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: Sarthak Bikram Rana

Group: N1

London Met ID: 20049228

College ID: NP01NT4S210129

Assignment Due Date: 10th September 2021

Assignment Submission Date: 10th September 2021

I confirm that I understand my coursework needs to be submitted online via Google Classroom under the relevant module page before the deadline in order for my assignment to be accepted and marked. I am fully aware that late submissions will be treated as non-submission and a marks of zero will be awarded.

Table of Contents

List of Figures

List of Tables

1. Introduction

This is our first coursework of the fundamentals of computing module. In this coursework we were assigned to develop a library management system by using the python programming language. Python is a general-purpose, multiparadigm, interpreted, high-level programming language. Python enables programmers to use a variety of programming styles to create simple or complex programs, achieve faster results, and write code almost as if speaking in a human language. (techopedia, 2019)

In a library, all the records are set manually whether a new book is added to the library or when someone borrows or returns a book. The staff must put more efforts in order to manage the library books and customer information, which is difficult and time-consuming. To overcome this difficulty, many fields use a library management system, which must record the details of books and customers in a systematic manner. The main goal of this coursework was to develop a similar library system which record stocks of the books in a file and allows customers to borrow and return books.

In terms of the overall scenario of this coursework, the main goal of this system was based on a library management system that contains all of the information about a specific book in a text file, such as its name, author, price, and quantity. This system can be used by the library to allow users to borrow books from its collection and even return books that they have previously borrowed. When a customer borrows a book, the quantity of that book in the book txt file is reduced by one, and a separate borrow bill is generated with the customer's name, the date and time the book was borrowed, the title of the book, and the total cost. When a customer returns a specific book, the book's quantity increases by one. When the book is returned, the system asks how long the person borrowed it for, and if more than ten days have passed, a fine of one dollar per day is assigned to the person's name. It also generates a separate bill with the customer's name, the date and time the book was returned, the title of the book, and the fine amount if the book was returned more than ten days late.

Python was a completely new programming language for me, and developing an entire library management system with it was a difficult task in and of itself. Python was learned from the basics to the advanced level by implementing different functions,

modules, using exception handling in the code while taking input from the users, and creating and using a list to create an easy-to-use and interactive user interface.

2. Discussion and Analysis

2.1 Algorithm

Step 1: Start

Step 2: Display welcome message

Step 3: Read Books.txt file

Step 4: Extract all the data from the Books.txt file and store it dlist

Step 5: Display dlist

Step 6: Initialize bookName to empty string, totalCost to 0 and valid to true

Step 7: Display 1 to borrow a book, 2 to return a book and 3 to exit

Step 8: Loop while fg equals to true

Step 9: Input inputNumber 1,2 or 3

Step 10: If inputNumber equals to 1

Step 11: Input the customer name

Step 12: Loop while valid equals to true

Step 13: Input enter the bookID

Step 14: If bookID equals to 1

Step 15: If quantity not equals to 0

Step 16: Subtract 1 from quantity of bookID 1

Step 17: Add cost of the book with ID 1 to totalCost

Step 18: Add name of the book with ID 1 to bookName

Step 19: Else display Out of stock message message

Step 20: Input enter the bookID

Step 21: Jump to step 14

Step 22: Input if you want to borrow type “Y” else “N”

Step 23: If Y Jump to step 13

Step 24: If N Create a new text file with Borrow_ and the customer name

Step 25: Write name of customer, date and time when book was borrowed, name of the book which was borrowed, total cost

Step 26: Break

Step 27: If bookID equals to 2

Step 28: If quantity not equals to 0

Step 29: Subtract 1 from quantity of bookID 2

Step 30: Add cost of the book with ID 2 to totalCost

Step 31: Add name of the book with ID 2 to bookName

Step 32: Else display Out of stock message message

Step 33: Input enter the bookID

Step 34: Jump to step 27

Step 35: Input if you want to borrow type “Y” else “N”

Step 36: If Y Jump to step 13

Step 37: If N Create a new text file with Borrow_ and the customer name

Step 38: Write name of customer, date and time when book was borrowed, name of the book which was borrowed, total cost

Step 39: Break

Step 40: If bookID equals to 3

Step 41: If quantity not equals to 0

Step 42: Subtract 1 from quantity of bookID 3

Step 43: Add cost of the book with ID 3 to totalCost

Step 44: Add name of the book with ID 3 to bookName

Step 45: Else display Out of stock message message

Step 46: Input enter the bookID

Step 47: Jump to step 40

Step 48: Input if you want to borrow type “Y” else “N”

Step 48: If Y Jump to step 13

Step 50: If N Create a new text file with Borrow_ and the customer name

Step 51: Write name of customer, date and time when book was borrowed, name of the book which was borrowed, total cost

Step 52: Break

Step 53: If bookID equals to 4

Step 54: If quantity not equals to 0

Step 55: Subtract 1 from quantity of bookID 4

Step 56: Add cost of the book with ID 4 to totalCost

Step 57: Add name of the book with ID 4 to bookName

Step 58: Else display Out of stock message message

Step 59: Input enter the bookID

Step 60: Jump to step 53

Step 61: Input if you want to borrow type “Y” else “N”

Step 62: If Y Jump to step 13

Step 63: If N Create a new text file with Borrow_ and the customer name

Step 64: Write name of customer, date and time when book was borrowed, name of the book which was borrowed, total cost

Step 65: Break

Step 66: If bookID equals to 5

Step 67: If quantity not equals to 0

Step 68: Subtract 1 from quantity of bookID 5

Step 69: Add cost of the book with ID 5 to totalCost

Step 70: Add name of the book with ID 5 to bookName

Step 71: Else display Out of stock message message

Step 72: Input enter the bookID

Step 73: Jump to step 66

Step 74: Input if you want to borrow type “Y” else “N”

Step 75: If Y Jump to step 13

Step 76: If N Create a new text file with Borrow_ and the customer name

Step 77: Write name of customer, date and time when book was borrowed, name of the book which was borrowed, total cost

Step 78: Break

Step 79: If bookID equals to 6

Step 80: If quantity not equals to 0

Step 81: Subtract 1 from quantity of bookID 6

Step 82: Add cost of the book with ID 6 to totalCost

Step 83: Add name of the book with ID 6 to bookName

Step 84: Else display Out of stock message message

Step 85: Input enter the bookID

Step 86: Jump to step 79

Step 87: Input if you want to borrow type “Y” else “N”

Step 88: If Y Jump to step 13

Step 89: If N Create a new text file with Borrow_ and the customer name

Step 90: Write name of customer, date and time when book was borrowed, name of the book which was borrowed, total cost

Step 91: Break

Step 92: If inputNumber equals to 2

Step 93: Input the customer name

Step 94: Input enter the bookID

Step 95: Input enter for how many days you have borrowed the book

Step 96: If bookID equals to 1

Step 97: Add1 from quantity of bookID 1

Step 98: Add name of the book with ID 1 to bookName

Step 99: If noOfDays > 10

Step 100: Add fineAmount of 1$ per day

Step 101: Create a new txt file with Return_ and the customer name

Step 102: Write name of customer, date and time when book was borrowed, name if the book which was borrowed, fine amount

Step 103: Else no fineAmount is added

Step 104: Create a new txt file with Return_ and the customer name

Step 105: Write name of customer, date and time when book was borrowed, name if the book which was borrowed, you returned the book on time so there is no fine

Step 106: If bookID equals to 2

Step 107: Add1 from quantity of bookID 2

Step 108: Add name of the book with ID 2 to bookName

Step 109: If noOfDays > 10

Step 110: Add fineAmount of 1$ per day

Step 111: Create a new txt file with Return_ and the customer name

Step 112: Write name of customer, date and time when book was borrowed, name if the book which was borrowed, fine amount

Step 113: Else no fineAmount is added

Step 114: Create a new txt file with Return_ and the customer name

Step 115: Write name of customer, date and time when book was borrowed, name if the book which was borrowed, you returned the book on time so there is no fine

Step 116: If bookID equals to 3

Step 117: Add1 from quantity of bookID 3

Step 118: Add name of the book with ID 3 to bookName

Step 119: If noOfDays > 10

Step 120: Add fineAmount of 1$ per day

Step 121: Create a new txt file with Return_ and the customer name

Step 122: Write name of customer, date and time when book was borrowed, name if the book which was borrowed, fine amount

Step 123: Else no fineAmount is added

Step 124: Create a new txt file with Return_ and the customer name

Step 125: Write name of customer, date and time when book was borrowed, name if the book which was borrowed, you returned the book on time so there is no fine

Step 126: If bookID equals to 4

Step 127: Add1 from quantity of bookID 4

Step 128: Add name of the book with ID 4 to bookName

Step 129: If noOfDays > 10

Step 130: Add fineAmount of 1$ per day

Step 131: Create a new txt file with Return_ and the customer name

Step 132: Write name of customer, date and time when book was borrowed, name if the book which was borrowed, fine amount

Step 133: Else no fineAmount is added

Step 134: Create a new txt file with Return_ and the customer name

Step 135: Write name of customer, date and time when book was borrowed, name if the book which was borrowed, you returned the book on time so there is no fine

Step 136: If bookID equals to 5

Step 137: Add1 from quantity of bookID 5

Step 138: Add name of the book with ID 5 to bookName

Step 139: If noOfDays > 10

Step 140: Add fineAmount of 1$ per day

Step 141: Create a new txt file with Return_ and the customer name

Step 142: Write name of customer, date and time when book was borrowed, name if the book which was borrowed, fine amount

Step 143: Else no fineAmount is added

Step 144: Create a new txt file with Return_ and the customer name

Step 145: Write name of customer, date and time when book was borrowed, name if the book which was borrowed, you returned the book on time so there is no fine

Step 146: If bookID equals to 6

Step 147: Add1 from quantity of bookID 6

Step 148: Add name of the book with ID 6 to bookName

Step 149: If noOfDays > 10

Step 150: Add fineAmount of 1$ per day

Step 151: Create a new txt file with Return_ and the customer name

Step 152: Write name of customer, date and time when book was borrowed, name if the book which was borrowed, fine amount

Step 153: Else no fineAmount is added

Step 154: Create a new txt file with Return_ and the customer name

Step 155: Write name of customer, date and time when book was borrowed, name if the book which was borrowed, you returned the book on time so there is no fine

