Module Code & Module Title

CS4051NI Fundamentals of Computing

Assessment Weightage & Type

60% Individual Coursework

Year and Semester

2020-21 Autumn

Student Name: Rhythm Sapkota

Group: C9

London Met ID: 20049417

College ID: np01cp4s210220

Assignment Due Date: 10TH September 2021

Assignment Submission Date: 10TH September 2021

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

This is the coursework report designed to offer a framework or application that manages the executive's library framework. The flowcharts were used to construct the software, which followed a precise guide of computations and pseudocodes.

Since the tasks were not straightforward. We must be forceful in order to show our capacity to comprehend and address questions, as well as exhibit our ability to explain them as needed. Because of the anticipated difficulties in completing tasks within the allowed time and making them valuable to the primary audience, all efforts are put forth until the finalized appearance is completed.

This project is similar to an inventory system in that it allows users to borrow and return books from the library. The library is a location where customers may borrow and return books at their leisure. Customers can borrow any book from the library's inventory.

Users may be wondering why we need to develop the proposed system if it isn't required. Because today's libraries are getting along just okay with their old methods, such as keeping track of a user's borrowed and returned items on paperwork instead of just using a computer software. The libraries operate properly even without the support of computer software. It is true, but we must constantly keep an optimistic future.

This project's main feature is that it will help the user comprehend the Python IDLE (Integrated Development and Learning Environment) and use it to its full capacity. Even individuals with no prior programming experience will be able to utilize this application to its full extent and without mistakes. This application helps the programmer grasp the programming language by allowing them to interact with various functions and statements.

This work may benefit the academic world, as well as a diverse group of users who are primarily interested in reading, publishing, and comprehending Python programming, such as developers, researchers, programmers, or people from other subjects. To finish the academic requirements, you'll need to have a variety of devices and programs. All of

the coursework requirements are completed on the computer. Microsoft Word is used for documenting while the Python programming language is being used for code development.

2. ALGORITHM

2.1 Algorithm for the program

2.1.1 Algorithm for opening the Holmes library

step 1: Start
step 2: Enter a number between 1 and 4.
step 3: The input number 1 is used to display a list of books,
step 4: The input number 2 is used to borrow books.
step 5: The input number for returning borrowed books is 3.
step6: The exit number from the library management system is 4.

2.1.2 Stepwise algorithm for borrowing books

step 1: Start
step 2: The input number 2 is used to borrow books.
step 3: enter the borrower's first name;
step 4: enter the borrower's last name;
step 5: select the book's title.
step 6: Compose a borrower.txt file with the borrower's identity as well as the book that was borrowed.
step 7: If no, then error and again climb to step 2.
step 8: Asking whether the book is available or not. If yes, add additional details to the borrower.txt file. If there isn't a print version, "the book isn't available" then fall to step 4.
step 9: Would you want to borrow another book too? If users answered yes, go back to **step 4:** if users answered no, proceed to step 10.
step 10: Exit

2.1.3 Stepwise Algorithm for returning books

step 1: Start

step 2: to return the borrowed books, enter 3 as the input number.

step 3: enter the borrower's name

step 4: If the name is correct, read the borrower.txt file; if not, print "no similar name found in the borrower's list." and return to step 2.

step 5: Show the borrower.txt file for the given name.

step 6: Modify the library file and produce the returner.txt file

step 7: Has the book's returning deadline passed? If you answered yes, move to step 8. If not, proceed to step 10.

step 8: Penalty fee for late returns has been imposed.

Step 9: Enter the number of late days of returning the book.

step 10: Fine is 2 times the no of late days

step 11: Sum the penalty fine and the price of the books.

step 12: A text file with the overall cost is created.

