LONDON METROPOLITAN UNIVERSITY

islington college
(इस्लिङ्टन कलेज)

Module Code & Module Title

CS4051NI Fundamentals of Computing

Assessment Weightage & Type

60% Individual Coursework

Year and Semester

2020-21 Autumn

Student Name: Rabina Shrestha

Group: C13

London Met ID: 20049416

College ID: NP01CP4S210039

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.

Contents

List of Figures

List of Tables

1. Introduction

1.1 Introduction to the project.

One of the modules that Computing students' study in Semester 2 is "Fundamentals of Computing". The assignment was given on the 6th week which weights 60% of the overall module. Based on the instructions given, we were required to develop an application. This coursework enables the students to develop a library management system. A library in itself is an organized system that makes borrowing and returning of books systematic. In contrast to the traditional method where the librarian keeps notes through pen and paper, it is much more convenient to use a library management system since the data stored is permanent and not easily spoiled or lost.

An application is developed to read the text file and display the books from the library which is stored in a text file. With each transaction a note is generated for the borrower. The note contains information about the borrower; name of the borrower, book information, issued date/time, last day of return, and sub-total cost while borrowing. Another note is generated after the book is returned which contains the above-mentioned information with additional late fees cost if the limited days are exceeded i.e., 10 days, along with the borrowed duration. Also, the stock of books is updated after every transaction.

1.2 Goals and Objectives.

With the use of algorithm and pseudocodes, we were expected to make a library management system in order to assist the management of books in the library. The aim of this coursework is to make a user friendly, easy to use library management system. This project provides a ground for the worker as well as the borrower to easily access information, make smooth transaction as well as provide more security in terms of data losses and frauds. The receipt contains clear information regarding the number of books borrowed by a particular person, the total cost which decreases the confusion regarding the end cost which occurs if the data is not stored in the way the library management system allows.

2. Discussion and Analysis

2.1 Algorithm.

One of the most important concepts in computer science is algorithm. Algorithms are the detailed set of instructions that are implemented to get the solution to a specific task. It can be written as a list of steps or as a diagram which represents the workflow of the program using shapes and arrows. (Sharma, 2020)

Stepwise Algorithm

Step 1: Start.
Step 2: Display the options for the users.
Step 3: Ask the user to input their choice from 1/2/3/4.

Step 4: If the user selects 1, which is to view the list of books; go to Step 5.
Step 5: Read “books.txt” file, display the information provided in books, and go back to Step 2.

Step 6: If the user selects 2, which is to borrow a book; go to Step 7.
Step 7: Ask the user to input their name.
Step 8: Ask the user to input the book ID of the book they want to borrow.
Step 9: Ask the user if they want to add another book; if “y” go back to Step 9 and if the user inputs “n” then go to Step 10.
Step 10: Create a borrow receipt in a text file, and print the exact same receipt in the shell. Go back to Step 2.

Step 11: If the user selects 3, which is to return a book; go to Step 12.
Step 12: Ask the user to input their name. If the name does not exist in borrow_Record; repeat this step.
Step 13: Ask the user to input the book ID of the book they borrowed. If the book ID does not match with borrow_Record; repeat this step.

Step 14: Ask the user to enter the borrowed duration of the book.

Step 15: Ask the user if they want to return another book. If the user inputs “y” then go back to Step 13, and if the user inputs “n”; go to Step 16.

Step 16: Create a return receipt in a text file, and print the exact same receipt in the shell. Go back to Step 2.

Step 17: If the user selects 4, which is to exit; print “Thank you and have a nice day!” and exit the program.