Step 156: Display Error

Step 157: Jump to step 94

Step 158: If inputNumber equals to 3

Step 159: Display you have exit the program

Step 160: Break

Step 161: Else Display You can only enter values from 1-3

Step 162: Stop

2.2 Flowchart

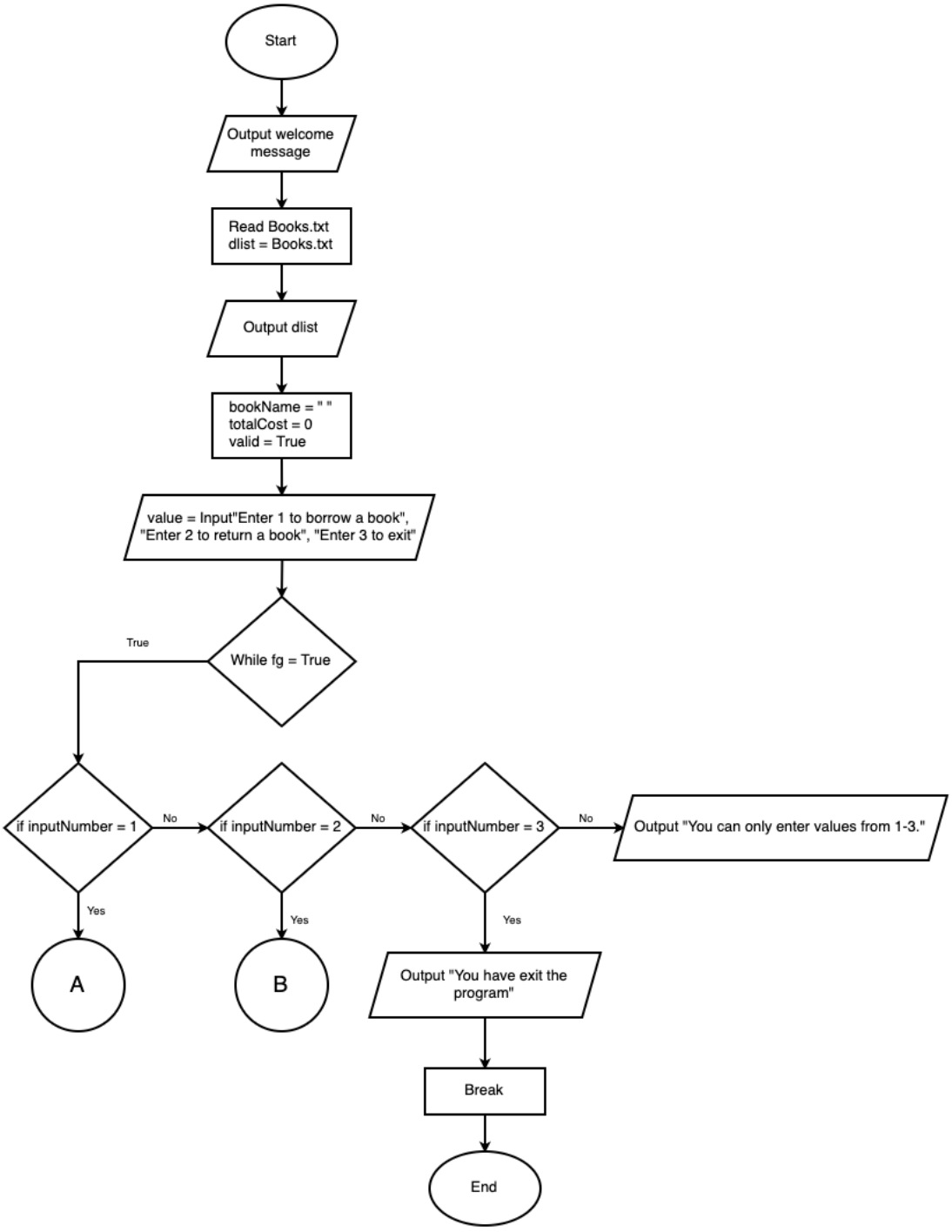

Figure 1: Flowchart of main module

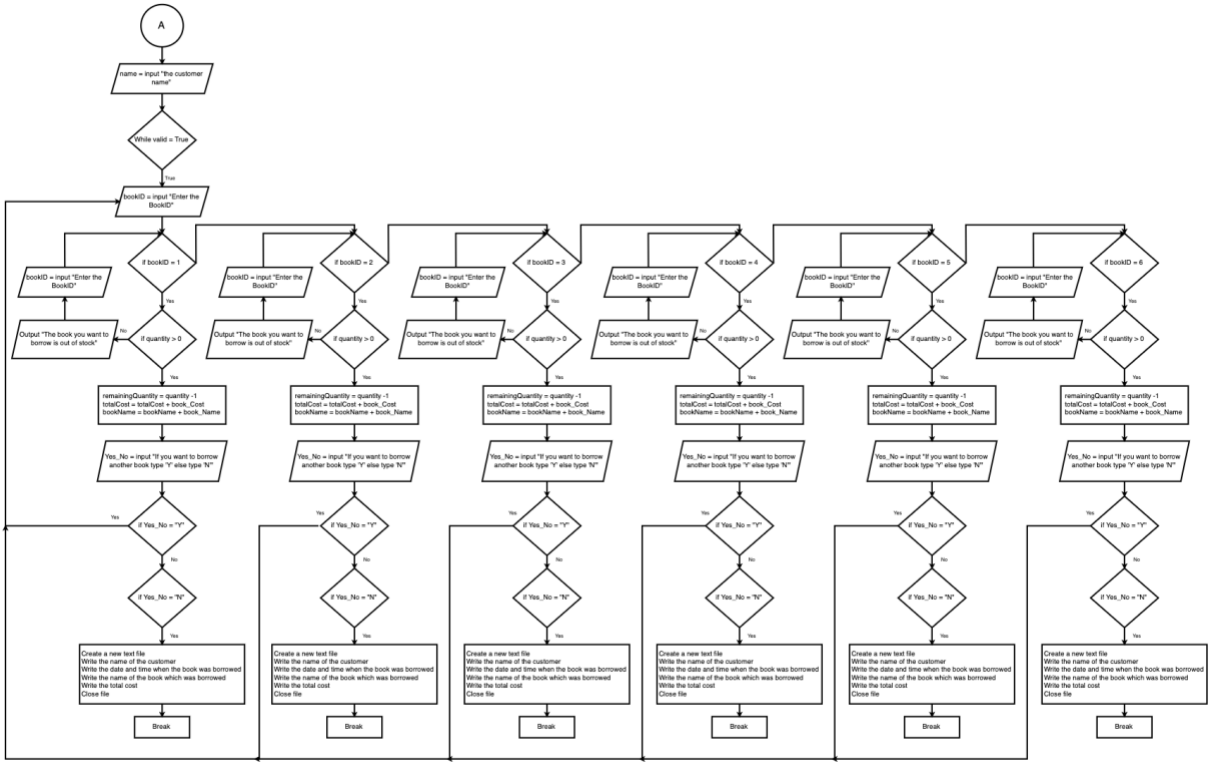

Figure 2: Flowchart of borrow module

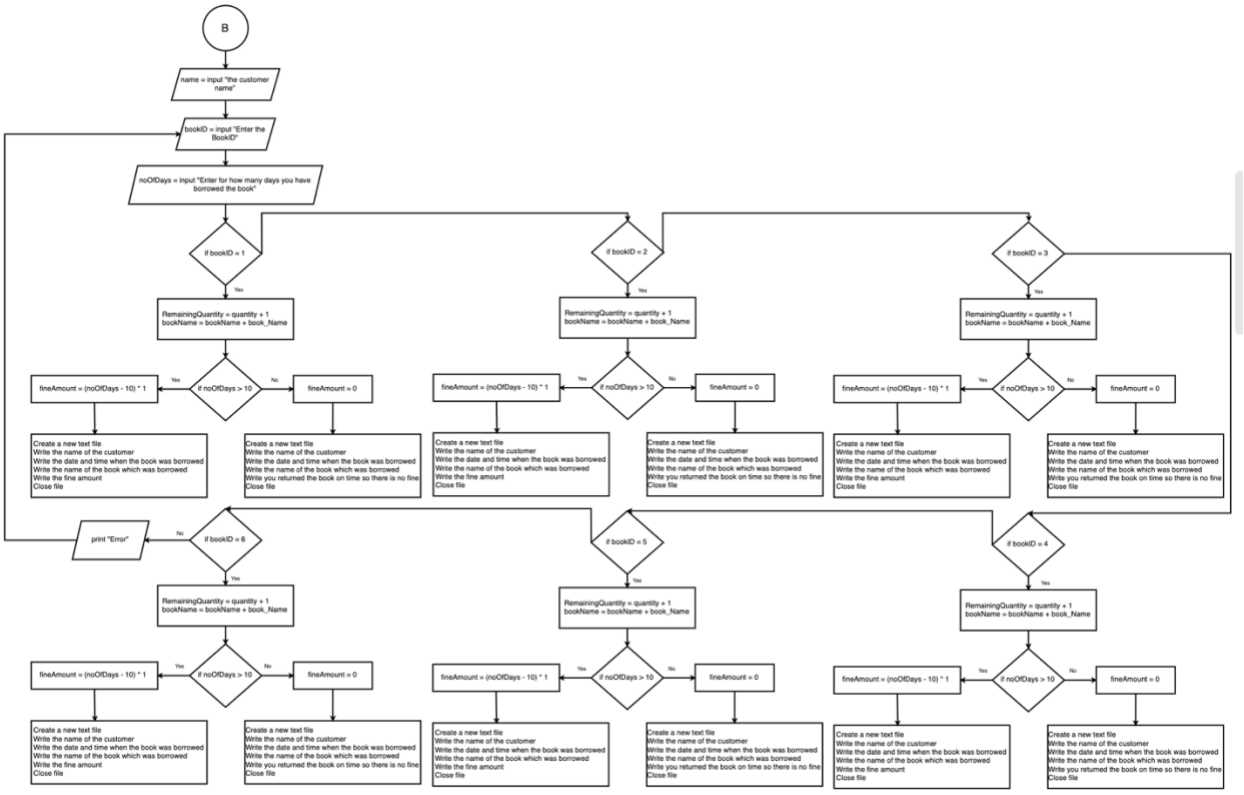

Figure 3: Flowchart of return module

2.3 Pseudocode

2.3.1 Module: Main.py

```
IMPORT Borrow
IMPORT Return
CALL OneDlist()
CALL Borrow.display(dlist)
SET bookName AS empty string
SET totalCost AS 0
SET inputNumber AS 0
SET valid AS TRUE
SET fg AS TRUE
WHILE fg AS TRUE
    PRINT “Enter 1 to borrow a book”
    PRINT “Enter 2 to return a book”
    PRINT “Enter 3 to exit”
    inputNumber ← INPUT(“Enter number: 1,2,3: ”)
    IF inputNumber EQUALS 1
        name ← “Enter the customer name: ”
        Cost, bookName ← CALL borrowContinuation(dlist, Borrow.display,
        totalCost, bookName, name)
        CALL Borrow.Bill(Cost, bookName, name)
    ELSE IF inputNumber EQUALS 2
        name ← “Enter the customer name: ”
        CALL Return.returnBook(dlist, name, Return.display)
    ELSE IF inputNumber EQUALS 3
        PRINT “You have exit the program”
    ELSE
        PRINT “You can only enter values from 1-3”
    END IF
END WHILE
```

2.3.2 Module: Borrow.py

```
IMPORT datetime
FUNCTION OneDlist
    OPENFILE “Books” FOR READ
    SET double_dlist AS Array
    SET single_dlist AS Array
    FOR line in file
        CHANGE line by replacing “\n” WITH empty string
    END FOR
    FOR I in range(double_dlist)
        FOR j in range (double_dlist[i])
            single_dlist ← double_dlist[I][j]
        END FOR
    END FOR
    CLOSEFILE “Books”
    RETURN single_dlist
END FUNCTION

FUNCTION display(dlist)
    PRINT “dlist[0], dlist[1], dlist[2], dlist[3], dlist[4]”
    PRINT “dlist[5], dlist[6], dlist[7], dlist[8], dlist[9]”
    PRINT “dlist[10], dlist[11], dlist[12], dlist[13], dlist[14]”
    PRINT “dlist[15], dlist[16], dlist[17], dlist[18], dlist[19]”
    PRINT “dlist[20], dlist[21], dlist[22], dlist[23], dlist[24]”
    PRINT “dlist[25], dlist[26], dlist[27], dlist[28], dlist[29]”
END FUNCTION

FUNCTION availableBooks(dlist,quantity,bookID)
    SET quantity1 ← dlist[4]
    SET quantity2 ← dlist[9]
    SET quantity3 ← dlist[14]
    SET quantity4 ← dlist[19]
    SET quantity5 ← dlist[24]
    SET quantity6 ← dlist[29]
    OPENFILE “Books” FOR WRITE
    IF bookID ← 1
        SET quantity1 AS (dlist[4]) - 1
        OPENFILE “Books” FOR WRITE
        ("1,JK Rowling,Harry Potter,$5,"+string(quantity1)+"\n2,James
        Clear,Atomic Habits,$15,"+quantity2+"\n3,Peter Thiel,Zero to
        One,$20,"+quantity3 +"\n4,Robert Iger,The ride of a
        lifetime,$15,"+quantity4+"\n5,Cal Newport,Deep
        Work,$10,"+quantity5+"\n6,Jim Collins,Good to
        Great,$10,"+quantity6+"")
```

```
ELSE IF bookID ← 2
    SET quantity2 AS (dlist[9]) - 1
    OPENFILE "Books" FOR WRITE
    ("1,JK Rowling,Harry Potter,$5,"+quantity1+"\n2,James Clear,Atomic
    Habits,$15,"+string(quantity2), "\n3,Peter Thiel,Zero to
    One,$20,"+quantity3 +"\n4,Robert Iger,The ride of a
    lifetime,$15,"+quantity4+"\n5,Cal Newport,Deep
    Work,$10,"+quantity5+"\n6,Jim Collins,Good to
    Great,$10,"+quantity6+"")
    CLOSEFILE "Books"

ELSE IF bookID ← 3
    SET quantity3 AS (dlist[14]) - 1
    OPENFILE "Books" FOR WRITE
    ("1,JK Rowling,Harry Potter,$5,"+quantity1+"\n2,James Clear,Atomic
    Habits,$15,"+quantity2, "\n3,Peter Thiel,Zero to
    One,$20,"+string(quantity3) +"\n4,Robert Iger,The ride of a
    lifetime,$15,"+quantity4+"\n5,Cal Newport,Deep
    Work,$10,"+quantity5+"\n6,Jim Collins,Good to
    Great,$10,"+quantity6+"")
    CLOSEFILE "Books"

ELSE IF bookID ← 4
    SET quantity4 AS (dlist[19]) - 1
    OPENFILE "Books" FOR WRITE
    ("1,JK Rowling,Harry Potter,$5,"+quantity1+"\n2,James Clear,Atomic
    Habits,$15,"+quantity2, "\n3,Peter Thiel,Zero to One,$20,"+quantity3
    +"\n4,Robert Iger,The ride of a lifetime,$15,"+string(quantity4)+"\n5,Cal
    Newport,Deep Work,$10,"+quantity5+"\n6,Jim Collins,Good to
    Great,$10,"+quantity6+"")
    CLOSEFILE "Books"

ELSE IF bookID ← 5
    SET quantity4 AS (dlist[24]) - 1
    OPENFILE "Books" FOR WRITE
    ("1,JK Rowling,Harry Potter,$5,"+quantity1+"\n2,James Clear,Atomic
    Habits,$15,"+string(quantity2), "\n3,Peter Thiel,Zero to
    One,$20,"+quantity3 +"\n4,Robert Iger,The ride of a
    lifetime,$15,"+quantity4+"\n5,Cal Newport,Deep
    Work,$10,"+string(quantity5)+"\n6,Jim Collins,Good to
    Great,$10,"+quantity6+"")
    CLOSEFILE "Books"
```

```
        ELSE IF bookID ← 6
            SET quantity4 AS (dlist[29]) - 1
            OPENFILE "Books" FOR WRITE
            ("1,JK Rowling,Harry Potter,$5,"+quantity1+"\n2,James Clear,Atomic
            Habits,$15,"+string(quantity2), "\n3,Peter Thiel,Zero to
            One,$20,"+quantity3 +"\n4,Robert Iger,The ride of a
            lifetime,$15,"+quantity4+"\n5,Cal Newport,Deep
            Work,$10,"+quantity5+"\n6,Jim Collins,Good to
            Great,$10,"+string(quantity6)+"")
            CLOSEFILE "Books"
        END IF
        CLOSEFILE "Books"

        SET remainingQuantity AS quantity - 1
        FOR I in range(dlist)
            FOR j in range (dlist[i])
                IF bookID ← 1
                SET dlist[4] AS RemainingQuantity

                ELSE IF bookID ← 2
                SET dlist[9] AS RemainingQuantity

                ELSE IF bookID ← 3
                SET dlist[14] AS RemainingQuantity

                ELSE IF bookID ← 4
                SET dlist[19] AS RemainingQuantity

                ELSE IF bookID ← 5
                SET dlist[24] AS RemainingQuantity

                ELSE IF bookID ← 6
                SET dlist[29] AS RemainingQuantity
                END IF
            END FOR
        END FOR
    END FUNCTION

    FUNCTION borrowBook(dlist)
        SET bookPrice AS 0
        SET book_Name AS empty string
        SET fg AS TRUE
        SET valid AS TRUE
        SET n_valid AS TRUE
                bookID ← INPUT("Enter the bookID: ")
```

```
                BREAK
                PRINT “The bookID should be an integer between 1-6”
        WHILE fg ← TRUE
            IF bookID ← 1
                SET quantity AS INTEGER(dlist[4])
                IF quantity > 0
                    CALL displayAvailable()
                    CALL availableBooks(dlist,quantity,bookID)
                    CALL display(dlist)
                    SET bookPrice AS Price(dlist,bookID)
                    SET book_Name ← N_book(dlist,bookID)
                    BREAK
                ELSE
                    bookID ← INPUT("The book you want to borrow is out of
                    stock, please enter another bookID if you want to borrow: ")
                    PRINT “The bookID should be an integer between 1-6”
                END IF

            ELSE IF bookID ← 2
                SET quantity AS INTEGER(dlist[9])
                IF quantity > 0
                    CALL displayAvailable()
                    CALL availableBooks(dlist,quantity,bookID)
                    CALL display(dlist)
                    SET bookPrice AS Price(dlist,bookID)
                    SET book_Name ← N_book(dlist,bookID)
                    BREAK
                ELSE
                    bookID ← INPUT("The book you want to borrow is out of
                    stock, please enter another bookID if you want to borrow: ")
                    PRINT “The bookID should be an integer between 1-6”
                END IF

            ELSE IF bookID ← 3
                SET quantity AS INTEGER(dlist[15])
                IF quantity > 0
                    CALL displayAvailable()
                    CALL availableBooks(dlist,quantity,bookID)
                    CALL display(dlist)
                    SET bookPrice AS Price(dlist,bookID)
                    SET book_Name ← N_book(dlist,bookID)
                    BREAK
                ELSE
                    bookID ← INPUT("The book you want to borrow is out of
                    stock, please enter another bookID if you want to borrow: ")
                    PRINT “The bookID should be an integer between 1-6”
```

```
        END IF

    ELSE IF bookID ← 4
        SET quantity AS INTEGER(dlist[19])
        IF quantity > 0
            CALL displayAvailable()
            CALL availableBooks(dlist,quantity,bookID)
            CALL display(dlist)
            SET bookPrice AS Price(dlist,bookID)
            SET book_Name ← N_book(dlist,bookID)
            BREAK
        ELSE
            bookID ← INPUT("The book you want to borrow is out of
            stock, please enter another bookID if you want to borrow: ")
            PRINT “The bookID should be an integer between 1-6”
        END IF

    ELSE IF bookID ← 5
        SET quantity AS INTEGER(dlist[24])
        IF quantity > 0
            CALL displayAvailable()
            CALL availableBooks(dlist,quantity,bookID)
            CALL display(dlist)
            SET bookPrice AS Price(dlist,bookID)
            SET book_Name ← N_book(dlist,bookID)
            BREAK
        ELSE
            bookID ← INPUT("The book you want to borrow is out of
            stock, please enter another bookID if you want to borrow: ")
            PRINT “The bookID should be an integer between 1-6”
        END IF

    ELSE IF bookID ← 6
        SET quantity AS INTEGER(dlist[29])
        IF quantity > 0
            CALL displayAvailable()
            CALL availableBooks(dlist,quantity,bookID)
            CALL display(dlist)
            SET bookPrice AS Price(dlist,bookID)
            SET book_Name ← N_book(dlist,bookID)
            BREAK
        ELSE
            bookID ← INPUT("The book you want to borrow is out of
            stock, please enter another bookID if you want to borrow: ")
            PRINT “The bookID should be an integer between 1-6”
        END IF
```

```
            ELSE
                PRINT "Error"
                BREAK
            END IF
        END WHILE
    RETURN bookPrice, book_Name
END FUNCTION

FUNCTION Price(dlist, bookID)
    IF bookID ← 1
        SET price AS dilst[3]
        CHANGE price by replacing "$" WITH empty string
    ELSE IF bookID ← 2
        SET price AS dilst[8]
        CHANGE price by replacing "$" WITH empty string
    ELSE IF bookID ← 3
        SET price AS dilst[13]
        CHANGE price by replacing "$" WITH empty string
    ELSE IF bookID ← 4
        SET price AS dilst[18]
        CHANGE price by replacing "$" WITH empty string
    ELSE IF bookID ← 5
        SET price AS dilst[23]
        CHANGE price by replacing "$" WITH empty string
    ELSE IF bookID ← 6
        SET price AS dilst[28]
        CHANGE price by replacing "$" WITH empty string
    END IF
    RETURN price
END FUNCTION

FUNCTION N_book(dlist, bookID)
    SET book_Name AS empty string
    IF bookID ← 1
        SET book_Name AS dlist[2]
    ELSE IF bookID ← 2
        SET book_Name AS dlist[7]
    ELSE IF bookID ← 3
        SET book_Name AS dlist[12]
    ELSE IF bookID ← 4
        SET book_Name AS dlist[17]
    ELSE IF bookID ← 5
        SET book_Name AS dlist[22]
    ELSE IF bookID ← 6
        SET book_Name AS dlist[27]
    END IF
```

```
        RETURN book_Name
END FUNCTION

FUNCTION borrowContinuation(dlist, diplay, totalCost, bookName, name)
        SET fg AS TRUE
        book_Cost, book_Name ← CALL borrowBook(dlist)
        SET totalCost ← totalCost + book_Cost AS INTEGER(book_Cost)
        SET bookName ← bookName + book_Name
        WHILE fg EQUALS True
                Yes_No ← INPUT("If you want to borrow another book type 'Y' else type
                'N':")
                IF Yes_No ← “Y”
                        book_Cost, book_Name ← CALL borrowBook(dlist)
                        SET totalCost ← totalCost + book_Cost AS INTEGER(book_Cost)
                        SET bookName ← bookName + book_Name
                ELSE IF Yes_No ← “N”
                        PRINT "Thankyou for borrowing books from us."
                        BREAK
                ELSE
                        PRINT “Invalid Input”
                END IF
        END WHILE
        RETURN totalCost, bookName
END FUNCTION

FUNCTION Bill(Cost, bookName, name)
        SET dateNtime AS datetime.datetime.now()
        SET minute AS String(datetime.datetime.now().minute)
        SET second AS String(datetime.datetime.now().second)
        SET microsecond AS String(datetime.datetime.now().microsecond)
        OPENFILE “Borrow_+name+” FOR WRITE
        WRITEFILE “Name of the Customer” &name
        WRITEFILE “Date and time when borrowed” &String(dateNtime)
        WRITEFILE “Name of the borrowed book is” &bookName
        WRITEFILE “The total cost is: $” &String(Cost)
        CL0SEFILE “Borrow_+name+”
END FUNCTION

FUNCTION displayAvailable()
        PRINT “The book is available”
END FUNCTION
```

