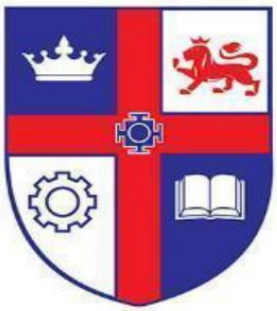
LONDON METROPOLITAN UNIVERSITY



islington college
(इस्लिङ्टन कलेज)

Module Code & Module Title

CS4001NI Programming

COURSEWORK -2

Assessment Weightage & Type

30% Individual Coursework

Semester and Year

Spring 2021

Student Name: Manish Hajur Khadka

Group: C10

London Met ID: 20048921

College ID: NP01CP4S210263

Assignment Due Date: 20th August, 2021

Assignment Submission Date: 20th August, 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 submission will be treated as non-submission and a mark of zero will be awarded.

Table of Contents

Tables of Figures

List of Tables

Introduction

Nowadays, computers and a multitude of programs are used to execute every task. This year's project is a Library Management System for Libraries, which allows a user or a customer to borrow and return books using the following project.

The Python Programming Language is used to write the program in IDLE. “IDLE is a simple Python-based integrated development environment (IDE).” It's a program that lets you type in and run your own programs.” (Heinold, 2012; Heinold, 2012). Python is a programming language, similar to Java and C++. Python, on the other hand, has a far more straightforward syntax than the other languages mentioned. As previously said, IDLE provides a simple development environment for writing Python programs. Similarly, draw.io was used to create the flowchart.

The following project was likewise made with the help of IDLE. The Project is a straightforward piece of software that allows individuals to choose whether to borrow or return a book. The application includes a lot of features that help users complete various tasks.

The Project’s goals and Objectives:

The project's major purpose is to develop a program that makes it simple for a user to borrow and return books, as well as to offer information about the books borrowed and returned. In addition, the following objectives are listed:

• Keep track of the cost of books and the number of copies available.

• When books are borrowed or returned, update the quantity.

• Adding primary keys to the books so that they may be found differently.

Algorithm of Project

An algorithm is a method for solving calculating problems in a predetermined number of steps. The program's algorithm is as follows:

• Step 1: Start

• Step 2: Enter d to display books, b to borrow books, r to return books, or q to exit the application

• Step 3: If the user enters "d":

Step 3.1: From the stock text file, the available books are shown.

• Step 4: If the user enters the letter "b":

Step 4.1: The book is displayed, together with its unique codes.

Step 4.2: The user will be prompted to input the code for the book they wish to borrow.

Step 4.3: The code entered by the user is compared to the code found in the books. If the code being compared is the same:

Step 4.3.1: The user is prompted to provide their name, book title, date, time, and pricing, which are kept in the variables fname, bookname, issuedate, issue time, and tot, respectively.

Step 4.3.2: A note is created that contains all of the information saved in the variables indicated previously.

Step 4.3.3: In the Stock file, the Quantity of the book being borrowed is modified.

Step 4.3.4: A message appears, indicating that the book has been borrowed.

Step 4.4: An error message appears if the code is not equal.

• Step 5: If the user types "r"

Step 5.1: The book is displayed, together with its unique codes.

Step 5.2: The user is prompted to input the code associated with the book they wish to return.

Step 5.3: The user is prompted to enter the number of days the book has been checked out for.

Step 5.4: If the two codes are identical:

Step 5.4.1: The user is prompted to provide their name, book title, date, time, and pricing, which are kept in the variables fname, bookname, issuedate, issuetime, and tot, respectively.

Step 5.4.2: If the book has not been returned after 10 days. The entire fee is increased by a specified amount of fine. Step 5.4.3: A note is formed that contains all of the information saved in the variables stated previously.

Step 5.4.4: In the Stock file, the Quantity of the book being borrowed is updated.

Step 5.4.5: A message appears, indicating that the book has been borrowed.

Step 5.5: An error message appears if the code is not equal.

Step 6: If the user types the letter "q,"

Step 6.1: The program displays a Thank You message and then closes.