2.2 Flowchart.

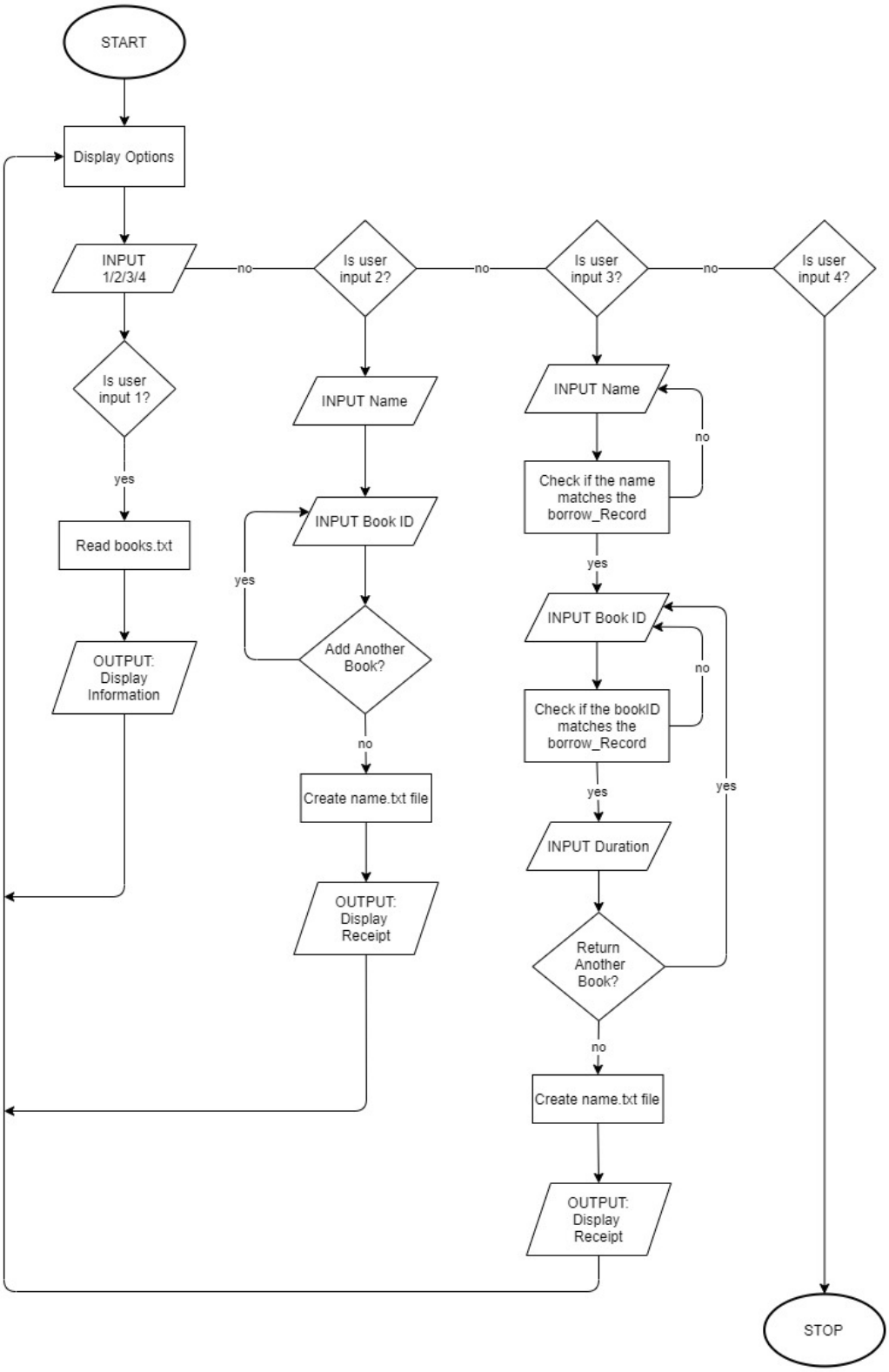


Figure 1: Flowchart

2.3 Pseudocode.

2.3.1. main.py

```
IMPORT os
IMPORT booksFile
IMPORT borrowBook
IMPORT returnBook
IMPORT dateTime

ASSIGN 0.5 TO LATE_FEES_PER_DAY.

DEFINE root function
DO
   PRINT ()
   PRINT ("=" 78 times)
   PRINT ("\t\t Welcome to Library Management System. \(^0^)/" + u"\U0001F4DA")
   WHILE True
   DO
      PRINT ("-" 78 times + "\n")
      PRINT (" To view the list of Books: PRESS 1")
      PRINT (" To borrow a Book          : PRESS 2")
      PRINT (" To return a Book          : PRESS 3")
      PRINT (" To Exit                   : PRESS 4")
      PRINT ("\n" + "-" 78 times)
      TRY
      DO
         INPUT choice (“Select your choice: ") AS int type.
         PRINT ("-" 78 times)
         IF choice is equal to 1
         DO
```

```
        PRINT ("_" 78 times)
        ASSIGN empty [] TO books_List
        ASSIGN ["BookID", "Book Name", "AuthorName", "Stock", "Cost"]TO columns

        WITH OPEN ("books.txt", 'r') AS file
        DO
            FOR line IN file
            DO
                ASSIGN line.split(',') TO book
                books_List.append(book)
            END FOR.
        END WITH.

        PRINT("{:<5s} {:<30s} {:<20s} {:<10s} {}" with format (
            columns [0], columns [1], columns [2], columns [3], columns [4]))
        PRINT ("_" 78 times +"\n")

        FOR book IN books_List:
        DO
            PRINT("{:<5s} {:<30s} {:<20s} {:<10s} {}" with format (
                " "+book [0], " "+book [1], book [2], book [3], u"\xA3" + book [4]), end="")
        END FOR.

        PRINT ("_" 78 times +"\n")
    END IF.

    ELIF choice is equal to 2
    DO
        ASSIGN empty [] TO add_to_receipt
        INPUT person_Name ("\n Enter your full name: ")
        WHILE True
```

```
DO
   INPUT book_ID ("\n"+" Enter the Book ID: ")
   WITH OPEN("books.txt","r") AS file
   DO
      SET exists AS False
      FOR line IN file
      DO
         ASSIGN book AS line.split(",")
         IF book [0] is equal to book_ID
         DO
            SET exists AS True
            BREAK
         END IF.
      END FOR.
      IF exists
      DO
         BREAK
      END IF.
      ELSE:
      DO
         PRINT ("\n" + "-" 78 times)
         PRINT (" Book not Found")
         PRINT ("-" 78 times)
      END ELSE.
   END WITH.
END WHILE.
IF borrowBook.borrow_Book(person_Name, book_ID)
DO
   add_to_receipt.append(book_ID)
END IF.
WHILE True
```

```
DO
    INPUT another (
        " Do you want to add another book? (y/n): ")
    IF another.lower() is equal to "n"
    DO
        BREAK
    END IF.
    ELIF another.lower() is equal to "y"
    DO
        INPUT next_Book ("\n Enter another Book ID: ")
        IF borrowBook.borrow_Book(person_Name, next_Book)
        DO
            add_to_receipt.append(next_Book)
        END IF.
    END ELIF.
    ELSE
    DO
        PRINT (" Enter a valid response!")
    END ELSE.
END WHILE.

ASSIGN data TO empty []
SET sub_total AS 0
WITH OPEN ("Books.txt", "r") AS file
DO
    FOR line IN file
    DO
        ASSIGN line.split(",") TO book
        FOR each_id IN add_to_receipt
        DO
            IF book[0] is equal to each_id:
```

```
                DO
                    data.append(
                        [book[0], book[1], book[2], book[4]])
                    sub_total += float(book[4])
                END IF.
            END FOR.
        END FOR.
    END WITH.

    FOR book IN data
    DO
        ASSIGN str(book[-1]).replace("\n", "") TO book[-1]
    END FOR.

    WITH OPEN(f"borrow_Receipt/{person_Name}.txt", "w+") AS file:
    DO
        file.WRITE("=" 72 times + "\n\n")
        file.WRITE("                    Library Management System" + "\n")
        file.WRITE("_" 72 times + "\n")
        file.WRITE("Date:"+dateTime.getDate()+"Time:"+dateTime.getTime() +"\n")
        file.WRITE("_" 72 times + "\n")
        file.WRITE(f"\n Borrower Name: {person_Name}\n\n")
        file.WRITE("-" 72 times + "\n")
        file.WRITE("{:<6s} {:<7s} {:<29s} {:<20s} {}" with format(
            " SNo.", "BookID","BookName", "AuthorsName", " Cost" +"\n"))
        file.WRITE("-" 72 times + "\n")
        SET SNo AS 0
        FOR book IN data
        DO
            SNo = SNo + 1
            file.WRITE(
```

```
        "{:<1s} {:<5d} {:<5s} {:<30s} {:<20s} {} \n"with format ("", SNo, book [0],
        book [1], book [2],"£"+book [3]))
END FOR.
file.WRITE("-" 72 times + "\n")
file.WRITE("\n Last date of Return: "+dateTime.getReturnDate()+"\n")
file.WRITE(
    f"Late fees per day after 10 days will be:£{LATE_FEES_PER_DAY}\n\n")
file.WRITE("_" 72 times + "\n\n")
file.WRITE(f"\t\t\t\t\t\t\t Sub-Total: £{sub_total} \n\n")
file.WRITE("="72 times)

PRINT ("\n  "+"-"*74)
PRINT (f"   borrow_Receipt/{person_Name}.txt")
PRINT ("   "+"=" 72 times)
PRINT ("\n                        Library Management System")
PRINT ("   "+"_"72 times)
PRINT ("\n   "+" Date: "+dateTime.getDate()+"Time: " +dateTime.getTime())
PRINT ("   "+"_"72 times)
PRINT (f"\n    Borrower Name: {person_Name}\n")
PRINT ("   "+"-"72 times)
PRINT("{:<6s} {:<7s} {:<29s} {:<20s} {}".FORmat(
        "    SNo.", "BookID","BookName", "AuthorsName", " Cost"))
PRINT ("   "+"-"72 times)
SET SNo AS 0
FOR book IN data
DO
    SNo = SNo + 1
    PRINT (
            "{:<1s} {:<5d} {:<5s} {:<30s} {:<20s} {} \n “with format ("", SNo,
            Book [0], book [1], book [2],"£"+book [3]))
END FOR.
```

```
            PRINT ("   "+"-"72 times)
            PRINT ("\n   "+" Last date of Return: "+dateTime.getReturnDate())
            PRINT (
                f"Latefees per day after10 days will be: £{LATE_FEES_PER_DAY}\n")
            PRINT ( "   "+"-" 72 times + "\n")
            PRINT (f"\t\t\t\t\t\t   Sub-Total: £{sub_total}\n")
            PRINT ("   "+"="72 times + "\n")
            PRINT ("  "+"-"*74 + "\n")
        END WITH.
    END ELIF.

    ELIF choice is equal to 3
    DO
        ASSIGN empty [] TO data
        SET sub_total AS 0
        SET total_late_fees AS 0
        ASSIGN empty [] TO add_to_receipt

        WHILE True
        DO
            INPUT person_Name ("\n Enter your full name: ")
            SET exists AS False
            WITH OPEN("borrow_Record.txt","r") AS file
            DO
                FOR line IN file:
                DO
                    ASSIGN lend AS line.split(",")
                    IF lend[1] is equal to person_Name
                    DO
                        SET exists AS True
                        BREAK
```

```
            END IF.
        END FOR.
    END WITH.

    IF exists
    DO
        BREAK
    END IF.
    ELSE
    DO
        PRINT ("\n" + "-" 78 times)
        PRINT (f" Cannot find borrow record with the name: {person_Name}")
        PRINT ("-" 78 times)
    END ELSE.
END WHILE.
WHILE True
DO
    INPUT book_ID ("\n"+" Enter the Book ID: ")
    WITH OPEN("borrow_Record.txt","r") AS file:
    DO
        SET exists AS False
        FOR line IN file
        DO
            ASSIGN book AS line.split(",")
            IF lend[0] is equal to book_ID
            DO
                SET exists AS True
                BREAK
            END IF.
        END FOR.
        IF exists
```

```
            DO
               BREAK
            END IF.
            ELSE
            DO
               PRINT ("\n" + "-" 78 times)
               PRINT (" Book not Found")
               PRINT ("-" 78 times)
            END ELSE.
         END WITH.
      END WHILE.

      IF returnBook.return_Book(person_Name, book_ID)
      DO
         WITH OPEN ("books.txt", "r") AS file:
         DO
            FOR line IN file
            DO
               ASSIGN book AS line.split(",")
               IF book [0] is equal to book_ID
               DO
                  WITH OPEN ("borrow_Record.txt", "r") AS borrow_Bundle
                  DO
                     ASSIGN borrow_Bundle.readlines() TO lines
                     ASSIGN empty [] TO new_Lines
                     FOR line IN lines
                     DO
                        ASSIGN book AS line.split (",")
                        SET (lend[-1]).replace("\n", "") TO lend[-1] AS str type
                        IF lend[0] is equal to book_ID and lend[1].lower() is equal to
                        person_Name.lower()
```

```
                    DO
                        data.append(
                            [book[0], book[1], book[2], lend[3], book[4]])
                        sub_total += float(book[4])
                        IF int(lend[3]) more than 10
                        DO
                            total_late_fees += (int(lend[3]) - 10) * \
                                LATE_FEES_PER_DAY
                        END IF.
                    END IF.
                    ELSE
                    DO
                        new_Lines.append(line)
                    END ELSE.
                END FOR.

                WITH OPEN ("borrow_Record.txt", "w") as file
                DO
                    file.writelines(new_Lines)
                END WITH.
            END WITH.
        END IF.
    END FOR.
  END WITH.
END IF.

WHILE True
DO
  INPUT another (" Do you want to return another book? (y/n): ")
  IF another.lower() is equal to "n"
  DO
```

```
        BREAK
    END IF.
    ELIF another.lower() is equal to "y"
    DO
        INPUT next_Book ("\n Enter another Book ID: ")
        IF returnBook.return_Book(person_Name, next_Book):
        DO
            WITH OPEN ("books.txt", "r") AS file
            DO
                FOR line IN file
                DO
                    ASSIGN book AS line.split(",")
                    IF book[0] is equal to next_Book
                    DO
                        WITH OPEN ("borrow_Record.txt", "r") AS borrow_Bundle
                        DO
                            ASSIGN borrow_Bundle.readlines() TO lines
                            ASSIGN new_Lines TO empty []
                            FOR line IN lines:
                            DO
                                ASSIGN lend AS line.split(",")
                                SET (lend[-1]).replace("\n", "") TO lend[-1] AS str type
                                IF lend[0] is equal to next_Book and lend[1].lower() is
                               equal to person_Name.lower()
                                DO
                                    data.append(
                                        [book[0], book[1], book[2], lend[3], book[4]])
                                    sub_total += float(book[4])
                                    IF int(lend[3]) is greater than 10:
                                    DO
                                        total_late_fees += (int(lend[3]) - 10) * \
```

```
                                LATE_FEES_PER_DAY
                            END IF.
                        END IF.
                        ELSE
                        DO
                            new_Lines.append(line)
                        END ELSE.
                    END FOR.
                    WITH OPEN ("borrow_Record.txt", "w") as file
                    DO
                        file.WRITElines(new_Lines)
                    END WITH.
                END WITH.
            END IF.
        END FOR.
    END WITH.
  END IF.
END ELIF.

ELSE
DO
  PRINT (" Enter a valid response!")
END ELSE.
END WHILE.

FOR book IN data
DO
  SET (book [-1]).replace("\n", "") TO book [-1] AS str type
END FOR.

WITH OPEN(f"return_Receipt/{person_Name}.txt", "w+") AS file
```

```
DO
    file.WRITE("=" 72 times + "\n\n")
    file.WRITE("                    Library Management System" + "\n")
    file.WRITE("_" 72 times + "\n")
    file.WRITE("Date:"+dateTime.getDate()+"Time:"+dateTime.getTime()+ "\n")
    file.WRITE("_" 72 times + "\n")
    file.WRITE(f"\n Returned By: {person_Name}\n\n")
    file.WRITE("-" 72 times + "\n")
    file.WRITE("{:<6s} {:<7s} {:<29s} {:<20s} {}".with format(
        " SNo.", "BookID","BookName", "AuthorsName", " Cost" +"\n"))
    file.WRITE("-" 72 times + "\n")
    SET SNo AS 0
    FOR book IN data
    DO
        SNo = SNo + 1
        file.WRITE(
            "{:<1s} {:<5d} {:<5s} {:<30s} {:<20s} {}\n ".with format ("", SNo, book[0],
            book [1], book [2],"£"+book [4]))
    END FOR.
    file.WRITE("-"72 times + "\n")
    file.WRITE("\n Borrowed Duration in days: "+book[3])
    file.WRITE(
        f"\n Late fees per day after 10 days will be:
        £{LATE_FEES_PER_DAY} \n\n")
    file.WRITE("_" 72 times + "\n\n")
    file.WRITE(f"\t\t\t\t\t\t\tSub-Total: £{sub_total}\n")
    file.WRITE(f"\t\t\t\t\t\t\tLate-Fees: £{total_late_fees}\n")
    total_fees = sub_total + total_late_fees
    two_float = "{:.2f}". with format(total_fees)
    file.WRITE(f"\t\t\t\t\t\t\tTotal    : £{total_fees}\n\n")
    file.WRITE("="72 times)
```

```
PRINT ("\n  "+"-" 74 times)
PRINT (f"   return_Receipt/{person_Name}.txt")
PRINT ("   "+"=" 72 times)
PRINT ("\n                        Library Management System")
PRINT ("   "+"_" 72 times)
PRINT ("\n"+" Date: "+dateTime.getDate()+"Time: " +dateTime.getTime())
PRINT ("   "+"_" 72 times)
PRINT (f"\n    Returned By: {person_Name}\n")
PRINT ("   "+"-" 72 times)
PRINT ("{:<6s} {:<7s} {:<29s} {:<20s} {}".FORmat(
        "    SNo.", "BookID","BookName", "AuthorsName", " Cost"))
PRINT ("   "+"-" 72 times)
SET SNo AS 0
FOR book IN data
DO
   SNo = SNo + 1
   PRINT (
         "{:<1s}{:<5d} {:<5s} {:<30s} {:<20s}{}\n".with format ("",SNo, book[0],
         book [1], book [2],"£"+book [4]))
END FOR.
PRINT ("   "+"-"72 times)
PRINT ("\n    Borrowed Duration in days: "+book [3])
PRINT (
        f"Latefees per day after10 days will be: £{LATE_FEES_PER_DAY}\n")
PRINT ("   "+"-"72 times + "\n")
PRINT (f"\t\t\t\t\t\t\t   Sub-Total: £{sub_total}")
PRINT (f"\t\t\t\t\t\t\t   Late-Fees: £{total_late_fees}")
SET sub_total + total_late_fees AS total_fees
SET two_float AS "{:.2f}". with format(total_fees)
PRINT (f"\t\t\t\t\t\t\t   Total    : £{total_fees}\n")
```

```
                PRINT ("   "+"="72 times + "\n")
                PRINT ("  "+"-"*74 + "\n")
            END WITH.
        END ELIF.
        ELIF choice is equal to 4:
        DO
            PRINT ("\t\t\t Thank You and Have a Nice Day!")
            PRINT ("="*78 times)
            BREAK
        END ELIF.

        ELSE
        DO
            PRINT (" Invalid choice. Try again.")
        END ELSE.

    EXCEPT ValueError
    DO
        PRINT ("\n" + "-" 78 times)
        PRINT (" Please enter 1/2/3/4.")
        PRINT ("-" 78 times + "\n")
    END EXCEPT.
  END WHILE.
END DO.
IF __name__ is equal to "__main__"
DO
  IF not os.path.exists("borrow_Receipt")
  DO
    os.makedirs("borrow_Receipt")
  END IF.
  IF not os.path.exists("return_Receipt")
```

```
    DO
        os.makedirs("return_Receipt")
    END IF.
    CALL root() function
END IF.
```

2.3.2. booksFile.py

```
IMPORT dateTime

INTIALIZE Book() AS class
DO
    DEFINE __init__ WITH (self, instance)
    DO
        SET self.book_ID AS instance[0]
        SET self.book_Name AS instance[1]
        SET self.book_Author AS instance[2]
        SET self.book_Stock AS instance[3]
        SET self.book_Cost AS instance[4]
    END DO.

    DEFINE check_Borrow WITH (self, person_Name)
    DO
        WITH OPEN ('borrow_Record.txt', 'r') AS file
        DO
            FOR line IN file
            DO
                ASSIGN line.split(',') TO lend
                IF lend[0] is equal to self.book_ID and lend[1].lower() is equal to
                person_Name.lower()
                DO
```

```
            RETURN True
          END IF.
        END FOR.
      END WITH.
    END DO.

    DEFINE update_Stock WITH (self)
    DO
      WITH OPEN ("books.txt", "r") AS file
      DO
        ASSIGN file.readlines() TO lines
        ASSIGN empty [] TO new_Lines
        FOR line IN lines
        DO
          ASSIGN line.split(',') TO book
          IF book[0] is equal to self.book_ID
          DO
            ASSIGN f" {self.book_ID}, {self.book_Name}, {self.book_Author},
            {self.book_Stock}, {self.book_Cost}" TO updated_Line
            new_Lines.append(updated_Line)
          END IF.
          ELSE:
          DO
            new_Lines.append(line)
          END ELSE.
        END FOR.
        WITH OPEN ("books.txt", "w") AS file
        DO
          file.writelines(new_Lines)
        END WITH
      END WITH.
```

```
END DO.

DEFINE borrow WITH (self, person_Name)
DO
    WITH OPEN("borrow_Record.txt", "a") AS file
    DO
        ASSIGN dateTime.getDate() TO date
        ASSIGN dateTime.getTime() TO time
        file.WRITE (f"{self.book_ID},{person_Name},{date}{time}\n")
    END WITH.
    ASSIGN int type (self.book_Stock) –1 TO self.book_Stock
    CALL self.update_Stock()
END DO.

DEFINE return_Back WITH (self, person_Name)
DO
    WHILE True
    DO
        TRY
        DO
            INPUT duration (" Enter the borrowed duration (in days): ")) AS int type
            WITH OPEN ("borrow_Record.txt", "r") AS file
            DO
                ASSIGN file.readlines() TO lines
                ASSIGN empty [] TO new_Lines
                WITH OPEN ("borrow_Record.txt", "w") AS file
                DO
                    FOR line IN lines
                    DO
                        ASSIGN lend AS line.split(",")
                        SET (lend[-1]).replace("\n", "") TO lend[-1] AS str type
```

```
                    IF lend[0]is equal to self.book_ID and lend[1] is equal to person_Name
                    DO
                        new_Lines.append(f" {lend[0]},{lend[1]},{lend[2]},{duration}\n")
                    END IF.
                    ELSE
                    DO
                        new_Lines.append(line)
                    END ELSE.
                END FOR.
                file.writelines(new_Lines)
            END WITH.
        END WITH.
        ASSIGN int type (self.book_Stock) + 1 TO self.book_Stock
        self.update_Stock()
        BREAK
    END TRY.
    EXCEPT ValueError
    DO
        PRINT ("\n" + "-" *78)
        PRINT (" Please input the borrowed duration in numeric value.")
        PRINT ("-" *78 + "\n")
    END EXCEPT.
  END WHILE.
 END DO.
END DO.
```

2.3.3. borrowBook.py

```
IMPORT booksFile

DEFINE borrow_Book WITH (person_Name, book_ID)
DO
   WITH OPEN ('books.txt', 'r') AS file
   DO
      FOR line IN file
      DO
         ASSIGN line.split(',') TO book
         IF book[0] is equal to book_ID
         DO
            ASSIGN booksFile.Book(book) TO book_ins
            IF NOT book_ins.check_Borrow(person_Name)
            DO
               IF int type (book_INs.book_Stock) is less than 1
               DO
                  PRINT ("-" 78 times)
                  PRINT (" Book out of Stock.")
                  PRINT ("-" 78 times+ "\n")
                  RETURN False
              END IF.
               book_ins.borrow(person_Name)
               RETURN True
            END IF NOT.
            ELSE:
            DO
               PRINT ("\n" +"-" 78 times)
               PRINT (" You have already borrowed this book.")
               PRINT ("-" 78 times+ "\n")
```

```
                RETURN False
            END ELSE.
        END IF.
    END FOR.
    PRINT ("\n"+"-” 78 times)
    PRINT (" Book not found.")
    PRINT ("-” 78 times + "\n")
  END WITH.
END DO.
```

2.3.4. returnBook.py

```
IMPORT booksFile

DEFINE return_Book WITH (person_Name, book_ID)
DO
  WITH OPEN ("books.txt", "r") AS file
  DO
    FOR line IN file
    DO
      ASSIGN line.split(',') TO book
      IF book[0] is equal to book_ID
      DO
        ASSIGN booksFile.Book(book) TO book_ins
        IF book_ins.check_Borrow(person_Name)
        DO
          book_ins.return_Back(person_Name)
          RETURN True
        END IF.
        ELSE
        DO
```

```
                PRINT ("\n" +"-" 78 times)
                PRINT (" Borrow record not found.")
                PRINT ("-" 78 times + "\n")
                RETURN False
            END ELSE.
        END IF.
    END FOR.
    PRINT ("\n" +"-" 78 times)
    PRINT (" Book not found.")
    PRINT ("-" 78 times + "\n")
  END WITH.
END DO.
```

2.3.5. dateTime.py

```
DEFINE getDate()
DO
  IMPORT datetime
  ASSIGN datetime.datetime.now TO Date
  RETURN in str type (Date().date())
END DO.

DEFINE getTime()
DO
  IMPORT datetime
  ASSIGN datetime.datetime.now TO Time
  RETURN in str type (Time().time())
END DO.

DEFINE getReturnDate()
DO
```

```
    IMPORT datetime
    FROM datetime IMPORT date, timedelta
    ASSIGN datetime.date.today() TO BorrowDate
    ASSIGN datetime.timedelta(days=10) TO DeltaTime
    RETURN in str type (BorrowDate + DeltaTime)
END DO.
```

2.4 Data Structures.

The second most fundamental concept in computer science is the data structures in Python. It is a way of organizing and storing the data so it works efficiently and can be accessed easily. The relationship between various logical operations and the data is defined with the help of data structure.