2.3.3 Module: Return.py

```
IMPORT datetime
FUNCTION OneDlist
    OPENFILE “Books” FOR READ
    SET double_dlist AS Array
    SET single_dlist AS Array
    FOR line in file
        CHANGE line by replacing “\n” WITH empty string
    END FOR
    FOR I in range(double_dlist)
        FOR j in range (double_dlist[i])
            single_dlist ← double_dlist[I][j]
        END FOR
    END FOR
    CLOSEFILE “Books”
    RETURN single_dlist
END FUNCTION

FUNCTION display(dlist)
    PRINT “dlist[0], dlist[1], dlist[2], dlist[3], dlist[4]”
    PRINT “dlist[5], dlist[6], dlist[7], dlist[8], dlist[9]”
    PRINT “dlist[10], dlist[11], dlist[12], dlist[13], dlist[14]”
    PRINT “dlist[15], dlist[16], dlist[17], dlist[18], dlist[19]”
    PRINT “dlist[20], dlist[21], dlist[22], dlist[23], dlist[24]”
    PRINT “dlist[25], dlist[26], dlist[27], dlist[28], dlist[29]”
END FUNCTION

FUNCTION addBooks(dlist,quantity,bookID)
    SET quantity1 ← dlist[4]
    SET quantity2 ← dlist[9]
    SET quantity3 ← dlist[14]
    SET quantity4 ← dlist[19]
    SET quantity5 ← dlist[24]
    SET quantity6 ← dlist[29]
    OPENFILE “Books” FOR WRITE
    IF bookID ← 1
        SET quantity1 AS (dlist[4]) + 1
        OPENFILE “Books” FOR WRITE
        ("1,JK Rowling,Harry Potter,$5,"+string(quantity1)+"\n2,James
        Clear,Atomic Habits,$15,"+quantity2+"\n3,Peter Thiel,Zero to
        One,$20,"+quantity3 +"\n4,Robert Iger,The ride of a
        lifetime,$15,"+quantity4+"\n5,Cal Newport,Deep
        Work,$10,"+quantity5+"\n6,Jim Collins,Good to
        Great,$10,"+quantity6+"")
```

ELSE IF bookID ← 2
SET quantity2 **AS** (dlist[9]) + 1
OPENFILE “Books” **FOR WRITE**
("1,JK Rowling,Harry Potter,$5,"+quantity1+"\n2,James Clear,Atomic Habits,$15,"+string(quantity2), "\n3,Peter Thiel,Zero to One,$20,"+quantity3 +"\n4,Robert Iger,The ride of a lifetime,$15,"+quantity4+"\n5,Cal Newport,Deep Work,$10,"+quantity5+"\n6,Jim Collins,Good to Great,$10,"+quantity6+"")
CLOSEFILE “Books”

ELSE IF bookID ← 3
SET quantity3 **AS** (dlist[14]) + 1
OPENFILE “Books” **FOR WRITE**
("1,JK Rowling,Harry Potter,$5,"+quantity1+"\n2,James Clear,Atomic Habits,$15,"+quantity2, "\n3,Peter Thiel,Zero to One,$20,”+string(quantity3) +"\n4,Robert Iger,The ride of a lifetime,$15,"+quantity4+"\n5,Cal Newport,Deep Work,$10,"+quantity5+"\n6,Jim Collins,Good to Great,$10,"+quantity6+"")
CLOSEFILE “Books”

ELSE IF bookID ← 4
SET quantity4 AS (dlist[19]) + 1
OPENFILE “Books” **FOR WRITE**
("1,JK Rowling,Harry Potter,$5,"+quantity1+"\n2,James Clear,Atomic Habits,$15,"+quantity2, "\n3,Peter Thiel,Zero to One,$20,"+quantity3 +"\n4,Robert Iger,The ride of a lifetime,$15,”+string(quantity4)+”\n5,Cal Newport,Deep Work,$10,"+quantity5+"\n6,Jim Collins,Good to Great,$10,"+quantity6+"")
CLOSEFILE “Books”

ELSE IF bookID ← 5
SET quantity4 AS (dlist[24]) + 1
OPENFILE “Books” **FOR WRITE**
("1,JK Rowling,Harry Potter,$5,"+quantity1+"\n2,James Clear,Atomic Habits,$15,"+string(quantity2), "\n3,Peter Thiel,Zero to One,$20,"+quantity3 +"\n4,Robert Iger,The ride of a lifetime,$15,”+quantity4+”\n5,Cal Newport,Deep Work,$10,"+string(quantity5)+”\n6,Jim Collins,Good to Great,$10,"+quantity6+"")
CLOSEFILE “Books”

ELSE IF bookID ← 6

```
            SET quantity4 AS (dlist[29]) + 1
            OPENFILE "Books" FOR WRITE
            ("1,JK Rowling,Harry Potter,$5,"+quantity1+"\n2,James Clear,Atomic
            Habits,$15,"+string(quantity2), "\n3,Peter Thiel,Zero to
            One,$20,"+quantity3 +"\n4,Robert Iger,The ride of a
            lifetime,$15,"+quantity4+"\n5,Cal Newport,Deep
            Work,$10,"+quantity5+"\n6,Jim Collins,Good to
            Great,$10,"+string(quantity6)+"")
            CLOSEFILE "Books"
        END IF
        CLOSEFILE "Books"

        SET remainingQuantity AS quantity + 1
        FOR i in range(dlist)
            FOR j in range (dlist[i])
                IF bookID ← 1
                SET dlist[4] AS RemainingQuantity

                ELSE IF bookID ← 2
                SET dlist[9] AS RemainingQuantity

                ELSE IF bookID ← 3
                SET dlist[14] AS RemainingQuantity

                ELSE IF bookID ← 4
                SET dlist[19] AS RemainingQuantity

                ELSE IF bookID ← 5
                SET dlist[24] AS RemainingQuantity

                ELSE IF bookID ← 6
                SET dlist[29] AS RemainingQuantity
                END IF
            END FOR
        END FOR
    END FUNCTION

    FUNCTION returnBook(dlist, name, display)
        SET fg AS True
        SET valid AS True
        SET n_valid AS True
                bookID ← INPUT("Enter number: 1,2,3: ")
                BREAK
                PRINT "The bookID should be an integer between 1-6"
                noOfDays ← INPUT("Enter for how many days you have borrowed
                the book: ")
```

```
                BREAK
                PRINT "Enter an integer"
        CALL bookFine(noOfDays, name, dlist, bookID)
        WHILE fg ← True
            IF bookID ← 1
                SET quantity AS Integer(dlist[4])
                CALL addBooks(dlist, quantity, bookID)
                CALL display(dlist)
                BREAK

            ELSE IF bookID ← 2
                SET quantity AS Integer(dlist[9])
                CALL addBooks(dlist, quantity, bookID)
                CALL display(dlist)
                BREAK

            ELSE IF bookID ← 3
                SET quantity AS Integer(dlist[14])
                CALL addBooks(dlist, quantity, bookID)
                CALL display(dlist)
                BREAK

            ELSE IF bookID ← 4
                SET quantity AS Integer(dlist[19])
                CALL addBooks(dlist, quantity, bookID)
                CALL display(dlist)
                BREAK

            ELSE IF bookID ← 5
                SET quantity AS Integer(dlist[24])
                CALL addBooks(dlist, quantity, bookID)
                CALL display(dlist)
                BREAK

            ELSE IF bookID ← 6
                SET quantity AS Integer(dlist[29])
                CALL addBooks(dlist, quantity, bookID)
                CALL display(dlist)
                BREAK

            ELSE
                PRINT "Error"
                BREAK
            END IF
        END WHILE
    END FUNCTION
```

```
    FUNCTION N_book(dlist, bookID)
        SET book_Name AS empty string
        IF bookID ← 1
            SET book_Name AS dlist[2]

        ELSE IF bookID ← 2
            SET book_Name AS dlist[7]

        ELSE IF bookID ← 3
            SET book_Name AS dlist[12]

        ELSE IF bookID ← 4
            SET book_Name AS dlist[17]

        ELSE IF bookID ← 5
            SET book_Name AS dlist[22]

        ELSE IF bookID ← 6
            SET book_Name AS dlist[27]
        END IF
        RETURN book_Name
    END FUNCTION

FUNCTION bookFine(noOfDays, name, dlist, bookID)
    IF Integer(noOfDays) > 10
        CALL fineBill(name, dlist, bookID, noOfDays)
    ELSE
        CALL noFineBill(name, dlist, bookID)
    END IF
END FUNCTION

FUNCTION fineBill(name, dlist, bookID, noOfDays)
    SET nameofbook ← N_book(dlist,bookID)
    SET fineAmount AS (Integer(noOfDays - 10)) * 1
    SET dateNtime AS datetime.datetime.now()
    SET minute AS String(datetime.datetime.now().minute)
    SET second AS String(datetime.datetime.now().second)
    SET microsecond AS String(datetime.datetime.now().microsecond)
    OPENFILE “Return_+name+” FOR WRITE
    WRITEFILE “Name of the Customer” &name
    WRITEFILE “Date and time when borrowed” &String(dateNtime)
    WRITEFILE “Name of the borrowed book is” &nameofbook
    WRITEFILE “The fine amount is: $” &String(fineAmount)
    CL0SEFILE “Return_+name+”
    PRINT “You have charged with a fine of 1$ per day.”
END FUNCTION
```

```
FUNCTION noFineBill(name, dlist, bookID)
    SET nameofbook ← N_book(dlist,bookID)
    SET dateNtime AS datetime.datetime.now()
    SET minute AS String(datetime.datetime.now().minute)
    SET second AS String(datetime.datetime.now().second)
    SET microsecond AS String(datetime.datetime.now().microsecond)
    OPENFILE “Return_+name+” FOR WRITE
    WRITEFILE “Name of the Customer” &name
    WRITEFILE “Date and time when borrowed” &String(dateNtime)
    WRITEFILE “Name of the borrowed book is” &nameofBook
    WRITEFILE “You returned the book on time, so there is no fine.”
    CL0SEFILE “Return_+name+”
END FUNCTION
```

2.4 Data Structures

A data structure is a collection of data type 'values' that are stored and organized in such a way that they can be accessed and modified efficiently. (McDonnell, 2019). In this coursework both primitive and non-primitive data structures have been used. Integer and String are used as primitive data structures whereas array and list are used as non-primitive data structures.

2.4.1 Primitive data structure

```
while valid == True:
    try:
        noOfDays = int(input("Enter for how many days you have borrowed the book: "))
        break
```

Figure 4: Screenshot of implementing int data type

As we know integer only stores mathematical values. Here in this code int data type is being used to store for how many days the user has borrowed the book.

```
def fineBill(name, dlist, bookID, noOfDays):
    nameofbook = N_book(dlist, bookID)
    fineAmount = (int(noOfDays) - 10) * 1
    dateNtime = datetime.datetime.now()
    minute = str(datetime.datetime.now().minute)
    second = str(datetime.datetime.now().second)
    microsecond = str(datetime.datetime.now().microsecond)
    randomNumber = minute+second+microsecond
    file = open("Return_"+name+".txt", "w")
    file.write("Name of the Costumer: "+name+"\n")
    file.write("The Date and Time when borrowed is: "+str(dateNtime)+"\n")
    file.write("The name of the borrowed book is: "+nameofbook+"\n")
    file.write("The Fine amount is: $"+str(fineAmount)+"\n")
    file.close()
```

Figure 5: Screenshot of implementing str data type

As we all know, a string is a collection of characters that can contain both numbers and words. String can be translated from any data type, including integer, float, and date. All of the parameters, such as date and time, are specified in string in order to display in the bill while making a bill.