Pseudocode

```
Class Library main


import loadingList
import functions
import noteGenerator
import listManipulation
list2 = loadingList.load('Books.txt','r')
INITIALIZE cart = []
INITIALIZE returned = []
DECLARE daysBorrowed = 0
DECLARE z = 1
while z>0
    CALL functions.menu()
    choice = input("Enter your choice: ")
    TRY
        IF choice == "D":
            DECLARE file = open("Books.txt","r")
            DECLARE text = file.read()
             print(text)
        ELIF choice == "L":


            CALL Id = functions.menu2(list2)
```

```
        userName = input("Enter your full name.: ")
        FOR i in list2:
            IF Id == i[0]:
                IF int(i[3])>0:
                    cart.append(i)
                  DECLARE  stock = int(i[3])
                    stock = stock - 1
                    i[3] = str(stock)
                ELSE;
                    print("\n----------Sorry the item is out of stock.----------\n")


    ELIF choice == "R":
    CALL userName = input("Enter you name: ")
        Id = functions.menu3(list2)
        daysBorrowed =  int(input("\nEnter the number of days you have had the book for: "))
        FOR i in list2:
            IF Id == i[0]:


                returned.append(i)
             DECLARE   stock = int(i[3])
             DECLARE   stock = stock + 1
                i[3] = str(stock)



ELIF choice == "Q":
        PRINT ("----------Thanks for visiting our Libary Keep coming----------")
```

```
            END ELIF
        ELSE
            PRINT ("\n----------------------Enter a valid Choice. --------------------\n")
    except Exception as e:
        PRINT (e)
#Creating a borrow note from noteGenerator module.
noteGenerator.returnNote(returned, daysBorrowed, userName)
RETURN noteGenerator.borrowNote(cart, userName)
APPEND listManipulation.appendList(list2)
```

PSEUDOCODE FOR FUNCTIONS

```
#Displays a menu which makes it easy for the user to carry out a function.
DEFmenu():
    PRINT ("Press D to display the book")
    PRINT ("Press L to lend the book")
    PRINT ("Press R to Return the book")
    PRINT ("Press Q to Quit the Program")

DEF menu1(list):
    PRINT ("\n\nBook Name\t\t\tPrice")
    FOR i in list:
       PRINT (i[1]+"\t\t\t"+i[4])

DEF menu2(list):
    PRINT ("\nBook ID\t\tBook Name\t\t\tPrice")
   FOR i in list:
        PRINT (i[0]+"\t\t"+i[1]+"\t\t\t"+i[4])
    bookId = input("\nEnter the book id of the book you choose to borrow: ")
    return bookId
```

```
DEF menu3(list):
    PRINT ("\nBook ID\t\tBook Name\t\t\tPrice")
    for I in list:
        PRINT (i[0]+"\t\t"+i[1]+"\t\t\t"+i[4])
    bookId = input("\nEnter the book id of the book you choose to return: ")
    return bookId
```

PSEUDOCORE FOR LIST MANIPULATION

```
DEF appendList(list):
   DECLARE file = open('Books.txt','w')
   FOR i in list:
        FOR j in range(len(i)):
            file.write(i[j])
           IF j < 4:
                file.write(",")
        file.write('\n')
```

PSEUDOCORE FOR LOADINGLIST

```
DEF load(filename, mode):
   DECLARE file1= open(filename, mode)
   DECLARE contents= file1.read()
    file1.close()
DECLARE    list1= contents.split('\n')
   INITIALIZE list2=[]
    FOR each_item in list1:
APPEND list2.append(each_item.split(","))
        list2.pop()
    RETURN list2
```

PSEUDOCORE FOR NOTE GENERATOR

```
import datetime
DEF borrowNote(list, name):
```

```
DECLARE    date_time = datetime.datetime.now()
 DECLARE   B_file = open('borrow_notice.txt','w')
   B_file.write("+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+\n")
   B_file.write(name+"\n")
   B_file.write("-----------------NOTICE FOR BORROWING THE BOOK: \n\n")

 DECLARE   grand_Total = 0

   FOR i in list:
    DECLARE   price = i[4].replace("$","")
     DECLARE price = float(price)
   DECARE    B_file.write("Name of the book is: "+ i[1])
      B_file.write("Price of the book is: "+ i[4])

     DECLARE  grand_Total = grand_Total + price
   B_file.write("---------------------------------------\n")
   B_file.write("Grand Total:-\t\t\t$"+str(grand_Total)+"\n\n")
   B_file.write("+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+\n")
   B_file.write("The date you issued book is : " + str(date_time))
   B_file.close()


DEF returnNote(list, days, name):
 DECLARE   date_time = datetime.datetime.now()
 DECLARE   R_file = open('return_notice.txt','w')
  DECLARE  total = 0
   R_file.write("+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+\n")
   R_file.write(name+"\n")
   R_file.write("--------------Notice For Returing the book----------------------------- \n\n")
```

```
FOR i in list:
  DECLARE  fine = 0
  DECLARE  price = i[4].replace("$","")
   DECLARE price = float(price)
  DECLARE  fineDays = 0
    IF days > 10:
       DECLARE fineDays = days - 10
      DECLARE fine = (1 / 0.6) * price * fineDays
    PRINT ("You are late in returing the book fine would up added")
    R_file.write("name of the book is: "+i[1])
    R_file.write("The price of the book is: "+i[4])
    R_file.write("The fine for this book is: "+str(fine)+"\n")

    DECLARE total = total + price + fine
R_file.write("----------------------------------------------------------------\n")
R_file.write("Grand Total:-\t\t\t$"+str(total)+"\n\n")
R_file.write("+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+\n")
R_file.write("The date you returned book is : " + str(date_time))
R_file.close()
```

