

**Module Code & Module Title**

**CS4051NI Fundamentals of Computing**

**Assessment Weightage & Type**

**60% Individual Coursework**

**Year and Semester**

**2020-21 Autumn**

**Student Name: Pukar Gautam**

**Group: N1**

**London Met ID: 20049200**

**College ID: NP01NT4S210015**

**Assignment Due Date: 10th Sept 2021**

**Assignment Submission Date: 10th Sept 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 4](#_Toc82176669)

[List of Tables 4](#_Toc82176670)

[Introduction 5](#_Toc82176671)

[Goals: 5](#_Toc82176672)

[Objectives: 5](#_Toc82176673)

[Discussion And Analysis: 6](#_Toc82176674)

[Python: 6](#_Toc82176675)

[Microsoft Word: 6](#_Toc82176676)

[Draw.io 7](#_Toc82176677)

[Algorithm: 8](#_Toc82176678)

[Algorithm for library management system: 8](#_Toc82176679)

[Flowchart: 9](#_Toc82176680)

[Flowchart for Library Management System 10](#_Toc82176681)

[PSEUDOCODE 11](#_Toc82176682)

[Pseudocode for main.py: 11](#_Toc82176683)

[Pseudocode for borrow.py 13](#_Toc82176684)

[Pseudocode For Return.py: 19](#_Toc82176685)

[Data Structure 25](#_Toc82176686)

[Integer: 25](#_Toc82176687)

[Float: 25](#_Toc82176688)

[String: 26](#_Toc82176689)

[Boolean: 26](#_Toc82176690)

[Lists: 26](#_Toc82176691)

[Dictionary: 27](#_Toc82176692)

[Files: 27](#_Toc82176693)

[Arrays: 28](#_Toc82176694)

[Tuples: 28](#_Toc82176695)

[Program: 29](#_Toc82176696)

[Implementation of Program: 29](#_Toc82176697)

[Showing Borrow Process: 30](#_Toc82176698)

[Showing Return process 32](#_Toc82176699)

[Bill of Borrow: 33](#_Toc82176700)

[Bill of Return: 33](#_Toc82176701)

[Creation Of text file 34](#_Toc82176702)

[Program Termination with Exit: 34](#_Toc82176703)

[Testing: 35](#_Toc82176704)

[Test 1: Show implementation of try, except 35](#_Toc82176705)

[Test 2: Selection of borrow and return 37](#_Toc82176706)

[Test 3: To Generate file of borrow, and output in shell 40](#_Toc82176707)

[Test 4: To generate file of return 43](#_Toc82176708)

[Test 5: Showing the updated stock: 1) Quantity Deducted for borrow 2) Quantity added for returning 47](#_Toc82176709)

[Conclusion: 50](#_Toc82176710)

[Appendix: 51](#_Toc82176711)

[-Main.py 51](#_Toc82176712)

[-borrow.py 52](#_Toc82176713)

[-Return.py 55](#_Toc82176714)

[References 59](#_Toc82176715)

# List of Figures

[Figure 1 Python Figure 6](#_Toc82176716)

[Figure 2 Word 7](#_Toc82176717)

[Figure 3 Draw.io Figure 7](#_Toc82176718)

[Figure 4 Flowchart 10](#_Toc82176719)

[Figure 5 Integer 25](#_Toc82176720)

[Figure 6 Float 25](#_Toc82176721)

[Figure 7 String 26](#_Toc82176722)

[Figure 8 Boolean Data Structure 26](#_Toc82176723)

[Figure 9 Lists 27](#_Toc82176724)

[Figure 10 Dictionary Data Structure 27](#_Toc82176725)

[Figure 11 Files Data Structure 28](#_Toc82176726)

[Figure 12 Implementation of Program 29](#_Toc82176727)

[Figure 13 Showing Borrow Process 31](#_Toc82176728)

[Figure 14 Showing Return Process 32](#_Toc82176729)

[Figure 15 Bill of Borrow 33](#_Toc82176730)

[Figure 16 Bill of return 33](#_Toc82176731)

[Figure 17 Creation Of txt file 34](#_Toc82176732)

[Figure 18 Termination Of program 34](#_Toc82176733)

[Figure 19 Screenshot of the test 1 with proper output has been shown below: 36](#_Toc82176734)

[Figure 20 Screenshot of the test 2 with proper output has been shown below: 38](#_Toc82176735)

[Figure 21 Screenshot of the test 2 with proper output has been shown below: for return 39](#_Toc82176736)

[Figure 22 Screenshot of the test 3 with the proper output has been shown below 41](#_Toc82176737)

[Figure 23 Bill of borrow in text file 42](#_Toc82176738)

[Figure 24 Screenshot of test 4 with the proper output has been shown: 44](#_Toc82176739)

[Figure 25 Screenshot of test 4 with the proper output has been shown: 45](#_Toc82176740)

[Figure 26 Bill of return 46](#_Toc82176741)

[Figure 27 Quantity deducted 48](#_Toc82176742)

[Figure 28 Quantity has been added 49](#_Toc82176743)

# List of Tables

[Table 1 Show implementation of try, except 35](#_Toc82176744)

[Table 2 Selection of borrow and return 37](#_Toc82176745)

[Table 3 To generate file of borrow, and output in shell 40](#_Toc82176746)

[Table 4 To generate file of return 43](#_Toc82176747)

[Table 5 Quantity Deducted , Quantity Added 47](#_Toc82176748)

# Introduction

The course work assigned to us was both hard and thrilling, since it was the first coursework for this subject and also our first time working with Python programming. Python is a high-level, general-purpose programming language that is interpreted. The use of substantial indentation in its design philosophy promotes code readability. Its language features and object-oriented approach are aimed at assisting programmers in writing clear, logical code for both small and large-scale projects.

In Fundamentals of computing, we were assigned the task of developing a library management system. A text file is used to keep track of a library's records. We were tasked with creating an application that would read a text file and show all of the books that could be borrowed. Then, for each transaction, a note including all of the information like the title of the borrowed book, the date and hour of its release, and the total money due for the loan If a person wishes to borrow several books, all of the borrowed volumes should be recorded on the note, and the total cost should be totalled up, it should be produced and saved in a file. When a person returns a book to the library, a note consisting of the borrower's name, the title of the book, and the date and hour of the return are all required. The loan period should be set to 10 days, and if the book is not returned on time, a daily fine should be levied, which should also be included in the produced file and written to the file again. After each transaction, the book stock should be updated as well.

# Goals:

* To

# Objectives:

# Discussion And Analysis:

## Python:

To make this library management system, programming language name Python, has been used. Python is one of the most widely used programming languages. Guido van Rossum created it, and it was released in 1991. It is used for, web development, software development, mathematics and system scripting. Python can be used to create web applications server. Python works on different platforms (Windows, Mac, Linux, Raspberry Pi, etc) (W3 Schools, 2021)



Figure 1 Python Figure

Making this coursework using python is something challenging. In this coursework, Library Management System has been made using python. The use of this system is to manage stock of the library, checks who borrow the book and checks fine and the date of returning of the book. It means it helps in management of library in very simple and fast way.

## Microsoft Word:

Microsoft Word, sometimes known as MS Word, is a graphical word processing tool that allows users to type. Microsoft is the firm that created it. Its primary function is to enable users to type and save documents. It has many helpful tools in it, spelling and grammar checker, word count, speech recognition. It has the facility of inserting pictures on the document. Web pages, graphs etc could be attached in MS Word. Tables can be created in the document. It is user friendly as well and works in every working environment. (Microsoft, 2021)



Figure 2 Word

Report has been made using Microsoft word where text, images and testing table has been properly showed in the report as well.

## Draw.io

Draw.io is the tool that is used to create flow charts and process diagram, org charts and mind maps, network and architecture diagrams, UML diagrams and mock-ups, floor plans, and much more. It is the free online tool to make such diagrams and process into documents. And can be migrated very easily. (Draw .io, 2021)

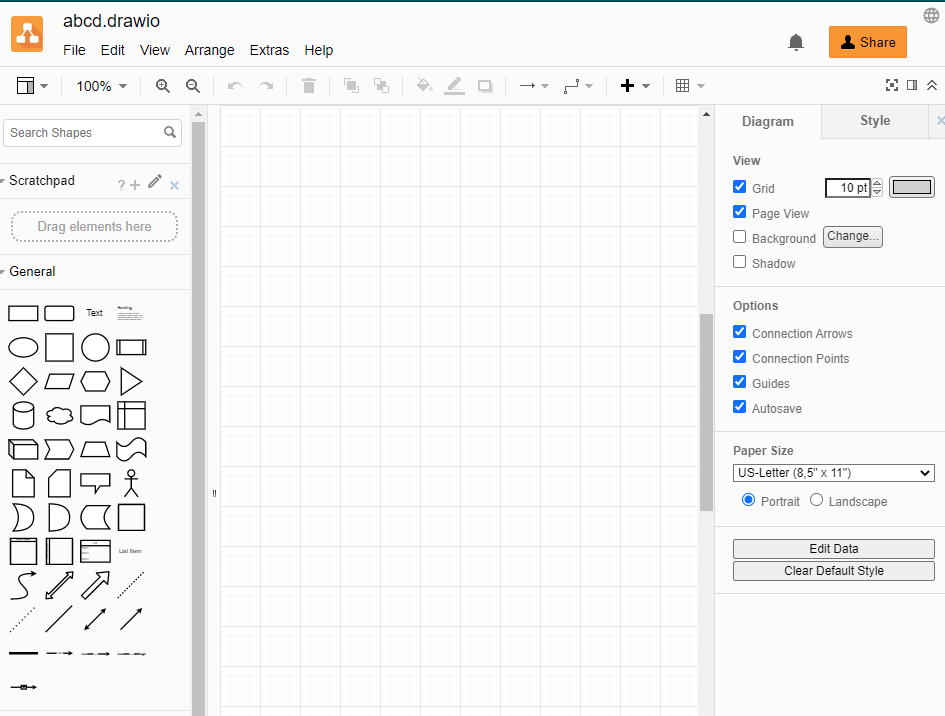


Figure 3 Draw.io Figure

The use of draw.io in this coursework is to make flowcharts. Making flow charts in draw.io is very easy works the connection of lines are easily connected through draw.io. All the shapes of the flowcharts are managed properly using draw.io.