step 13: End

3. Flowchart

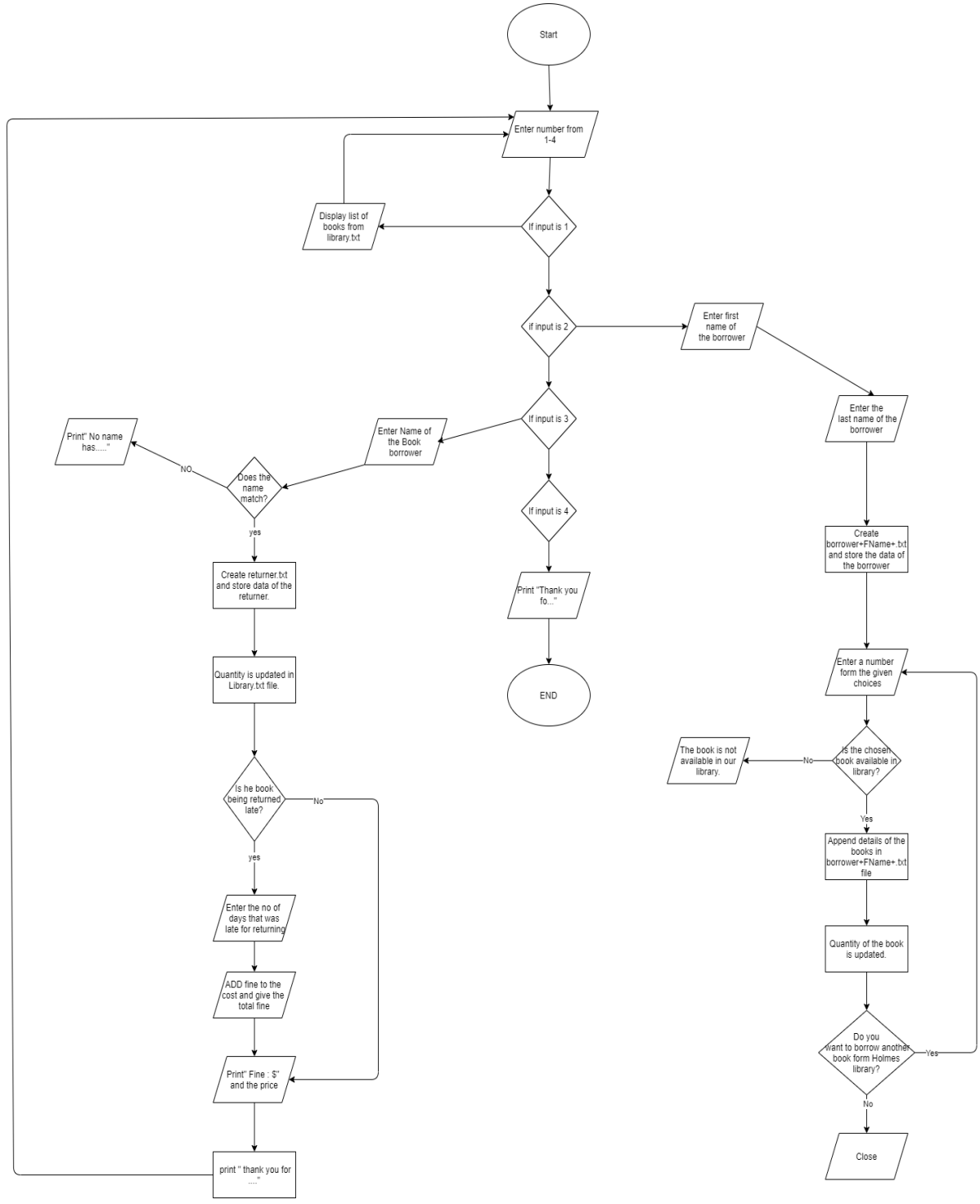

Figure 1: Flowchart of the program

4. PESUDOCODE

4.1 Pseudocode for main

```
IMPORT  module ReturnBook
IMPORT module BorrowBook
IMPORT module Lists
IMPORT module Date

CREATE function Library
ASSIGN boolean value True to Repeat
        WHILE Repeat is true
                PRINT "\n"
                PRINT "------------------------------------------------------------------------"
                PRINT "-----------------Welcome to the Holmes Library----------------------"
                PRINT "------------------------------------------------------------------"
                PRINT "Enter 1 to display the available books."
                PRINT "Enter 2 to borrow a Book from Holmes library."
                PRINT "Enter 3 to return the Book to Holmes library."
                PRINT" Enter 4 to exit the Holmes library"

                TRY
                        INPUT "Please, choose a number from 1-4 and enter the number in
                        the required field: "in int datatype and STOCK in variable id

                        IF id is 1
                                CALL list function from Lists module
                                PRINT "The available books are:"
                                OPEN library.txt in read mode

                                END FOR

                                PRINT read
                                CLOSE library.txt file
                        END IF

                        ELIF id is 2


                                CALL list function from Lists module
                                CALL Borrow function from BorrowBook module
                        END ELIF

                        ELIF a is 3
                                CALL list function from Lists module
                                CALL Return function from ReturnBook module
```

```
                    END ELIF

                    ELIF a is 4
                        PRINT "Thank you for using our service!!"
                        SET menu to False
                    END ELIF

                    ELSE
                        PRINT "Please enter a number from the choices."
                    END ELSE

                END TRY

                EXCEPT ValueError
                    PRINT "Invalid! Only integers are allowed to enter in this section."
                END EXCEPT

END Function Library
```

4.2 Pesudocode for Lists

```
CREATE empty list for Books, Author, Quantity, price
CREATE function lists
  OPEN library.txt file in read mode
  READ the lines in the file and
  REMOVE new lines in the and STORE in list LINE
  FOR i to length of LINE
    ASSIGN variable Y as 0
    FOR j in index "i" of LINE SPLIT by ","
      IF variable Y is 0
        APPEND j in list booksnames
      ELIF variable Y is 1
        APPEND j in list Author
      ELIF variable Y is 2
        APPEND j in list Quantity
      ELIF variable Y is 3
        APPEND j in list cost and REMOVE "$"
```

END IF

ASSIGN variable Y as Y+1

4.3 Pseudocode for date

CREATE function datetime

IMPORT module datetime

ASSIGN current date and time to variable dte

RETURN date from variable dte in string datatype

4.4 Pseudocode for borrow books

IMPORT module Date
IMPORT module Lists

CREATE function Borrow
ASSIGN boolean value False to isborrowed
WHILE boolean is True
INPUT “Enter the first name of the borrower” and **STORE** in variable FName
IF FName is alphabet
BREAK
PRINT “Please input alphabet from A-Z”
END WHILE

WHILE boolean is True
INPUT “Input last name of borrower” and **STORE** in variable LName
IF lastname is alphabet
BREAK
PRINT “Please input alphabet from A-Z”
END WHILE

CREATE variable store and **ASSIGN** “Borrower“ and value of FName and “.txt”
OPEN Borrower file in read and write mode
WRITE “Details of the borrower and books borrowed”
WRITE “Borrowed By: ” and value of FName and LName
WRITE “Date and Time: ” and value returned by function getDateTime from Date Module
WRITE “S.N. Book Name Author Name Price”
CLOSE Borrow file

CREATE variable total and **ASSIGN** 0 as float datatype

WHILE Borrowed is False
PRINT “Please select an option below:”