Data structures is generally categorized into two:

1. Primitive Data Type:

These are the basic data structures which contains pure, simple values of a data, serving as the building blocks for data manipulation. (Sharma, 2020)

Variables with primitive data type are listed below:

- Integers: This data type represents numeric data, from positive whole numbers to negative. For example: 5, 7, -3, or -2.

Int used in the program:

```
print(" To view the list of Books: PRESS 1")
print(" To borrow a Book         : PRESS 2")
print(" To return a Book         : PRESS 3")
print(" To Exit                  : PRESS 4")
print("\n" + "-" *78)
# Exception handling is used.
try:
    choice = int(input(" Select your choice: "))
    print("-" *78)
    if choice == 1:
        print("_" *78)
        # View logic.
        books_List = []
        columns = [" BookID" , " Book Name","AuthorName", "Stock" , "  Cost"]
```

Figure 2: Integer example 1.

The main module which is main.py, contains int which is a primitive data type. It asks the user to select from the given choices as shown above; 1,2,3, or 4. int(input()) ensures that we will get an input which is a whole number. If the user inputs a string data type, then an error message will be printed.

```
for line in lines:
    lend = line.split(",")
    lend[-1] = str(lend[-1]).replace("\n", "")
    if lend[0] == book_ID and lend[1].lower() == person_Name.lower():
        data.append(
            [book[0], book[1], book[2], lend[3], book[4]])
        sub_total += float(book[4])
        if int(lend[3]) > 10:
            total_late_fees += (int(lend[3]) - 10) * \
                LATE_FEES_PER_DAY
    else:
        new_Lines.append(line)
with open("borrow_Record.txt", "w") as file:
    file.writelines(new_Lines)
```

Figure 3: Integer example 2.

The main module: main.py includes another int data type. If the given lend[3] which contains the duration of the borrowed book is greater than 10 then the user has to pay the added late fees. In order to check if the value is greater than 10 int data type is used in lend[3] to return an integer object and to compare. If lend[3] is greater than 10 then the number stored in it is subtracted with 10 as late fees per days is executed once the borrowed duration is more than 10 days.

- Float: This data type float stands for “Floating Point Number”. It represents the rational numbers containing decimal points. For example: 1.11, 5.23, or 7.25.

Float used in the program:

```
for line in lines:
    lend = line.split(",")
    lend[-1] = str(lend[-1]).replace("\n", "")
    if lend[0] == book_ID and lend[1].lower() == person_Name.lower():
        data.append(
            [book[0], book[1], book[2], lend[3], book[4]])
        sub_total += float(book[4])
        if int(lend[3]) > 10:
            total_late_fees += (int(lend[3]) - 10) * \
                LATE_FEES_PER_DAY
    else:
        new_Lines.append(line)
with open("borrow_Record.txt", "w") as file:
    file.writelines(new_Lines)
```

Figure 4: Float Example.

The main module: main.py includes float data type. Here book[4] refers to the cost of a certain book. As cost of the books are in decimal numbers, float data type is used in order to add them and store them in sub_total. When the user adds more than one book the float value of the cost is added.

- String: This data type denotes words, collection of alphabets or other characters. A string can be created by including series of characters within ‘’ and “”. It can be concatenated by applying + operations on two or more strings and can be repeated by applying *. Strings can also be sliced, capitalized and retrieved using different operations.

String used in the program:

```
print("=" *78)
print("\t\tWelcome to Library Management System.\(^0^)/" + u"\U0001F4DA")

while True:
    ''' Shows the services provided by the system.
        Asks the user to select a choice.'''
    print("-" *78 + "\n")
    print(" To view the list of Books: PRESS 1")
    print(" To borrow a Book         : PRESS 2")
    print(" To return a Book         : PRESS 3")
    print(" To Exit                  : PRESS 4")
    print("\n" + "-" *78)
```

Figure 5: String Example.

The main module: main.py includes str data type. It is used to display the things you see in the shell. + and * operations are used. - * 78 repeats the "- "78 times to create a line effect. + "\n" is used to end of a line text, concatenating it will add another line after the – is repeated 78 times.

- Boolean: This data type takes in boolean values which is either True or False, it can be interchanged with the integers 1 for True and 0 for False. It is used in comparison and conditional expressions.

Boolean used in the program:

```
# Checks whether book ID exists.
while True:
    book_ID = input("\n"+" Enter the Book ID: ")
    with open("borrow_Record.txt","r") as file:
        exists = False
        for line in file:
            book = line.split(",")
            if lend[0] == book_ID:
                exists = True
                break
        if exists:
            break
        else:
            print("\n" + "-" *78)
            print(" Book not Found")
            print("-" *78)
```

Figure 6: Boolean Example.

The main module: main.py includes boolean data type. Here “exists” is assigned as False. It is used as a conditional expression. In the next statement once the lend[0] and the book_ID matches with each other exists is assigned as True and the code breaks. If lend[0] is not equal to book_ID then the program executes the else part. It is used here to check if the person returning a book with that certain book_ID matches the previously borrowed record.

2. Non-Primitive Data Type:

In contrast to the previous data type, non-primitive data types store a collection of values in different formats, not just a value like the primitive data type. (Jaiswal, 2017)

Variables with primitive data type are listed below:

- Lists: One of the most versatile data structures in Python is the list. It is used to store a collection of heterogeneous and homogeneous elements which is written as a list of comma-separated elements enclosed within [] square brackets. Its content can be changed, keeping the identity intact which is why they are called mutable. Many methods are provided by Python to manipulate and work with lists. Add new items, remove some, sort out, extend, append, and many more can be done.

Lists used in the program:

```
if choice == 1:
    print("_" *78)
    # View logic.
    books_List = []
    columns = [" BookID" , " Book Name","AuthorName", "Stock" , "  Cost"]

    with open("books.txt", 'r') as file:
        for line in file:
            book = line.split(',')
            books_List.append(book)

    print("{:<5s} {:<30s} {:<20s} {:<10s} {}".format(
        columns[0], columns[1], columns[2], columns[3], columns[4]))
    print("_" *78 +"\n")

    for book in books_List:
        print("{:<5s} {:<30s} {:<20s}  {:<10s}{}".format(
            " "+book[0], " "+book[1],book[2], book[3], u"\xA3" + book[4]), end="")
```

Figure 7: List Example.

The main module: main.py includes many lists. One of them being books_List which is an empty list and another being columns which includes the headings required for a two-dimensional list. After the list has been created the code will open a text file named books, split it wherever there is a ‘,’ and append the split data into the books_List. Line.split is used to split a string into a list and. append is used to add items to the end of an existing list.

- Dictionaries: A non-primitive data type consisting of key-value pairs. Key used to identify an item, and value used to store a value of the item. Key separated by the value using colons, items separated by commas, which is enclosed within {} curly brackets. Unlike lists, dictionaries are not mutable but the value can be of any type.

Example:

```
>>>
>>> dict_1 = {"First Name":"Rabina" , "Last Name":"Shrestha" , "Age":"18"}
>>> print(dict_1)
{'First Name': 'Rabina', 'Last Name': 'Shrestha', 'Age': '18'}
>>>
>>> |
```

Figure 8: Dictionaries Example.

- Sets: An unordered collection of distinct elements is known as Sets. It is mutable but no two values can be same. It is written within [] square brackets.

Example:

```
>>>
>>> set_1 = {2, 5, 7, 10, 2}
>>> print(set_1)
{2, 10, 5, 7}
>>>
>>> |
```

Figure 9: Sets Example.

- Tuples: Another standard sequence data type is a tuple. Unlike lists, tuples are immutable. Once you define then you cannot add, delete, or edit any element inside.

Example:

```
>>>
>>> tup_1 = (7, "Hi", 0.57)
>>> print(tup_1)
(7, 'Hi', 0.57)
>>>
>>> |
```

Figure 10: Tuples Example.

3. Program.

The program consists of five modules i.e., main.py, booksFile.py, borrowBook.py, returnBook.py and dateTime.py. Each module consists of different functions which is responsible for the working of the program. The data of the books is store in books.txt file.

main.py

This module comprises of four modules which handles the functioning aspect of the system. Without this, the system may not be able to work.

The function kept in this module is root() which is the main function of the project. It prints out the welcome page and asks the user to choose from 1/2/3/4 i.e., to view the list of books, to borrow a book, to return a book, and to exit respectively.

```
==============================================================================
                Welcome to Library Management System.\(^0^)/
------------------------------------------------------------------------------

 To view the list of Books: PRESS 1
 To borrow a Book         : PRESS 2
 To return a Book         : PRESS 3
 To Exit                  : PRESS 4

------------------------------------------------------------------------------
 Select your choice:
```

Figure 11: Display.

The welcome page can be seen in figure 11. The user has to type in their choice from the above options.

'Try' and 'except' is used to run the code under the try statement, if in case it does not execute properly, the except code will run. For example: If the user inputs a string value when the data type assigned is int, the execute code will run and if the user inputs a

value i.e., anything except 1/2/3/4, then the else code which is kept before the execute code runs.

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

 To view the list of Books: PRESS 1
 To borrow a Book         : PRESS 2
 To return a Book         : PRESS 3
 To Exit                  : PRESS 4

--------------------------------------------------------------------------------
 Select your choice: e

--------------------------------------------------------------------------------
 Please enter 1/2/3/4.
--------------------------------------------------------------------------------
```

Figure 12: Try and except.

The above image shows an instance where the user has chosen e accidentally instead of the given options.

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

 To view the list of Books: PRESS 1
 To borrow a Book         : PRESS 2
 To return a Book         : PRESS 3
 To Exit                  : PRESS 4

--------------------------------------------------------------------------------
 Select your choice: 5
--------------------------------------------------------------------------------
 Invalid choice. Try again.
--------------------------------------------------------------------------------
```

Figure 13: Wrong Input Result.

Here, since the user has chosen 5 which is not an option, the ‘Invalid choice. Try again message pops up.

Option 1.

If the user chooses 1 then, the book.txt file is read and displayed using split, and appending it to the existing empty list named books_List. The headings list which is columns is also printed before the books_List to separate book ID, Name, Author Name, Stock available and the cost of the book.

```
==============================================================================
                Welcome to Library Management System.\(^0^)/
------------------------------------------------------------------------------

 To view the list of Books: PRESS 1
 To borrow a Book         : PRESS 2
 To return a Book         : PRESS 3
 To Exit                  : PRESS 4

------------------------------------------------------------------------------
 Select your choice: 1
------------------------------------------------------------------------------

 BookID  Book Name                      AuthorName          Stock        Cost

 01      I Want to Eat Your Pancreas    Yoru Sumino         50        £ 11.99
 02      Wolf Children: Ame & Yuki      Mamoru Hosoda       50        £ 14.99
 03      Solo Leveling                  Chugong             50        £ 11.99
 04      Toradora                       Yuyuko Takemiya     50        £ 10.99
 05      Naruto                         Masashi Kishimoto   50        £ 10.99
 06      Black Butler                   Yana Toboso         50        £ 09.99
 07      Assassination Classroom        Yusei Matsui        50        £ 07.99
 08      Tokyo Ghoul                    Sui Ishida          50        £ 08.99
 09      City of Bones                  Cassandra Clare     50        £ 08.99
 10      The Haunting of Hill House     Shirley Jackson     50        £ 09.99

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

Figure 14: 1: To view the list of books.

Option 2.

If the user chooses 2 then, similar process as above is followed with an exception of only the selected book ID's details is shown. If the user tries to add a non-existing book, 'Book not Found' is printed and the user is asked to enter the book ID again. Once a book is added, 'Do you want to add another book?' option is shown in which if you type y, the process is repeated but if the input is n, then a borrow receipt is created.

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

 To view the list of Books: PRESS 1
 To borrow a Book         : PRESS 2
 To return a Book         : PRESS 3
 To Exit                  : PRESS 4

--------------------------------------------------------------------------------
 Select your choice: 2
--------------------------------------------------------------------------------

 Enter your full name: Mira Shrestha

 Enter the Book ID: 22

--------------------------------------------------------------------------------
 Book not Found
--------------------------------------------------------------------------------

 Enter the Book ID: e

--------------------------------------------------------------------------------
 Book not Found
--------------------------------------------------------------------------------

 Enter the Book ID: 08
 Do you want to add another book? (y/n): y

 Enter another Book ID: 08

--------------------------------------------------------------------------------
 You have already borrowed this book.
--------------------------------------------------------------------------------

 Do you want to add another book? (y/n): y

 Enter another Book ID: 07
 Do you want to add another book? (y/n): n
```

Figure 15: Invalid Inputs and Borrowing the same book twice.

In the above shown figure, invalid inputs are addressed along with a user trying to borrow the same book twice. ‘Book not found’ and ‘You have already borrowed this book’ message is printed respectively.

```
borrow_Receipt/Mira Shrestha.txt

                    Library Management System

 Date: 2021-09-10                                Time: 00:51:20.043553

 Borrower Name: Mira Shrestha

 SNo. BookID  BookName                      AuthorsName          Cost
 1    07      Assassination Classroom       Yusei Matsui       £ 07.99

 2    08      Tokyo Ghoul                   Sui Ishida         £ 08.99

 Last date of Return: 2021-09-20
 Late fees per day after 10 days will be: £0.5

                                                Sub-Total: £16.98
```

Figure 16: Borrow Receipt is Printed.

Mira Shrestha - Notepad

File Edit Format View Help

```
=========================================================================

                      Library Management System

_________________________________________________________________________
 Date: 2021-09-10                                 Time: 00:51:20.022558
_________________________________________________________________________

 Borrower Name: Mira Shrestha

-------------------------------------------------------------------------
 SNo.  BookID  BookName                      AuthorsName           Cost
-------------------------------------------------------------------------
 1     07      Assassination Classroom       Yusei Matsui       £ 07.99
 2     08      Tokyo Ghoul                   Sui Ishida         £ 08.99
-------------------------------------------------------------------------

 Last date of Return: 2021-09-20
 Late fees per day after 10 days will be: £0.5


_________________________________________________________________________

                                                Sub-Total: £16.98

=========================================================================
```

Ln 1, Col 1 | 100% | Windows (CRLF) | ANSI

Figure 17: Borrow Receipt Text File.