## Algorithm:

Algorithm is a process or set of rules to be followed in calculations or other problem-solving operations, especially by a computer. Algorithm is written in high level languages. Algorithm is the best way to deal with the complex program in the step wise calculations. (Tech Target, 2021)

### Algorithm for library management system:

## Flowchart:

A flowchart depicts the individual steps of a process in a logical order. It's a generic tool that may be used to define a range of processes, including manufacturing, administrative, and service processes, and project plans. One of the seven core quality tools, it's a frequent process analysis tool. The use of flow chart is to develop understanding of how a process is done. To study a process of improvement. To communicate to others how a process is done. To document process and many more uses of flow chart. (Geeks For Geeks, 2021)

### Flowchart for Library Management System

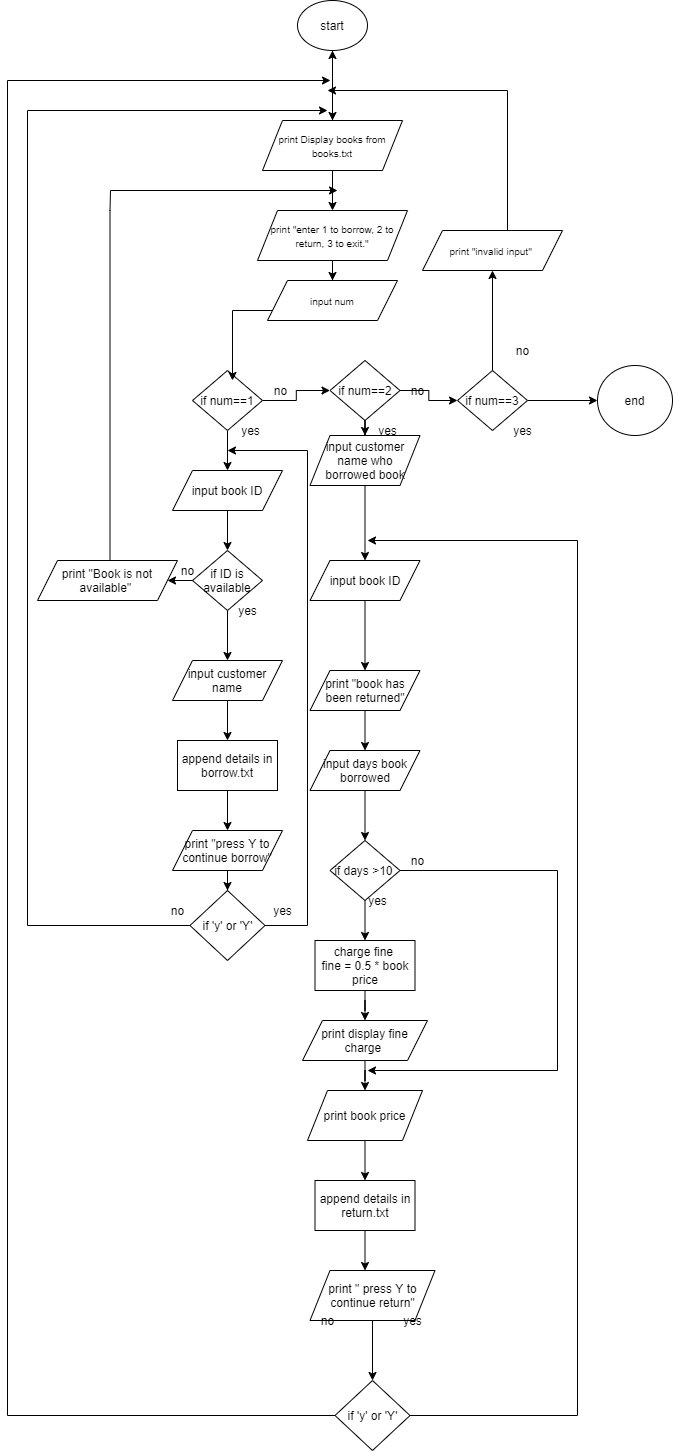


Figure 4 Flowchart

## PSEUDOCODE

Pseudocode is an artificial and informal language that helps programmers develop algorithms. Pseudocode is a “text-based” detail algorithmic design tool. The rules of Pseudocode are reasonably straight forward. All statements showing dependency are to be intended. These include while, do, for, if, switch. Examples below will illustrate this notion. (UNF , 2021)

### Pseudocode for main.py:

**IMPORT** borrow file which contains borrow code

**IMPORT** Return file which contains return code

**DEFINE** **FUNCTION** starting():

**OUTPUT**("+++++++++++++++++++++++++++++++++++++++++++++++++++")

**OUTPUT**("Hello and Welcome to our library management system")

**OUTPUT**("+++++++++++++++++++++++++++++++++++++++++++++++++++")

**OUTPUT**("\n")

**OUTPUT**("The Available Books IN our library are : \n")

borrow.book\_list()

**OUTPUT**("\n")

while(True):

**OUTPUT**("Enter 1 to borrow a book: ")

**OUTPUT**("Enter 2 to RETURN a book: ")

**OUTPUT**("Enter 3 to exit")

**TRY**:

**SET** num **TO** int(**INPUT**("Enter a value here: "))

**IF** num **EQUALS** 1:

borrow.borrowBook()

borrow.book\_list()

**ELSEIF** num **EQUALS** 2:

Return.**RETURN**\_book()

Return.book\_list()

**ELSEIF** num **EQUALS** 3:

**OUTPUT**("\n")

**OUTPUT**("+++++++++++++++++++++++++++++++++++++++++++++++++++")

**OUTPUT**("Thank You For Using Library Management System")

**OUTPUT**("+++++++++++++++++++++++++++++++++++++++++++++++++++++ \n")

break

**ELSE**:

**OUTPUT**("\n")

**OUTPUT**("+++++++++++++++++++++++++++++++++++++++++++++++++++")

**OUTPUT**("Invalid Input detected! Please INPUT valid Number")

**OUTPUT**("+++++++++++++++++++++++++++++++++++++++++++++++++++++ \n")

**EXCEPT**:

**OUTPUT**("\n")

**OUTPUT**("+++++++++++++++++++++++++++++++++++++++++++++++++++")

**OUTPUT**("Invalid Input detected! Please INPUT valid Number")

**OUTPUT**("+++++++++++++++++++++++++++++++++++++++++++++++++++++ \n”)

### Pseudocode for borrow.py

**DEFINE** **FUNCTION** lists():

**SET** file **TO** open("Books.txt", "r")

**SET** listBooks **TO** []

**SET** dictionary **TO** {}

**FOR** each **IN** file:

**SET** line **TO** each.replace("\n", "")

listBooks.append(line.split(","))

**FOR** i IN range(1, len(listBooks)+1):

**SET** each **TO** i

**SET** dictionary[each] **TO** listBooks[i-1]

file.close()

**RETURN** dictionary

**DEFINE** **FUNCTION** borrowBook():

**TRY**:

**SET** I **TO** int(INPUT("Enter ID of the book you want to borrow: "))

**SET** listBooks **TO** lists()

**IF** I >= 0 and I <= 5:

**IF** int(listBooks[I][2]) > 0:

**OUTPUT**("\n+++++++++++++++++++++")

**OUTPUT**("Book is available")

**OUTPUT**("+++++++++++++++++++++ \n")

**SET** name **TO** **INPUT**(

"Enter the name of person who want to borrow the books: ")

change\_quantity(I)

**SET** dt **TO** datetime.datetime.now()

**SET** t **TO** dt.strftime("%H:%M:%S")

**SET** d **TO** dt.strftime("%d/%m/%Y")

**SET** s **TO** 0

while(True):

**SET** YesNo **TO** INPUT(

"If 'Yes' please enter 'Y' or No FOR 'N' to cancel this borrowing process")

**IF** YesNo **EQUALS** "y" or YesNo **EQUALS** "Y":

**SET** s **TO** s+1

**SET** ID **TO** int(

**INPUT**("Enter id of the book that you want to borrow: "))

# validation(ID)

**SET** listBooks TO lists()

IF ID >= 0 and ID <= 5:

**IF** int(listBooks[ID][2]) > 0:

**OUTPUT**("\n +++++++++++++++++++++")

**OUTPUT**("Book is available")

**OUTPUT**("+++++++++++++++++++++ \n")

change\_quantity(ID)

bill(name, I, ID, t, d)

**OUTPUT**("\n")

total(I, ID)