IF a is less than 0
PRINT “Please enter a valid number given above”
END IF

ELSE
TRY

IF index ‘a’ in Quantity from Lists module of int datatype is more than 0

PRINT “The book you’ve selected is in stock”
OPEN Borrow file in append mode
WRITE “1.” and **CALL** Books in index ‘i’ from Lists module and
CALL Author in index ‘i’ from Lists module and **CALL** Quantity in index ‘i’ from Lists module and “$” **CALL** Price in index ‘i’ from List module
CLOSE Borrow file
DECREASE the value of index ‘a’ in Quantity from Lists module by 1
ADD int value of index ‘a’ Price from Lists module to total

ASSIGN boolean value True to variable multi
ASSIGN variable cnt to 1

WHILE multi is TRUE

INPUT "If you want to borrow another book then press Y for Yes if not then press N for No:” in uppercase and **STORE** in question **IF** question is Y
PRINT “TO BORROW A BOOK, Please enter a number form below:”

FOR i of length 4
PRINT “Input” and ‘i’ and “to borrow book” and **CALL** Books in index ‘i’ from Lists module
END FOR
TRY

TAKE Input and **STORE** in variable as chck int datatype

ELSE
TRY

IF index ‘chck’ in Quantity from Lists module of int datatype is more than 0

PRINT “Fortunately,The book is available for borrowing.”
INCREASE cnt by 1
OPEN Borrower file in append mode
WRITE value of cnt in string datatype and **CALL** BookName in index ‘i’ from Lists module and **CALL** Author in index ‘i’ from Lists module and **CALL** Quantity in index ‘i’ from Lists module and “$” **CALL** Price in index ‘i’ from Lists module
CLOSE Borrower file
DECREASE the value of index ‘chck’ in Quantity from Lists module by 1
ADD int value of index ‘chck’ Price from Lists module to total

END TRY

EXCEPT IndexError
PRINT "Please choose the books according to the numbers given."
END EXCEPT

END TRY

EXCEPT ValueError
PRINT “Invalid!! Choose as it is suggested.”
END EXCEPT
END IF

ELSE
PRINT “Thank you for borrowing books from Holmes Library.\n\n”
SET multi to False
SET isborrowed to True
END ELIF

ELSE
PRINT “Please input as suggested”
END ELSE

ELSE
PRINT “Book is not available right now. Please visit us after some time.”
END ELSE
END TRY

```
        EXCEPT IndexError
    PRINT "P Please choose the books according to the numbers given."
        END EXCEPT

      END ELSE

   END TRY

     EXCEPT ValueError
    PRINT "Invalid!! Choose as it is suggested."
END EXCEPT
```

4.5 Pseudocode for returning the book

```
IMPORT module Lists
IMPORT module Date

CREATE function Return
  INPUT "Enter the name of the borrower who is returning the book:" and STORE in
variable Name
  CREATE variable 'txtfil' and ASSIGN "Borrower" and Name and ".txt"
  TRY
    OPEN 'txtfil' file in read mode
    READ 'txtfil' file and STORE in info
    PRINT info   END TRY
 EXCEPT
    PRINT "No similar name found in the borrwer list to return a book."
    CALL function Return
END EXCEPT

  CREATE variable store and ASSIGN "Returner" and Name and ".txt"
  OPEN store file in read and write mode
  WRITE "Details of the returner and books returned"
  WRITE "Returned by: " and value of Name
  WRITE "Date and Time of returning the book: " and value returned by function
   getDateTime from Date Module
  WRITE ""\t\tS.N.\t\t Name of the Book\t\tCost\n"
  CLOSE store file
  ASSIGN variable price as 0.0 float datatype

  FOR i of length 4
```

```
    IF index 'i' of Books of Lists is in data
      INCREASE Count by 1
      OPEN store file in append mode
      WRITE cnt in string datatype CALL Books in index 'i' from Lists module ","
       CALL Author in index 'i' from Lists module "," CALL Quantity in index 'i'
       from Lists module ", $" CALL Price in index 'i' from Lists module
      CLOSE  Return file
      INCREASE the value of index 'i' in Quantity from Lists module by 1
      ADD int value of index 'a' Price from Lists module to total
 END IF
END FOR

 ASSIGN boolean value False to Check
 ASSIGN boolean value False to Fine

 WHILE Check is False
   PRINT "The book return date expires in 10 days. Has the book return date already
    expired?
   PRINT "Press Y for Yes and N for No"
   TAKE input in uppercase and STORE in choice

   WHILE Fine is False
     IF choice is Y
       PRINT "By how many days was the book returned late?"
    TRY
         TAKE input as int datatype and STORE in days variable
         ASSIGN variable fine as 2*day
         OPEN store file
         WRITE "Fine: $" and value of fine in string datatype
         CLOSE store file
         ADD fine in total variable
         SET Check to True
         SET Fine to True
 END TRY

 OPEN library.txt in read and write mode

 FOR i of length 4
   WRITE CALL Books in index 'i' from Lists module "," CALL Author in index 'i' from
    Lists module "," CALL Quantity in index 'i' from Lists
    module ", $" CALL Price in index 'i' from Lists module
    END FOR

 CLOSE Library.txt
```

5. DATA STRUCTURE

To execute different operations in Python for input/output and data storage, this project makes significant use of the collection data type list. Integers, strings, Booleans, and floats are just a few of the data kinds that Python can handle. To store, modify, and execute different actions on the data, several data types and data structures were used when creating the software. This program makes use of the following data kinds and structures: Integer , float , String , Boolean and List.

