at: www.w3Schools.com



5/13/22, 12:45 PM Report_Format_CW1 (3)

Originality report

COURSE NAME
CS4051NI - Fundamentals of Computing

STUDENT NAME
ANISH LAMICHHANE BSc (Hons) in Computing

FILE NAME
Report_Format_CW1 (3)

REPORT CREATED
13 May 2022

Summary

Flagged passages	7	2%
Cited/quoted passages	0	0%

Web matches

coursehero.com	1	0.7%
codegrepper.com	4	0.5%
toppr.com	1	0.3%
coursera.org	1	0.2%

1 of 7 passages
Student passage FLAGGED

I confirm that I understand my coursework needs to be submitted online via Google Classroom under the relevant module page before the deadline in order for my assignment to be accepted and marked. I...

Top web match

I confirm that I understand my coursework needs to be submitted online via Google Classroom under the relevant module page before the deadline in order for my assignment to be accepted and marked. I...

Coursework Cover Page (London Met) (For all Multimedia Modules
... https://www.coursehero.com/file/77813162/Coursework-Cover-Page-London-Met-For-all-Multimedia-Modules-1docx/

https://classroom.google.com/g/w/NDcwMDgyMzYwMjE1/NDcyNjEwMzk5OTE0/1ZQibdn-BKCLio1Qe7Wl3jKyOROunxEME3TQ_D8NUUw 1/3