2.4.2 Non-Primitive data structure

```
#To convert the text file into 1D list
def OneDlsit():
    file = open("Books.txt", "r")
    double_dlist = []
    single_dlist = []
    for line in file:
        line = line.replace("\n", "")
        double_dlist.append(line.split(","))
    for i in range(len(double_dlist)):
        for j in range(len(double_dlist[i])):
            single_dlist.append(double_dlist[i][j])
    return single_dlist
```

Figure 6: Screenshot of converting text file into 1D list

The overall records and data from the books.txt file are extracted and stored in a 2D list in the function OneDlist(). Finally, all of the data in the 2D list is converted to a 1D list.

```
def display(dlist):
    print("+++++++++++++\nBook ID          Author          Book Name                Price          Quantity\n++++++++++++++++++++")
    print("  ", dlist[0], "          ", dlist[1], "      ", dlist[2], "              ", dlist[3], "            ", dlist[4])
    print("  ", dlist[5], "          ", dlist[6], "      ", dlist[7], "              ", dlist[8], "            ", dlist[9])
    print("  ", dlist[10], "         ", dlist[11], "     ", dlist[12], "              ", dlist[13], "           ", dlist[14])
    print("  ", dlist[15], "         ", dlist[16], "     ", dlist[17], "    ", dlist[18], "          ", dlist[19])
    print("  ", dlist[20], "         ", dlist[21], "     ", dlist[22], "                ", dlist[23], "           ", dlist[24])
    print("  ", dlist[25], "         ", dlist[26], "     ", dlist[27], "              ", dlist[28], "           ", dlist[29])
    print("++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++")
```

Figure 7: Screenshot of displaying data in a list

The data can be shown in whatever way you like using an array list. The data is displayed in a table format here utilizing a 1D list.

```
'''for displaying available books'''
def availableBooks(dlist, quantity, bookID):
    quantity1 = dlist[4]
    quantity2 = dlist[9]
    quantity3 = dlist[14]
    quantity4 = dlist[19]
    quantity5 = dlist[24]
    quantity6 = dlist[29]
    file = open("Books.txt", "w")
    if bookID == 1:
```

Figure 8: Screenshot of appending value of list

Append is primarily used to add a single item to an existing list. The usage of append allows programmers to simply edit code. Here, using appending, quantity1 is transformed into quantity2, quantity3, quantity4, quantity5, and quantity6.

3. Program

3.1 Borrow process

```
>>>
============ RESTART: /Users/sarthakrana/Desktop/SBR_PYTHOPN/Main.py ===========
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
Book ID          Author          Book Name                Price          Quantity
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
   1           JK Rowling        Harry Potter              $5               10
   2           James Clear       Atomic Habits             $15              15
   3           Peter Thiel       Zero to One               $20              5
   4           Robert Iger       The ride of a lifetime    $15              10
   5           Cal Newport       Deep Work                 $10              7
   6           Jim Collins       Good to Great             $10              15
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
Enter 1 to borrow a book
Enter 2 to return a book
Enter 3 to exit
Enter number: 1,2,3 :
```

Figure 9: Screenshot displaying when the program is run in the shell

At first, when the program was run in the shell it displays all the records of the books in a tabular format. Then it takes input from the users as three different options as 1 to borrow a book, 2 to return a book and 3 to exit the program.

```
Enter 1 to borrow a book
Enter 2 to return a book
Enter 3 to exit
Enter number: 1,2,3 : 1
Enter the costumer name: Tobi
Enter the BookID: 6
The book is available
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
Book ID          Author          Book Name                Price          Quantity
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
   1           JK Rowling        Harry Potter              $5               10
   2           James Clear       Atomic Habits             $15              15
   3           Peter Thiel       Zero to One               $20              5
   4           Robert Iger       The ride of a lifetime    $15              10
   5           Cal Newport       Deep Work                 $10              7
   6           Jim Collins       Good to Great             $10              14
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

Figure 10: Screenshot displaying after borrowing the book with bookID 6 by Tobi

When the users enter the option 1 which means to borrow a book, the program asks the user to enter the customer name. After the users enter the customer name the program asks to enter the book id. Here “6” is entered as the book id which is referred to the “Good to Great” book. Since, the book is on the records and it has still 15 quantities left the book is successfully borrowed. A single quantity is decreased from the particular book in the records which is displayed in the shell.

```
If you want to borrow another book type 'Y' else type 'N':Y
Enter the BookID: 1
The book is available
+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
Book ID          Author          Book Name                Price          Quantity
+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
   1          JK Rowling       Harry Potter              $5               9
   2          James Clear      Atomic Habits             $15              15
   3          Peter Thiel      Zero to One               $20              5
   4          Robert Iger      The ride of a lifetime    $15              10
   5          Cal Newport      Deep Work                 $10              7
   6          Jim Collins      Good to Great             $10              14
+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
If you want to borrow another book type 'Y' else type 'N':N
Thankyou for borrowing books from us.
Enter 1 to borrow a book
Enter 2 to return a book
Enter 3 to exit
Enter number: 1,2,3 :
```

Figure 11: Screenshot displaying again after borrowing the book with bookID 1 by Tobi

After the book is borrowed the program asks the user whether he/she wants to borrow another book with an option of ‘Y’ and ‘N’. Here user enter ‘Y’ and the program again asks the user for another book id, when the user enters “1” as book id which is referred to the “Harry Potter” book the program accepts the book id since the book is on the records and it has still 10 quantities left the book is successfully borrowed. A single quantity is decreased from the particular book in the records which is displayed in the shell. Then, the program again asks the user whether he/she wants to borrow another book with an option of ‘Y’ and ‘N’. Here the user enter ‘N’ and the loop is breaks and a message saying “Thankyou for borrowing books from us” is displayed. The program again asks the user to input three different options as 1 to borrow a book, 2 to return a book and 3 to exit the program.

Borrow_Tobi.txt

```
Name of the Customer: Tobi
The Date and Time when borrowed is: 2021-09-09 13:08:48.755098
The name of the borrowed book is: Good to Great
Harry Potter

The total Cost is: $15
```

Figure 12: Screenshot of borrow bill in a txt file of Tobi

A borrow bill is generated in the name of the customer after the user borrows the book which consists the name of the customer which is “Tobi”, the date and time when the book was borrowed which is “2021-09-09” and “13:08:48:755098”, the name of the

borrowed books which is “Good to Great” and “Harry Potter” and at last the total cost of the books is displayed which is “$15” in total.

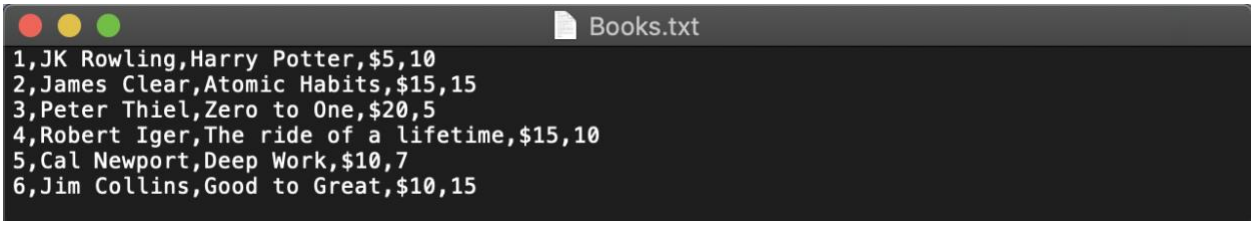

Figure 13: Screenshot of the records before borrowing

This is screenshot before borrowing of the books.txt file where all the records of the books are stored. Here the quantity of the book “Harry Potter” is ‘10’ and the quantity of the book “Good to Great” is “15”.

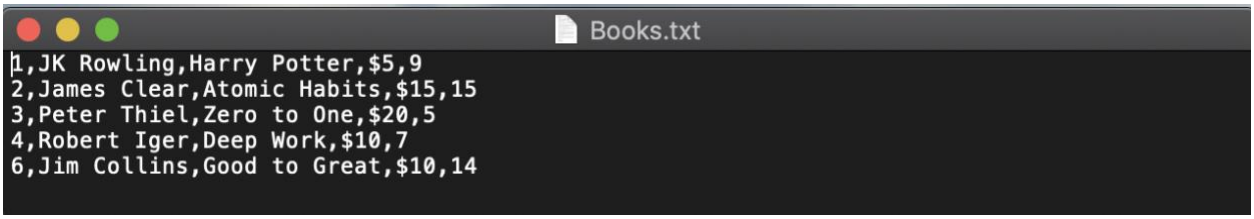

Figure 14: Screenshot of the records after borrowing

This is screenshot after borrowing of the books.txt file where all the records of the books are stored and the quantity gets decreased by one of the particular borrowed books. Here the quantity of the book “Harry Potter” is decreased by one which is ‘9’ and the quantity of the book “Good to Great” is also decreased by one which is “14”.

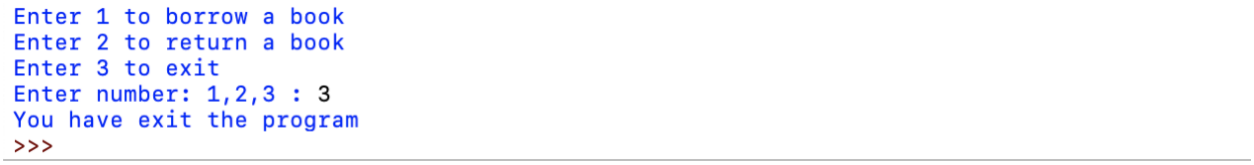

Figure 15: Screenshot after terminating the program

At last the program again asks the user to input three different options as 1 to borrow a book, 2 to return a book and 3 to exit the program. After entering “3” finally user exits from the program and a message saying “You have exit the program” is displayed.

3.2 Return Process

```
>>>
============ RESTART: /Users/sarthakrana/Desktop/SBR_PYTHOPN/Main.py ===========
+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
Book ID          Author           Book Name                  Price          Quantity
+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
   1           JK Rowling        Harry Potter               $5               10
   2           James Clear       Atomic Habits              $15              15
   3           Peter Thiel       Zero to One                $20              5
   4           Robert Iger       The ride of a lifetime     $15              10
   5           Cal Newport       Deep Work                  $10              7
   6           Jim Collins       Good to Great              $10              15
+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
Enter 1 to borrow a book
Enter 2 to return a book
Enter 3 to exit
Enter number: 1,2,3 : |
```

Figure 16: Screenshot displaying when the program is run in the shell

At first, when the program was run in the shell it displays all the records of the books in a tabular format. Then it takes input from the users as three different options as 1 to borrow a book, 2 to return a book and 3 to exit the program.

```
Enter 1 to borrow a book
Enter 2 to return a book
Enter 3 to exit
Enter number: 1,2,3 : 2
Enter the costumer name: Justin
Enter BookID: 3
Enter for how many days you have borrowed the book: 8
+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
Book ID          Author           Book Name                  Price          Quantity
+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
   1           JK Rowling        Harry Potter               $5               10
   2           James Clear       Atomic Habits              $15              15
   3           Peter Thiel       Zero to One                $20              6
   4           Robert Iger       The ride of a lifetime     $15              10
   5           Cal Newport       Deep Work                  $10              7
   6           Jim Collins       Good to Great              $10              15
+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

Figure 17: Screenshot displaying after returning the book with bookID 3 by Justin without fine

When the users enter the option 2 which means to return a book, the program asks the user to enter the customer name. After the users enter the customer name the program asks to enter the book id which he/she has borrowed previously from the library. Here “3” is entered as the book id which is referred to the “Zero to one” book. The program asks the user to enter for how many long days he/she has borrowed the book. Since, the user borrowed the book only for ‘8’ days and returned on time so he/she doesn’t gets charged for the fine. A single quantity is increased from the particular book in the records which is displayed in the shell.

```
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
Book ID          Author           Book Name                 Price          Quantity
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
   1           JK Rowling        Harry Potter               $5               10
   2           James Clear       Atomic Habits              $15              15
   3           Peter Thiel       Zero to One                $20              6
   4           Robert Iger       The ride of a lifetime     $15              10
   5           Cal Newport       Deep Work                  $10              7
   6           Jim Collins       Good to Great              $10              15
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
Enter 1 to borrow a book
Enter 2 to return a book
Enter 3 to exit
Enter number: 1,2,3 :
```

Figure 18: Screenshot displaying when the program exits from the loop and again asks for the input