The integer data type was used in the software to store all of the numerical values entered by users, as well as the outcomes of different actions. It's used to store the number of books in the variable quantity[index], for example.

the program uses the float data type to store decimal numbers. It is used to hold the entire cost price of the books in the variable total.

Likewise, text document with a series of one or more characters is stored using the string data type. Many operations, such as choice and stat, are done in string. In our.py code, the most of the output statements are concatenated and of the data type string. To record user decisions, our.py module uses string data types in various methods.

Boolean is also used to hold a single character (either true or false). It is kept in variables such as loop and success and is utilized throughout the program to test conditions.

In our.py file, list is also used to keep track of values for booksnames, author, quantity, and cost.Since the list is a variable collection data type, the values on the list may be added or eliminated as required by using append() and extend() functions, that have been effectively included within the multiple modules of our.py file.

6. Program

6.1 How the program executes:

This program contains five python files and one text file, with programs written in each. The main.py file contains the program's main code and is used to run it.

The Lists.py file is used to keep track of the name, author, quantity, and price of books in a list. The date and time for when the book is borrowed or returned are implemented using the date. py module. The library.txt file provides information on the book's title, author, quantity, and price. Lastly, the BorrowBook.py and ReturnBook.py files are used to write the codes for the borrowing and returning of books.

Whenever the main.py code is executed, the user is shown what to do. The software is stuck in an infinite loop and will not stop unless the user commands it to. The user is asked to enter a number between 1 and 4. The lists function is called from the Lists module when **number 1** is entered, and it uses the for loop to show the books accessible in the library. After showing the books, the loop resumes. If an incorrect number is entered, the program shows a specific message and the loop continues; similarly, if a string or float value is provided, the program prints a specific message and the loop continues. The software is designed to handle user's exceptions.

The borrow function from the borrow module, and also the list function from the lists module, are called when **number 2** is entered. The user is next prompted to provide their first and last names. After entering the user's name, the user can borrow a book by entering the book's number. If a number other than the one specified in the software is entered, a notice will appear. After the user enters the book number, the program scans to see whether the book is in inventory. A notice will be displayed if the book is not available. If the book is available, it will notify the user that it is available and asking if they want to borrow some other books. In every instance, Y and N can be entered. If Y is entered, the software will repeat the process of borrowing another book. The loop will continue until N is entered or the books are no longer available. The program produces a thank you message when N is entered.

The text file is named Borrower and the borrower's initial name and saved in the very same folder as the code files. The user's borrowed books are appended to the text file that was previously produced, as well as the total price. In the Library.txt file, the number of books borrowed has been reduced by one.

Again, When the **number 3** is entered, the return function from the return module, as well as the list function from the lists module, are called. After that, the user is needed to

provide the borrower's name. The user's name is compared to the text file generated throughout the borrowing procedure. If the name matches, the application will proceed; if it does not, a notice will be displayed. When the names are matched, the program opens the borrower text file, which contains the total cost produced during the borrowing procedure. After the user enters their name, a text file is produced, similar to the borrow procedure.

The program asks the user if such book has been returned late after reviewing the file. If it is being returned late, the user must enter Yes, and if it is not being returned late, the user must enter No. If the user enters Yes, the program will ask how many days late is the book returner. For the days late, the user must enter a number. If you return the book late, you will be penalized by the program. The fine is simply calculated and added to the final cost by the program.The software then produces a thank you message and reads the text file that has been generated. Then Following the display of the text file, the program returns to the first loop. If No is entered, the program displays the total price and reads the text file generated. In the library.txt file, the number of books borrowed has been reduced by one.

When the **number 4** is entered, the program comes to an end. Before the application terminates, a message is displayed saying thank you.

7. Testing

7.1 Test 1 – Test for showing try and except implementation

Test Number	**1**
Objective:	Showing implementation for ty and except
Action:	→ shdh2 is being entered in the section.
Expected Result:	The program should carry the user's incorrect input and display a message.
Actual Result:	The program carried wrong input of the user and a message was displayed.
Conclusion:	The test is successful.

Table 1: Test table for showing try and expect implementation.

Output Result:

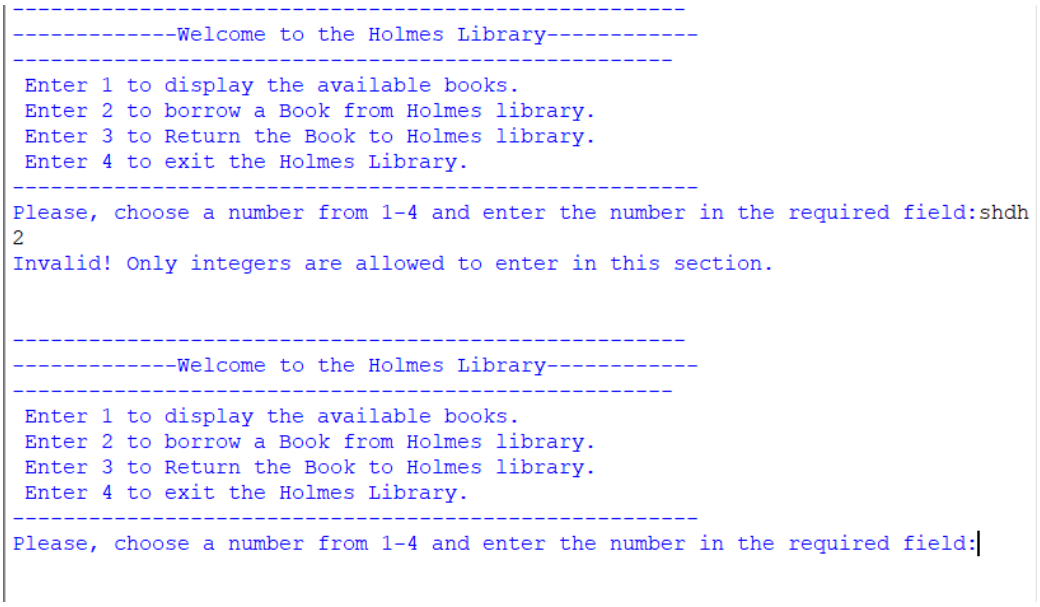

Figure 2: A message is shown saying only integers are allowed.