**OUTPUT**("customers details: ")

**ELSE**:

**OUTPUT**("Book is not available")

**ELSE**:

**OUTPUT("**Please Provide a valid number")

**ELSE**:

**OUTPUT**("\n \n")

**OUTPUT**("Thank You "+name+" For Borrowing Books")

break

**ELSE**:

**OUTPUT**("Book is not available")

**ELSE**:

**OUTPUT**("Please Provide a valid number")

**EXCEPT**:

**OUTPUT**("\n")

**OUTPUT**("++++++++++++++++++++++++++++++++++++++")

**OUTPUT**("Invalid Input detected")

**OUTPUT**("++++++++++++++++++++++++++++++++++++++")

**DEFINE** **FUNCTION** book\_list():

**OUTPUT**("--------------------------------------------------------------------------------------")

**OUTPUT**("Book ID"+" "+"Book Name"+" "+" Author" +

" "+"Quantity"+" "+"Price")

**OUTPUT**("------------------------------------------------------------------------------------")

**SET** listBooks TO lists()

**FOR** key, value IN listBooks.items():

**OUTPUT**(" ", key, " ",

value[0], " ", value[1], " ", value[2], " ", value[3])

**OUTPUT**("------------------------------------------------------------------------------------")

**DEFINE** **FUNCTION** change\_quantity(val):

**SET** listBooks **TO** lists()

**SET** quantity **TO** int(listBooks[val][2])-1

**SET** listBooks[val][2] **TO** str(quantity).zfill(2)

**SET** file **TO** open("Books.txt", "w")

**FOR** key, value IN listBooks.items():

file.write(value[0]+","+value[1]+","+value[2]+","+value[3]+"\n")

file.close()

**OUTPUT**("Books ID: "+str(val) + "has been successfully borrowed")

**DEFINE** **FUNCTION** total(I, val):

**SET** listBooks **TO** lists()

**SET** price1 **TO** (listBooks[I][3])

**SET** price2 **TO** (listBooks[val][3])

**SET** p1 **TO** float(price1.strip("$"))

**SET** p2 **TO** float(price2.strip("$"))

**SET** total **TO** round(p1+p2, 2)

**RETURN** str(total)

**DEFINE** **FUNCTION** bill(name, I, ID, t, d):

**SET** listBooks **TO** lists()

**SET** year **TO** str(datetime.datetime.now().year)

**SET** month **TO** str(datetime.datetime.now().month)

**SET** day **TO** str(datetime.datetime.now().day)

**SET** second **TO** str(datetime.datetime.now().second)

with open("Borrow-"+name+""+year+""+month+""+day+""+second+".txt", "w+") as f:

**WRITE** in file("+++++++++++++++++++++++++++++++++++ \n")

**WRITE** in file (" Library Management System \n")

**WRITE** in file ("\n Book is borrowed by: "+name+"\n")

**WRITE** in file ("The time of borrowed book is: "+t+"\n")

**WRITE** in file ("The date of borrowed book is : "+d+"\n")

**WRITE** in file (“The borrowed book is: "+listBooks[I][0])

**WRITE** in file (", "+listBooks[ID][0]+", ")

### Pseudocode For Return.py:

**DEFINE** **FUNCTION** lists():

**SET** file **TO** open("Books.txt", "r")

**SET** listBooks **TO** []

**SET** dictionary **TO** {}

**FOR** each **IN** file:

**SET** line **TO** each.replace("\n", "")

listBooks.append(line.split(","))

**FOR** i IN range(1, len(listBooks)+1):

**SET** each TO i

**SET** dictionary[each] TO listBooks[i-1]

file.close()

**RETURN** dictionary

**DEFINE** **FUNCTION** RETURN\_book():

**SET** listsBooks **TO** lists()

**SET** name **TO** INPUT("Enter the name of the person who borrowed the book: ")

**SET** **ID** **TO** int(INPUT("Enter the ID of the book you want to RETURN: "))

**SET** dt **TO** datetime.datetime.now()

**SET** t **TO** dt.strftime("%H:%M:%S")

**SET** d **TO** dt.strftime("%d/%m/%Y")

change\_quantity(ID)

book\_list()

**SET** number **TO** int(INPUT("Enter the number of days you borrow the book: "))

**IF** number > 10:

**SET** minus TO number - 10

**SET** fine TO 0.5

**SET** fine TO minus \* fine

**SET** price TO (listsBooks[ID][3])

**SET** p1 TO float(price.strip("$"))

**SET** total TO round(fine + p1,2)

**OUTPUT**(str(fine)+ "$ is your fine. ")

**OUTPUT**("Total cost of the book is",str(total))

bill(name, ID, t, d, fine, price, total)

**ELSE**:

**SET** total TO 0

**SET** fine TO 0

**SET** listsBooks TO lists()

**SET** price TO (listsBooks[ID][3])

**OUTPUT**("The total price of the book is",str(price))

bill(name, ID, t, d, fine, price, total)

while(True):

**SET** YesNo **TO** INPUT("Do you want to RETURN more books. 'Y' FOR Yes and 'N' FOR No: ")

**IF** YesNo EQUALS "Y" or YesNo EQUALS "y":

**SET** ID **TO** int(INPUT("Enter the ID of the book you want to RETURN: "))

change\_quantity(ID)

book\_list()

**SET** number **TO** int(INPUT("Enter the number of days you borrow the book: "))

**IF** number > 10:

**SET** minus TO number - 10

**SET** fine TO 0.5

**SET** fine TO minus \* fine

**SET** price TO (listsBooks[ID][3])

**SET** p1 TO float(price.strip("$"))

**SET** total TO round(fine + p1,2)

**OUTPUT**(str(fine)+ "$ is your fine. ")

**OUTPUT**("Total cost of the book is ",str(total))

bill(name, ID, t, d, fine, price, total)

**ELSE**:

**SET** fine TO 0

**SET** listsBooks TO lists()

**SET** price TO (listsBooks[ID][3])

**OUTPUT**("The total price of the book is ",str(price))

bill(name, ID, t, d, fine, price, total)

**ELSE**:

break

**DEFINE** **FUNCTION** book\_list():

**OUTPUT**("--------------------------------------------------------------------------------------")

**OUTPUT**("Book ID"+" "+"Book Name"+" "+" Author" +

" "+"Quantity"+" "+"Price")

**OUTPUT**("------------------------------------------------------------------------------------")

**SET** listBooks **TO** lists()

**FOR** key, value **IN** listBooks.items():

**OUTPUT**(" ", key, " ",

value[0], " ", value[1], " ", value[2], " ", value[3])

**OUTPUT**("------------------------------------------------------------------------------------")

**DEFINE** **FUNCTION** change\_quantity(val):

**SET** listBooks **TO** lists()

**SET** quantity **TO** int(listBooks[val][2])+1

**SET** listBooks[val][2] **TO** str(quantity).zfill(2)

**SET** file **TO** open("Books.txt", "w")

**FOR** key, value IN listBooks.items():

file.write(value[0]+","+value[1]+","+value[2]+","+value[3]+"\n")

file.close()

**OUTPUT**("Books ID: "+str(val) + "has been successfully RETURNed! Please Proceed to cost calculation")

**DEFINE** **FUNCTION** bill(name, ID, t, d, fine, price, total):

**SET** listBooks **TO** lists()

**SET** year **TO** str(datetime.datetime.now().year)

**SET** month **TO** str(datetime.datetime.now().month)

**SET** day **TO** str(datetime.datetime.now().day)

**SET** second **TO** str(datetime.datetime.now().second)

with open("Returned\_By-"+name+""+year+""+month+""+day+""+second+".txt", "w+") as f:

**WRITE** in file ("+++++++++++++++++++++++++++++++++++ \n")

**WRITE** in file (" Library Management System \n")

**WRITE** in file ("\n Book is RETURN by: "+name+"\n")

**WRITE** in file ("The time of RETURN book is: "+t+"\n")

**WRITE** in file ("The date of RETURN book is : "+d+"\n")

**WRITE** in file ("The RETURN book is: "+listBooks[ID][0]+"\n")

**WRITE** in file ("The fine is: $"+str(fine)+"\n")

**WRITE** in file ("The price of the book is: "+price+"\n")

**WRITE** in file ("The total price of the book is: "+str(total)+"\n")

## Data Structure

Data structures are code frameworks for storing and organizing data that make it simpler to change, explore, and retrieve data. Data structures determine how data is collected, what operations we may do, and how data is connected. From operating systems to front-end development to machine learning, data structures are utilized in nearly all aspects of computer science. The following are examples of several data structures used in Python: (Corporate Finance Institute, 2021)

### Integer:

Integers, or whole integers ranging from negative infinity to infinity, such as 1,2,-1, can be used to represent numeric data.

**Screenshot of the data structure integer used in code of library management system:**

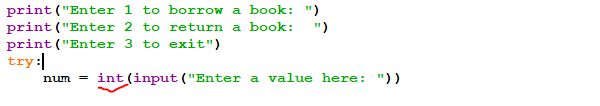


Figure 5 Integer

The data structure integer is used in program as int which converts the user input value as integer.

### Float:

The floating-point number is referred to as a "float." It can be used to represent rational numbers that end in a decimal, such as 2.22 or 2.34, and so on.

**Screenshot of the data structure float used in code of library management system:**

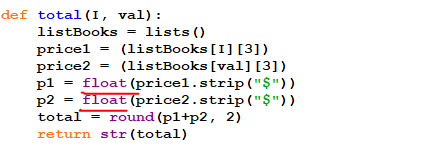


Figure 6 Float

The data structure float is used in the program as float() which converts into float data types.