Here in the program again runs back the loop and asks the user to give input as three different options as 1 to borrow a book, 2 to return a book and 3 to exit the program.

```
Enter 1 to borrow a book
Enter 2 to return a book
Enter 3 to exit
Enter number: 1,2,3 : 2
Enter the costumer name: Olivia
Enter BookID: 2
Enter for how many days you have borrowed the book: 25
You have charged with a fine of 1$ per day.
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
Book ID          Author           Book Name                 Price          Quantity
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
   1           JK Rowling        Harry Potter               $5               10
   2           James Clear       Atomic Habits              $15              16
   3           Peter Thiel       Zero to One                $20              6
   4           Robert Iger       The ride of a lifetime     $15              10
   5           Cal Newport       Deep Work                  $10              7
   6           Jim Collins       Good to Great              $10              15
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

Figure 19: Screenshot displaying after returning the book with bookID 2 by Olivia with fine

When the users enter the option 2 which means to return a book, the program asks the user to enter the customer name. After the users enter the customer name the program asks to enter the book id which he/she has borrowed previously from the library. Here “2” is entered as the book id which is referred to the “Atomic Habits” book. The program asks the user to enter for how many long days he/she has borrowed the book. Since, the user borrowed the book for ‘25’ days and returned late so he/she gets charged for the fine of a single dollar each day. A single quantity is increased from the particular book in the records which is displayed in the shell.

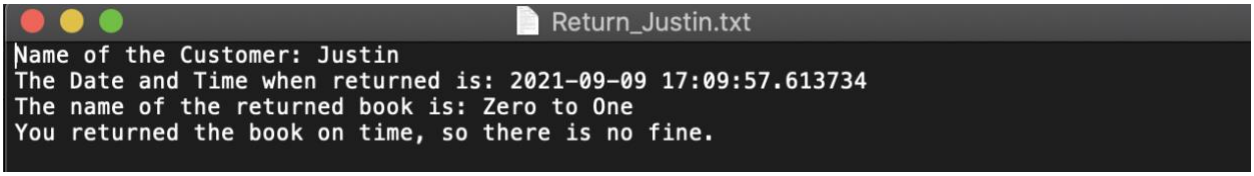

Figure 20: Screenshot displaying return bill in a txt file of Justin without fine

A return bill is generated in the name of the customer after the user returns the book which consists the name of the customer which is “Justin”, the date and time when the book was returned which is “2021-09-09” and “17:09:57:613734”, the name of the returned books which is “Zero to One” and at last “You returned the book on time, so there is no fine” message is displayed.

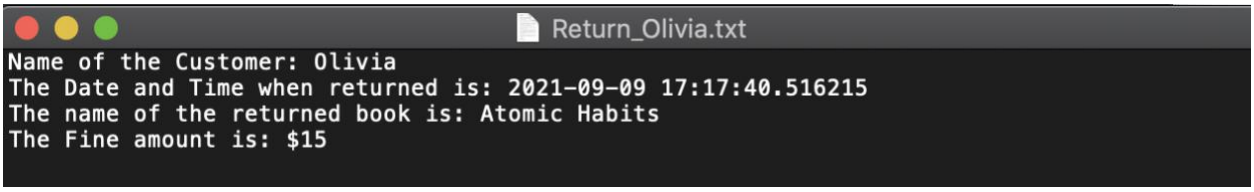

Figure 21: Screenshot displaying return bill in a txt file of Olivia with fine

A return bill is generated in the name of the customer after the user returns the book which consists the name of the customer which is “Olivia”, the date and time when the book was returned which is “2021-09-09” and “17:17:40:516215”, the name of the returned books which is “Atomic Habits” and at last the fine amount as per the code which is set as one dollar per day.

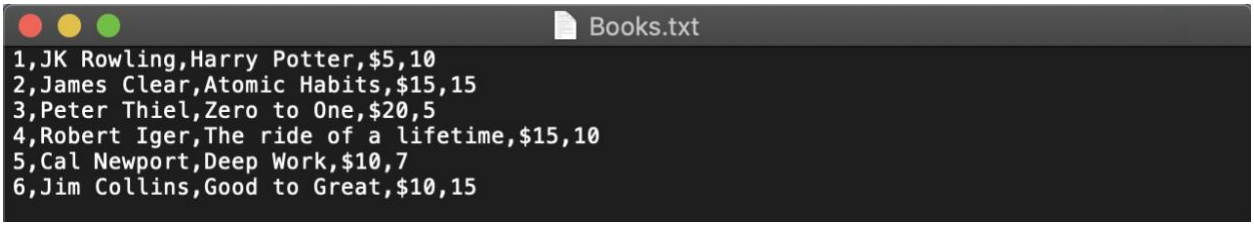

Figure 22: Screenshot of the records before returning

This is screenshot before returning of the books.txt file where all the records of the books are stored. Here the quantity of the book “Atomic Habits” is ‘15’ and the quantity of the book “Zero to one is “5”.

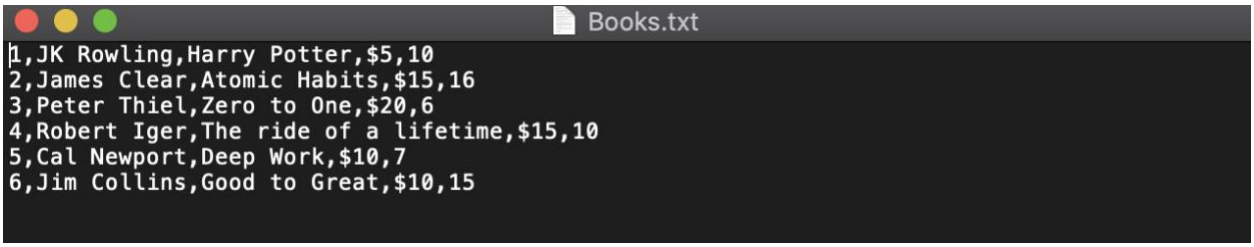

Figure 23: Screenshot of the records after returning

This is screenshot after returning of the books.txt file where all the records of the books are stored and the quantity gets increased by one of the particular returned books. Here the quantity of the book “Atomic Habits” is ‘16’ and the quantity of the book “Good to Great” is “6”.

4. Testing

4.1 Test 1

Table 1: Testing the implementation of try and except

Objective	Testing implementation of try and except.
Action	- The program was run in shell. - An integer was entered as invalid value while assigning. - "fr" was assigned .
Expected Result	An error message saying "The given value is not an integer" should display.
Actual Result	An error message saying "The given value is not an integer" was displayed.
Conclusion	Test successful.

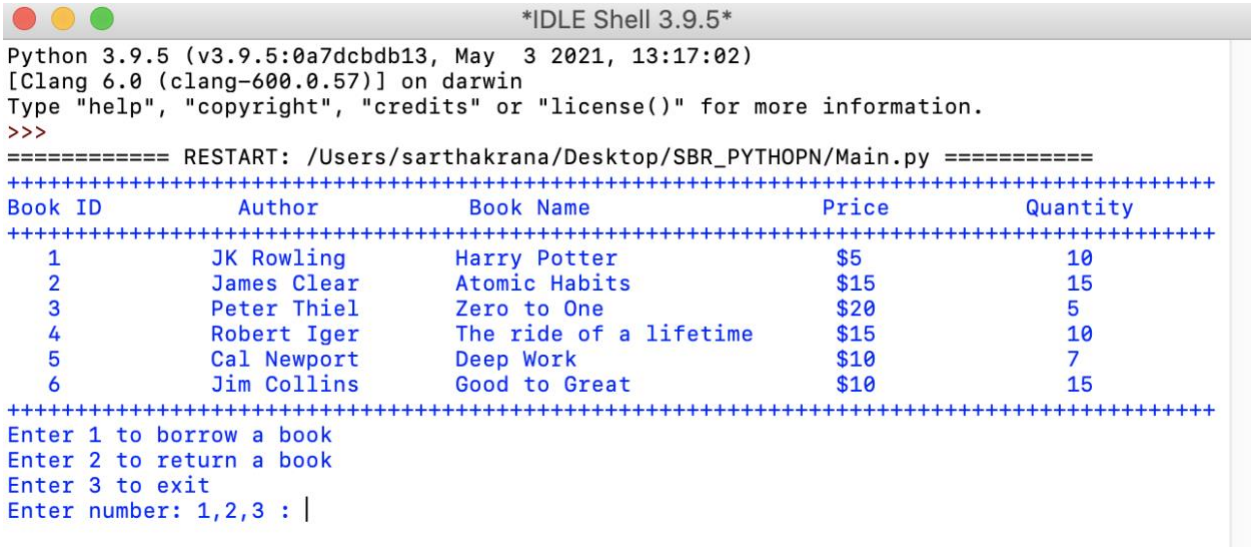

Figure 24: Screenshot displaying when the program is run in the shell

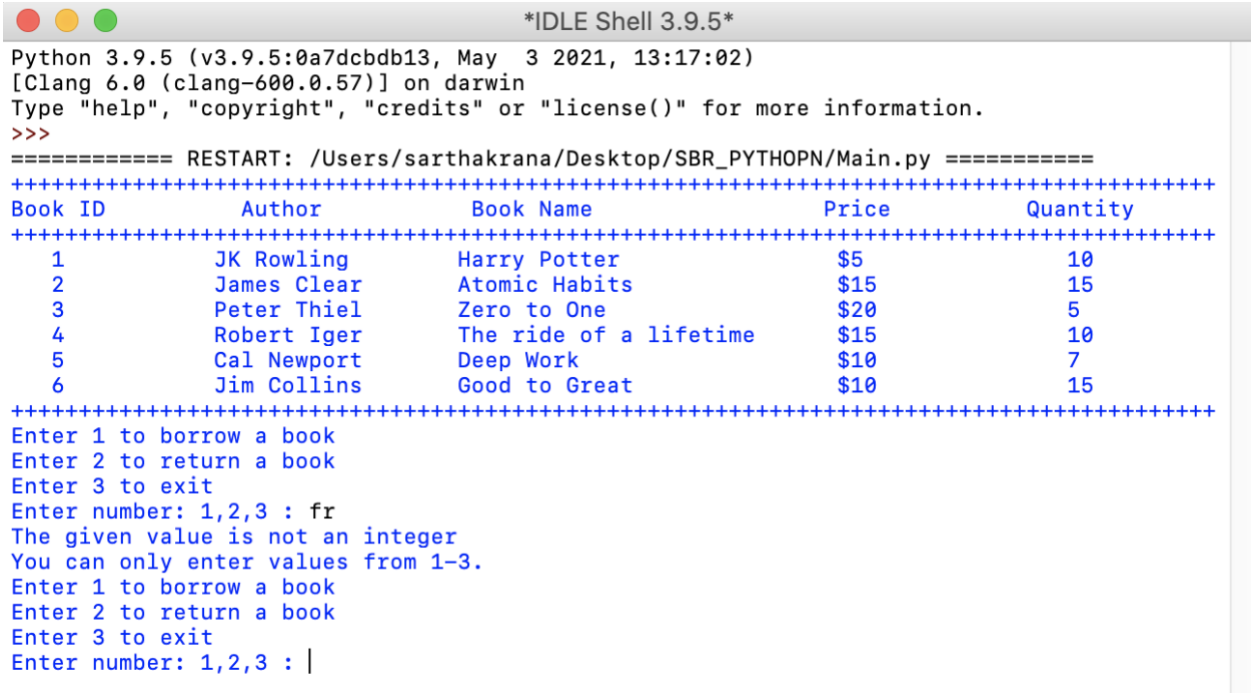

Figure 25: Screenshot of implementation of try and except

4.2 Test 2

Table 2: Testing the selection borrow and return option

Objective	Testing selection borrow and return option.
Action	- The program was run in shell. - A negative value was entered while assigning. - A non-existing value was entered while assigning. - “-3” was assigned. - “7” was assigned.
Expected Result	An error message saying “You can only enter values from 1-3” should display.
Actual Result	An error message saying “You can only enter values from 1-3” was displayed.
Conclusion	Test successful.

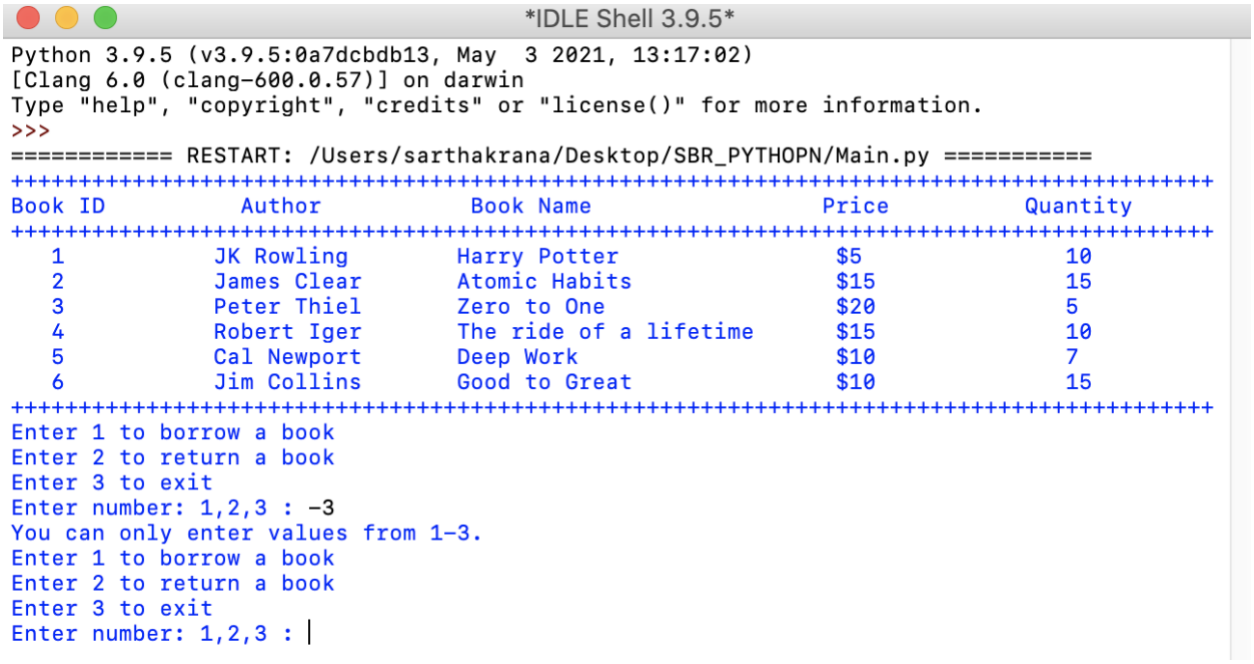

Figure 26: Screenshot of entering a negative value

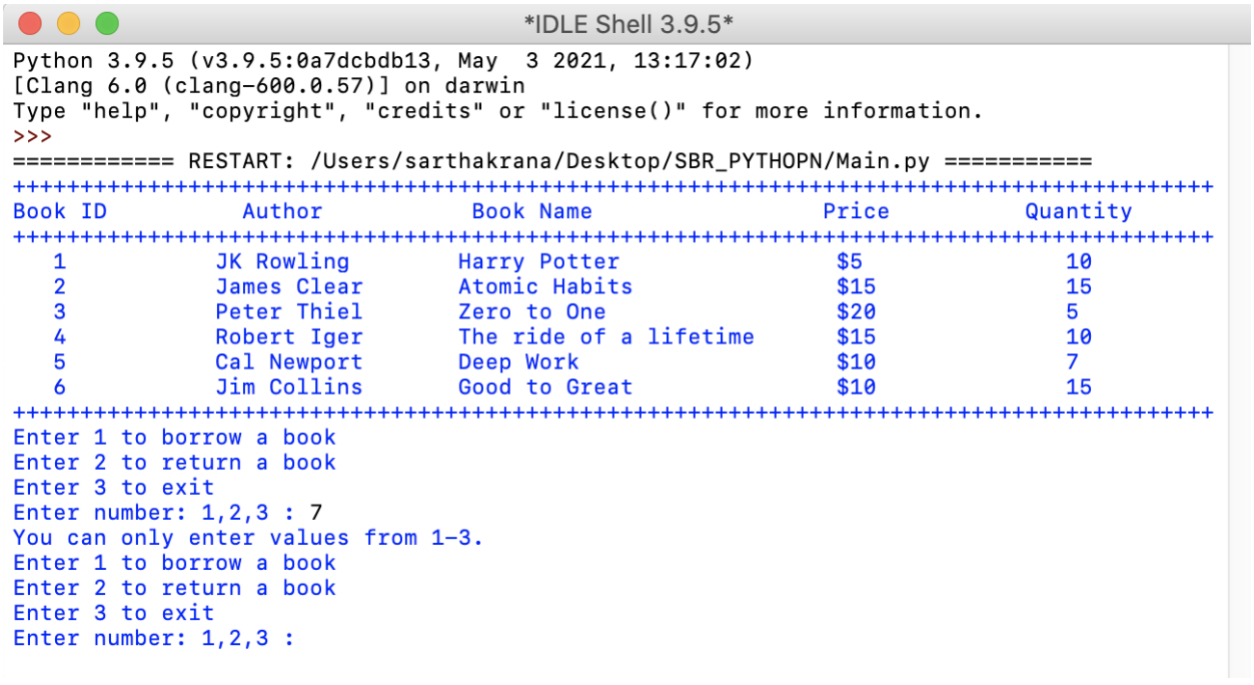

Figure 27: Screenshot of entering a non-existing value

4.3 Test 3

Table 3: Testing the file generation of borrow

Objective	Testing file generation of borrow.
Action	- The program was run in shell. - Complete borrow process was carried out. - The book whose Book ID is 1 was borrowed. - Decrease in quantity of that book in shell was shown. - A new borrow text file was shown.
Expected Result	The book should be borrowed successfully with a decrease in quantity and a borrow txt should generate.
Actual Result	The book was borrowed successfully with a decrease in quantity and a borrow txt file was generated.
Conclusion	Test successful.

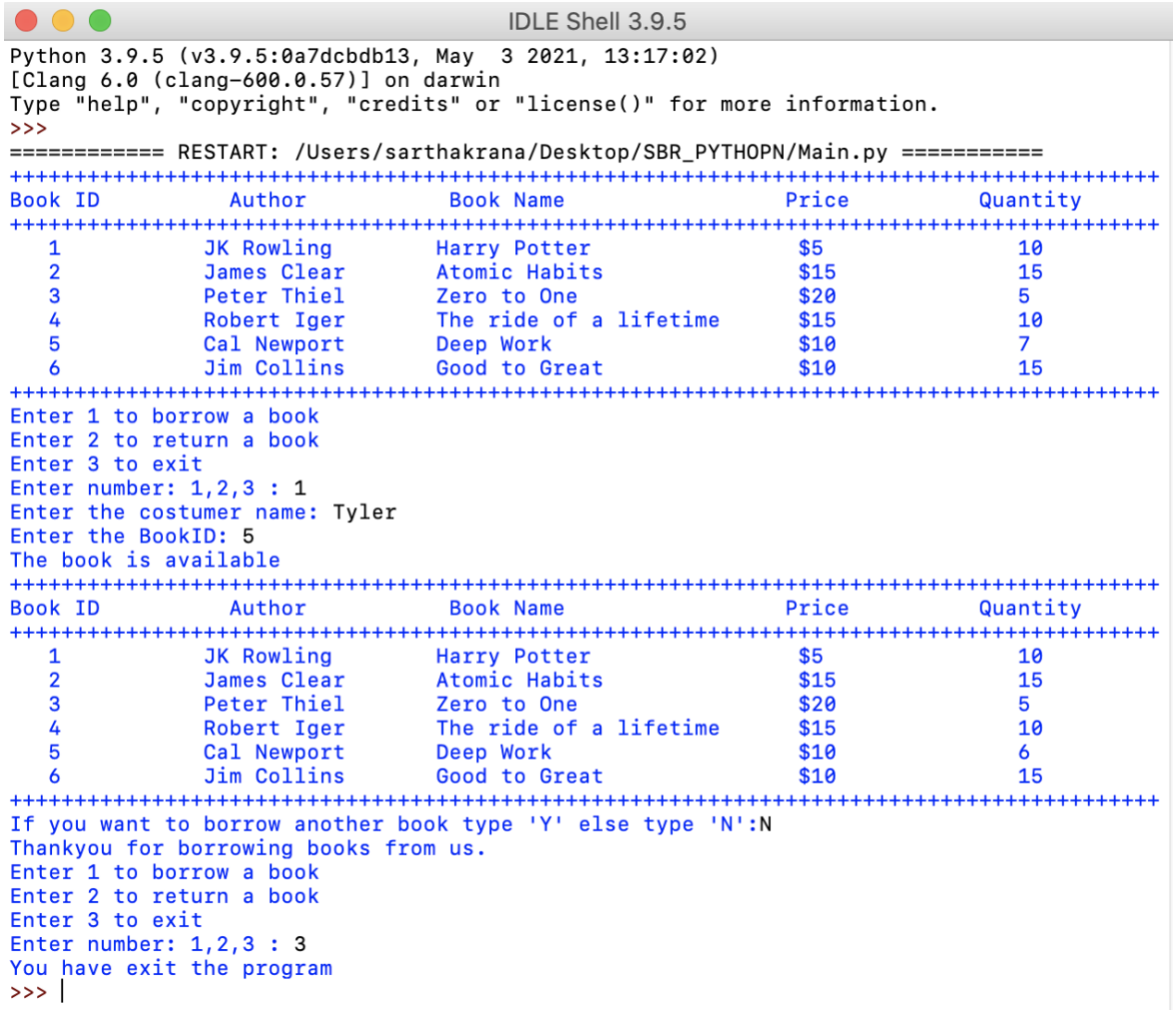

Figure 28: Screenshot of complete borrow process