7.2 Test no 2 Error message while wrong input in selection and return option.

Test Number	**2**
Objective:	borrow and return option selection
Action:	→ 2 is inputted in the input section to go borrow a book. → First name and last name are entered. → 10 value is being input. → sjhda2 value is being entered.
Expected Result:	The program should carry the user's incorrect input and display a message.
Actual Result:	The program carried wrong input of the user and a message was displayed.
Conclusion:	The test is successful.

Table 2: Test table for testing borrow and return option selection.

Output Result:

```
------------------------------------------------------
------------Welcome to the Holmes Library-------------
------------------------------------------------------
 Enter 1 to display the available books.
 Enter 2 to borrow a Book from Holmes library.
 Enter 3 to Return the Book to Holmes library.
 Enter 4 to exit the Holmes Library.
------------------------------------------------------
Please, choose a number from 1-4 and enter the number in the required field:2
Enter the first name of the borrower:Rhythm
Enter the last name of the borrower:Sapkota


TO BORROW A BOOK, Please enter a number form below:
To borrow A journey to the centre of the Earth Enter the number 0
To borrow Romeo and Juleit Enter the number 1
To borrow A Story of Moonlight Enter the number 2
To borrow Pyramids: The history Enter the number 3
Enter the number:10
Please choose the books according to the numbers given.
```

Figure 3 : Program showing message to choose according to the given numbers.

```
Please choose the books according to the numbers given.
TO BORROW A BOOK, Please enter a number form below:
To borrow A journey to the centre of the Earth Enter the number 0
To borrow Romeo and Juleit Enter the number 1
To borrow A Story of Moonlight Enter the number 2
To borrow Pyramids: The history Enter the number 3
Enter the number:sjhda2
Invalid!! Choose as it is suggested.
```

Ln: 51 Col: 17

Figure 4: Program showing invalid message for other datatype input.

7.3 Test no 3: Creating borrowers file

Test Number	3
Objective:	Creating Borrowers file
Action:	→ To borrow a book no 2 is entered. → First name and last name are being entered. → From the given list of books, no 2 is chosen. → To borrow another book, Y is entered. → Again, from the given list of books, no 2 is chosen. → To stop borrowing more books, N is entered. → The borrower+FName+.txt file created is opened.
Expected Result:	The program should be able to let users borrow the available books from the library.
Actual Result:	The Books were borrowed by using the program.
Conclusion:	The test is successful.

Table 3 :Test table for creating borrowers file

Output Result:

```
TO BORROW A BOOK, Please enter a number form below:
To borrow A journey to the centre of the Earth Enter the number 0
To borrow Romeo and Juleit Enter the number 1
To borrow A Story of Moonlight Enter the number 2
To borrow Pyramids: The history Enter the number 3
Enter the number:2
Fortunately,The book is available for borrowing.
Do you want to borrow another book? Type Y for yes and N for no Y
TO BORROW A BOOK, Please enter a number form below:
To borrow A journey to the centre of the Earth Enter the number 0
To borrow Romeo and Juleit Enter the number 1
To borrow A Story of Moonlight Enter the number 2
To borrow Pyramids: The history Enter the number 3
Enter the number:1
Fortunately,The book is available for borrowing.
Do you want to borrow another book? Type Y for yes and N for no N
Thank you for borrowing books from Holmes Library.
```

Figure 5: Program generating borrowers file.

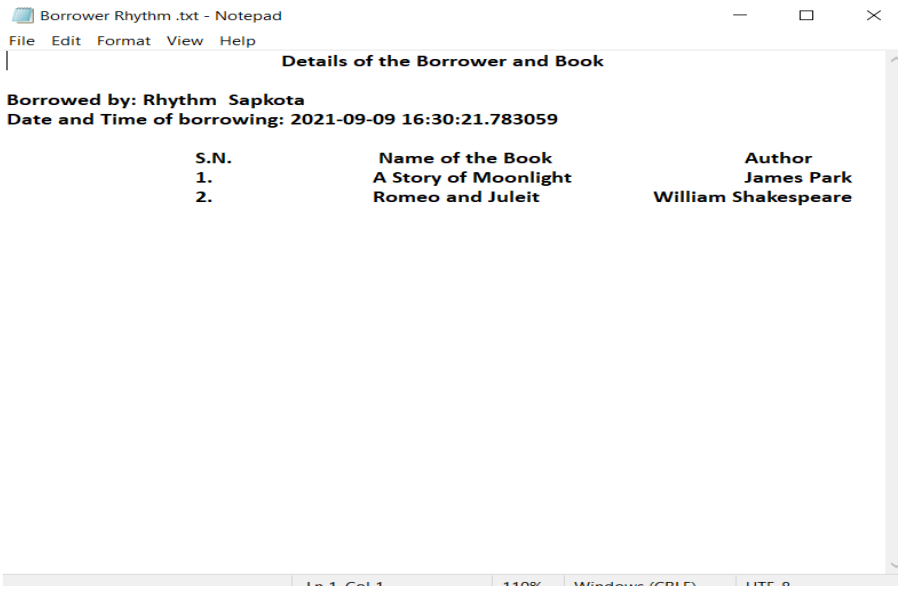

Figure 6: Generated borrower.txt file by the program.