Flowchart

Fig.1 Flowchart

Data Structures:

The method of organizing or managing data in a software so that it can be used efficiently is referred to as data structure. The following are some of the data structures utilized in the program:

• Int:

The int data structure is made up of entire numbers that are either positive or negative. The int class is used to represent it.

• String: Unicode characters are represented in the string data structure as arrays of bytes. A string is a collection of data wrapped in single or double quotation marks and including one or more characters. str is the symbol for it.

• Lists:

A list is a collection of elements enclosed in square brackets. The elements contained within the list can be accessed using their index, which starts at 0. For instance, if a = [1,2,3,4,5,6] is a list, we can access the variables by using indexes such as a[0], which returns 1. Likewise, a[5] yields the element 6.
They are changeable, which means that the list's elements can be altered.

• Dictionary:

In Python, a dictionary is an unordered collection of lists that are used as key-value pairs. Every value is associated with a unique key that cannot be modified. The keys in the dictionary are not mutable, but the values are. Curly brackets are used to indicate dictionaries rather than lists. The keys in the dictionary cannot be duplicated, but the values in the dictionary can.

• Float:

In Python, a float is a basic data structure. Float, unlike int, is used to store real numbers, such as decimal values or fractional numbers. For instance, f=float (2.5).
As previously stated, int can only represent whole numbers, so we use float to represent real values.

• Sets:

A set is an unsorted collection of elements that always includes unique elements.

Curly brackets, similar to those used in dictionaries, are used to express sets. Sets, on the other hand, can only store unique components and do not tolerate duplicates. Sets are also mutable, allowing us to add and remove elements.

• Tuples: Tuples are an ordered series of elements with attributes similar to lists, except that the element contained within the tuples cannot be modified. As a result, the items contained within Tuples are unchangeable. Small brackets are used to signify multiples (). Append, extend, and other operations are not supported. As a result, the elements within the tuples are stored indefinitely. For example: {1,2,3,4,5}

Program

When the application is run for the first time, it displays the program's main interface, which provides the user with four options. The program then asks the user to supply input so that the function can be executed according to the choice picked by the user.

The application then asks the user to choose one of the following options:

"Press d to see what books are available,"

"Press b to borrow books,"

"Press r to return borrowed books,"

"Press q to exit the program."

```
Press D to display the book
Press L to lend the book
Press R to Return the book
Press Q to Quit the Program
Enter your choice:
```

Fig.2 Main UI of the program

If the user types in "b," which stands for "borrow book," a list of books with their unique codes appears. The codes are constructed using dictionaries and are used to uniquely identify the books. The user is then prompted to enter the code of the book they wish to borrow.

```
Enter your choice: D
B001, Harry Potter, JK Rowling,16, $2
B002, Start With Why, Simon Sinek,0 , $1.5
B003, Programming with Python, John Smith,14, $1.5
```

Fig.3 Displaying the books along with their codes

If the book code is legitimate and assigned to a book, the computer will prompt the user for their name in order to issue a borrow notice/invoice.

```
Book ID         Book Name                       Price
B001             Harry Potter                    $2
B002             Start With Why                  $1.5
B003             Programming with Python                          $1.5

Enter the book id of the book you choose to borrow: B001
Enter your full name.: Manish Hajur Khadka
```

Fig.4 Successfully borrowing books

After borrow function closes, a text file is created as an invoice displaying the details of the book borrowed.

```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
Manish Hajur Khadka
-----------------NOTICE FOR BORROWING THE BOOK:

Name of the book is:  Harry PotterPrice of the book is:  $2-----------
Grand Total:-                    $2.0

+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
The date you issued book is : 2021-09-10 08:32:55.508198
```

Fig.5 Note generated after borrowing the books

```
B001, Harry Potter, JK Rowling,16, $2
B002, Start With Why, Simon Sinek,0 , $1.5
B003, Programming with Python, John Smith,14, $1.5
```

Fig.6 Update on Stock file after borrowing

We can see that the stock file has been updated as well.