Book ID	Author	Book Name	Price	Quantity
1	JK Rowling	Harry Potter	$5	10
2	James Clear	Atomic Habits	$15	15
3	Peter Thiel	Zero to One	$20	5
4	Robert Iger	The ride of a lifetime	$15	10
5	Cal Newport	Deep Work	$10	7
6	Jim Collins	Good to Great	$10	15

Figure 29: Screenshot displaying records in shell before borrowing the book

Book ID	Author	Book Name	Price	Quantity
1	JK Rowling	Harry Potter	$5	10
2	James Clear	Atomic Habits	$15	15
3	Peter Thiel	Zero to One	$20	5
4	Robert Iger	The ride of a lifetime	$15	10
5	Cal Newport	Deep Work	$10	6
6	Jim Collins	Good to Great	$10	15

Figure 30: Screenshot displaying records in shell after borrowing the book

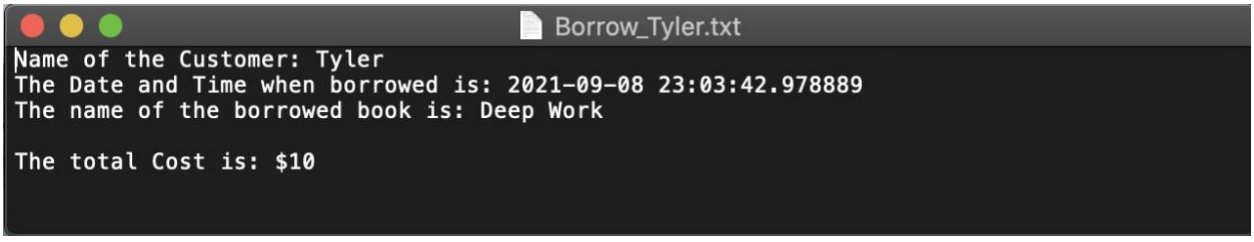

Figure 31: Screenshot of borrow bill in a txt file of Tyler

4.4 Test 4

Table 4: Testing the file generation of return

Objective	Testing file generation of return
Action	- The program was run in shell. - Complete return process was carried out. - The book whose Book ID is 3 was borrowed. - Increase in quantity of that book in shell was shown. - A new return text file was shown.
Expected Result	The book should be returned successfully with an increase in quantity and a return txt should generate.
Actual Result	The book was returned successfully with an increase in quantity and a return txt file was generated.
Conclusion	Test successful.

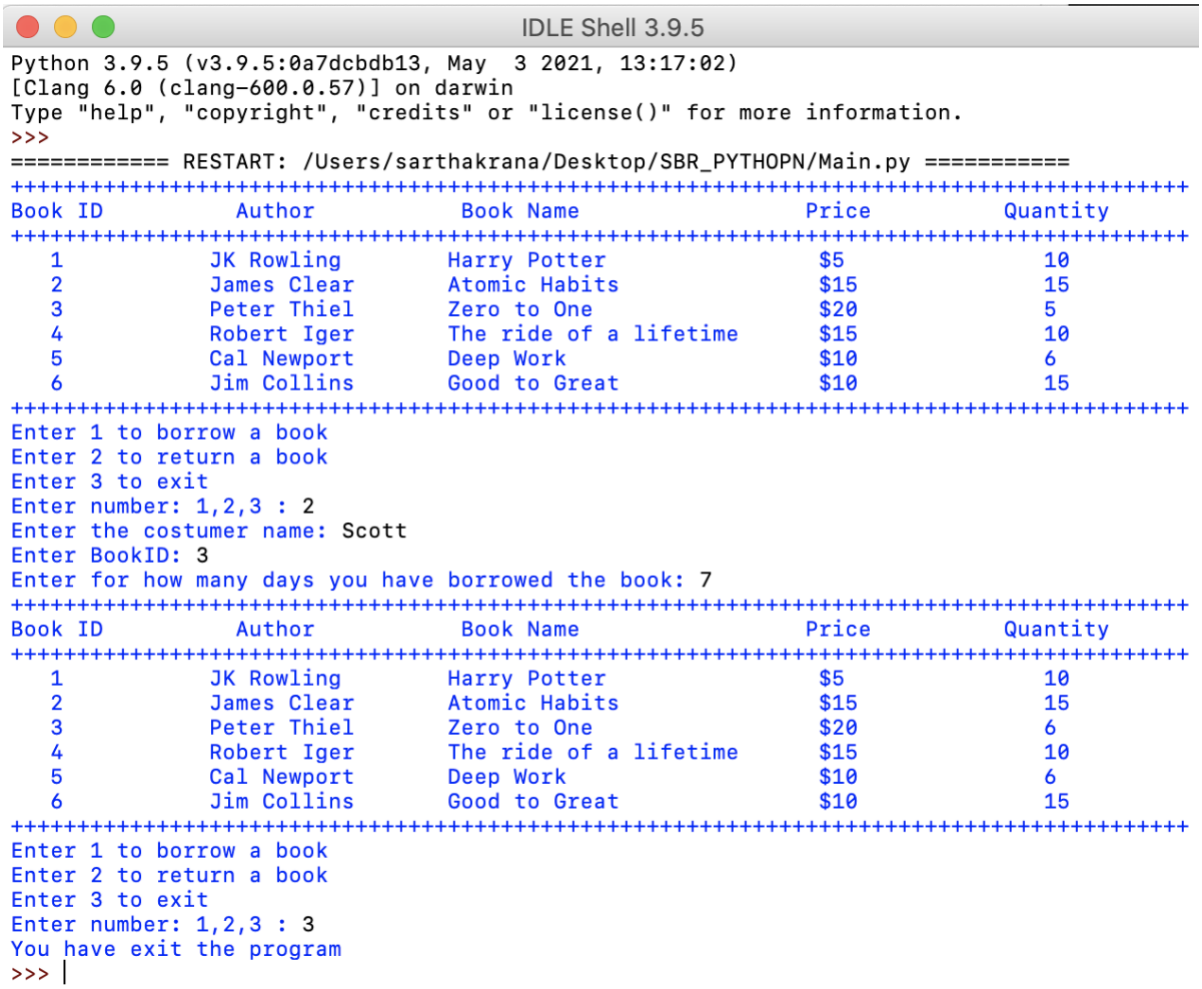

Figure 32: Screenshot of complete return process

Book ID	Author	Book Name	Price	Quantity
1	JK Rowling	Harry Potter	$5	10
2	James Clear	Atomic Habits	$15	15
3	Peter Thiel	Zero to One	$20	5
4	Robert Iger	The ride of a lifetime	$15	10
5	Cal Newport	Deep Work	$10	6
6	Jim Collins	Good to Great	$10	15

Figure 33: Screenshot displaying records in shell before returning the book

Book ID	Author	Book Name	Price	Quantity
1	JK Rowling	Harry Potter	$5	10
2	James Clear	Atomic Habits	$15	15
3	Peter Thiel	Zero to One	$20	6
4	Robert Iger	The ride of a lifetime	$15	10
5	Cal Newport	Deep Work	$10	6
6	Jim Collins	Good to Great	$10	15

Figure 34: Screenshot displaying records in shell after returning the book

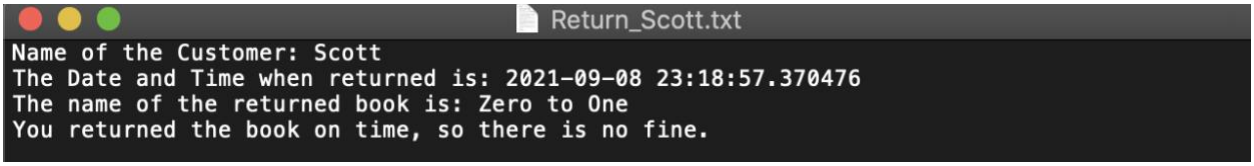

Figure 35: Screenshot of return bill in a txt file of Scott without fine

4.5 Test 5

Table 5: Testing the update in stock

Objective	Testing update in stock
Action	- The program was run in shell. - The stock of the books in books.txt file before borrowing and returning was shown. - The book whose Book ID is 3 was borrowed. - The book whose Book ID is 6 was returned.
Expected Result	The quantity of the books should decrease and increase while borrowing and returning in the books.txt file.
Actual Result	The quantity of the books was decreased and increased while the book was borrowed and returned in the books.txt file.
Conclusion	Test successful.

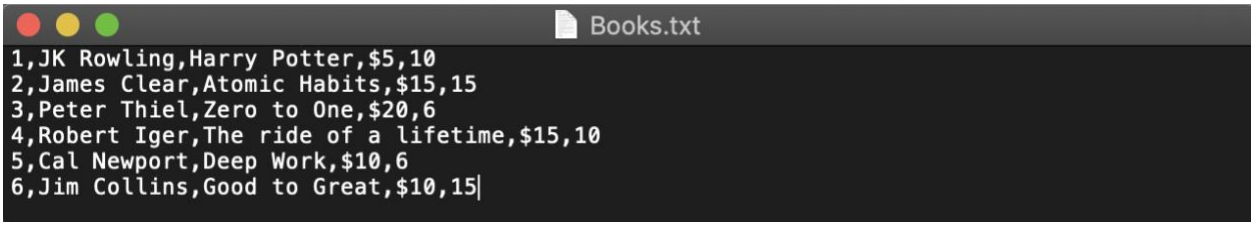

Figure 36: Screenshot of the stocks before borrowing

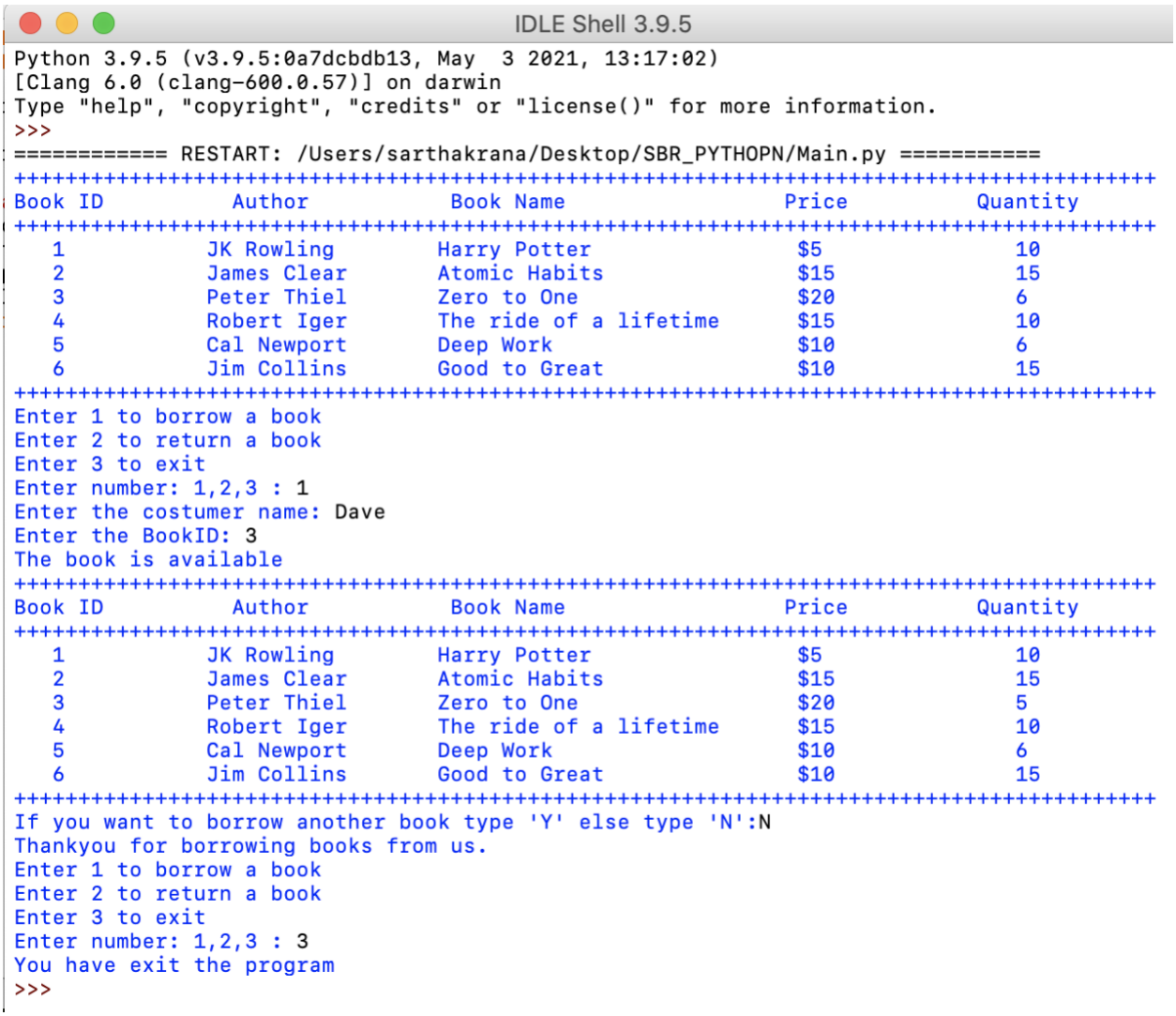

Figure 37: Screenshot of complete borrow process

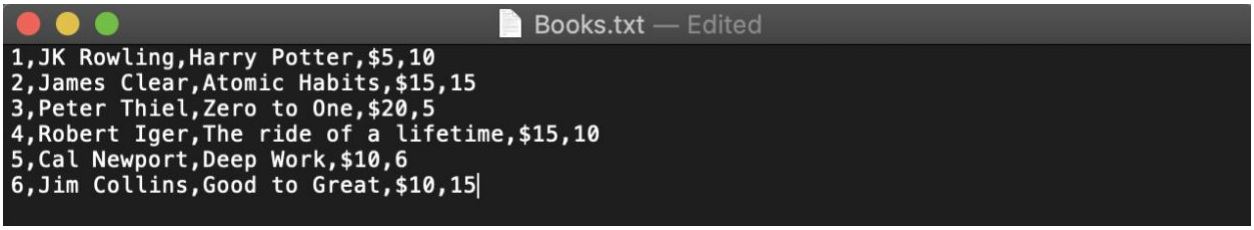

Figure 38: Screenshot displaying quantity change of bookID 3 in stock after borrowing

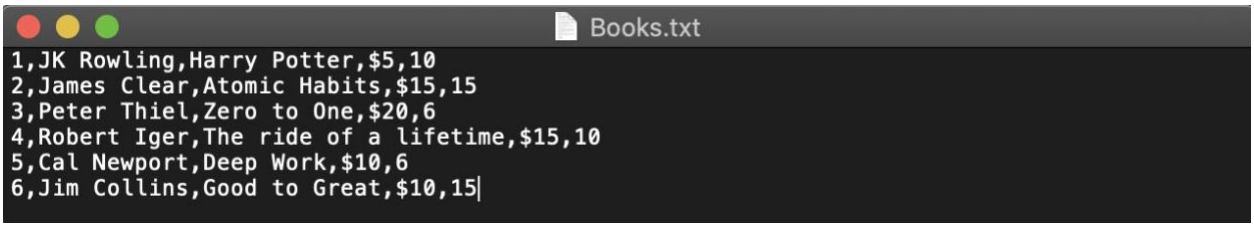

Figure 39: Screenshot of the stocks before returning

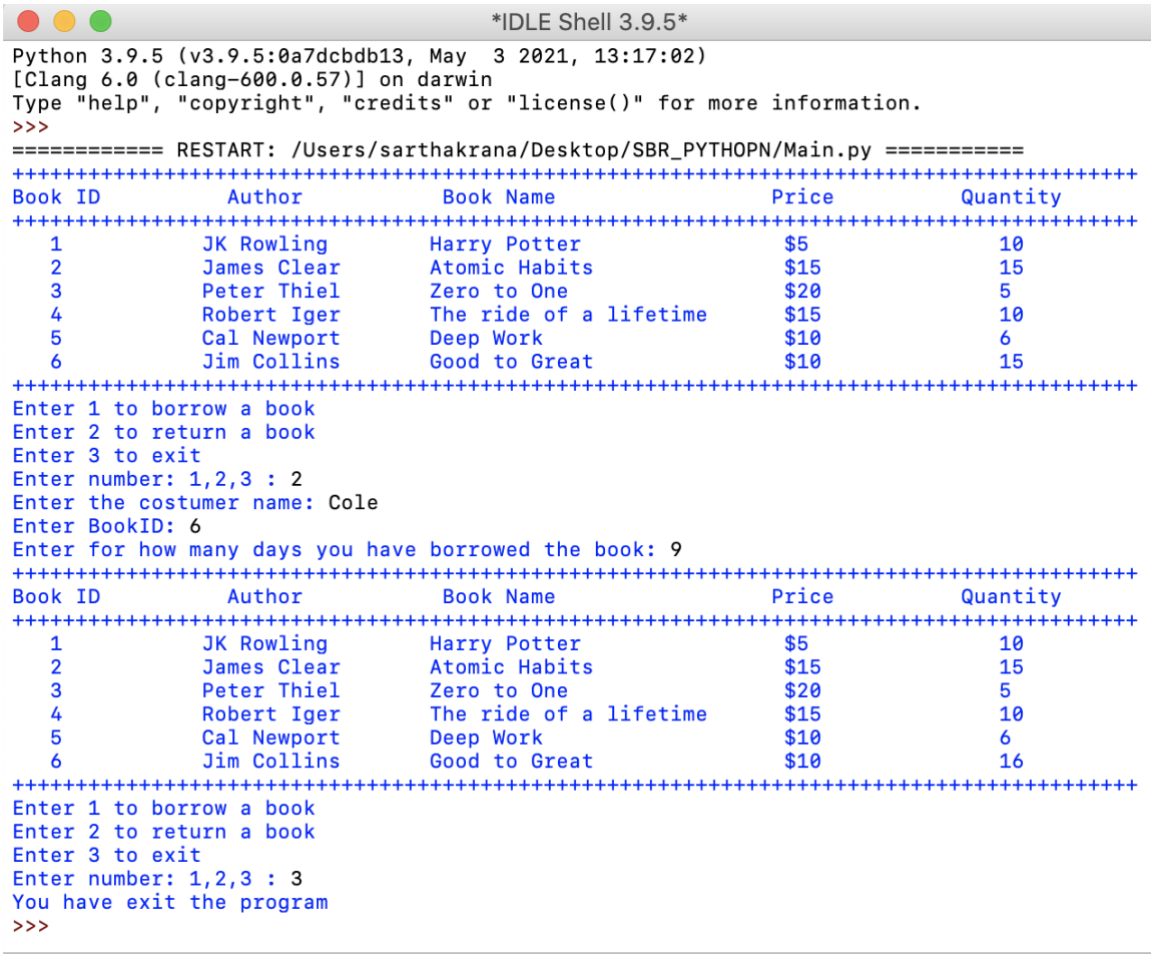

Figure 40: Screenshot of complete return process

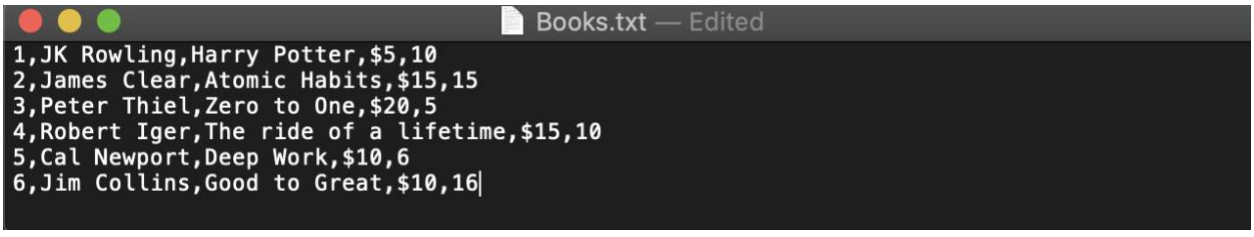

Figure 41: Screenshot displaying quantity change of bookID 6 in stock after returning

5. Conclusion

Finally, the library management system development was completed successfully. This library system saves all of the book's data and stocks in a text file, which is accessed in the program via a one-dimensional list. The user can borrow and return books to the library using the system. This was the first time I had learned and used the Python programming language at a higher level to create a system like this. This course covered the proper usage of functions, lists, appends, imports, and a variety of other topics.

I completed this project using resources provided by our teacher as well as several online references. Throughout the module, the weekly assignment made the concept of functions and lists crystal clear. Using the draw.io application, a flowchart of the entire code was created while developing the report for this coursework. Many problems were encountered while developing the system, especially when storing the quantity in the list. It was made easier by the use of append. Because the concept of exception handling was new to me and implementing it was required in the coursework, I had to do a lot of research through online sites and videos before thoroughly going through the contents provided by our teacher helped me to perform it. Many logical errors occurred throughout the program, particularly when the program was run in the shell, and particularly where exception handling was used, resulting in the program crashing and displaying the same message multiple times due to the lack of use of ‘break'. These types of errors were discovered during the program's testing, and they were corrected.

This course covers a variety of Python programming language topics such as functions, lists, append, import, and many more. In a program, functions are used to bundle a set of instructions that we want to use repeatedly throughout the program, list to store the data, append to add a single item to an existing list, import to link the various files and access their data, and import the real world date and time. This aided in the development of an appropriate working library management system based on the overall scenario of the coursework.

Appendix

- Module: Main.py