7.4 Test no 4- Creating returners file

Test Number	**4**
Objective:	Creating returners file.
Action:	→ The number 3 is entered in the input section to return a book to Holmes library. → The borrower's First Name is being entered. → As books are returned late, Yes is entered. →The book is returned 8 days late, so 8 is entered. → The returner.txt file generated is opened.
Expected Result:	The program should allow the user to return the books that were borrowed.
Actual Result:	The program allowed borrowed books to be returned.
Conclusion:	The test is successful.

Table 4: Test table for creating returners file.

Output Result:

```
--------------------------------------------------
------------Welcome to the Holmes Library----------
--------------------------------------------------
 Enter 1 to display the available books.
 Enter 2 to borrow a Book from Holmes library.
 Enter 3 to Return the Book to Holmes library.
 Enter 4 to exit the Holmes Library.
--------------------------------------------------
Please, choose a number from 1-4 and enter the number in the required field:3
Enter the name of the borrower who is returning the book:Rhythm
                      Details of the Borrower and Book

Borrowed by: Rhythm  Sapkota
Date and Time of borrowing: 2021-09-09 16:30:21.783059

                S.N.            Name of the Book               Author
                1.             A Story of Moonlight            James Park
                2.             Romeo and Juleit                William Shakes
peare

                Total Price= $9.0


Is the book is returned back late than permitted time? Type YES for yes and NO
for no: Yes
How many days late is the returner? 8
You have returned the book late. Therefore, you have to pay a fine.
Total after fine: $25.0
```

Figure 7: Program creating a returner file of the borrower and showing total price after fine.

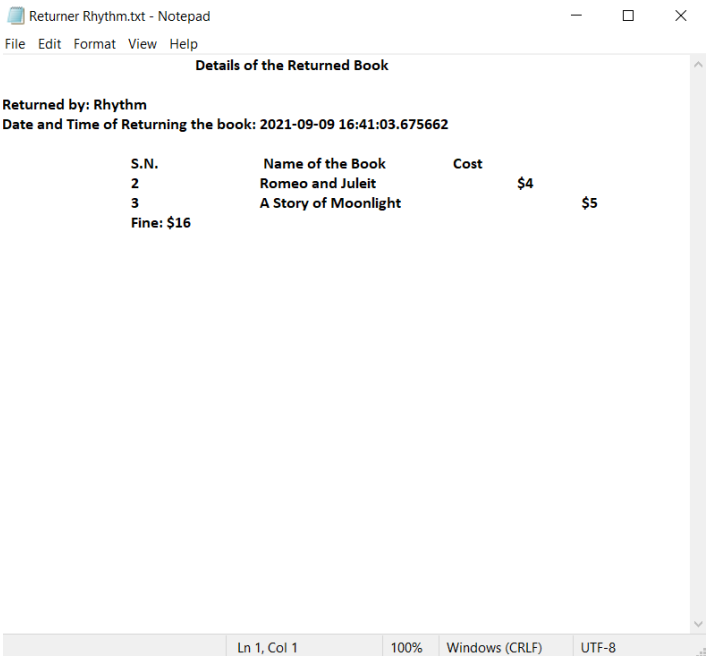

Figure 8: Created text file of the returner containing the final cost as fine.

7.5 Test 5- Quantity being updated in library.txt file

Test Number	5
Objective:	Quantity being updated in library.txt file
Action:	→ The library.txt file containing the books stored is opened before borrowing the books. → Again, after borrowing the books, library.txt file is being opened. → Library.txt file is again opened after returning the borrowed books.
Expected Result:	After borrowing and returning books, the quantity and volumes of the books should be updated.
Actual Result:	After borrowing and returning the books, the quantity of the remaining available books were changed and updated.
Conclusion:	The test is successful.

Table 5: Test table to check if the quantity is being updated or not in library.txt file.

Output Result:

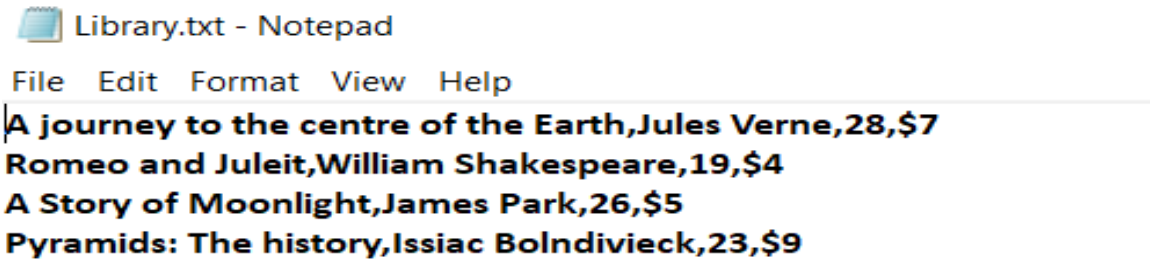

Figure 9: Library.txt before borrowing a book.

Figure 10: Library.txt after borrowing the book.