We can now return the book after we have borrowed it. To return the book, the user must select the "r" option, which activates the return function. After selecting the return option, the user is prompted to write the code of the book they are trying to return, similar to how they are asked to write the code of the book they are borrowing.

```
Book ID          Book Name                          Price
B001              Harry Potter                       $2
B002              Start With Why                     $1.5
B003              Programming with Python

Enter the book id of the book you choose to return: B001
```

Fig.7 Returning the book borrowed

```
B001, Harry Potter, JK Rowling,17, $2
B002, Start With Why, Simon Sinek,0 , $1.5
B003, Programming with Python, John Smith,14, $1.5
```

Fig.8 Records after returning the book

Now, if the book has been returned after the borrowed time, the fine will be added. The invoice showing fine is shown as follows:

```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
Manish Hajur Khadka
-------------Notice For Returing the book-----------------------------

name of the book is:  Harry PotterThe price of the book is:  $2The fine for this book is: 16.666666666666668
----------------------------------------------------------------
Grand Total:-                   $18.666666666666668

+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
The date you returned book is : 2021-09-10 08:32:55.504135
```

Fig.9 A note showing the description of book return

Finally, if a user wants to exit the software, they can do so by inputting the letter "q" as indicated in the interface. The software is terminated with the notice shown below after the choice is selected.

```
Press D to display the book
Press L to lend the book
Press R to Return the book
Press Q to Quit the Program
Enter your choice: Q
------------Thanks for visiting our Libary Keep coming----------------
```

Fig.10 Quitting the program

Testing

To show implementation of try, except

Test No	a
Objective	To Show Implementation of try, except
Action	Program is executed B is selected to borrow a book A1 is added as the book code.
Expected Results	Exception would be created, and the following message should be displayed: “Please give the correct input.”
Actual Results	Exception was created and the following message was displayed. “Please give the correct input.”
Conclusion	The test is successful.

Table.1 Table for Test A

Screenshot of Test A

```
Enter the code of the book you want to borrow: A1
Please give the correct input.
```

Fig.11 Screenshot of Test A

To show selection of borrow and return when negative value is given as an input

Test No	b

Objective	To Show selection of Borrow and Return Option when negative values is given as input.
Action	Program is executed -3 is selected as the choice
Expected Results	Error message saying “Enter a valid choice” should be displayed.
Actual Results	Error message saying “Enter the valid choice” was displayed.
Conclusion	The test is successful.

Table.2 Test B

Screenshot of Test B

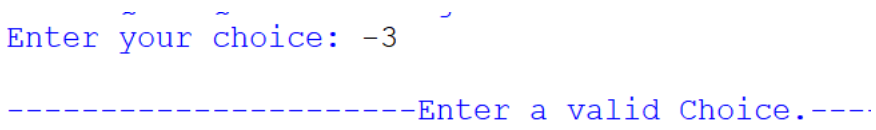


Fig.12 Screenshot of Test I

To show selection of borrow and Return option when nonexistent values are taken as an input

Test No	c
Objective	To Show selection of Borrow and Return Option when non existing values is given as input.
Action	Program is executed h is selected as the choice
Expected Results	Error message saying “Enter the valid choice” should be displayed.
Actual Results	Error message saying “Enter the valid choice” was displayed.
Conclusion	The test is successful.

Table3. Test C

Screenshot of Test C

```
Enter your choice: h

----------------------Enter a valid Choice.---------
```

Fig.13 Screenshot of Test C

To show the file generation of borrow

Test No	d
Objective	To show the File generation of borrow
Action	Program is executed B is selected which is borrow B001 is selected as the Book code ➢Manish Hajur Khadka is kept as the name ➢ Only one book is selected.
Expected Results	A Text File displaying the information of the books should be created
Actual Results	A Text File displaying the information of the books was created
Conclusion	The test is successful.