```
import Borrow #importing Borrow.py
import Return #importing Return.py

'''Initializing dlist in borrow 1D list'''
dlist = Borrow.OneDlsit()
'''CallingDisplay function'''
Borrow.display(dlist)

'''Main Function'''
bookName = "" #initializing bookName to empty string
totalCost = 0 #initializing totalCost to 0
inputNumber = 0 #initializing inoutNumber to 0
valid = True
while inputNumber != 3: #when inputNumber is not equal to 3
    print("Enter 1 to borrow a book")
    print("Enter 2 to return a book")
    print("Enter 3 to exit")
    while valid == True:
        '''Exception Handling'''
        try:
            inputNumber = int(input("Enter number: 1,2,3 : "))
            break
        except:
            print("The given value is not an integer")
            break
    if inputNumber == 1: #when inoutNumber equals to 1
        name = input("Enter the costumer name: ")
        Cost, bookName = Borrow.borrowContinuation(dlist, Borrow.display, totalCost,
bookName, name)
        Borrow.Bill(Cost, bookName, name)

    elif inputNumber == 2: #when inoutNumber equals to 2
        name = input("Enter the costumer name: ")
        Return.returnBook(dlist, name, Return.display)

    elif inputNumber == 3: #when inoutNumber equals to 3
        print("You have exit the program")

    else:
        print("You can only enter values from 1-3.")
```

- Module: Borrow.py

```
'''importing date and time'''
import datetime

'''creating functions'''
'''To convert the text file into 1D list'''
def OneDlsit():
   file = open("Books.txt", "r")
   double_dlist = []
   single_dlist = []
   for line in file:
      line = line.replace("\n", "")
      double_dlist.append(line.split(","))
   for i in range(len(double_dlist)):
      for j in range(len(double_dlist[i])):
         single_dlist.append(double_dlist[i][j])
   return single_dlist

'''To display values of dictionary in a table format'''
def display(dlist):
   '''Column name'''

print("++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
+++++++++++++++++++++++++\nBook ID         Author          Book Name
Price
Quantity\n++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++++++++++++++")
   '''All the data of the list in the table'''
   print("  ", dlist[0], "        ", dlist[1], "      ", dlist[2], "            ", dlist[3], "            ", dlist[4])
   print("  ", dlist[5], "        ", dlist[6], "     ", dlist[7], "            ", dlist[8], "           ", dlist[9])
   print("  ", dlist[10], "        ", dlist[11], "     ", dlist[12], "              ", dlist[13], "            ",
dlist[14])
   print("  ", dlist[15], "        ", dlist[16], "     ", dlist[17], "    ", dlist[18], "            ", dlist[19])
   print("  ", dlist[20], "        ", dlist[21], "     ", dlist[22], "                ", dlist[23], "            ",
dlist[24])
   print("  ", dlist[25], "        ", dlist[26], "     ", dlist[27], "            ", dlist[28], "           ",
dlist[29])

print("++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
+++++++++++++++++++++++++")

'''for displaying available books'''
def availableBooks(dlist, quantity, bookID):
   '''Appending value'''
   quantity1 = dlist[4]
```

```
    quantity2 = dlist[9]
    quantity3 = dlist[14]
    quantity4 = dlist[19]
    quantity5 = dlist[24]
    quantity6 = dlist[29]
    file = open("Books.txt", "w") #Open the books.txt file and write the following
    if bookID == 1:
        quantity1 = int(dlist[4])-1
        file.write("1,JK Rowling,Harry Potter,$5,"+str(quantity1)+"\n2,James Clear,Atomic
Habits,$15,"+quantity2+"\n3,Peter Thiel,Zero to One,$20,"+quantity3 +
                "\n4,Robert Iger,The ride of a lifetime,$15,"+quantity4+"\n5,Cal
Newport,Deep Work,$10,"+quantity5+"\n6,Jim Collins,Good to
Great,$10,"+quantity6+"")
    elif bookID == 2:
        quantity2 = int(dlist[9])-1
        file.write("1,JK Rowling,Harry Potter,$5,"+quantity1+"\n2,James Clear,Atomic
Habits,$15,"+str(quantity2)+ "\n3,Peter Thiel,Zero to One,$20,"+quantity3 +
                "\n4,Robert Iger,The ride of a lifetime,$15,"+quantity4+"\n5,Cal
Newport,Deep Work,$10,"+quantity5+"\n6,Jim Collins,Good to
Great,$10,"+quantity6+"")
    elif bookID == 3:
        quantity3 = int(dlist[14])-1
        file.write("1,JK Rowling,Harry Potter,$5,"+quantity1+"\n2,James Clear,Atomic
Habits,$15,"+quantity2+"\n3,Peter Thiel,Zero to One,$20,"+str(quantity3) +
                "\n4,Robert Iger,The ride of a lifetime,$15,"+quantity4+"\n5,Cal
Newport,Deep Work,$10,"+quantity5+"\n6,Jim Collins,Good to
Great,$10,"+quantity6+"")
    elif bookID == 4:
        quantity4 = int(dlist[19])-1
        file.write("1,JK Rowling,Harry Potter,$5,"+quantity1+"\n2,James Clear,Atomic
Habits,$15,"+quantity2+"\n3,Peter Thiel,Zero to One,$20,"+quantity3 +
                "\n4,Robert Iger,The ride of a lifetime,$15,"+str(quantity4)+"\n5,Cal
Newport,Deep Work,$10,"+quantity5+"\n6,Jim Collins,Good to
Great,$10,"+quantity6+"")
    elif bookID == 5:
        quantity5 = int(dlist[24])-1
        file.write("1,JK Rowling,Harry Potter,$5,"+quantity1+"\n2,James Clear,Atomic
Habits,$15,"+quantity2+"\n3,Peter Thiel,Zero to One,$20,"+quantity3 +
                "\n4,Robert Iger,The ride of a lifetime,$15,"+quantity4+"\n5,Cal
Newport,Deep Work,$10,"+str(quantity5)+"\n6,Jim Collins,Good to
Great,$10,"+quantity6+"")
    elif bookID == 6:
        quantity6 = int(dlist[29])-1
        file.write("1,JK Rowling,Harry Potter,$5,"+quantity1+"\n2,James Clear,Atomic
Habits,$15,"+quantity2+"\n3,Peter Thiel,Zero to One,$20,"+quantity3 +
```

```
                "\n4,Robert Iger,The ride of a lifetime,$15,"+quantity4+"\n5,Cal
Newport,Deep Work,$10,"+quantity5+"\n6,Jim Collins,Good to
Great,$10,"+str(quantity6)+"")
    file.close()
    RemainingQuantity = quantity - 1
    for i in range(len(dlist)):
        for j in range(len(dlist[i])):
            if bookID == 1:
                dlist[4] = str(RemainingQuantity)
            elif bookID == 2:
                dlist[9] = str(RemainingQuantity)
            elif bookID == 3:
                dlist[14] = str(RemainingQuantity)
            elif bookID == 4:
                dlist[19] = str(RemainingQuantity)
            elif bookID == 5:
                dlist[24] = str(RemainingQuantity)
            elif bookID == 6:
                dlist[29] = str(RemainingQuantity)