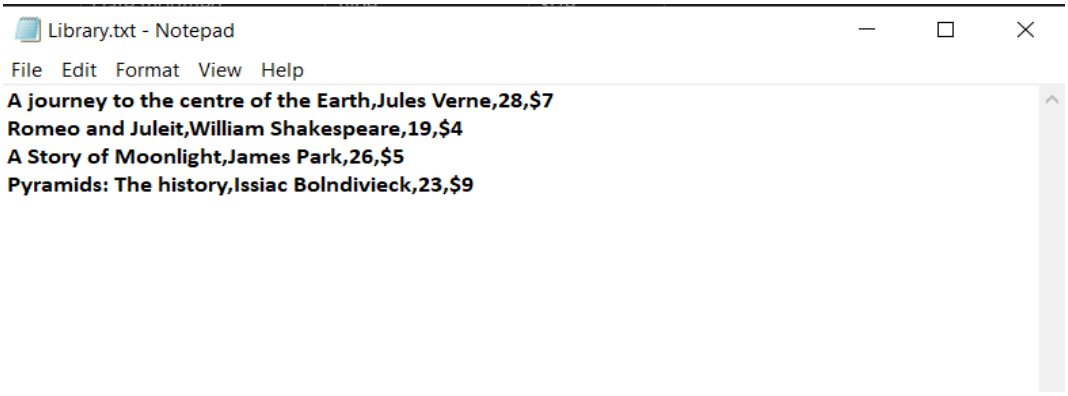

Figure 11: Library.txt file after returning the book.

8. Conclusion

I had to conduct some research before I could begin working on this project. Information can be found in publications, reference materials, and on the Web. They may be able to give knowledge on how others have resolved or partially solved comparable difficulties, allowing them to solve or partly fix the problems. To accomplish this project, we have used a wide range of materials.

The goal of this project was to create an easy-to-use digital library management system. Learning Python data structures was the most crucial element of building this app. Because the data would need to be retrieved several times, I thought that utilizing lists as the project's foundation would be a suitable solution. We were required to know looping over lists, utilizing if-elif statements, and a number of other aspects of the Programming in addition to data structures.

As this coursework was a bit hard and complex, I took reference from various sites and journals and also took some help from brother and my friends. I used draw.io for building flowchart and MS Word for completing my report. Other different sites like w3schools, Dataflair, Edureka, etc, helped me in my coding as well as report. Also the journal by D.Necaise was really useful for my algorithm.

I concentrated on refining my time management skills in addition to refining the programming skills. Multitasking numerous jobs from different modules proved to be very useful to me. This project also helped me prioritize activities according to their importance. Not just did I obtain knowledge of the Python programming language, but also i developed a lot of skills which will come in handy in the upcoming future. I have found this coursework to be really very useful in general. After working on numerous projects, I'm more comfortable in Python abilities and knowledge that i acquired while doing this coursework.

So, in this way I completed my coursework of python programming language which was really useful for enhancing and boosting my self-confidence for further coding in python. This coursework was hard and complex but at the same time it was challenging and really fun to work with.

Bibliography

D. Necaise, R., 2021. *Data Structures and Algorithms Using Python*. [online] Home.ustc.edu.cn. Available at: <http://home.ustc.edu.cn/~huang83/ds/Data%20Structures%20and%20Algorithms%20Using%20Python.pdf> [Accessed 29 August 2021].

W3schools.com. 2021. *Python Built-in Functions*. [online] Available at: <https://www.w3schools.com/python/python_ref_functions.asp> [Accessed 3 September 2021].

Data flair. 2021. *Python Programs*. [online] Available at: <https://data-flair.training/blogs/library-management-system-python-project/> [Accessed 5 September 2021].

Programiz.com. 2021. *Learn Python Programming*. [online] Available at: <https://www.programiz.com/python-programming> [Accessed 2 September 2021].

Edureka. 2021. *Data Structures in Python | List, Tuple, Dict, Sets, Stack, Queue | Edureka*. [online] Available at: <https://www.edureka.co/blog/data-structures-in-python/#:~:text=on%20them%20accordingly.-,Types%20of%20Data%20Structures%20in%20Python,full%20control%20over%20their%20functionality.> [Accessed 26 August 2021].

App.diagrams.net. 2021. *Flowchart Maker & Online Diagram Software*. [online] Available at: <https://app.diagrams.net/> [Accessed 7 September 2021].

APPENDIX

Main.py

```
import ReturnBook
import BorrowBook
import Lists
import Date

def main():
    Repeat=True
    while(Repeat==True):
        print("\n")
        print ("-----------------------------------------------------")
        print("-------------Welcome to the Holmes Library------------")
        print ("-----------------------------------------------------")
        print(" Enter 1 to display the available books.")
        print(" Enter 2 to borrow a Book from Holmes library.")
        print(" Enter 3 to Return the Book to Holmes library.")
        print(" Enter 4 to exit the Holmes Library.")
        print("-----------------------------------------------------")
        try:
            id=int(input("Please, choose a number from 1-4 and enter the number in the required
field:"))
            if(id==1):
                G=open("Library.txt","r")
                read=G.read()
                print(read)
                G.close()
            elif(id==2):
                Lists.lists()
                BorrowBook.borrow()
            elif(id==3):

                Lists.lists()
                ReturnBook.Return()
            elif(id==4):
                print("Thank you for using our service!!")
                break
            else:
                print("Please enter a number from the choices.")
        except ValueError:
            print("Invalid! Only integers are allowed to enter in this section.")
main()
```

Lists.py

```
Books=[]
Author=[]
Quantity=[]
Price=[]
def lists():
    G=open("Library.txt","r")
    LINE=G.readlines()
    LINE=[a.strip("\n") for a in LINE]
    for i in range(len(LINE)):
        Y=0
        for j in LINE[i].split(","):
            if(Y==0):
                Books.append(j)
            elif(Y==1):
                Author.append(j)
            elif(Y==2):
                Quantity.append(j)
            elif(Y==3):
                Price.append(j.strip("$"))
            Y=Y+1
```