### String:

A string is a logically ordered collection of alphabets, words, or other characters. In Python, a string is formed by surrounding a sequence of characters in a pair of single or double quotes. For example, "biscuit," "tea," and so on.

**Screenshot of the data structure string used in code of library management system:**

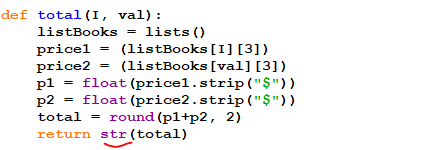


Figure 7 String

The data structure String is used in the program as str() which converts integer value to the string.

### Boolean:

The values True and False, which are commonly confused with the integers 1 and 0, are stored in this built-in data type. Booleans are used in conditional and comparison statements.

**Screenshot of the data structure Boolean used in code of library management system:**

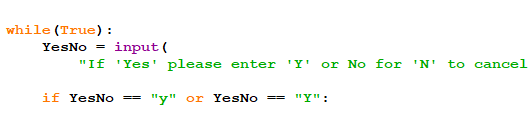


Figure 8 Boolean Data Structure

As mention above Boolean represent True and False in the program data structure Boolean is used as True.

### Lists:

In Python, lists are used to keep track of a collection of different elements. These are changeable, meaning they can have their content updated without losing their identity. The square brackets are used to distinguish lists. [ ], which is a comma-separated list of elements.

**Screenshot of the data structure Lists used in code of library management system:**

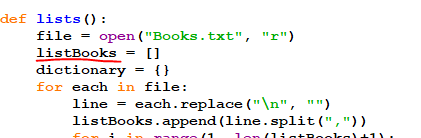


Figure 9 Lists

The data structure Lists in program is used as listBooks [] where it collects different elements.

### Dictionary:

You'll need dictionaries if you want to create something similar to a phone book. Except for the dictionary, none of the data structures are suitable for a phone directory. In this case, a dictionary will come in handy. The value of an item is represented by key-value pairs, as the name implies.

**Screenshot of the data structure Dictionary used in code of library management system.**

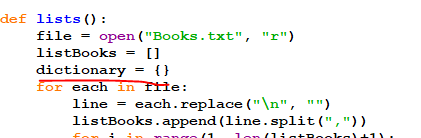


Figure 10 Dictionary Data Structure

The data structure dictionary in the program is used to store file in key value pair in the dictionary as {}

### Files:

Files are commonly found in data structures. Files have traditionally been an element of data structures. Despite the fact that big data is common in data science, a programming language that doesn't allow you to save and restore previously saved data isn't very useful. You must continue to use all of the data saved in files across databases, which you will learn how to do. The syntax for reading and writing files in Python is similar to those of other programming languages, although it is more user-friendly.

**Screenshot of data structure Files used in code of library Management system**

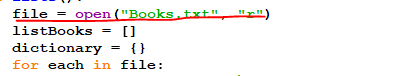


Figure 11 Files Data Structure

The data structure file is used to open Books.txt in the program.

### Arrays:

The array module, which must be installed before you can start creating and manipulating arrays in Python, makes arrays possible. The data types of array elements are restricted.

### Tuples:

Tuples are a type of variable that allows you to store several elements in a single variable. Tuple is one of Python's four built-in data types for storing collections of data; the other three are List, Set, and Dictionary, all of which have different properties and applications. A tuple is a collection of items that is both ordered and immutable.

# Program:

## Implementation of Program:

When first time program runs, main file is executed it shows the welcome message , book list and navigation with appropriate message.

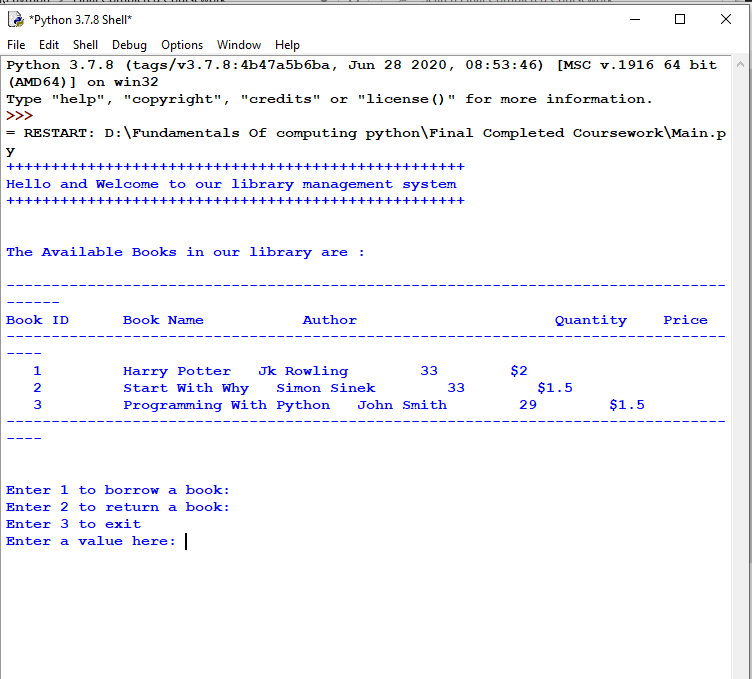
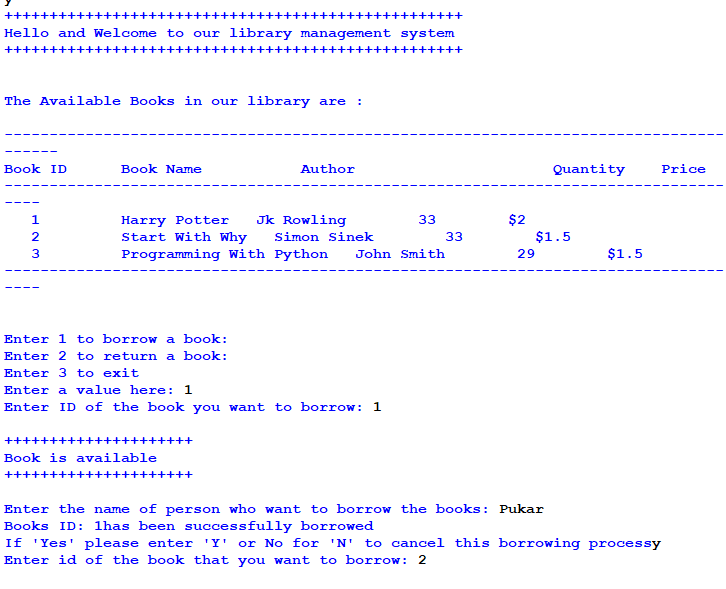


Figure 12 Implementation of Program

## Showing Borrow Process:



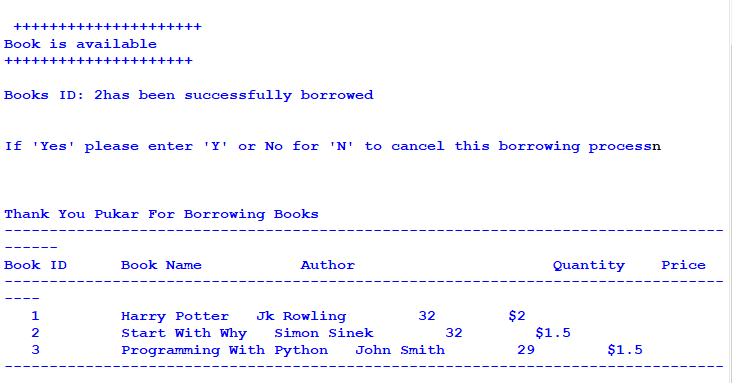
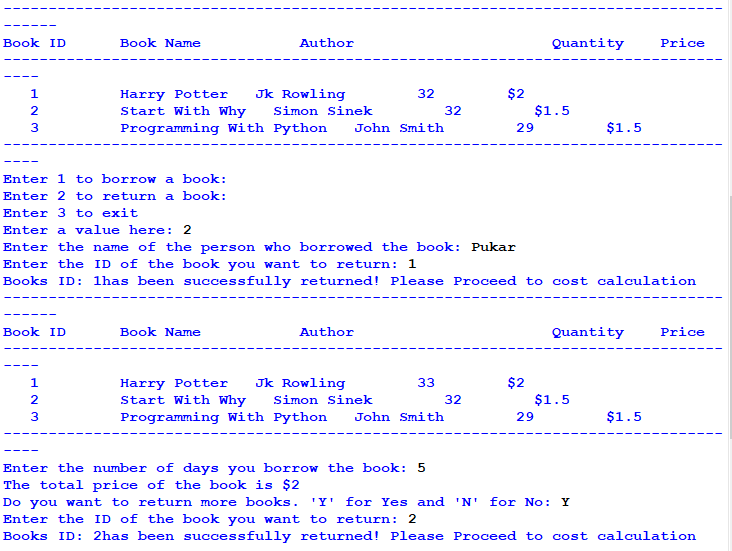


Figure 13 Showing Borrow Process

Hereafter when borrow is complete the quantity of the selected book has been deducted

## Showing Return process



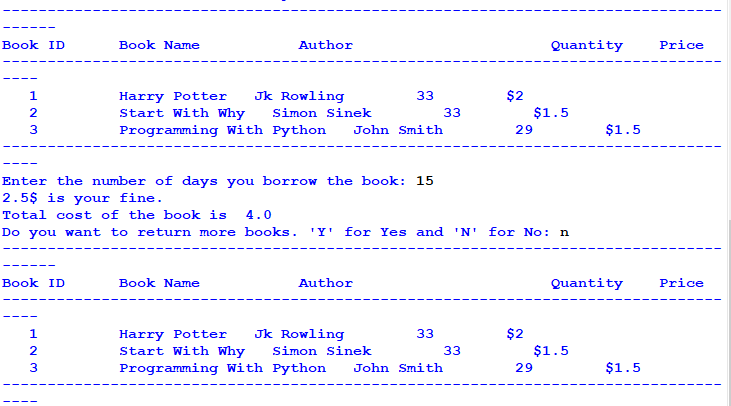


Figure 14 Showing Return Process

Hereafter when return process is complete the quantity of the selected book has been added in the stock.

