جامعة بيرزيت

BIRZEIT UNIVERSITY

FACULTY OF ENGINEERING AND TECHNOLOGY

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

LINUX LABORATORY

ENCS313

Python PROJECT

INSTRUCTOR : DR.Aziz Qaroush.

STUDENTS :-

1. **Momen Salem (#1200034).**

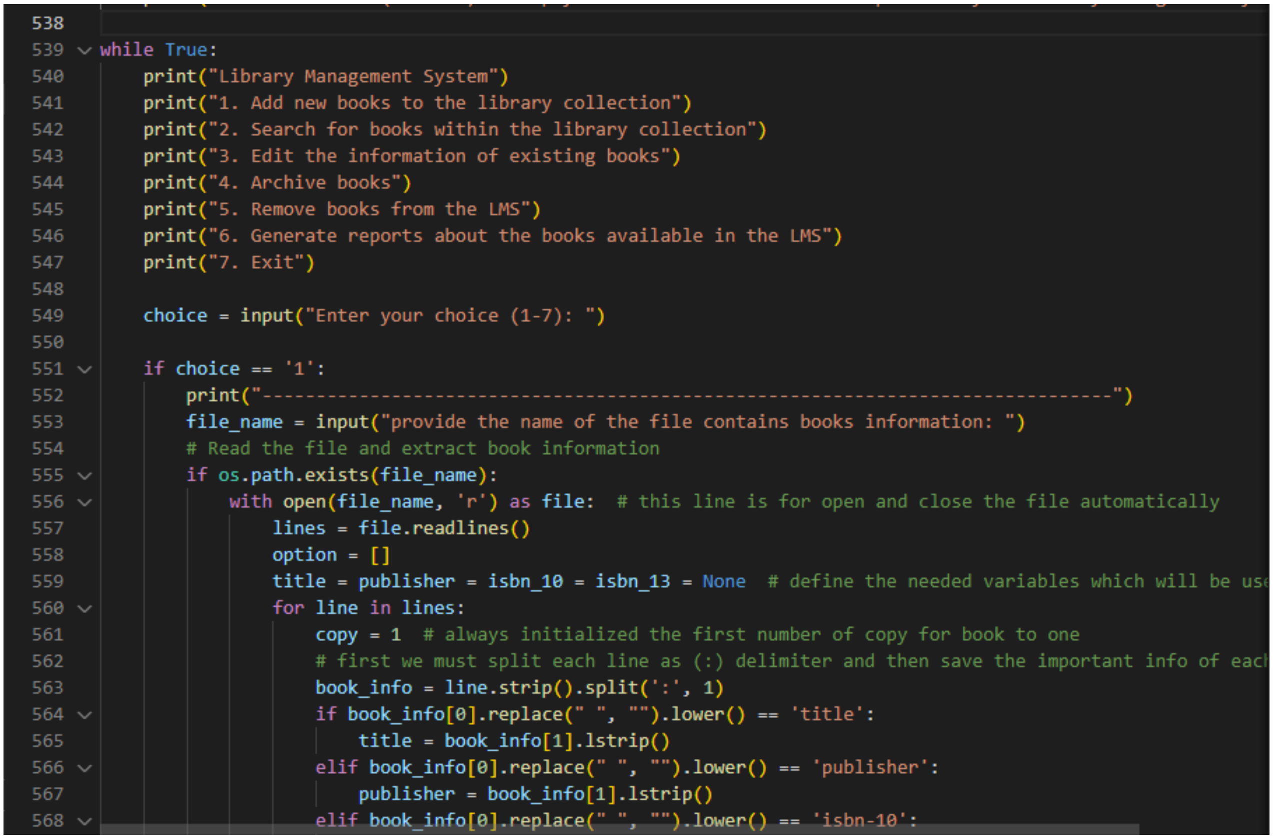
2. **Mohammad Dallash(#1200937).**

Section : 2

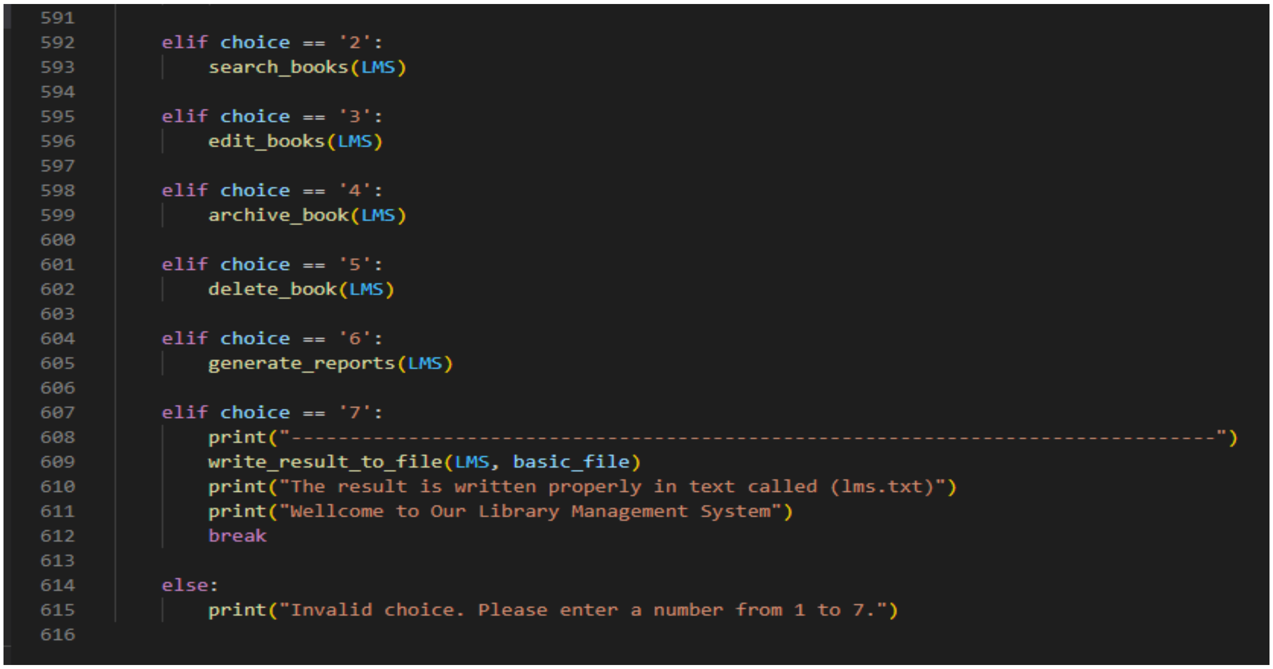
Contents

Tasks for python project (Code & expansion) :

✞ **Main :**



```python
while True:
    print("Library Management System")
    print("1. Add new books to the library collection")
    print("2. Search for books within the library collection")
    print("3. Edit the information of existing books")
    print("4. Archive books")
    print("5. Remove books from the LMS")
    print("6. Generate reports about the books available in the LMS")
    print("7. Exit")

    choice = input("Enter your choice (1-7): ")

    if choice == '1':
        print("--------------------------------------------------------------------------------")
        file_name = input("provide the name of the file contains books information: ")
        # Read the file and extract book information
        if os.path.exists(file_name):
            with open(file_name, 'r') as file:  # this line is for open and close the file automatically
                lines = file.readlines()
                option = []
                title = publisher = isbn_10 = isbn_13 = None  # define the needed variables which will be use
                for line in lines:
                    copy = 1  # always initialized the first number of copy for book to one
                    # first we must split each line as (:) delimiter and then save the important info of each
                    book_info = line.strip().split(':', 1)
                    if book_info[0].replace(" ", "").lower() == 'title':
                        title = book_info[1].lstrip()
                    elif book_info[0].replace(" ", "").lower() == 'publisher':
                        publisher = book_info[1].lstrip()
                    elif book_info[0].replace(" ", "").lower() == 'isbn-10':
```



```python

    elif choice == '2':
        search_books(LMS)

    elif choice == '3':
        edit_books(LMS)

    elif choice == '4':
        archive_book(LMS)

    elif choice == '5':
        delete_book(LMS)

    elif choice == '6':