Once the user has confirmed all the necessary details, the borrow receipt is created and printed in the above shown format.

Choice 3.

If the user chooses 3 then, similar process as option 2 is followed with an exception of if the user enters a name or book ID which is not recorded in the borrow_Record then the user is asked to enter the name or book ID again assuming that there has been a mistake. Once a book is returned, 'Do you want to return another book?' option is shown in which if you type y, the process is repeated but if the input is n, then a return receipt is created along with the total amount including the late fees if the borrowed duration exceeds 10 days.

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

 To view the list of Books: PRESS 1
 To borrow a Book         : PRESS 2
 To return a Book         : PRESS 3
 To Exit                  : PRESS 4

--------------------------------------------------------------------------------
 Select your choice: 3
--------------------------------------------------------------------------------

 Enter your full name: Mira Shresthaa

--------------------------------------------------------------------------------
 Cannot find borrow record with the name: Mira Shresthaa
--------------------------------------------------------------------------------

 Enter your full name: Mira Shrestha

 Enter the Book ID: 10

--------------------------------------------------------------------------------
 Book not Found
--------------------------------------------------------------------------------

 Enter the Book ID: 08
 Enter the borrowed duration (in days): 10
 Do you want to return another book? (y/n): y

 Enter another Book ID: 07
 Enter the borrowed duration (in days): 15
 Do you want to return another book? (y/n): n
```

Figure 18: Name and Book ID Error.

In the above shown figure,

When a user tries to enter an unregistered name, ‘Cannot find borrow record with the name: unregistered name’ is printed. Whereas, when an invalid bookID and unregistered bookID is typed in then it shows, ‘Book not Found.’.

```
---------------------------------------------------------------------------
 return_Receipt/Mira Shrestha.txt
 ==========================================================================

                        Library Management System

 ___________________________________________________________________________

  Date: 2021-09-10                                  Time: 00:52:23.328161

 ___________________________________________________________________________

  Returned By: Mira Shrestha

 --------------------------------------------------------------------------
  SNo. BookID  BookName                      AuthorsName           Cost
 --------------------------------------------------------------------------
  1    08      Tokyo Ghoul                   Sui Ishida         £ 08.99

  2    07      Assassination Classroom       Yusei Matsui       £ 07.99

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

  Borrowed Duration in days: 15
  Late fees per day after 10 days will be: £0.5

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

                                                  Sub-Total: £16.98
                                                  Late-Fees: £2.5
                                                  Total    : £19.48

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

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

Figure 19: Return Receipt is Printed.

Mira Shrestha - Notepad

File Edit Format View Help

```
=====================================================================

                      Library Management System

_____________________________________________________________________
 Date: 2021-09-10                                Time: 00:52:23.303600
_____________________________________________________________________

 Returned By: Mira Shrestha

---------------------------------------------------------------------
 SNo.  BookID  BookName                      AuthorsName          Cost
---------------------------------------------------------------------
 1     08      Tokyo Ghoul                   Sui Ishida        £ 08.99
 2     07      Assassination Classroom       Yusei Matsui      £ 07.99
---------------------------------------------------------------------

 Borrowed Duration in days: 15
 Late fees per day after 10 days will be: £0.5


_____________________________________________________________________

                                            Sub-Total: £16.98
                                            Late-Fees: £2.5
                                            Total    : £19.48

=====================================================================
```

Ln 1, Col 1 | 100% | Windows (CRLF) | ANSI

Figure 20: Return Receipt Text File.

Once the user has confirmed all the necessary details, the return receipt is created and printed in the above shown format.

Choice 4.

If the user chooses 4 then a ‘Thank You and Have a Nice Day!’ is printed and the program closes shortly.

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

 To view the list of Books: PRESS 1
 To borrow a Book         : PRESS 2
 To return a Book         : PRESS 3
 To Exit                  : PRESS 4

-------------------------------------------------------------------------------
 Select your choice: 4
-------------------------------------------------------------------------------
                        Thank You and Have a Nice Day!
===============================================================================
>>>
```

Figure 21: Exit.

The last function, if __name__ == “main” prevents the code being run when the module is imported and in this block in case the borrow_Receipt or return_Receipt folder does not exist it creates the receipt destination.

main.py codes:

main.py - C:\Users\Rabina\Desktop\main funda\main.py (3.9.6)

File Edit Format Run Options Window Help

```python
# Importing all the modules.
import os
import booksFile
import borrowBook
import returnBook
import dateTime

LATE_FEES_PER_DAY = 0.5

def root():
    """ Main function of the project."""
    print()
    print("=" *78)
    print("\t\tWelcome to Library Management System.\(^0^)/" + u"\U0001F4DA")

    while True:
        ''' Shows the services provided by the system.
            Asks the user to select a choice.'''
        print("-" *78 + "\n")
        print(" To view the list of Books: PRESS 1")
        print(" To borrow a Book         : PRESS 2")
        print(" To return a Book         : PRESS 3")
        print(" To Exit                  : PRESS 4")
        print("\n" + "-" *78)
        # Exception handling is used.
        try:
            choice = int(input(" Select your choice: "))
            print("-" *78)
            if choice == 1:
                print("_" *78)
                # View logic.
                books_List = []
                columns = [" BookID" , " Book Name","AuthorName", "Stock" , "  Cost"]

                with open("books.txt", 'r') as file:
                    for line in file:
                        book = line.split(',')
                        books_List.append(book)

                print("{:<5s} {:<30s} {:<20s} {:<10s} {}".format(
                    columns[0], columns[1], columns[2], columns[3], columns[4]))
                print("_" *78 +"\n")

                for book in books_List:
                    print("{:<5s} {:<30s} {:<20s}  {:<10s}{}".format(
                        " "+book[0], " "+book[1],book[2], book[3], u"\xA3" + book[4]), end="")
```

```python
            print("_" *78 +"\n")

        elif choice == 2:
            # Borrow logic.
            add_to_receipt = []
            person_Name = input("\n Enter your full name: ")
            while True:
                book_ID = input("\n"+" Enter the Book ID: ")
                with open("books.txt","r") as file:
                    exists = False
                    for line in file:
                        book = line.split(",")
                        if book[0] == book_ID:
                            exists = True
                            break
                    if exists:
                        break
                    else:
                        print("\n" + "-" *78)
                        print(" Book not Found")
                        print("-" *78)

            if borrowBook.borrow_Book(person_Name, book_ID):
                add_to_receipt.append(book_ID)
            while True:
                another = input(
                    " Do you want to add another book? (y/n): ")
                if another.lower() == "n":
                    break
                elif another.lower() == "y":
                    next_Book = input("\n Enter another Book ID: ")
                    if borrowBook.borrow_Book(person_Name, next_Book):
                        add_to_receipt.append(next_Book)
                else:
                    print(" Enter a valid response!")

            data = []
            sub_total = 0
            with open("Books.txt", "r") as file:
                for line in file:
                    book = line.split(",")
                    for each_id in add_to_receipt:
                        if book[0] == each_id:
                            data.append(
                                [book[0], book[1], book[2], book[4]])
                            sub_total += float(book[4])
            for book in data:
                book[-1] = str(book[-1]).replace("\n", "")

            # Creating borrow receipt in txt and shell.
            with open(f"borrow_Receipt/{person_Name}.txt", "w+") as file:
                file.write("="*72 + "\n\n")
                file.write("                        Library Management System" + "\n")
                file.write("_"*72 + "\n")
                file.write(" Date: "+dateTime.getDate()+"                                    Time: " +dateTime.getTime()+"\n")
                file.write("_"*72 + "\n")
                file.write(f"\n Borrower Name: {person_Name}\n\n")
                file.write("-"*72 + "\n")
                file.write("{:<6s} {:<7s} {:<29s} {:<20s} {}".format(
                    " SNo.", "BookID","BookName", "AuthorsName", " Cost" +"\n"))
                file.write("-"*72 + "\n")
                SNo = 0
                for book in data:
                    SNo = SNo + 1
                    file.write(
                        "{:<1s}{:<5d} {:<5s} {:<30s} {:<20s}{}\n".format("",SNo, book[0], book[1], book[2],"£"+book[3]))
                file.write("-"*72 + "\n")
                file.write("\n Last date of Return: "+dateTime.getReturnDate()+"\n")
                file.write(
                    f" Late fees per day after 10 days will be: £{LATE_FEES_PER_DAY}\n\n")
                file.write("_"*72 + "\n\n")
                file.write(f"\t\t\t\t\t\tSub-Total: £{sub_total} \n\n")
                file.write("="*72)

                print("\n  "+"-"*74)
                print(f"   borrow_Receipt/{person_Name}.txt")
                print("   "+"="*72)
                print("\n                        Library Management System")
                print("   "+"_"*72)
                print("\n   "+" Date: "+dateTime.getDate()+"                                    Time: " +dateTime.getTime())
                print("   "+"_"*72 )
                print(f"\n    Borrower Name: {person_Name}\n")
                print("   "+"-"*72)
                print("{:<6s} {:<7s} {:<29s} {:<20s} {}".format(
                        "    SNo.", "BookID","BookName", "AuthorsName", " Cost"))
                print("   "+"-"*72)
                SNo = 0
                for book in data:
                    SNo = SNo + 1
                    print(
                            "{:<1s}   {:<5d}{:<5s} {:<30s} {:<20s}{}\n".format("",SNo, book[0], book[1], book[2],"£"+book[3]))
                print("   "+"-"*72)
                print("\n   "+" Last date of Return: "+dateTime.getReturnDate())
                print(
```

```
                        f"    Late fees per day after 10 days will be: £{LATE_FEES_PER_DAY}\n")
                print("   "+"-"*72 + "\n")
                print(f"\t\t\t\t\t\t   Sub-Total: £{sub_total}\n")
                print("   "+"="*72 + "\n")
                print("  "+"-"*74 + "\n")

        elif choice == 3:
            # Return logic.
            data = []
            sub_total = 0
            total_late_fees = 0
            add_to_receipt = []

            # Checks whether name exists.
            while True:
                person_Name = input("\n Enter your full name: ")
                exists = False
                with open("borrow_Record.txt","r") as file:
                    for line in file:
                        lend = line.split(",")

                        if lend[1] == person_Name :
                            exists = True
                            break
                if exists:
                    break
                else:
                    print("\n" + "-" *78)
                    print(f" Cannot find borrow record with the name: {person_Name}")
                    print("-" *78)

            # Checks whether book ID exists.
            while True:
                book_ID = input("\n"+" Enter the Book ID: ")
                with open("borrow_Record.txt","r") as file:
                    exists = False
                    for line in file:
                        book = line.split(",")
                        if lend[0] == book_ID:
                            exists = True
                            break
                    if exists:
                        break
                    else:
                        print("\n" + "-" *78)
                        print(" Book not Found")
                        print("-" *78)
```

```
            if returnBook.return_Book(person_Name, book_ID):
                with open("books.txt", "r") as file:
                    for line in file:
                        book = line.split(",")
                        if book[0] == book_ID:
                            with open("borrow_Record.txt", "r") as borrow_Bundle:
                                lines = borrow_Bundle.readlines()
                                new_Lines = []
                                for line in lines:
                                    lend = line.split(",")
                                    lend[-1] = str(lend[-1]).replace("\n", "")
                                    if lend[0] == book_ID and lend[1].lower() == person_Name.lower():
                                        data.append(
                                            [book[0], book[1], book[2], lend[3], book[4]])
                                        sub_total += float(book[4])
                                        if int(lend[3]) > 10:
                                            total_late_fees += (int(lend[3]) - 10) * \
                                                LATE_FEES_PER_DAY
                                    else:
                                        new_Lines.append(line)
                                with open("borrow_Record.txt", "w") as file:
                                    file.writelines(new_Lines)

            while True:
                another = input(" Do you want to return another book? (y/n): ")
                if another.lower() == "n":
                    break
                elif another.lower() == "y":
                    next_Book = input("\n Enter another Book ID: ")
                    if returnBook.return_Book(person_Name, next_Book):
                        with open("books.txt", "r") as file:
                            for line in file:
                                book = line.split(",")
                                if book[0] == next_Book:
                                    with open("borrow_Record.txt", "r") as borrow_Bundle:
                                        lines = borrow_Bundle.readlines()
                                        new_Lines = []
                                        for line in lines:
                                            lend = line.split(",")
                                            lend[-1] = str(lend[-1]).replace("\n", "")
                                            if lend[0] == next_Book and lend[1].lower() == person_Name.lower():
                                                data.append(
                                                    [book[0], book[1], book[2], lend[3], book[4]])
                                                sub_total += float(book[4])
                                                if int(lend[3]) > 10:
                                                    total_late_fees += (int(lend[3]) - 10) * \
                                                        LATE_FEES_PER_DAY
```

```python
                                        else:
                                            new_Lines.append(line)
                                    with open("borrow_Record.txt", "w") as file:
                                        file.writelines(new_Lines)

                    else:
                        print(" Enter a valid response!")

                for book in data:
                    book[-1] = str(book[-1]).replace("\n", "")