Table4. Test D

Screenshot of Test D

```
Press D to display the book
Press L to lend the book
Press R to Return the book
Press Q to Quit the Program
Enter your choice: L

Book ID         Book Name                       Price
B001             Harry Potter                    $2
B002             Start With Why                  $1.5
B003             Programming with Python                          $1.5

Enter the book id of the book you choose to borrow: B001
Enter your full name.: Manish Hajur Khadka
```

Figure.14 Screenshot of Test D

To show the file generation of return

Test No	e
Objective	To show the File generation of Return
Action	• Program is executed • R is selected which is borrow ➢ B001 is selected as the Book code ➢ 15 is kept as no of days. • Manish Hajur Khadka is kept as the name • Only one book is selected to return
Expected Results	A Text File displaying the information of the books should be created
Actual Results	A Text File displaying the information of the books was created
Conclusion	The test is successful.

Table.5 Test E

Screenshot of Test E

```
Enter your full name.: Manish Hajur Khadka
Press D to display the book
Press L to lend the book
Press R to Return the book
Press Q to Quit the Program
Enter your choice: R
Enter you name: Manish Hajur Khadka

Book ID         Book Name                       Price
B001             Harry Potter                    $2
B002             Start With Why                  $1.5
B003             Programming with Python

Enter the book id of the book you choose to return: B001

Enter the number of days you have had the book for: 15
```

Fig.15 Screenshot of Test E

Conclusion

The following project on Library Management System draws to a finish in this manner. Throughout the assignment, I learned a number of topics that have helped me have a better understanding of Python and IDLE. I've also learned how to set up a management system and how to update text files with Python. I've also learned how to make a flowchart, an algorithm, and Pseudocode. I've learnt more about how to break a program into independent modules using functions and how to make it easier to debug and test.

I'd like to thank our instructors, Sukrit Shakya sir for all of the lecture slides and for helping us learn more during lecture sessions, as well as Shreeyash Mool Sir for helping me grasp how different sorts of programs function in Tutorial Classes. Despite the fact that I had a good understanding of the following topic module. I had a minor stumbling block when invoking the routines. However, little research and watching internet videos assisted me in overcoming my difficulties, and I was able to accomplish my assignment. Python, in comparison, is a fairly simple language that is much easier to comprehend than languages like Java and C++. Python's syntax is more human-like than that of other languages, making it easier to comprehend. As a result, I was enthralled by this language because it was more approachable.

So, as I near the finish of my report, I'd want to point out that while the Library Management System project was challenging, it was also intriguing and thrilling

to study because it allowed me and possibly other students to understand more about the programming language and how programs are created.

Appendix

Functions.py

```
#Displays a menu which makes it easy for the user to carry out a function.
def menu():
    print("Press D to display the book")
    print("Press L to lend the book")
    print("Press R to Return the book")
    print("Press Q to Quit the Program")


#Displays the name of book and the price.
def menu1(list):
    print("\n\nBook Name\t\t\tPrice")
    for i in list:
        print(i[1]+"\t\t\t"+i[4])


#Displays the Book ID, the name of the book and the price of the book.
#Asks for the book ID.
def menu2(list):
    print("\nBook ID\t\tBook Name\t\t\tPrice")
    for i in list:
        print(i[0]+"\t\t"+i[1]+"\t\t\t"+i[4])
    bookId = input("\nEnter the book id of the book you choose to borrow: ")
```

```
    return bookId


#Displays the Book ID, the name of the book and the price of the book.
#Asks for the book ID.
def menu3(list):
    print("\nBook ID\t\tBook Name\t\t\tPrice")
    for i in list:
        print(i[0]+"\t\t"+i[1]+"\t\t\t"+i[4])
    bookId = input("\nEnter the book id of the book you choose to return: ")
    return bookId
```