## Bill of Borrow:

When borrow is done the creation of bill of borrow is shown here

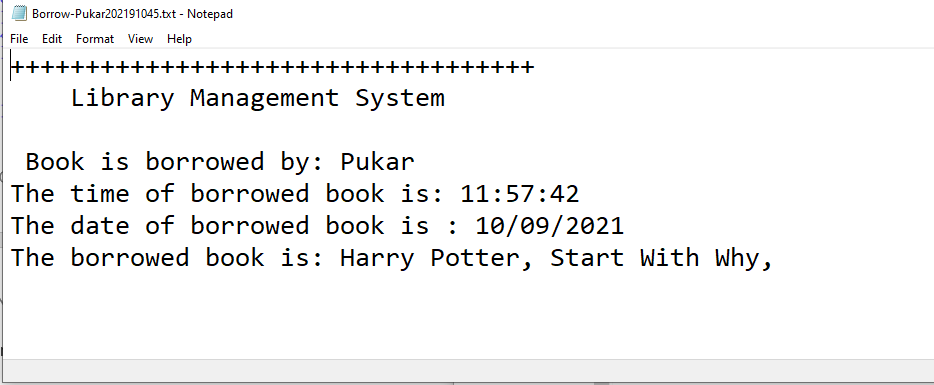


Figure 15 Bill of Borrow

## Bill of Return:

When return is done the creation of bill of return with total is shown here

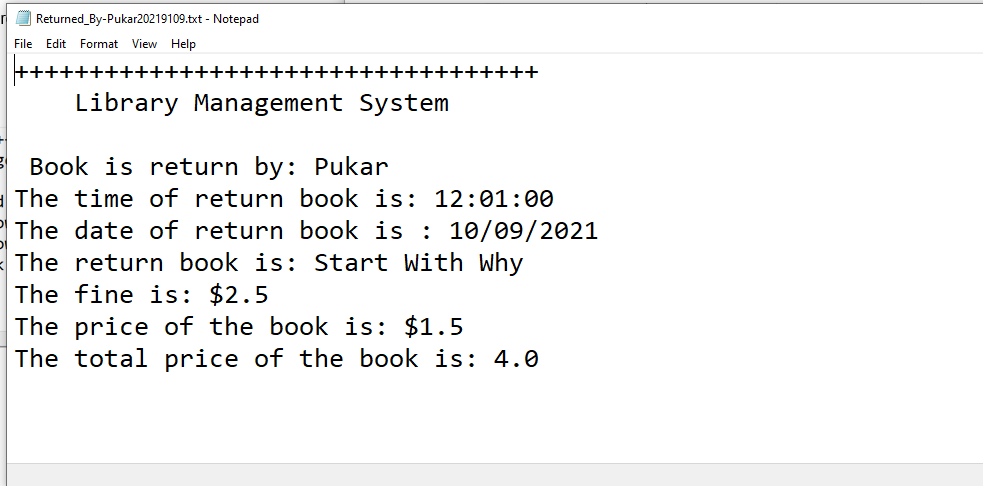


Figure 16 Bill of return

## Creation Of text file

Here when borrow and return is done the text file has been created.

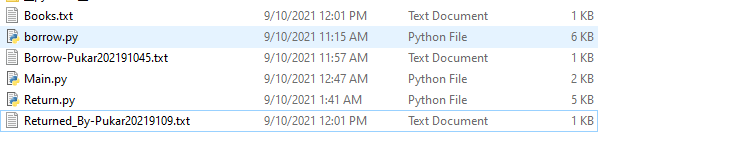


Figure 17 Creation Of txt file

## Program Termination with Exit:

When program is executed the value 3 has been given and the program has been successfully terminated

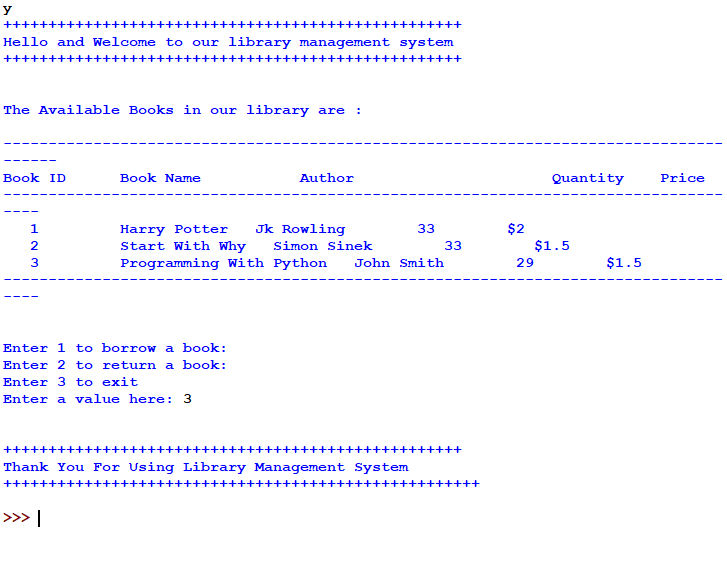


Figure 18 Termination Of program

# Testing:

## Test 1: Show implementation of try, except

|  |  |
| --- | --- |
| Test No | 1 |
| Objective | To show the implementation of try, except  To provide invalid input and showing the message |
| Action: | * The main.py file has been run in the idle. * Then after the program ask as 1 2 and 3 for input * To provide invalid input keyword ‘a’ has been enter then after the program shows: * Invalid Input Detected |
| Expected Result | Invalid Input Detected |
| Actual Result | Try Except should work properly . |
| Conclusion | The test is successful. |

Table 1 Show implementation of try, except

**Screenshot of the test 1 with proper output has been shown below:**

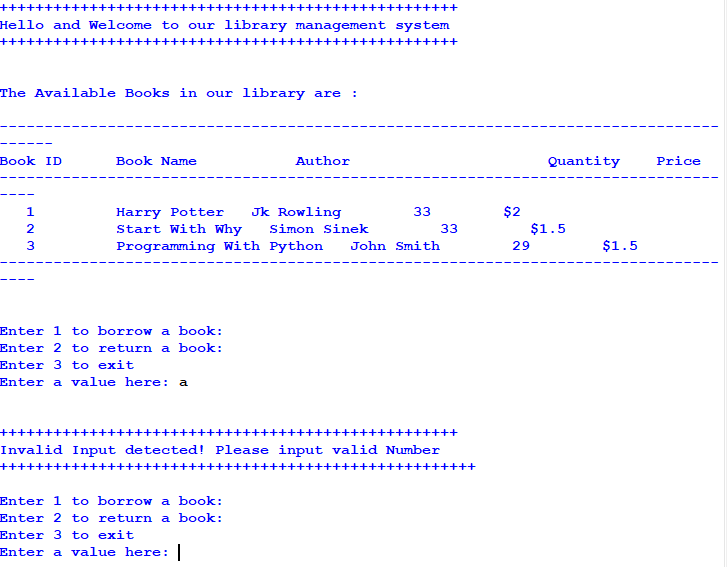


Figure 19 Screenshot of the test 1 with proper output has been shown below:

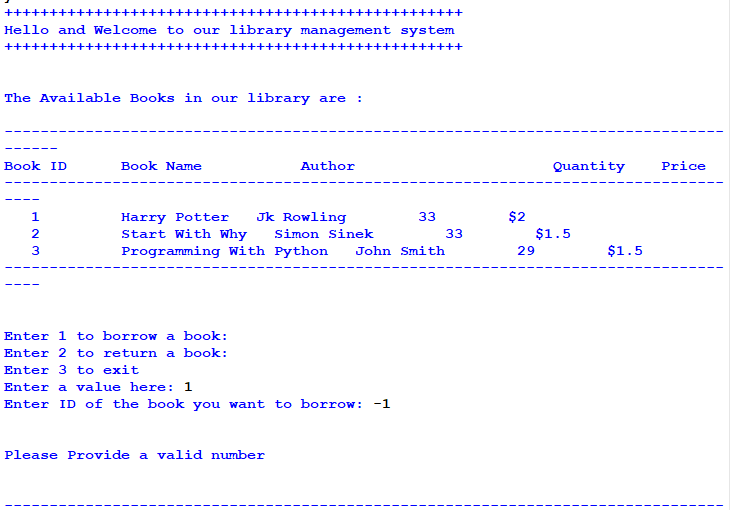
## Test 2: Selection of borrow and return

|  |  |
| --- | --- |
| Test No | 2 |
| Objective | To show selection process of borrow and return  To make program successfully run after invalid input |
| Action: | * The main.py file has been run in the idle. * Then after the program ask as 1 2 and 3 for input * 1 has been given for borrow process * In Id of book -1 has been given * In the book list there are only 3 books not existed value 5 has been given * Then after for return 2 has been given * In id of book -1 has been given * In the book list there are only 3 books not existed value 5 has been given |
| Expected Result | Program show appropriate message and helps in successfully run without exception |
| Actual Result | Borrow and return has been selected and program is successfully run . |
| Conclusion | The test is successful. |