                # Creating return receipt in txt and shell.
                with open(f"return_Receipt/{person_Name}.txt", "w+") as file:
                    file.write("="*72 + "\n\n")
                    file.write("                    Library Management System" + "\n")
                    file.write("_"*72 + "\n")
                    file.write(" Date: "+dateTime.getDate()+"                                Time: " +dateTime.getTime()+"\n")
                    file.write("_"*72 + "\n")
                    file.write(f"\n Returned By: {person_Name}\n\n")
                    file.write("-"*72 + "\n")
                    file.write("{:<6s} {:<7s} {:<29s} {:<20s} {}".format(
                        " SNo.", "BookID","BookName", "AuthorsName", " Cost" +"\n"))
                    file.write("-"*72 + "\n")
                    SNo = 0
                    for book in data:
                        SNo = SNo + 1
                        file.write(
                            "{:<1s}{:<5d} {:<5s} {:<30s} {:<20s}{}\n".format("",SNo, book[0], book[1], book[2],"£"+book[4]))
                    file.write("-"*72 + "\n")
                    file.write("\n Borrowed Duration in days: "+book[3])
                    file.write(
                        f"\n Late fees per day after 10 days will be: £{LATE_FEES_PER_DAY}\n\n")
                    file.write("_"*72 + "\n\n")
                    file.write(f"\t\t\t\t\t\tSub-Total: £{sub_total}\n")
                    file.write(f"\t\t\t\t\t\tLate-Fees: £{total_late_fees}\n")
                    total_fees = sub_total + total_late_fees
                    two_float = "{:.2f}".format(total_fees)
                    file.write(f"\t\t\t\t\t\tTotal    : £{total_fees}\n\n")
                    file.write("="*72)

                    print("\n  "+"-"*74)
                    print(f"   return_Receipt/{person_Name}.txt")
                    print("   "+"="*72)
                    print("\n                    Library Management System")
                    print("   "+"_"*72)
                    print("\n   "+" Date: "+dateTime.getDate()+"                                Time: " +dateTime.getTime())
                    print("   "+"_"*72)
                    print(f"\n   Returned By: {person_Name}\n")
                    print("   "+"-"*72)
                    print("{:<6s} {:<7s} {:<29s} {:<20s} {}".format(
                            "    SNo.", "BookID","BookName", "AuthorsName", " Cost"))
                    print("   "+"-"*72)
                    SNo = 0
                    for book in data:
                        SNo = SNo + 1
                        print(
                                "{:<1s}   {:<5d}{:<5s} {:<30s} {:<20s}{}\n".format("",SNo, book[0], book[1], book[2],"£"+book[4]))
                    print("   "+"-"*72)
                    print("\n    Borrowed Duration in days: "+book[3])
                    print(
                            f"    Late fees per day after 10 days will be: £{LATE_FEES_PER_DAY}\n")
                    print("   "+"-"*72 + "\n")
                    print(f"\t\t\t\t\t\t   Sub-Total: £{sub_total}")
                    print(f"\t\t\t\t\t\t   Late-Fees: £{total_late_fees}")
                    total_fees = sub_total + total_late_fees
                    two_float = "{:.2f}".format(total_fees)
                    print(f"\t\t\t\t\t\t   Total    : £{total_fees}\n")
                    print("   "+"="*72 + "\n")
                    print("  "+"-"*74 + "\n")

            elif choice == 4:
                # Exit logic.
                print("\t\t\tThank You and Have a Nice Day!")
                print("="*78)
                break

            else:
                print(" Invalid choice. Try again.")

        except ValueError:
            print("\n" + "-" *78)
            print(" Please enter 1/2/3/4.")
            print("-" *78 + "\n")

if __name__ == "__main__":
    ''' Creates receipt destination if it does not exist.'''
    if not os.path.exists("borrow_Receipt"):
        os.makedirs("borrow_Receipt")
    if not os.path.exists("return_Receipt"):
        os.makedirs("return_Receipt")
    root()
```

Figure 22: main.py Codes.

bookFile.py

Here, a class named Book() is created to represent the books. The __init__ method is used for the bookID, Name, AuthorsName, Stock, and Cost which is stored in an instance. By using the 'self' keyword, the attributes and methods can be accessed. The check_Borrow function checks the borrow_Record and returns a Boolean value 'True' if the book ID matches the user input.

The update_Stock function automatically updates the stock. The borrow function reduces the stock when a customer borrower a book. The return_Back function restocks when a customer returns the book.

BookID	Book Name	AuthorName	Stock	Cost
01	I Want to Eat Your Pancreas	Yoru Sumino	50	£ 11.99
02	Wolf Children: Ame & Yuki	Mamoru Hosoda	50	£ 14.99
03	Solo Leveling	Chugong	50	£ 11.99
04	Toradora	Yuyuko Takemiya	50	£ 10.99
05	Naruto	Masashi Kishimoto	50	£ 10.99
06	Black Butler	Yana Toboso	50	£ 09.99
07	Assassination Classroom	Yusei Matsui	49	£ 07.99
08	Tokyo Ghoul	Sui Ishida	49	£ 08.99
09	City of Bones	Cassandra Clare	50	£ 08.99
10	The Haunting of Hill House	Shirley Jackson	50	£ 09.99

Figure 23: Stock when a book is borrowed.

BookID	Book Name	AuthorName	Stock	Cost
01	I Want to Eat Your Pancreas	Yoru Sumino	50	£ 11.99
02	Wolf Children: Ame & Yuki	Mamoru Hosoda	50	£ 14.99
03	Solo Leveling	Chugong	50	£ 11.99
04	Toradora	Yuyuko Takemiya	50	£ 10.99
05	Naruto	Masashi Kishimoto	50	£ 10.99
06	Black Butler	Yana Toboso	50	£ 09.99
07	Assassination Classroom	Yusei Matsui	50	£ 07.99
08	Tokyo Ghoul	Sui Ishida	50	£ 08.99
09	City of Bones	Cassandra Clare	50	£ 08.99
10	The Haunting of Hill House	Shirley Jackson	50	£ 09.99

Figure 24: Stock when a book is returned.

The customer is asked to enter the borrowed duration and ‘try’ and ‘except’ is used here to ensure that the customer inputs an integer data type.

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

 To view the list of Books: PRESS 1
 To borrow a Book         : PRESS 2
 To return a Book         : PRESS 3
 To Exit                  : PRESS 4

--------------------------------------------------------------------------------
 Select your choice: 3
--------------------------------------------------------------------------------

 Enter your full name: Duration Except

 Enter the Book ID: 07
 Enter the borrowed duration (in days): e

--------------------------------------------------------------------------------
 Please input the borrowed duration in numeric value.
--------------------------------------------------------------------------------

 Enter the borrowed duration (in days): 10
 Do you want to return another book? (y/n): n
```

Figure 25: Duration Try, Except.

If any value except for integer is entered in ‘Enter the borrowed duration’ field, ‘Please input the borrowed duration in numeric value’ is shown. The above image portrays this very attempt.

bookFile.py codes:

booksFile.py - C:\Users\Rabina\Desktop\main funda\booksFile.py (3.9.6)

File Edit Format Run Options Window Help

```python
# Importing the dateTime module.
import dateTime

class Book():
    """ A class used to represent Books."""
    def __init__(self, instance):
        self.book_ID = instance[0]
        self.book_Name = instance[1]
        self.book_Author = instance[2]
        self.book_Stock = instance[3]
        self.book_Cost = instance[4]

    def check_Borrow(self, person_Name):
        ''' Checks borrow record and returns a boolean value.'''
        with open('borrow_Record.txt', 'r') as file:
            for line in file:
                lend = line.split(",")
                if lend[0] == self.book_ID and lend[1].lower() == person_Name.lower():
                    return True

    def update_Stock(self):
        ''' Updates the stocks.'''
        with open("books.txt", "r") as file:
            lines = file.readlines()
            new_Lines = []
            for line in lines:
                book = line.split(",")
                if book[0] == self.book_ID:
                    updated_Line = f"{self.book_ID},{self.book_Name},{self.book_Author},{self.book_Stock},{self.book_Cost}"
                    new_Lines.append(updated_Line)
                else:
                    new_Lines.append(line)
            with open("books.txt", "w") as file:
                file.writelines(new_Lines)

    def borrow(self, person_Name):
        ''' Reduces the stock.'''
        with open("borrow_Record.txt", "a") as file:
            date = dateTime.getDate()
            time = dateTime.getTime()
            file.write(f"{self.book_ID},{person_Name},{date}{time}\n")
        self.book_Stock = int(self.book_Stock) - 1
        self.update_Stock()

    def return_Back(self, person_Name):
        ''' Exception handling is used here.
            It increases the stock and deletes the records when returned.'''
        while True:
            try:
                duration = int(input(" Enter the borrowed duration (in days): "))
                with open("borrow_Record.txt", "r") as file:
                    lines = file.readlines()
                    new_Lines = []
                    with open("borrow_Record.txt", "w") as file:
                        for line in lines:
                            lend = line.split(",")
                            lend[-1] = str(lend[-1]).replace("\n", "")
                            if lend[0] == self.book_ID and lend[1] == person_Name:
                                new_Lines.append(
                                    f"{lend[0]},{lend[1]},{lend[2]},{duration}\n")
                            else:
                                new_Lines.append(line)
                        file.writelines(new_Lines)
                self.book_Stock = int(self.book_Stock) + 1
                self.update_Stock()
                break
            except ValueError:
                print("\n" + "-" *78)
                print(" Please input the borrowed duration in numeric value.")
                print("-" *78 + "\n")
```

Figure 26: bookFile codes.

borrowBook.py

The borrow_Book function validates the borrowing user input. It opens the books.txt file and checks the negatives of the users' commands i.e., if the book the customer is trying to borrow is out of stock or if they have already borrowed the book or if the book, they are trying to borrow has incorrect bookID.

borrowBook.py codes:

borrowBook.py - C:\Users\Rabina\Desktop\main funda\borrowBook.py (3.9.6)

File Edit Format Run Options Window Help

```python
# Importing booksFile module.
import booksFile

def borrow_Book(person_Name, book_ID):
    ''' Validates borrowing user input.'''
    with open('books.txt', 'r') as file:
        for line in file:
            book = line.split(",")
            if book[0] == book_ID:
                book_ins = booksFile.Book(book)
                if not book_ins.check_Borrow(person_Name):
                    if int(book_ins.book_Stock) < 1:
                        print("-" *78)
                        print(" Book out of Stock.")
                        print("-" *78+ "\n")
                        return False
                    book_ins.borrow(person_Name)
                    return True
                else:
                    print("\n" +"-" *78)
                    print(" You have already borrowed this book.")
                    print("-" *78+ "\n")
                    return False
        print("\n"+"-" *78)
        print(" Book not found.")
        print("-" *78 + "\n")

```

Ln: 1 Col: 0

Figure 27: borrow.py Codes.

returnBook.py

The return_Book function validates the returning user input. It opens the books.txt file and checks the negatives of the users' commands i.e., if the bookID or the name of the borrower is incorrect then, 'Borrow record not found' is shown.

returnBook.py codes:

returnBook.py - C:\Users\Rabina\Desktop\main funda\returnBook.py (3.9.6)

File Edit Format Run Options Window Help

```python
# Importing booksFile module.
import booksFile

def return_Book(person_Name, book_ID):
    ''' Validates returning user input.'''
    with open("books.txt", "r") as file:
        for line in file:
            book = line.split(",")
            if book[0] == book_ID:
                book_ins = booksFile.Book(book)
                if book_ins.check_Borrow(person_Name):
                    book_ins.return_Back(person_Name)
                    return True
                else:
                    print("\n" +"-" *78)
                    print(" Borrow record not found.")
                    print("-" *78 + "\n")
                    return False
        print("\n" +"-" *78)
        print(" Book not found.")
        print("-" *78 + "\n")
```

Ln: 1 Col: 0

Figure 28: returnBook.py Codes.

dateTime.py

The getDate function imports the date from datetime and stores only the date in number format. The getTime function stores the time from datetime, and the getReturnDate function stores the timedelta from datetime which shows the last date of return in the transaction by calculating the number of days starting from the issued date.

dateTime.py Codes.

dateTime.py - C:\Users\Rabina\Desktop\main funda\dateTime.py (3.9.6)

File Edit Format Run Options Window Help

```python

def getDate():
    # Importing date from datetime.
    import datetime
    Date = datetime.datetime.now
    return str(Date().date())

def getTime():
    # Importing time from datetime.
    import datetime
    Time = datetime.datetime.now
    return str(Time().time())

def getReturnDate():
    # Importing timedelta from datetime.
    import datetime
    from datetime import date, timedelta
    BorrowDate = datetime.date.today()
    DeltaTime = datetime.timedelta(days=10)
    return str(BorrowDate + DeltaTime)

```

Ln: 21 Col: 0

Figure 29: dateTime.py Codes.

4. Testing.

4.1. Test 1: To show implementation of try, except.

Test No.	**1**
Objective:	To test the implementation of try, except.
Action:	Test 1.1 ➢ Open the main file or use the terminal to run the module. ➢ When the program asks you to select your choice input anything except 1,2,3, and 4. ➢ To test the implementation of try and except, “invalid input error” which is a string is entered. Test 1.2 ➢ The program asks you to enter the borrowed duration while returning a book. ➢ Instead of entering an integer number, input a string.
Expected Result:	An appropriate message should pop up asking you to choose between 1-4.
Actual Result:	An appropriate message pops up.
Conclusion:	The test was successful.

Table 1: Implementation of try, except.

```
C:\Windows\py.exe
==============================================================================
                Welcome to Library Management System.\(^0^)/
------------------------------------------------------------------------------

 To view the list of Books: PRESS 1
 To borrow a Book         : PRESS 2
 To return a Book         : PRESS 3
 To Exit                  : PRESS 4

------------------------------------------------------------------------------
 Select your choice: invalid input error

------------------------------------------------------------------------------
 Please enter 1/2/3/4.
------------------------------------------------------------------------------


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

 To view the list of Books: PRESS 1
 To borrow a Book         : PRESS 2
 To return a Book         : PRESS 3
 To Exit                  : PRESS 4

------------------------------------------------------------------------------
 Select your choice:
```

Figure 30: Try, Except 1.