library.py

```
#Main file of the code.
#importing functions from different modules.
import loadingList
import functions
import noteGenerator
import listManipulation
#Saves the imported list as list2.
list2 = loadingList.load('Books.txt','r')
# userName = input("Enter your full name.: ")
#Setting up variables
cart = []
returned = []
daysBorrowed = 0
z = 1
#Creating infinite loop which breaks when user input is 4.
while z>0:
    #Calling function menu from functions module.
    functions.menu()
    choice = input("Enter your choice: ")
    try:
```

```
    if choice == "D":
    # #Calling function menu1 from functions module.
    # functions.menu1(list2)
        file = open("Books.txt","r")
        text = file.read()
        print(text)
    elif choice == "L":


        #Calling function menu2 from functions module.
        Id = functions.menu2(list2)
        userName = input("Enter your full name.: ")
        for i in list2:
            if Id == i[0]:
                if int(i[3])>0:
                    #Adding the file to cart and reducing the stock.
                    cart.append(i)
                    stock = int(i[3])
                    stock = stock - 1
                    i[3] = str(stock)
                else:
                    print("\n----------Sorry the item is out of stock.----------
--\n")



    elif choice == "R":
    #Calling function menu3 from functions module.
        userName = input("Enter you name: ")
        Id = functions.menu3(list2)
        daysBorrowed =  int(input("\nEnter the number of days you have had the book for: "))
        for i in list2:
            if Id == i[0]:
            #Adding the file to returned list and increasing the stock.
```

```
                returned.append(i)
                stock = int(i[3])
                stock = stock + 1
                i[3] = str(stock)



        elif choice == "Q":
            print("------------Thanks for visiting our Libary Keep coming----------")
            break
        else:
            print("\n----------------------Enter a valid Choice.--------------------\n")
    except Exception as e:
        print(e)
#Creating a borrow note from noteGenerator module.
noteGenerator.returnNote(returned, daysBorrowed, userName)
#Creating a return note from noteGenerator module.
noteGenerator.borrowNote(cart, userName)
#Appending the list by calling appendList function from listManipulation module and
manipulating the stock of the books
listManipulation.appendList(list2)
```

listManipulation.py

```
#Rewrites the initial text file with the new data gained through the user interactions.
def appendList(list):
    file = open('Books.txt','w')
    for i in list:
        for j in range(len(i)):
            file.write(i[j])
            if j < 4:
                file.write(",")
        file.write('\n')
```

loadinglist.py

```
#Loads the data from the text file.
def load(filename, mode):
    file1= open(filename, mode)
    contents= file1.read()
    file1.close()
    #Splits the list in places where lines separate.
    list1= contents.split('\n')
    list2=[]
    for each_item in list1:
        #Splits the list where there is a comma.
        list2.append(each_item.split(","))
    #Since the last element of the text file is an empty space it gets removed.
    list2.pop()
    return list2
```

note generator.py

```
#Creates a borrow note.
import datetime
def borrowNote(list, name):
    date_time = datetime.datetime.now()
    B_file = open('borrow_notice.txt','w')
    B_file.write("+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+\n")
    B_file.write(name+"\n")
    B_file.write("-----------------NOTICE FOR BORROWING THE BOOK: \n\n")

    grand_Total = 0
    #Calculation of the grand total and writing the data onto the text B_file.
    for i in list:
        price = i[4].replace("$","")
        price = float(price)
```

```
        B_file.write("Name of the book is: "+ i[1])
        B_file.write("Price of the book is: "+ i[4])


        grand_Total = grand_Total + price
    B_file.write("---------------------------------------\n")
    B_file.write("Grand Total:-\t\t\t$"+str(grand_Total)+"\n\n")
    B_file.write("+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+\n")
    B_file.write("The date you issued book is : " + str(date_time))
    B_file.close()



#Creates a return note.
def returnNote(list, days, name):
    date_time = datetime.datetime.now()
    R_file = open('return_notice.txt','w')
    total = 0
    R_file.write("+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+\n")
    R_file.write(name+"\n")
    R_file.write("--------------Notice For Returing the book----------------------------- \n\n")

    #Calculates the grand total after returning and fining and writes it to a text B_file.
    for i in list:
        fine = 0
        price = i[4].replace("$",'')
        price = float(price)
        fineDays = 0
        if days > 10:
            fineDays = days - 10
            fine = (1 / 0.6) * price * fineDays
        print("You are late in returing the book fine would up added")
        R_file.write("name of the book is: "+i[1])
```

```
        R_file.write("The price of the book is: "+i[4])
        R_file.write("The fine for this book is: "+str(fine)+"\n")

        total = total + price + fine
    R_file.write("-----------------------------------------------------------------\n")
    R_file.write("Grand Total:-\t\t\t$"+str(total)+"\n\n")
    R_file.write("+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+\n")
    R_file.write("The date you returned book is : " + str(date_time))
    R_file.close()
```