'''for displaying borrowBook'''
def borrowBook(dlist):
    bookPrice = 0 #initializing bookPrice to 0
    book_Name = "" #initializing book_Name to empty string
    fg = True
    valid = True
    n_valid = True
    while valid == True:
        '''Exception Handling'''
        try:
            bookID = int(input("Enter the BookID: "))
            break
        except:
            print("The bookID should be an integer between 1-6")
    while fg == True:
        if bookID == 1:
            quantity = int(dlist[4])
            if quantity > 0:
                displayAvailable()
                availableBooks(dlist, quantity, bookID)
                display(dlist)
                bookPrice = Price(dlist, bookID)
                book_Name = N_book(dlist, bookID)
                break
            else:
                while n_valid == True:
```

```
                '''Exception Handling'''
                try:
                    bookID = int(input("The book you want to borrow is out of stock, please
enter another bookID if you want to borrow: "))
                    break
                except:
                    print("ID should be a integer between 1-6")
        elif bookID == 2:
            quantity = int(dlist[9])
            if quantity > 0:
                displayAvailable()
                availableBooks(dlist, quantity, bookID)
                display(dlist)
                bookPrice = Price(dlist, bookID)
                book_Name = N_book(dlist, bookID)
                break
            else:
                while n_valid == True:
                    '''Exception Handling'''
                    try:
                        bookID = int(input("The book you want to borrow is out of stock, please
enter another bookID if you want to borrow: "))
                        break
                    except:
                        print("ID should be a integer between 1-6")
        elif bookID == 3:
            quantity = int(dlist[14])
            if quantity > 0:
                displayAvailable()
                availableBooks(dlist, quantity, bookID)
                display(dlist)
                bookPrice = Price(dlist, bookID)
                book_Name = N_book(dlist, bookID)
                break
            else:
                while n_valid == True:
                    '''Exception Handling'''
                    try:
                        bookID = int(input("The book you want to borrow is out of stock, please
enter another bookID if you want to borrow: "))
                        break
                    except:
                        print("ID should be a integer between 1-6")
        elif bookID == 4:
            quantity = int(dlist[19])
            if quantity > 0:
```

```
            displayAvailable()
            availableBooks(dlist, quantity, bookID)
            display(dlist)
            bookPrice = Price(dlist, bookID)
            book_Name = N_book(dlist, bookID)
            break
        else:
            while n_valid == True:
                '''Exception Handling'''
                try:
                    bookID = int(input("The book you want to borrow is out of stock, please
enter another bookID if you want to borrow: "))
                    break
                except:
                    print("ID should be a integer between 1-6")
    elif bookID == 5:
        quantity = int(dlist[24])
        if quantity > 0:
            displayAvailable()
            availableBooks(dlist, quantity, bookID)
            display(dlist)
            bookPrice = Price(dlist, bookID)
            book_Name = N_book(dlist, bookID)
            break
        else:
            while n_valid == True:
                '''Exception Handling'''
                try:
                    bookID = int(input("The book you want to borrow is out of stock, please
enter another bookID if you want to borrow: "))
                    break
                except:
                    print("ID should be a integer between 1-6")
    elif bookID == 6:
        quantity = int(dlist[29])
        if quantity > 0:
            displayAvailable()
            availableBooks(dlist, quantity, bookID)
            display(dlist)
            bookPrice = Price(dlist, bookID)
            book_Name = N_book(dlist, bookID)
            break
        else:
            while n_valid == True:
                '''Exception Handling'''
                try:
```

```
                    bookID = int(input("The book you want to borrow is out of stock, please
enter another bookID if you want to borrow: "))
                    break
                except:
                    print("ID should be a integer between 1-6")
        else:
            print("Error")
            break
    return bookPrice, book_Name

'''for seting price of the books of each book'''
def Price(dlist, bookID):
    if bookID == 1:
        price = dlist[3]
        price = price.replace("$", "")
    elif bookID == 2:
        price = dlist[8]
        price = price.replace("$", "")
    elif bookID == 3:
        price = dlist[13]
        price = price.replace("$", "")
    elif bookID == 4:
        price = dlist[18]
        price = price.replace("$", "")
    elif bookID == 5:
        price = dlist[23]
        price = price.replace("$", "")
    elif bookID == 6:
        price = dlist[28]
        price = price.replace("$", "")
    return price

'''for displaying Number of books'''
def N_book(dlist, bookID):
    book_Name = ""
    if bookID == 1:
        book_Name = dlist[2]
    elif bookID == 2:
        book_Name = dlist[7]
    elif bookID == 3:
        book_Name = dlist[12]
    elif bookID == 4:
        book_Name = dlist[17]
    elif bookID == 5:
        book_Name = dlist[22]
    elif bookID == 6:
```

```
        book_Name = dlist[27]
    return book_Name

'''when the user wants to continue borrow'''
def borrowContinuation(dlist, display, totalCost, bookName, name):
    fg = True
    book_Cost, book_Name = borrowBook(dlist) #call borrowBook
    totalCost += int(book_Cost) #set total cost to the total cost + book_Cost
    bookName += book_Name+"\n" #set bookName to the bookName + book_Name
    while fg == True:
        Yes_No = input("If you want to borrow another book type 'Y' else type 'N':")
        if Yes_No == "Y":
            book_Cost, book_Name = borrowBook(dlist)
            totalCost += int(book_Cost)
            bookName += book_Name+"\n"
        elif Yes_No == "N":
            print("Thankyou for borrowing books from us.")
            break
        else:
            print("Invalid Input")
    return totalCost, bookName

'''to develop borrow bill with a unique name'''
def Bill(Cost, bookName, name):
    dateNtime = datetime.datetime.now()
    minute = str(datetime.datetime.now().minute)
    second = str(datetime.datetime.now().second)
    microsecond = str(datetime.datetime.now().microsecond)
    file = open("Borrow_"+name+".txt", "w")
    file.write("Name of the Customer: "+name+"\n")
    file.write("The Date and Time when borrowed is: "+str(dateNtime)+"\n")
    file.write("The name of the borrowed book is: "+bookName+"\n")
    file.write("The total Cost is: $"+str(Cost)+"\n")
    file.close()

'''When the quantity of the book is left on the records'''
def displayAvailable():
    print("The book is available")
```

- Module: Return.py

```
'''importing date and time'''
import datetime

'''ccreating function'''
```

```
'''To convert the text file imtp 1D list'''
def OneDlsit():
    file = open("Books.txt", "r")
    double_dlist = []
    single_dlist = []
    for line in file:
        line = line.replace("\n", "")
        double_dlist.append(line.split(","))
    for i in range(len(double_dlist)):
        for j in range(len(double_dlist[i])):
            single_dlist.append(double_dlist[i][j])
    return single_dlist

'''To display values of dictionary in a table format'''
def display(dlist):
    '''Column name'''

print("++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++\nBook ID          Author          Book Name                                Price
Quantity\n++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++")
    '''All the data of the list in the table'''
    print("  ", dlist[0], "          ", dlist[1], "       ", dlist[2], "              ", dlist[3], "              ", dlist[4])
    print("  ", dlist[5], "          ", dlist[6], "      ", dlist[7], "              ", dlist[8], "              ", dlist[9])
    print("  ", dlist[10], "          ", dlist[11], "      ", dlist[12], "                ", dlist[13], "              ",
dlist[14])
    print("  ", dlist[15], "          ", dlist[16], "      ", dlist[17], "    ", dlist[18], "              ", dlist[19])
    print("  ", dlist[20], "          ", dlist[21], "      ", dlist[22], "                ", dlist[23], "              ",
dlist[24])
    print("  ", dlist[25], "          ", dlist[26], "      ", dlist[27], "              ", dlist[28], "              ",
dlist[29])

print("++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++")


'''for displaying available books after returning'''
def addBooks(dlist, quantity, bookID):
    '''Appending value'''
    quantity1 = dlist[4]
    quantity2 = dlist[9]
    quantity3 = dlist[14]
    quantity4 = dlist[19]
    quantity5 = dlist[24]
    quantity6 = dlist[29]
```

```
    file = open("Books.txt", "w") #Open the books.txt file and write the following
    if bookID == 1:
        quantity1 = int(dlist[4])+1
        file.write("1,JK Rowling,Harry Potter,$5,"+str(quantity1)+"\n2,James Clear,Atomic
Habits,$15,"+quantity2+"\n3,Peter Thiel,Zero to One,$20,"+quantity3 +
                "\n4,Robert Iger,The ride of a lifetime,$15,"+quantity4+"\n5,Cal
Newport,Deep Work,$10,"+quantity5+"\n6,Jim Collins,Good to
Great,$10,"+quantity6+"")
    elif bookID == 2:
        quantity2 = int(dlist[9])+1
        file.write("1,JK Rowling,Harry Potter,$5,"+quantity1+"\n2,James Clear,Atomic
Habits,$15,"+str(quantity2)+ "\n3,Peter Thiel,Zero to One,$20,"+quantity3 +
                "\n4,Robert Iger,The ride of a lifetime,$15,"+quantity4+"\n5,Cal
Newport,Deep Work,$10,"+quantity5+"\n6,Jim Collins,Good to
Great,$10,"+quantity6+"")
    elif bookID == 3:
        quantity3 = int(dlist[14])+1
        file.write("1,JK Rowling,Harry Potter,$5,"+quantity1+"\n2,James Clear,Atomic
Habits,$15,"+quantity2+"\n3,Peter Thiel,Zero to One,$20,"+str(quantity3) +
                "\n4,Robert Iger,The ride of a lifetime,$15,"+quantity4+"\n5,Cal
Newport,Deep Work,$10,"+quantity5+"\n6,Jim Collins,Good to
Great,$10,"+quantity6+"")
    elif bookID == 4:
        quantity4 = int(dlist[19])+1
        file.write("1,JK Rowling,Harry Potter,$5,"+quantity1+"\n2,James Clear,Atomic
Habits,$15,"+quantity2+"\n3,Peter Thiel,Zero to One,$20,"+quantity3 +
                "\n4,Robert Iger,The ride of a lifetime,$15,"+str(quantity4)+"\n5,Cal
Newport,Deep Work,$10,"+quantity5+"\n6,Jim Collins,Good to
Great,$10,"+quantity6+"")
    elif bookID == 5:
        quantity5 = int(dlist[24])+1
        file.write("1,JK Rowling,Harry Potter,$5,"+quantity1+"\n2,James Clear,Atomic
Habits,$15,"+quantity2+"\n3,Peter Thiel,Zero to One,$20,"+quantity3 +
                "\n4,Robert Iger,The ride of a lifetime,$15,"+quantity4+"\n5,Cal
Newport,Deep Work,$10,"+str(quantity5)+"\n6,Jim Collins,Good to
Great,$10,"+quantity6+"")
    elif bookID == 6:
        quantity6 = int(dlist[29])+1
        file.write("1,JK Rowling,Harry Potter,$5,"+quantity1+"\n2,James Clear,Atomic
Habits,$15,"+quantity2+"\n3,Peter Thiel,Zero to One,$20,"+quantity3 +
                "\n4,Robert Iger,The ride of a lifetime,$15,"+quantity4+"\n5,Cal
Newport,Deep Work,$10,"+quantity5+"\n6,Jim Collins,Good to
Great,$10,"+str(quantity6)+"")
    file.close()
    RemainingQuantity = quantity + 1
    for i in range(len(dlist)):
```

```
        for j in range(len(dlist[i])):
            if bookID == 1:
                dlist[4] = str(RemainingQuantity)
            elif bookID == 2:
                dlist[9] = str(RemainingQuantity)
            elif bookID == 3:
                dlist[14] = str(RemainingQuantity)
            elif bookID == 4:
                dlist[19] = str(RemainingQuantity)
            elif bookID == 5:
                dlist[24] = str(RemainingQuantity)
            elif bookID == 6:
                dlist[29] = str(RemainingQuantity)

'''for displaying returnBook'''
def returnBook(dlist, name, display):
    fg = True
    valid = True
    n_valid = True
    while valid == True:
        '''Exception Handling'''
        try:
            bookID = int(input("Enter BookID: "))
            break
        except:
            print("The bookID should be an integer between 1-6")
    while valid == True:
        '''Exception Handling'''
        try:
            noOfDays = int(input("Enter for how many days you have borrowed the book: "))
            break
        except:
            print("Enter an integer")
    bookFine(noOfDays, name, dlist, bookID)
    while fg == True:
        if bookID == 1:
            quantity = int(dlist[4])
            addBooks(dlist, quantity, bookID)
            display(dlist)
            break
        elif bookID == 2:
            quantity = int(dlist[9])
            addBooks(dlist, quantity, bookID)
            display(dlist)
            break
        elif bookID == 3:
```

```
            quantity = int(dlist[14])
            addBooks(dlist, quantity, bookID)
            display(dlist)
            break
        elif bookID == 4:
            quantity = int(dlist[19])
            addBooks(dlist, quantity, bookID)
            display(dlist)
            break
        elif bookID == 5:
            quantity = int(dlist[24])
            addBooks(dlist, quantity, bookID)
            display(dlist)
            break
        elif bookID == 6:
            quantity = int(dlist[29])
            addBooks(dlist, quantity, bookID)
            display(dlist)
            break
        else:
            print("Error")
            break

'''for Number of books'''
def N_book(dlist, bookID):
    book_Name = "" #initialize book_Name to empty string
    if bookID == 1:
        book_Name = dlist[2]
    elif bookID == 2:
        book_Name = dlist[7]
    elif bookID == 3:
        book_Name = dlist[12]
    elif bookID == 4:
        book_Name = dlist[17]
    elif bookID == 5:
        book_Name = dlist[22]
    elif bookID == 6:
        book_Name = dlist[27]
    return book_Name


'''function for bookFine'''
def bookFine(noOfDays, name, dlist, bookID):
    if int(noOfDays) > 10:
        fineBill(name, dlist, bookID, noOfDays) #when noOfDays > 10, display fineBill
    else:
```

```
        noFineBill(name, dlist, bookID) #when noOfDays < 10, display noFineBill

'''to develop return bill with a unique name when there is fine'''
def fineBill(name, dlist, bookID, noOfDays):
    nameofbook = N_book(dlist, bookID)
    fineAmount = (int(noOfDays) - 10) * 1
    dateNtime = datetime.datetime.now()
    minute = str(datetime.datetime.now().minute)
    second = str(datetime.datetime.now().second)
    microsecond = str(datetime.datetime.now().microsecond)
    file = open("Return_"+name+".txt", "w")
    file.write("Name of the Customer: "+name+"\n")
    file.write("The Date and Time when returned is: "+str(dateNtime)+"\n")
    file.write("The name of the returned book is: "+nameofbook+"\n")
    file.write("The Fine amount is: $"+str(fineAmount)+"\n")
    file.close()
    print("You have charged with a fine of 1$ per day.")



'''to develop return bill with a unique name when there is no fine'''
def noFineBill(name, dlist, bookID):
    nameofbook = N_book(dlist, bookID)
    dateNtime = datetime.datetime.now()
    minute = str(datetime.datetime.now().minute)
    second = str(datetime.datetime.now().second)
    microsecond = str(datetime.datetime.now().microsecond)
    file = open("Return_"+name+".txt", "w")
    file.write("Name of the Customer: "+name+"\n")
    file.write("The Date and Time when returned is: "+str(dateNtime)+"\n")
    file.write("The name of the returned book is: "+nameofbook+"\n")
    file.write("You returned the book on time, so there is no fine.")
    file.close()
```

Bibliography

techopedia, 2019. *techopedia.* [Online]
Available at: https://www.techopedia.com/definition/3533/python
[Accessed 2 September 2021].

McDonnell, M., 2019. *integralist.* [Online]
Available at: https://www.integralist.co.uk/posts/data-types-and-data-structures/#data-structures
[Accessed 4 September 2021].