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

 To view the list of Books: PRESS 1
 To borrow a Book         : PRESS 2
 To return a Book         : PRESS 3
 To Exit                  : PRESS 4

------------------------------------------------------------------------------
 Select your choice: 3
------------------------------------------------------------------------------

 Enter your full name: Rabina Shrestha

 Enter the Book ID: 05
 Enter the borrowed duration (in days): y

------------------------------------------------------------------------------
 Please input the borrowed duration in numeric value.
------------------------------------------------------------------------------

 Enter the borrowed duration (in days):
```

Figure 31: Try, Except 2.

4.2. Test 2: Selection borrow and return option.

Test No.	**2**
Objective:	To test the selection process of borrow and return.
Action:	Borrow ➢ Enter 2 for borrow and enter your full name. ➢ Write a negative value, and a non-existing value when the program asks you to input the Book ID. ➢ -01 is entered for the negative value and 11 is entered for a non-existing value. Return ➢ Enter 3 for return and enter your full name. ➢ Write a negative value, and a non-existing value when the program asks you to input the Book ID. ➢ -07 is entered for the negative value and 57 is entered for a non-existing value.
Expected Result:	An appropriate message should pop up.
Actual Result:	An appropriate message pops up.
Conclusion:	The test was successful.

Table 2: Selection borrow and return option.

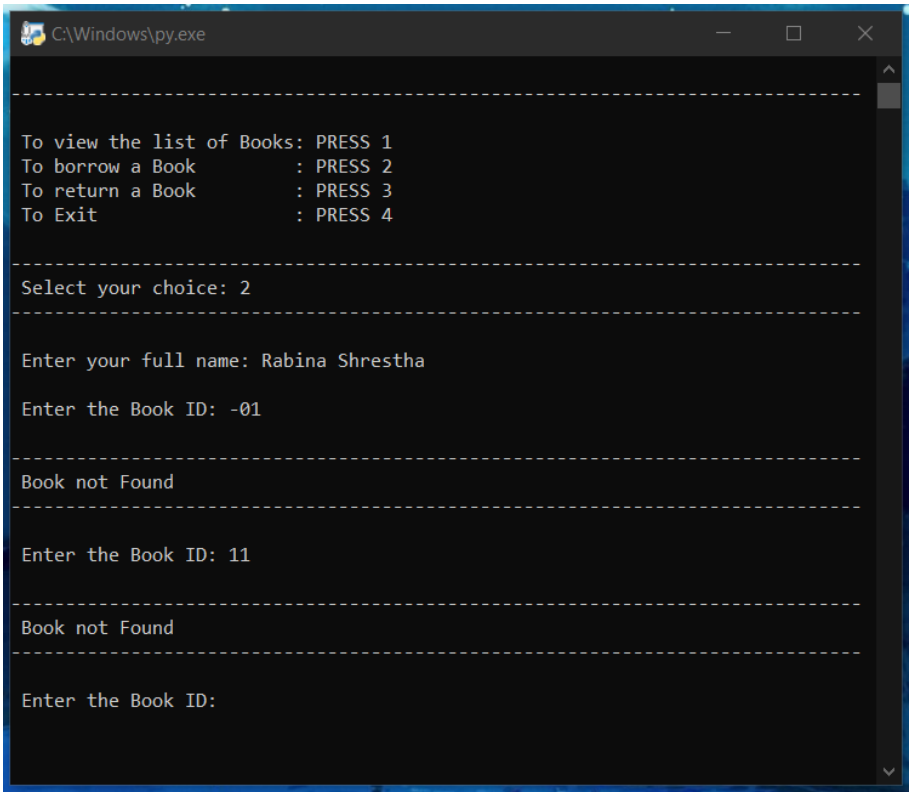


Figure 32: Providing Negative and Non-Existing Value as input in Borrow.

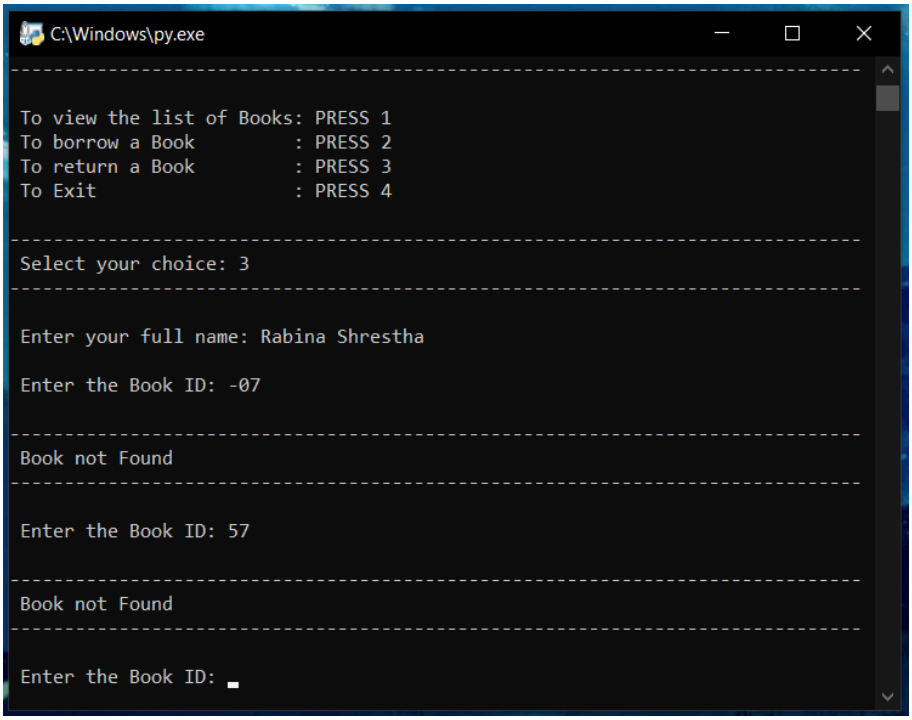


Figure 33: Providing Negative and Non-Existing Value as input in Return.

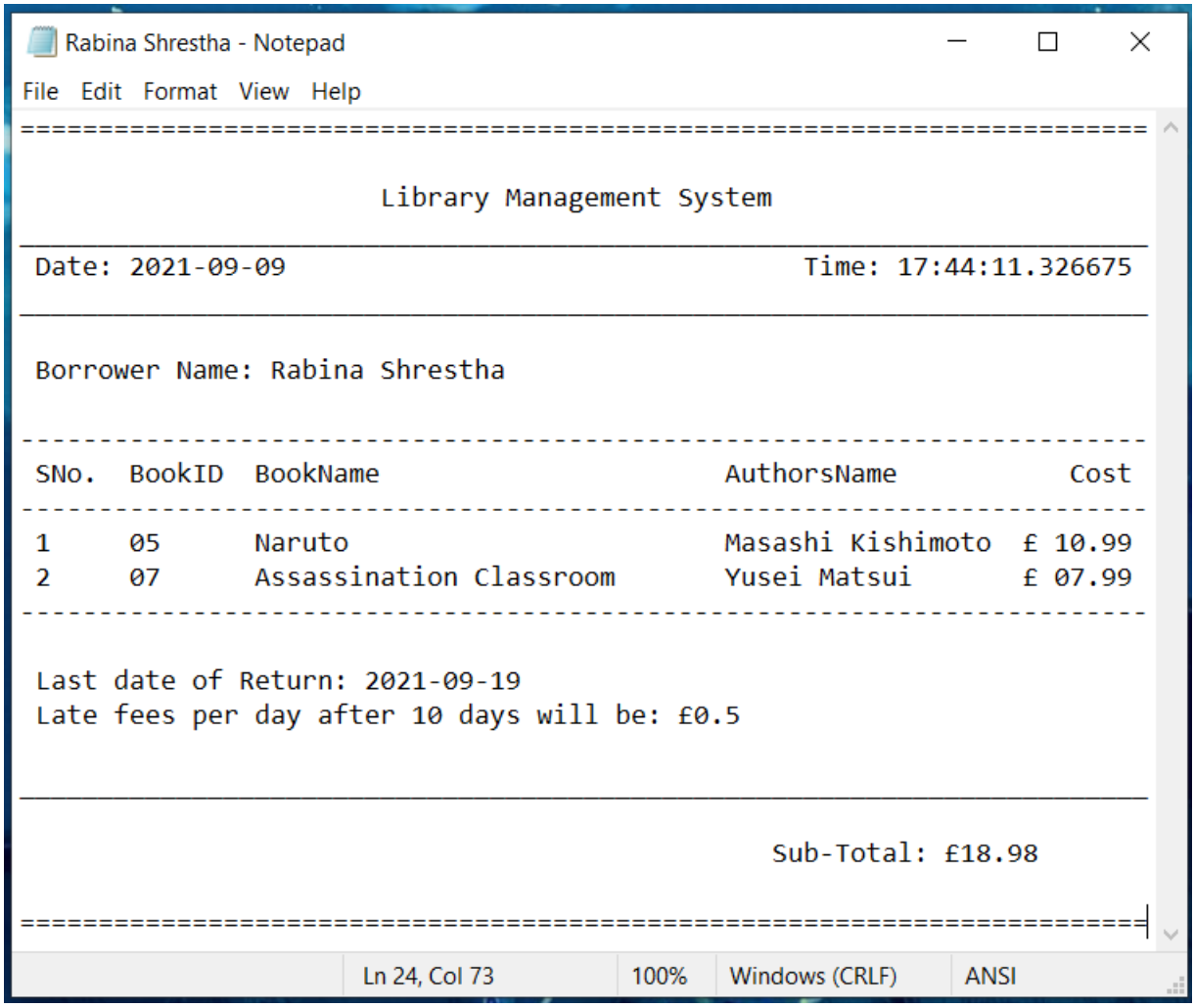
4.3. Test 3: File generation of borrow.

Test No.	**3**
Objective:	To test the file generation while borrowing a book.
Action:	Borrow ➢ Enter 2 for borrow. ➢ Enter your full name and the Book ID of the book you want to borrow. ➢ If you want to borrow some more books enter “y” when the program asks you if you want to add another book. ➢ Once “n” is entered indicating the user is done borrowing a book. ➢ A receipt should be printed in the shell along with an actual receipt being created in a text file in the borrow_Receipt folder.
Expected Result:	A receipt should be printed in the shell along with it a text file should be created in the borrow_Receipt folder.
Actual Result:	A receipt is printed in the shell along with it a text file that is created in the borrow_Receipt folder.
Conclusion:	The test was successful.

Table 3: File generation of borrow.

```
C:\Windows\py.exe

==============================================================================
                Welcome to Library Management System.\(^0^)/
------------------------------------------------------------------------------

 To view the list of Books: PRESS 1
 To borrow a Book         : PRESS 2
 To return a Book         : PRESS 3
 To Exit                  : PRESS 4

------------------------------------------------------------------------------
 Select your choice: 2
------------------------------------------------------------------------------

 Enter your full name: Rabina Shrestha

 Enter the Book ID: 05
 Do you want to add another book? (y/n): y

 Enter another Book ID: 07
 Do you want to add another book? (y/n): n

  ---------------------------------------------------------------------------
   borrow_Receipt/Rabina Shrestha.txt
   ==========================================================================

                          Library Management System

   ________________________________________________________________________

    Date: 2021-09-09                                  Time: 17:44:11.339220

   ________________________________________________________________________

    Borrower Name: Rabina Shrestha

    ---------------------------------------------------------------------
    SNo. BookID  BookName                      AuthorsName          Cost
   ---------------------------------------------------------------------
    1    05      Naruto                        Masashi Kishimoto  £ 10.99

    2    07      Assassination Classroom       Yusei Matsui       £ 07.99

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

    Last date of Return: 2021-09-19
    Late fees per day after 10 days will be: £0.5

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

                                                   Sub-Total: £18.98
```

Figure 34: Complete Borrow Process and Output in Shell.



```
Rabina Shrestha - Notepad
File  Edit  Format  View  Help
==========================================================================

                        Library Management System

Date: 2021-09-09                                  Time: 17:44:11.326675

Borrower Name: Rabina Shrestha

--------------------------------------------------------------------------
SNo.  BookID  BookName                      AuthorsName          Cost
--------------------------------------------------------------------------
1     05      Naruto                        Masashi Kishimoto  £ 10.99
2     07      Assassination Classroom       Yusei Matsui       £ 07.99
--------------------------------------------------------------------------

Last date of Return: 2021-09-19
Late fees per day after 10 days will be: £0.5

                                                Sub-Total: £18.98

==========================================================================
Ln 24, Col 73   100%   Windows (CRLF)   ANSI
```

Figure 35: Borrow Note in Text File.

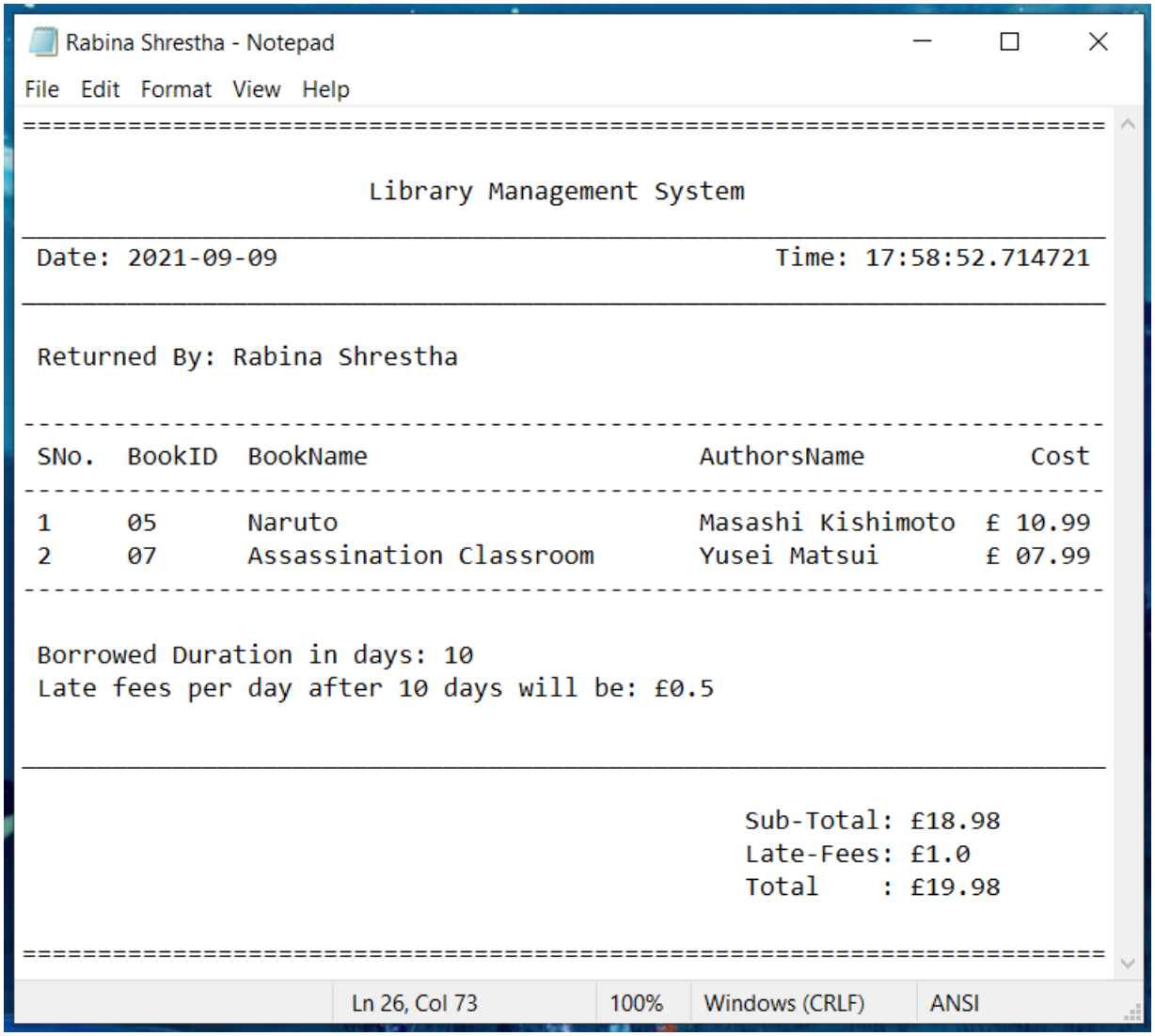
4.4. Test 4: File generation of return.

Test No.	**4**
Objective:	To test the file generation while borrowing a book.
Action:	Borrow ➢ Enter 3 for return. ➢ Enter your full name and the Book ID of the book you borrowed. ➢ If you want to return more than one book enter “y” when the program asks you if you want to return another book. ➢ Once “n” is entered indicating the user is done returning a book. ➢ A receipt should be printed in the shell along with an actual receipt being created in a text file in the return_Receipt folder.
Expected Result:	A receipt should be printed in the shell along with it a text file should be created in the return_Receipt folder.
Actual Result:	A receipt is printed in the shell along with it a text file that is created in the return_Receipt folder.
Conclusion:	The test was successful.

Table 4: File generation of return.