BorrowBook.py

```
import Date
import Lists

def borrow():
    isborrowed=False
    #Using while loop
    while(True):
        FName=input("Enter the first name of the borrower:")
        LName=input("Enter the last name of the borrower:")
        print("\n")
        break
    store="Borrower "+FName+".txt"
    G=open(store,"w+")
    G.write("\t\t\tDetails of the Borrower and Book \n\n")
    G.write("Borrowed by: "+FName+ " "+LName+ "\n")
    G.write("Date and Time of borrowing: "+Date.datetime()+"\n\n")
    G.write("\t\t S.N. \t\t Name of the Book \t\t Author\n")
    G.close()
    while isborrowed==False:
        print("TO BORROW A BOOK, Please enter a number form below: ")
        for i in range(len(Lists.Books)):
            print("To borrow", Lists.Books[i],"Enter the number",i)
        try:
            number=int(input("Enter the number:"))
```

```
        try:
            if(int(Lists.Quantity[number])>0):
                print("Fortunately,The book is available for borrowing.")
                G=open(store,"a")
                G.write("\t\t 1. \t\t"+Lists.Books[number]+"\t\t "+Lists.Author[number]+"\n")
                G.close()
                Lists.Quantity[number]=int(Lists.Quantity[number])-1
                G=open("Library.txt","w+")
                for i in range(4):

G.write(Lists.Books[i]+","+Lists.Author[i]+","+str(Lists.Quantity[i])+","+"$"+Lists.Price[i]+"\n")
                G.close()

                multi=True
                cnt=1
                while multi==True:
                    question=input("Do you want to borrow another book? Type Y for yes and N for
no ").upper()
                    if(question=="Y"):
                        cnt=cnt+1
                        print("TO BORROW A BOOK, Please enter a number form below: ")
                        for i in range(len(Lists.Books)):
                            print("To borrow", Lists.Books[i],"Enter the number",i)
                        try:
                            chck=int(input("Enter the number:"))
                            try:
                                if(int(Lists.Quantity[chck])>0):
                                    print("Fortunately,The book is available for borrowing.")
                                    G=open(store,"a")
                                    G.write("\t\t "+str(cnt)+"."+"\t\t"+Lists.Books[chck]+"\t\t
"+Lists.Author[chck]+"\n")
                                    G.close()
                                    Lists.Quantity[chck]=int(Lists.Quantity[chck])-1
                                    G=open("Library.txt","w+")
                                    for i in range(4):

G.write(Lists.Books[i]+","+Lists.Author[i]+","+str(Lists.Quantity[i])+","+"$"+Lists.Price[i]+"\n")
                                    G.close()
                                else:
                                    multi=False
                                    break
                            except IndexError:
                                print("Please choose the books according to the numbers given.")
                        except ValueError:
                            print("Invalid!! Choose as it is suggested.")
                    else:
                        print("Thank you for borrowing books from Holmes Library.\n\n")
                        multi=False
                        isborrowed=True
        except IndexError:
            print("Please choose the books according to the numbers given.")
```

```
        except ValueError:
            print("Invalid!! Choose as it is suggested.")
```

ReturnBook.py

```
import Lists
import Date

def Return():

    Name=input("Enter the name of the borrower who is returning the book:")
    txtfil="Borrower "+Name+".txt"
    try:
        G=open(txtfil,"r")
        k=G.readlines()
        k=[txtfil.strip("$") for txtfil in k]#Removing the $ using strip keyword
        G.close()
        G=open(txtfil,"r")
        info=G.read()
        print(info)
    except:
        print("No similar name found in the borrwer list to return a book.")
        Return()
    store="Returner "+Name+".txt"
    G=open(store,"w+")
    G.write("\t\t\tDetails of the Returned Book \n\n")
    G.write("Returned by: "+Name+"\n")
    G.write("Date and Time of Returning the book: "+Date.datetime()+"\n\n")
    G.write("\t\tS.N.\t\t Name of the Book\t\tCost\n")
    G.close()

    price=0.0
    for i in range(4):
        if Lists.Books[i] in info:
            G=open(store,"a")
            G.write("\t\t"+str(i+1)+"\t\t"+Lists.Books[i]+"\t\t\t$"+Lists.Price[i]+"\n")
            G.close()
            Lists.Quantity[i]=int(Lists.Quantity[i])+1
            price=price+float(Lists.Price[i])


    print("\t\t Total Price= "+"$"+str(price)+"\n\n")
    chck=input("Is the book is returned back late than permitted time? Type YES for yes and NO
for no: ").upper()
    if chck=="YES":
        days=int(input("How many days late is the returner?"))
        if days>=1:
            print("You have returned the book late. Therefore, you have to pay a fine.")
```

```
            fine=2*days
            G=open(store,"a")
            G.write("\t\tFine: $"+str(fine)+"\n")
            price=price+fine
        print("Total after fine: $"+str(price))
    G=open("Library.txt","w+")
    for i in range(4):
        G.write(Lists.Books[i]+","+Lists.Author[i]+","+str(Lists.Quantity[i])+","+"$"+Lists.Price[i]+"\n")
    G.close()
```

Date.py

```
def datetime():
    import datetime
    dte=datetime.datetime.now()
    return str(dte)
```

Library.txt

```
A journey to the centre of the Earth,Jules Verne,28,$7
Romeo and Juleit,William Shakespeare,19,$4
A Story of Moonlight,James Park,26,$5
Pyramids: The history,Issiac Bolndivieck,23,$9
```

END