Table 2 Selection of borrow and return

**Screenshot of the test 2 with proper output has been shown below:**

**For Borrow:**



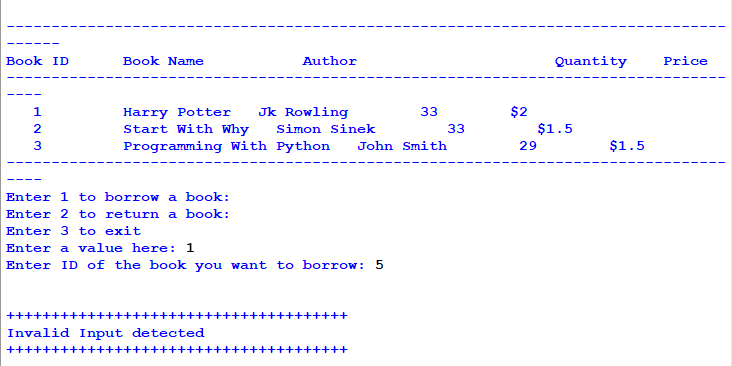
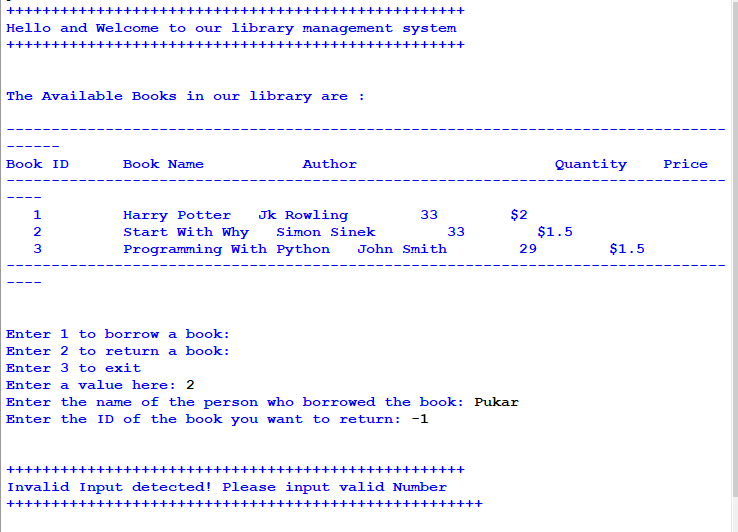


Figure 20 Screenshot of the test 2 with proper output has been shown below:

**Screenshot of the test 2 with proper output has been shown below:**

**For Return:**



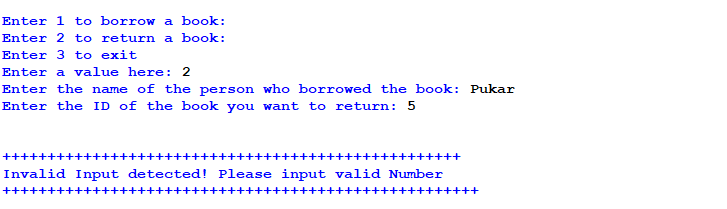


Figure 21 Screenshot of the test 2 with proper output has been shown below: for return

## Test 3: To Generate file of borrow, and output in shell

|  |  |
| --- | --- |
| Test No | 3 |
| Objective | To show borrow process completely, to show them in shell, and make the borrow note |
| Action: | * The main.py file has been run in the idle. * Then after the program ask as 1 2 and 3 for input * 1 has been given for borrow process * 1 has been input for the book ID * Name has been given “Pukar” * Borrow has been shown in the shell * Borrow note is generated |
| Expected Result | Complete borrow process, show in shell, and make the borrow note. |
| Actual Result | Borrow process and note has been successfully created . |
| Conclusion | The test is successful. |

Table 3 To generate file of borrow, and output in shell

**Screenshot of the test 3 with the proper output has been shown below:**

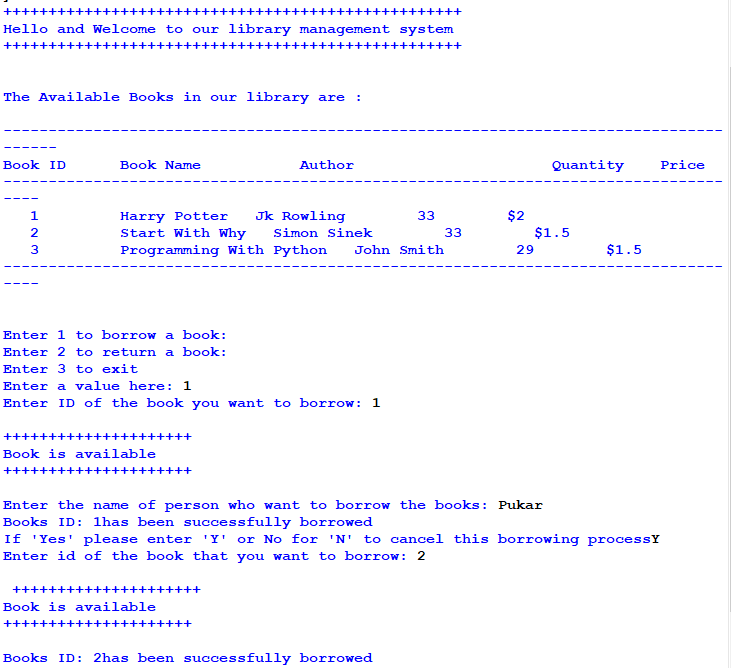


Figure 22 Screenshot of the test 3 with the proper output has been shown below

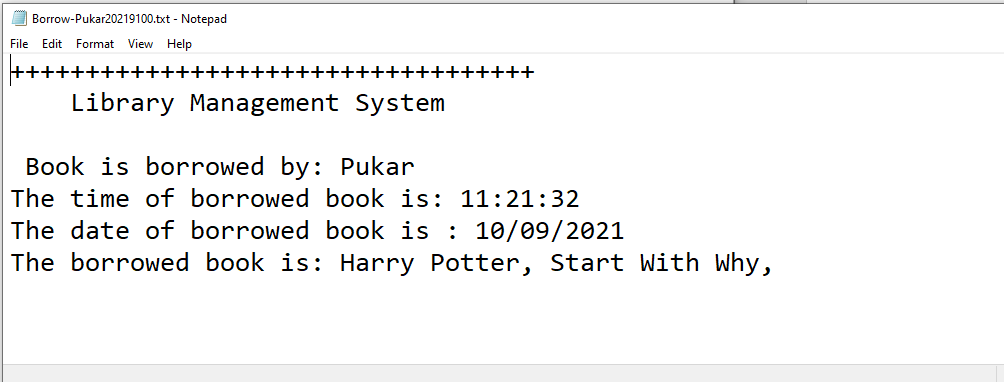


Figure 23 Bill of borrow in text file

## Test 4: To generate file of return

|  |  |
| --- | --- |
| Test No | 4 |
| Objective | To show return process completely, to show them in shell, and make the return note |
| Action: | * The main.py file has been run in the idle. * Then after the program ask as 1 2 and 3 for input * 2 has been given for return process * 1 has been input for the book ID * Name has been given “Pukar” * Return has been shown in the shell * Return note is generated |
| Expected Result | Complete return process, show in shell, and make the return note. |
| Actual Result | Return process and note has been successfully created . |
| Conclusion | The test is successful. |

Table 4 To generate file of return

**Screenshot of test 4 with the proper output has been shown below:**

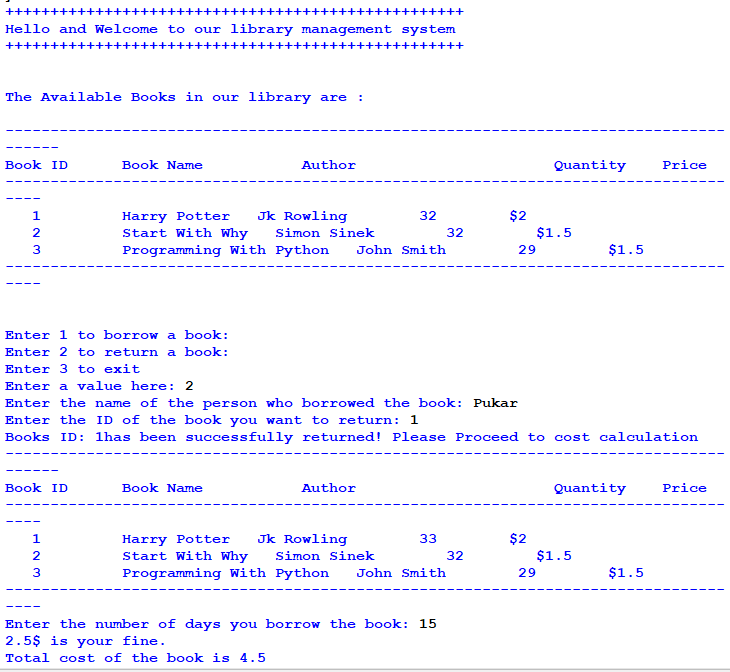


Figure 24 Screenshot of test 4 with the proper output has been shown:

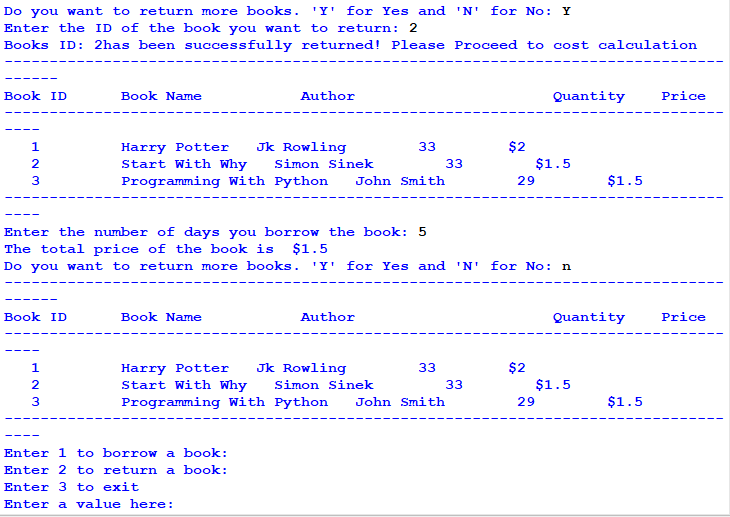


Figure 25 Screenshot of test 4 with the proper output has been shown:

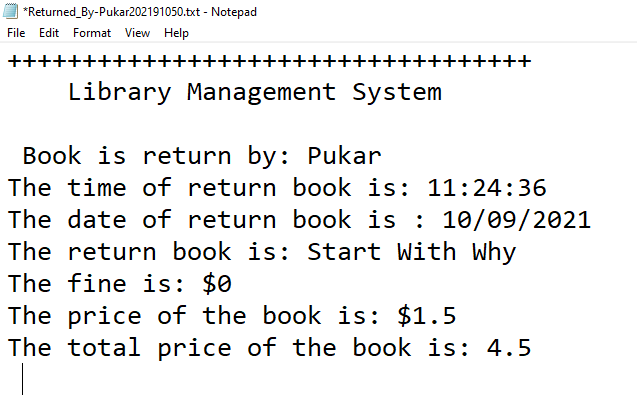


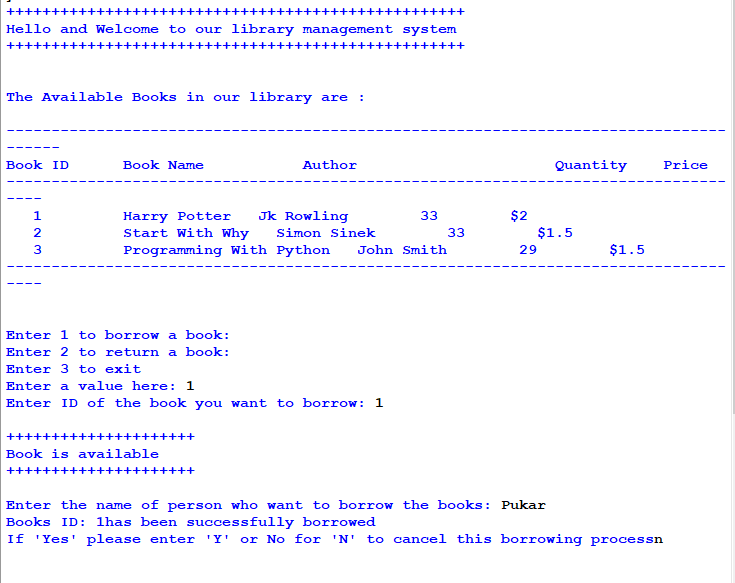
Figure 26 Bill of return

## Test 5: Showing the updated stock: 1) Quantity Deducted for borrow 2) Quantity added for returning

|  |  |
| --- | --- |
| Test No | 5 |
| Objective | To show the quantity deducted for borrow and quantity added for returning |
| Action: | * The main.py file has been run in the idle. * Then after the program ask as 1 2 and 3 for input * 1 has been given for borrow process * 1 has been input for the book ID * Name has been given “Pukar” * After borrow the quantity deducted has been showed. * Then running for return * 2 has been given for return process * 1 has been given for Book ID to be returned * Name has been given Pukar * After return the quantity added has been showed. |
| Expected Result | Showing the quantity deducted for borrow and quantity added for returning. |
| Actual Result | Quantity has been changed |
| Conclusion | The test is successful. |

Table 5 Quantity Deducted , Quantity Added

**Screenshot of the test 5 with the proper output has been showed below For borrow:**



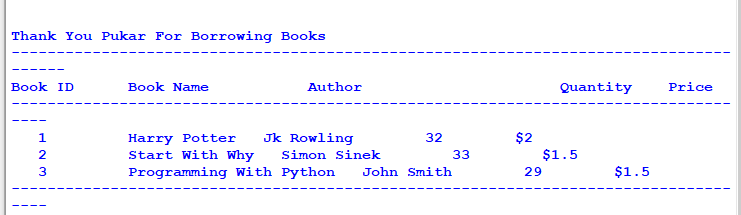
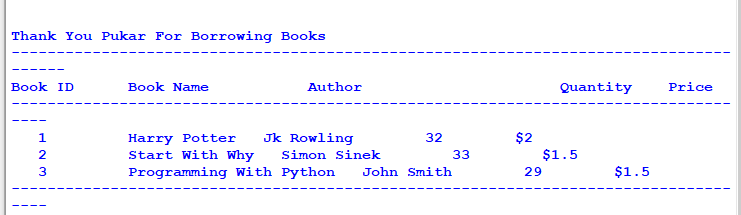


Figure 27 Quantity deducted

Here, the quantity has been deducted.

**Screenshot of the test 5 with the proper output has been showed below for return:**



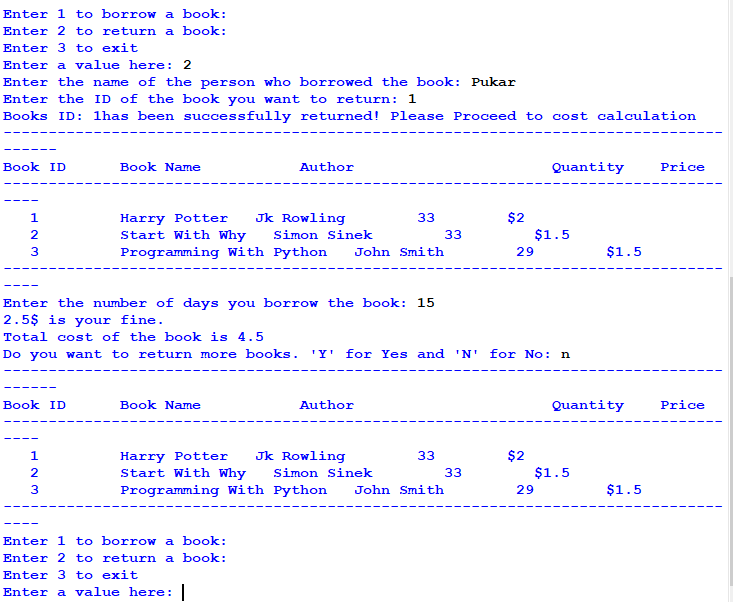


Figure 28 Quantity has been added

After that the quantity has been added after the return process.

# Conclusion:

I wasn’t much familiar with this programming language and time was limited; the coursework provided to us was highly instructive yet long. I had studied python in high school, which is a flexible programming language, and we had previously worked on BlueJ, so I had a rudimentary understanding of it, even if I was still a beginner when it came to python. I made a few blunders on the work, but I couldn't figure out where I went wrong, so I decided to postpone it for now. Then I started looking into the problems that I was having while performing the work, and I discovered that the most common problem was indentation and syntax errors. I was able to identify where I went wrong and correct it after learning about the many sorts and descriptions of errors. As a consequence, I was able to effectively complete the development phase of my software and create a library management system.

We had to create an algorithm, pseudo code, flow chart, data structures, and a program that included Correctness, Implementation, Programming Style, Exception Handling, User Interface, and Program Usability for the report section. Following that, testing should be carried out to demonstrate that the software is operating properly as evidenced by evidences. While this was difficult at first, I was able to do it in less time since I was familiar with it, and I sought assistance from my teacher or friends whenever I became lost. Finally, this training aided in the development of my essential programming abilities; the project, albeit little, was critical. I gained a lot of knowledge about the programming language as well as how to handle problems. This has sparked my curiosity in pursuing a programming profession. I'd want to express my gratitude to my instructors and the institution for providing me with such a wonderful and varied learning environment.