```
C:\Windows\py.exe

To view the list of Books: PRESS 1
To borrow a Book         : PRESS 2
To return a Book         : PRESS 3
To Exit                  : PRESS 4

--------------------------------------------------------------------------------
Select your choice: 3
--------------------------------------------------------------------------------

Enter your full name: Rabina Shrestha

Enter the Book ID: 05
Enter the borrowed duration (in days): 12
Do you want to return another book? (y/n): y

Enter another Book ID: 07
Enter the borrowed duration (in days): 10
Do you want to return another book? (y/n): n

 ---------------------------------------------------------------------------
  return_Receipt/Rabina Shrestha.txt
  ===========================================================================

                        Library Management System
  ___________________________________________________________________________

   Date: 2021-09-09                                Time: 17:58:52.714721
  ___________________________________________________________________________

   Returned By: Rabina Shrestha

  ---------------------------------------------------------------------------
   SNo. BookID  BookName                      AuthorsName           Cost
  ---------------------------------------------------------------------------
   1    05      Naruto                        Masashi Kishimoto  £ 10.99

   2    07      Assassination Classroom       Yusei Matsui       £ 07.99

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

   Borrowed Duration in days: 10
   Late fees per day after 10 days will be: £0.5

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

                                                   Sub-Total: £18.98
                                                   Late-Fees: £1.0
                                                   Total    : £19.98
```

Figure 36: Complete Return Process and Output in Shell.



```
Rabina Shrestha - Notepad
File  Edit  Format  View  Help
==========================================================================

                        Library Management System

Date: 2021-09-09                                  Time: 17:58:52.714721

Returned By: Rabina Shrestha

--------------------------------------------------------------------------
SNo.  BookID  BookName                        AuthorsName          Cost
--------------------------------------------------------------------------
1     05      Naruto                          Masashi Kishimoto  £ 10.99
2     07      Assassination Classroom         Yusei Matsui       £ 07.99
--------------------------------------------------------------------------

Borrowed Duration in days: 10
Late fees per day after 10 days will be: £0.5

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

                                              Sub-Total: £18.98
                                              Late-Fees: £1.0
                                              Total    : £19.98

==========================================================================
Ln 26, Col 73   100%   Windows (CRLF)   ANSI
```

Figure 37: Borrow Note in Text File.

4.5. Test 5: Show the update in stock.

Test No.	**5**
Objective:	To test if the stock updates automatically.
Action:	➢ First choose 1 to view the list of books and see the number of stock available. ➢ Then choose 2 to borrow a book. ➢ Enter your name, the book ID of the book you want to borrow. ➢ Once the receipt is created choose 1 to see if the number of stocks has decreased by 1. ➢ After you have checked, to return the book choose 3. ➢ Fill in all the necessary details, once the receipt is created, choose 1 to check if the stock of that particular book has increased by 1.
Expected Result:	The number of stocks should decrease by 1 when borrowed, and increase by 1 when returned.
Actual Result:	The number of stocks decreases by 1 when borrowed, and increases by 1 when returned.
Conclusion:	The test was successful.

Table 5: Update in Stock.

```
Select C:\Windows\py.exe

==============================================================================
                Welcome to Library Management System.\(^0^)/
------------------------------------------------------------------------------

To view the list of Books: PRESS 1
To borrow a Book         : PRESS 2
To return a Book         : PRESS 3
To Exit                  : PRESS 4

------------------------------------------------------------------------------
Select your choice: 1
------------------------------------------------------------------------------

BookID  Book Name                    AuthorName          Stock        Cost

01      I Want to Eat Your Pancreas  Yoru Sumino         50        £ 11.99
02      Wolf Children: Ame & Yuki    Mamoru Hosoda       50        £ 14.99
03      Solo Leveling                Chugong             50        £ 11.99
04      Toradora                     Yuyuko Takemiya     50        £ 10.99
05      Naruto                       Masashi Kishimoto   50        £ 10.99
06      Black Butler                 Yana Toboso         50        £ 09.99
07      Assassination Classroom      Yusei Matsui        50        £ 07.99
08      Tokyo Ghoul                  Sui Ishida          50        £ 08.99
09      City of Bones                Cassandra Clare     50        £ 08.99
10      The Haunting of Hill House   Shirley Jackson     50        £ 09.99

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

Figure 38: Stock Beforehand.

```
C:\Windows\py.exe

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

To view the list of Books: PRESS 1
To borrow a Book         : PRESS 2
To return a Book         : PRESS 3
To Exit                  : PRESS 4

--------------------------------------------------------------------------------
Select your choice: 2
--------------------------------------------------------------------------------

Enter your full name: Rize Shrestha

Enter the Book ID: 08
Do you want to add another book? (y/n): n

 ------------------------------------------------------------------------------
  borrow_Receipt/Rize Shrestha.txt
  ==============================================================================

                        Library Management System
  ------------------------------------------------------------------------------

   Date: 2021-09-09                                Time: 18:03:01.242735
  ------------------------------------------------------------------------------

   Borrower Name: Rize Shrestha

  ------------------------------------------------------------------------------
   SNo. BookID  BookName                      AuthorsName           Cost
  ------------------------------------------------------------------------------
   1    08      Tokyo Ghoul                   Sui Ishida         £ 08.99

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

   Last date of Return: 2021-09-19
   Late fees per day after 10 days will be: £0.5

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

                                                     Sub-Total: £8.99

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

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

Figure 39: Borrowing the Book.

```
Select C:\Windows\py.exe
--------------------------------------------------------------------------------

To view the list of Books: PRESS 1
To borrow a Book         : PRESS 2
To return a Book         : PRESS 3
To Exit                  : PRESS 4

--------------------------------------------------------------------------------
Select your choice: 1
--------------------------------------------------------------------------------

BookID  Book Name                      AuthorName          Stock        Cost

01      I Want to Eat Your Pancreas    Yoru Sumino         50        £ 11.99
02      Wolf Children: Ame & Yuki      Mamoru Hosoda       50        £ 14.99
03      Solo Leveling                  Chugong             50        £ 11.99
04      Toradora                       Yuyuko Takemiya     50        £ 10.99
05      Naruto                         Masashi Kishimoto   50        £ 10.99
06      Black Butler                   Yana Toboso         50        £ 09.99
07      Assassination Classroom        Yusei Matsui        50        £ 07.99
08      Tokyo Ghoul                    Sui Ishida          49        £ 08.99
09      City of Bones                  Cassandra Clare     50        £ 08.99
10      The Haunting of Hill House     Shirley Jackson     50        £ 09.99

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

To view the list of Books: PRESS 1
To borrow a Book         : PRESS 2
To return a Book         : PRESS 3
To Exit                  : PRESS 4

--------------------------------------------------------------------------------
Select your choice:
```

Figure 40: Stock Decreases When Borrowed.

```
C:\Windows\py.exe

To view the list of Books: PRESS 1
To borrow a Book         : PRESS 2
To return a Book         : PRESS 3
To Exit                  : PRESS 4

--------------------------------------------------------------------------------
Select your choice: 3
--------------------------------------------------------------------------------

Enter your full name: Rize Shrestha

Enter the Book ID: 08
Enter the borrowed duration (in days): 10
Do you want to return another book? (y/n): n

 ----------------------------------------------------------------------------
  return_Receipt/Rize Shrestha.txt
  ==========================================================================

                        Library Management System

  __________________________________________________________________________

   Date: 2021-09-09                                Time: 18:08:15.978527
  __________________________________________________________________________

   Returned By: Rize Shrestha

  ----------------------------------------------------------------------------
   SNo. BookID  BookName                      AuthorsName           Cost
  ----------------------------------------------------------------------------
   1    08      Tokyo Ghoul                   Sui Ishida         £ 08.99

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

   Borrowed Duration in days: 10
   Late fees per day after 10 days will be: £0.5

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

                                                   Sub-Total: £8.99
                                                   Late-Fees: £0
                                                   Total    : £8.99

  ==========================================================================
```

Figure 41: Returning the Book.

```
C:\Windows\py.exe

To view the list of Books: PRESS 1
To borrow a Book         : PRESS 2
To return a Book         : PRESS 3
To Exit                  : PRESS 4

--------------------------------------------------------------------------------
Select your choice: 1
--------------------------------------------------------------------------------

BookID  Book Name                      AuthorName          Stock        Cost

01      I Want to Eat Your Pancreas    Yoru Sumino         50        £ 11.99
02      Wolf Children: Ame & Yuki      Mamoru Hosoda       50        £ 14.99
03      Solo Leveling                  Chugong             50        £ 11.99
04      Toradora                       Yuyuko Takemiya     50        £ 10.99
05      Naruto                         Masashi Kishimoto   50        £ 10.99
06      Black Butler                   Yana Toboso         50        £ 09.99
07      Assassination Classroom        Yusei Matsui        50        £ 07.99
08      Tokyo Ghoul                    Sui Ishida          50        £ 08.99
09      City of Bones                  Cassandra Clare     50        £ 08.99
10      The Haunting of Hill House     Shirley Jackson     50        £ 09.99

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

To view the list of Books: PRESS 1
To borrow a Book         : PRESS 2
To return a Book         : PRESS 3
To Exit                  : PRESS 4

--------------------------------------------------------------------------------
Select your choice:
```

Figure 42: Increase in Stock when Returned.

5. Conclusion.

The report is based on a project given to the students in hopes for us to create a working library management system. This project specifically aims to make the system more accessible and user-friendly in order for the people who have basic knowledge about computers taste the ease of working with it. It consists of five different modules in which four of the modules are imported into a single main module. The functioning aspect of the system is handled by the single main.py module without which the system cannot be operated. After understanding the concept of Python, we were able to implement the use of it in a real-life scenario. Although restricted by errors and mistakes along the way, with the help of teachers, the lecture slides, recordings provided to us and sites we were able to see the end of it.

Through this project we were able to learn and develop our skills in the field of Python. Algorithm, Flowchart and Pseudocode helped us create a reference sketch of the code which made the coding step considerably easier and the data flow clear. Algorithms was a subject new to me but through thorough investigation and practice I feel like I have gotten the hang of its basics.

Creating the library management system was an interesting project yet proved to be very challenging. The shapes of the flowchart were a subject I took lots of time in since I did not understand what shape to be allocated in which place. Another aspect which took much of my time was in context of borrowing multiple books; even if I were to put in data where the book ID was same it allowed the transaction. This problem prolonged to show error when I tried to fix it. 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 os
import booksFile
import borrowBook
import returnBook
import dateTime

LATE_FEES_PER_DAY = 0.5

def root():
    """ Main function of the project."""
    print()
    print("=" *78)
    print("\t\tWelcome to Library Management System.\(^0^)/" + u"\U0001F4DA")

    while True:
        ''' Shows the services provided by the system.
            Asks the user to select a choice.'''
        print("-" *78 + "\n")
        print(" To view the list of Books: PRESS 1")
        print(" To borrow a Book         : PRESS 2")
        print(" To return a Book         : PRESS 3")
        print(" To Exit                  : PRESS 4")
        print("\n" + "-" *78)
        # Exception handling is used.
        try:
            choice = int(input(" Select your choice: "))
            print("-" *78)
            if choice == 1:
```

```
                print("_" *78)
                # View logic.
                books_List = []
                columns = [" BookID" , " Book Name","AuthorName", "Stock" , "  Cost"]

                with open("books.txt", 'r') as file:
                    for line in file:
                        book = line.split(',')
                        books_List.append(book)

                print("{:<5s} {:<30s} {:<20s} {:<10s} {}".format(
                    columns[0], columns[1], columns[2], columns[3], columns[4]))
                print("_" *78 +"\n")

                for book in books_List:
                    print("{:<5s} {:<30s} {:<20s}  {:<10s}{}".format(
                        "  "+book[0],  "  "+book[1],book[2],  book[3],  u"\xA3"  +  book[4]),
end="")

                print("_" *78 +"\n")

            elif choice == 2:
                # Borrow logic.
                add_to_receipt = []
                person_Name = input("\n Enter your full name: ")
                while True:
                    book_ID = input("\n"+" Enter the Book ID: ")
                    with open("books.txt","r") as file:
                        exists = False
                        for line in file:
                            book = line.split(",")
```

```
                if book[0] == book_ID:
                    exists = True
                    break
            if exists:
                break
            else:
                print("\n" + "-" *78)
                print(" Book not Found")
                print("-" *78)

        if borrowBook.borrow_Book(person_Name, book_ID):
            add_to_receipt.append(book_ID)
        while True:
            another = input(
                " Do you want to add another book? (y/n): ")
            if another.lower() == "n":
                break
            elif another.lower() == "y":
                next_Book = input("\n Enter another Book ID: ")
                if borrowBook.borrow_Book(person_Name, next_Book):
                    add_to_receipt.append(next_Book)
            else:
                print(" Enter a valid response!")

        data = []
        sub_total = 0
        with open("Books.txt", "r") as file:
            for line in file:
                book = line.split(",")
                for each_id in add_to_receipt:
                    if book[0] == each_id:
```

```
                    data.append(
                        [book[0], book[1], book[2], book[4]])
                    sub_total += float(book[4])
            for book in data:
                book[-1] = str(book[-1]).replace("\n", "")

            # Creating borrow receipt in txt and shell.
            with open(f"borrow_Receipt/{person_Name}.txt", "w+") as file:
                file.write("="*72 + "\n\n")
                file.write("                    Library Management System" + "\n")
                file.write("_"*72 + "\n")
                file.write(" Date: "+dateTime.getDate()+"                                Time: "
+dateTime.getTime()+"\n")
                file.write("_"*72 + "\n")
                file.write(f"\n Borrower Name: {person_Name}\n\n")
                file.write("-"*72 + "\n")
                file.write("{:<6s} {:<7s} {:<29s} {:<20s} {}".format(
                    " SNo.", "BookID","BookName", "AuthorsName", " Cost" +"\n"))
                file.write("-"*72 + "\n")
                SNo = 0
                for book in data:
                    SNo = SNo + 1
                    file.write(
                        "{:<1s}{:<5d} {:<5s} {:<30s} {:<20s}{}\n".format("",SNo, book[0],
book[1], book[2],"£"+book[3]))
                file.write("-"*72 + "\n")
                file.write("\n Last date of Return: "+dateTime.getReturnDate()+"\n")
                file.write(
                    f" Late fees per day after 10 days will be:
£{LATE_FEES_PER_DAY}\n\n")
                file.write("_"*72 + "\n\n")
```

```
                    file.write(f"\t\t\t\t\t\tSub-Total: £{sub_total} \n\n")
                    file.write("="*72)

                    print("\n  "+"-"*74)
                    print(f"   borrow_Receipt/{person_Name}.txt")
                    print("   "+"="*72)
                    print("\n                    Library Management System")
                    print("   "+"_"*72)
                    print("\n                    "+"    Date:    "+dateTime.getDate()+"
Time: " +dateTime.getTime())
                    print("   "+"_"*72 )
                    print(f"\n    Borrower Name: {person_Name}\n")
                    print("   "+"-"*72)
                    print("{:<6s} {:<7s} {:<29s} {:<20s} {}".format(
                        "    SNo.", "BookID","BookName", "AuthorsName", " Cost"))
                    print("   "+"-"*72)
                    SNo = 0
                    for book in data:
                        SNo = SNo + 1
                        print(
                            "{:<1s}    {:<5d}{:<5s}   {:<30s}   {:<20s}{}\n".format("",SNo,
book[0], book[1], book[2],"£"+book[3]))
                    print("   "+"-"*72)
                    print("\n   "+" Last date of Return: "+dateTime.getReturnDate())
                    print(
                        f"    Late fees per day after 10 days will be:
£{LATE_FEES_PER_DAY}\n")
                    print("   "+"-"*72 + "\n")
                    print(f"\t\t\t\t\t\t   Sub-Total: £{sub_total}\n")
                    print("   "+"="*72 + "\n")
                    print("  "+"-"*74 + "\n")
```

```
        elif choice == 3:
            # Return logic.
            data = []
            sub_total = 0
            total_late_fees = 0
            add_to_receipt = []

            # Checks whether name exists.
            while True:
                person_Name = input("\n Enter your full name: ")
                exists = False
                with open("borrow_Record.txt","r") as file:
                    for line in file:
                        lend = line.split(",")

                        if lend[1] == person_Name :
                            exists = True
                            break
                if exists:
                    break
                else:
                    print("\n" + "-" *78)
                    print(f" Cannot find borrow record with the name: {person_Name}")
                    print("-" *78)

            # Checks whether book ID exists.
            while True:
                book_ID = input("\n"+" Enter the Book ID: ")
                with open("borrow_Record.txt","r") as file:
```

```
                    exists = False
                    for line in file:
                        book = line.split(",")
                        if lend[0] == book_ID:
                            exists = True
                            break
                    if exists:
                        break
                    else:
                        print("\n" + "-" *78)
                        print(" Book not Found")
                        print("-" *78)


            if returnBook.return_Book(person_Name, book_ID):
                with open("books.txt", "r") as file:
                    for line in file:
                        book = line.split(",")
                        if book[0] == book_ID:
                            with open("borrow_Record.txt", "r") as borrow_Bundle:
                                lines = borrow_Bundle.readlines()
                                new_Lines = []
                                for line in lines:
                                    lend = line.split(",")
                                    lend[-1] = str(lend[-1]).replace("\n", "")
                                    if lend[0] == book_ID and lend[1].lower() ==
person_Name.lower():
                                        data.append(
                                            [book[0], book[1], book[2], lend[3], book[4]])
                                        sub_total += float(book[4])
                                        if int(lend[3]) > 10:
```

```
                                    total_late_fees += (int(lend[3]) - 10) * \
                                        LATE_FEES_PER_DAY
                                else:
                                    new_Lines.append(line)
                            with open("borrow_Record.txt", "w") as file:
                                file.writelines(new_Lines)

                while True:
                    another = input(" Do you want to return another book? (y/n): ")
                    if another.lower() == "n":
                        break
                    elif another.lower() == "y":
                        next_Book = input("\n Enter another Book ID: ")
                        if returnBook.return_Book(person_Name, next_Book):
                            with open("books.txt", "r") as file:
                                for line in file:
                                    book = line.split(",")
                                    if book[0] == next_Book:
                                        with open("borrow_Record.txt", "r") as borrow_Bundle:
                                            lines = borrow_Bundle.readlines()
                                            new_Lines = []
                                            for line in lines:
                                                lend = line.split(",")
                                                lend[-1] = str(lend[-1]).replace("\n", "")
                                                if lend[0] == next_Book and lend[1].lower() ==
person_Name.lower():
                                                    data.append(
                                                        [book[0], book[1], book[2], lend[3], book[4]])
                                                    sub_total += float(book[4])
                                                    if int(lend[3]) > 10:
                                                        total_late_fees += (int(lend[3]) - 10) * \
```

```
                                        LATE_FEES_PER_DAY
                                    else:
                                        new_Lines.append(line)
                                with open("borrow_Record.txt", "w") as file:
                                    file.writelines(new_Lines)

                else:
                    print(" Enter a valid response!")

            for book in data:
                book[-1] = str(book[-1]).replace("\n", "")

            # Creating return receipt in txt and shell.
            with open(f"return_Receipt/{person_Name}.txt", "w+") as file:
                file.write("="*72 + "\n\n")
                file.write("                    Library Management System" + "\n")
                file.write("_"*72 + "\n")
                file.write(" Date: "+dateTime.getDate()+"                                Time: "
+dateTime.getTime()+"\n")
                file.write("_"*72 + "\n")
                file.write(f"\n Returned By: {person_Name}\n\n")
                file.write("-"*72 + "\n")
                file.write("{:<6s} {:<7s} {:<29s} {:<20s} {}".format(
                    " SNo.", "BookID","BookName", "AuthorsName", " Cost" +"\n"))
                file.write("-"*72 + "\n")
                SNo = 0
                for book in data:
                    SNo = SNo + 1
                    file.write(
                        "{:<1s}{:<5d} {:<5s} {:<30s} {:<20s}{}\n".format("",SNo, book[0],
book[1], book[2],"£"+book[4]))
```

```
                file.write("-"*72 + "\n")
                file.write("\n Borrowed Duration in days: "+book[3])
                file.write(
                    f"\n Late fees per day after 10 days will be: £{LATE_FEES_PER_DAY}\n\n")
                file.write("_"*72 + "\n\n")
                file.write(f"\t\t\t\t\t\t\tSub-Total: £{sub_total}\n")
                file.write(f"\t\t\t\t\t\t\tLate-Fees: £{total_late_fees}\n")
                total_fees = sub_total + total_late_fees
                two_float = "{:.2f}".format(total_fees)
                file.write(f"\t\t\t\t\t\t\tTotal    : £{total_fees}\n\n")
                file.write("="*72)

                print("\n  "+"-"*74)
                print(f"   return_Receipt/{person_Name}.txt")
                print("   "+"="*72)
                print("\n                    Library Management System")
                print("   "+"_"*72)
                print("\n                    "+"    Date:    "+dateTime.getDate()+" Time: " +dateTime.getTime())
                print("   "+"_"*72)
                print(f"\n    Returned By: {person_Name}\n")
                print("   "+"-"*72)
                print("{:<6s} {:<7s} {:<29s} {:<20s} {}".format(
                    "    SNo.", "BookID","BookName", "AuthorsName", " Cost"))
                print("   "+"-"*72)
                SNo = 0
                for book in data:
                    SNo = SNo + 1
                    print(
```

```
                    "{:<1s}        {:<5d}{:<5s}   {:<30s}   {:<20s}{}\n".format("",SNo,
book[0], book[1], book[2],"£"+book[4]))
                print("   "+"-"*72)
                print("\n    Borrowed Duration in days: "+book[3])
                print(
                    f"            Late  fees  per  day  after  10  days  will  be:
£{LATE_FEES_PER_DAY}\n")
                print("   "+"-"*72 + "\n")
                print(f"\t\t\t\t\t\t   Sub-Total: £{sub_total}")
                print(f"\t\t\t\t\t\t   Late-Fees: £{total_late_fees}")
                total_fees = sub_total + total_late_fees
                two_float = "{:.2f}".format(total_fees)
                print(f"\t\t\t\t\t\t   Total    : £{total_fees}\n")
                print("   "+"="*72 + "\n")
                print("  "+"-"*74 + "\n")

            elif choice == 4:
                # Exit logic.
                print("\t\t\tThank You and Have a Nice Day!")
                print("="*78)
                break

            else:
                print(" Invalid choice. Try again.")

        except ValueError:
            print("\n" + "-" *78)
            print(" Please enter 1/2/3/4.")
            print("-" *78 + "\n")

if __name__ == "__main__":
```

```
    ''' Creates receipt destination if it does not exist.'''
    if not os.path.exists("borrow_Receipt"):
        os.makedirs("borrow_Receipt")
    if not os.path.exists("return_Receipt"):
        os.makedirs("return_Receipt")
    root()
```

6.2. booksFile.py

```
# Importing the dateTime module.
import dateTime

class Book():
    """ A class used to represent Books."""
    def __init__(self, instance):
        self.book_ID = instance[0]
        self.book_Name = instance[1]
        self.book_Author = instance[2]
        self.book_Stock = instance[3]
        self.book_Cost = instance[4]

    def check_Borrow(self, person_Name):
        ''' Checks borrow record and returns a boolean value.'''
        with open('borrow_Record.txt', 'r') as file:
            for line in file:
                lend = line.split(",")
                if lend[0] == self.book_ID and lend[1].lower() == person_Name.lower():
                    return True

    def update_Stock(self):
        ''' Updates the stocks.'''
        with open("books.txt", "r") as file:
```

```
            lines = file.readlines()
            new_Lines = []
            for line in lines:
                book = line.split(",")
                if book[0] == self.book_ID:
                    updated_Line =
f"{self.book_ID},{self.book_Name},{self.book_Author},{self.book_Stock},{self.book_Cost}"
                    new_Lines.append(updated_Line)
                else:
                    new_Lines.append(line)
            with open("books.txt", "w") as file:
                file.writelines(new_Lines)

        def borrow(self, person_Name):
            ''' Reduces the stock.'''
            with open("borrow_Record.txt", "a") as file:
                date = dateTime.getDate()
                time = dateTime.getTime()
                file.write(f"{self.book_ID},{person_Name},{date}{time}\n")
            self.book_Stock = int(self.book_Stock) - 1
            self.update_Stock()

        def return_Back(self, person_Name):
            ''' Exception handling is used here.
                It increases the stock and deletes the records when returned.'''
            while True:
                try:
                    duration = int(input(" Enter the borrowed duration (in days): "))
                    with open("borrow_Record.txt", "r") as file:
                        lines = file.readlines()
```

```
                    new_Lines = []
                    with open("borrow_Record.txt", "w") as file:
                        for line in lines:
                            lend = line.split(",")
                            lend[-1] = str(lend[-1]).replace("\n", "")
                            if lend[0] == self.book_ID and lend[1] == person_Name:
                                new_Lines.append(
                                    f"{lend[0]},{lend[1]},{lend[2]},{duration}\n")
                            else:
                                new_Lines.append(line)
                        file.writelines(new_Lines)
                self.book_Stock = int(self.book_Stock) + 1
                self.update_Stock()
                break
            except ValueError:
                print("\n" + "-" *78)
                print(" Please input the borrowed duration in numeric value.")
                print("-" *78 + "\n")
```

6.3. borrowBook.py

```
# Importing booksFile module.
import booksFile

def borrow_Book(person_Name, book_ID):
    ''' Validates borrowing user input.'''
    with open('books.txt', 'r') as file:
        for line in file:
            book = line.split(",")
            if book[0] == book_ID:
                book_ins = booksFile.Book(book)
                if not book_ins.check_Borrow(person_Name):
```

```
                    if int(book_ins.book_Stock) < 1:
                        print("-" *78)
                        print(" Book out of Stock.")
                        print("-" *78+ "\n")
                        return False
                    book_ins.borrow(person_Name)
                    return True
                else:
                    print("\n" +"-" *78)
                    print(" You have already borrowed this book.")
                    print("-" *78+ "\n")
                    return False
            print("\n"+"-" *78)
            print(" Book not found.")
            print("-" *78 + "\n")
```

6.4. returnBook.py

```
# Importing booksFile module.
import booksFile

def return_Book(person_Name, book_ID):
    ''' Validates returning user input.'''
    with open("books.txt", "r") as file:
        for line in file:
            book = line.split(",")
            if book[0] == book_ID:
                book_ins = booksFile.Book(book)
                if book_ins.check_Borrow(person_Name):
                    book_ins.return_Back(person_Name)
                    return True
                else:
```

```
            print("\n" +"-" *78)
            print(" Borrow record not found.")
            print("-" *78 + "\n")
            return False
        print("\n" +"-" *78)
        print(" Book not found.")
        print("-" *78 + "\n")
```

6.5. dateTime.py

```
def getDate():
    # Importing date from datetime.
    import datetime
    Date = datetime.datetime.now
    return str(Date().date())

def getTime():
    # Importing time from datetime.
    import datetime
    Time = datetime.datetime.now
    return str(Time().time())

def getReturnDate():
    # Importing timedelta from datetime.
    import datetime
    from datetime import date, timedelta
    BorrowDate = datetime.date.today()
    DeltaTime = datetime.timedelta(days=10)
    return str(BorrowDate + DeltaTime)
```

7. Bibliography

Jaiswal, S., 2017. *datacamp.* [Online]
Available at: https://www.datacamp.com/community/tutorials/data-structures-python?utm_source=adwords_ppc&utm_campaignid=1455363063&utm_adgroupid=65083631748&utm_device=c&utm_keyword=&utm_matchtype=b&utm_network=g&utm_adpostion=&utm_creative=278443377095&utm_targetid=au
[Accessed 5 September 2021].

Meinecke, L., 2016. *study.com.* [Online]
Available at: https://study.com/academy/lesson/what-is-an-algorithm-in-programming-definition-examples-analysis.html
[Accessed 5 September 2021].

Sharma, R., 2020. *upGrad blog.* [Online]
Available at: https://www.upgrad.com/blog/data-structures-algorithm-in-python/
[Accessed 1 September 2021].