# Appendix:

## -Main.py

import borrow

import Return

def starting():

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

print("Hello and Welcome to our library management system")

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

print("\n")

print("The Available Books in our library are : \n")

borrow.book\_list()

print("\n")

while(True):

print("Enter 1 to borrow a book: ")

print("Enter 2 to return a book: ")

print("Enter 3 to exit")

try:

num = int(input("Enter a value here: "))

if num == 1:

borrow.borrowBook()

borrow.book\_list()

elif num == 2:

Return.return\_book()

Return.book\_list()

elif num == 3:

print("\n")

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

print("Thank You For Using Library Management System")

print("+++++++++++++++++++++++++++++++++++++++++++++++++++++ \n")

break

else:

print("\n")

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

print("Invalid Input detected! Please input valid Number")

print("+++++++++++++++++++++++++++++++++++++++++++++++++++++ \n")

except:

print("\n")

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

print("Invalid Input detected! Please input valid Number")

print("+++++++++++++++++++++++++++++++++++++++++++++++++++++ \n")

starting()

## -borrow.py

import datetime

def lists():

file = open("Books.txt", "r")

listBooks = []

dictionary = {}

for each in file:

line = each.replace("\n", "")

listBooks.append(line.split(","))

for i in range(1, len(listBooks)+1):

each = i

dictionary[each] = listBooks[i-1]

file.close()

return dictionary

def borrowBook():

try:

I = int(input("Enter ID of the book you want to borrow: "))

listBooks = lists()

if I >= 0 and I <= 5:

if int(listBooks[I][2]) > 0:

print("\n+++++++++++++++++++++")

print("Book is available")

print("+++++++++++++++++++++ \n")

name = input(

"Enter the name of person who want to borrow the books: ")

change\_quantity(I)

# we have to generate bill here so will do it later

dt = datetime.datetime.now()

t = dt.strftime("%H:%M:%S")

d = dt.strftime("%d/%m/%Y")

s = 0

while(True):

YesNo = input(

"If 'Yes' please enter 'Y' or No for 'N' to cancel this borrowing process")

if YesNo == "y" or YesNo == "Y":

s = s+1

ID = int(

input("Enter id of the book that you want to borrow: "))

# validation(ID)

listBooks = lists()

if ID >= 0 and ID <= 5:

if int(listBooks[ID][2]) > 0:

print("\n +++++++++++++++++++++")

print("Book is available")

print("+++++++++++++++++++++ \n")

change\_quantity(ID)

bill(name, I, ID, t, d)

# quantity change or updates

print("\n")

total(I, ID)

# total costs chainxha yetaa

print("customers details: ")

else:

print("Book is not available")

else:

print("Please Provide a valid number")

else:

print("\n \n")

print("Thank You "+name+" For Borrowing Books")

break

# second time book borrow garna ko lagi

# ani book borrow hune code esmai hunxa

else:

print("Book is not available")

else:

print("Please Provide a valid number")

except:

print("\n")

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

print("Invalid Input detected")

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

def book\_list():

print("--------------------------------------------------------------------------------------")

print("Book ID"+" "+"Book Name"+" "+" Author" +

" "+"Quantity"+" "+"Price")

print("------------------------------------------------------------------------------------")

listBooks = lists()

for key, value in listBooks.items():

print(" ", key, " ",

value[0], " ", value[1], " ", value[2], " ", value[3])

print("------------------------------------------------------------------------------------")

def change\_quantity(val):

listBooks = lists()

quantity = int(listBooks[val][2])-1

listBooks[val][2] = str(quantity).zfill(2)

file = open("Books.txt", "w")

for key, value in listBooks.items():

file.write(value[0]+","+value[1]+","+value[2]+","+value[3]+"\n")

file.close()

print("Books ID: "+str(val) + "has been successfully borrowed")

def total(I, val):

listBooks = lists()

price1 = (listBooks[I][3])

price2 = (listBooks[val][3])

p1 = float(price1.strip("$"))

p2 = float(price2.strip("$"))

total = round(p1+p2, 2)

return str(total)

def bill(name, I, ID, t, d):

listBooks = lists()

year = str(datetime.datetime.now().year)

month = str(datetime.datetime.now().month)

day = str(datetime.datetime.now().day)

second = str(datetime.datetime.now().second)

with open("Borrow-"+name+""+year+""+month+""+day+""+second+".txt", "w+") as f:

f.write("+++++++++++++++++++++++++++++++++++ \n")

f.write(" Library Management System \n")

f.write("\n Book is borrowed by: "+name+"\n")

f.write("The time of borrowed book is: "+t+"\n")

f.write("The date of borrowed book is : "+d+"\n")

f.write("The borrowed book is: "+listBooks[I][0])

f.write(", "+listBooks[ID][0]+", ")

## -Return.py

import datetime

def lists():

file = open("Books.txt", "r")

listBooks = []

dictionary = {}

for each in file:

line = each.replace("\n", "")

listBooks.append(line.split(","))

for i in range(1, len(listBooks)+1):

each = i

dictionary[each] = listBooks[i-1]

file.close()

return dictionary

def return\_book():

listsBooks = lists()

name = input("Enter the name of the person who borrowed the book: ")

ID = int(input("Enter the ID of the book you want to return: "))

dt = datetime.datetime.now()

t = dt.strftime("%H:%M:%S")

d = dt.strftime("%d/%m/%Y")

change\_quantity(ID)

book\_list()

number = int(input("Enter the number of days you borrow the book: "))

if number > 10:

minus = number - 10

fine = 0.5

fine = minus \* fine

price = (listsBooks[ID][3])

p1 = float(price.strip("$"))

total = round(fine + p1,2)

print(str(fine)+ "$ is your fine. ")

print("Total cost of the book is",str(total))

bill(name, ID, t, d, fine, price, total)

else:

total = 0

fine = 0

listsBooks = lists()

price = (listsBooks[ID][3])

print("The total price of the book is",str(price))

bill(name, ID, t, d, fine, price, total)

while(True):

YesNo = input("Do you want to return more books. 'Y' for Yes and 'N' for No: ")

if YesNo == "Y" or YesNo == "y":

ID = int(input("Enter the ID of the book you want to return: "))

change\_quantity(ID)

book\_list()

number = int(input("Enter the number of days you borrow the book: "))

if number > 10:

minus = number - 10

fine = 0.5

fine = minus \* fine

price = (listsBooks[ID][3])

p1 = float(price.strip("$"))

total = round(fine + p1,2)

print(str(fine)+ "$ is your fine. ")

print("Total cost of the book is ",str(total))

bill(name, ID, t, d, fine, price, total)

else:

fine = 0

listsBooks = lists()

price = (listsBooks[ID][3])

print("The total price of the book is ",str(price))

bill(name, ID, t, d, fine, price, total)

else:

break

def book\_list():

print("--------------------------------------------------------------------------------------")

print("Book ID"+" "+"Book Name"+" "+" Author" +

" "+"Quantity"+" "+"Price")

print("------------------------------------------------------------------------------------")

listBooks = lists()

for key, value in listBooks.items():

print(" ", key, " ",

value[0], " ", value[1], " ", value[2], " ", value[3])

print("------------------------------------------------------------------------------------")

def change\_quantity(val):

listBooks = lists()

quantity = int(listBooks[val][2])+1

listBooks[val][2] = str(quantity).zfill(2)

file = open("Books.txt", "w")

for key, value in listBooks.items():

file.write(value[0]+","+value[1]+","+value[2]+","+value[3]+"\n")

file.close()

print("Books ID: "+str(val) + "has been successfully returned! Please Proceed to cost calculation")

def bill(name, ID, t, d, fine, price, total):

listBooks = lists()

year = str(datetime.datetime.now().year)

month = str(datetime.datetime.now().month)

day = str(datetime.datetime.now().day)

second = str(datetime.datetime.now().second)

with open("Returned\_By-"+name+""+year+""+month+""+day+""+second+".txt", "w+") as f:

f.write("+++++++++++++++++++++++++++++++++++ \n")

f.write(" Library Management System \n")

f.write("\n Book is return by: "+name+"\n")

f.write("The time of return book is: "+t+"\n")

f.write("The date of return book is : "+d+"\n")

f.write("The return book is: "+listBooks[ID][0]+"\n")

f.write("The fine is: $"+str(fine)+"\n")

f.write("The price of the book is: "+price+"\n")

f.write("The total price of the book is: "+str(total)+"\n")

# References

Corporate Finance Institute, 2021. *Corporate Finance Institute.* [Online]   
Available at: https://corporatefinanceinstitute.com/resources/knowledge/other/python-data-structures/  
[Accessed 09 Sept 2021].

Draw .io, 2021. *Draw.io.* [Online]   
Available at: www.draw.io  
[Accessed 09 Sept 2021].

Geeks For Geeks, 2021. *Geeks For Geeks.* [Online]   
Available at: https://www.geeksforgeeks.org/an-introduction-to-flowcharts/#:~:text=Flowchart%20is%20a%20graphical%20representation,is%20known%20as%20%E2%80%9Cflowcharting%E2%80%9D.  
[Accessed 09 Sept 2021].

Microsoft, 2021. *Microsoft.* [Online]   
Available at: https://support.microsoft.com/en-us/office/what-s-new-in-word-for-microsoft-365-4219dfb5-23fc-4853-95aa-b13a674a6670?ui=en-us&rs=en-us&ad=us  
[Accessed 09 September 2021].

Tech Target, 2021. *Tech Target.* [Online]   
Available at: https://whatis.techtarget.com/definition/algorithm  
[Accessed 09 Sept 2021].

UNF , 2021. *UNF.* [Online]   
Available at: https://www.unf.edu/~broggio/cop2221/2221pseu.htm  
[Accessed 09 Sept 2021].

W3 Schools, 2021. *W3 Schools.* [Online]   
Available at: https://www.w3schools.com/python/python\_intro.asp  
[Accessed 09 Sept 2021].