        generate_reports(LMS)

    elif choice == '7':
        print("------------------------------------------------------------------------------")
        write_result_to_file(LMS, basic_file)
        print("The result is written properly in text called (lms.txt)")
        print("Wellcome to Our Library Management System")
        break

    else:
        print("Invalid choice. Please enter a number from 1 to 7.")

```

The (Main) function is mainly used to call the other functions from each task, the main gives the user 7 options to choose from which are:

- **Option (1) : Adding new books to the library collection :**
  This option enables the user to add new books to the library collection. The user will be able to select this option and enter the name of a file containing information about the new books. The LMS will verify the availability and accessibility of the file. The new books' information will be presented in a specific format, including mandatory details such as Title, Publisher, and ISBNs. Additionally, the LMS will display the book information on the screen as it loads the data from the file. Each book will be assigned a unique ISBN, and the initial "number of copies" for each new book will be set to 1.

- **Option (2) : Searching for books within the library collection:-**
  The second enhancement focuses on improving the book search functionality. Users will be able to search for books using various parameters, including the optional ones. The LMS will display the search results on the screen and provide an option to store the results in a text file for future reference.

- **Option (3) : . Editing the information of existing books:-**
  The third enhancement allows users to edit the information of existing books. When selecting this option, the LMS will prompt the user to provide either the file name or ISBN number of the book they want to update. After making the necessary changes, the LMS will ask for confirmation before saving the updated data.

- **Option (4) : Archiving books:-**
  The fourth enhancement enables users to archive books that are rarely used. Users can select this option and enter the ISBNs of the books they wish to archive. The LMS will ask for confirmation and provide the flexibility to choose the number of copies to be archived if multiple copies are available.

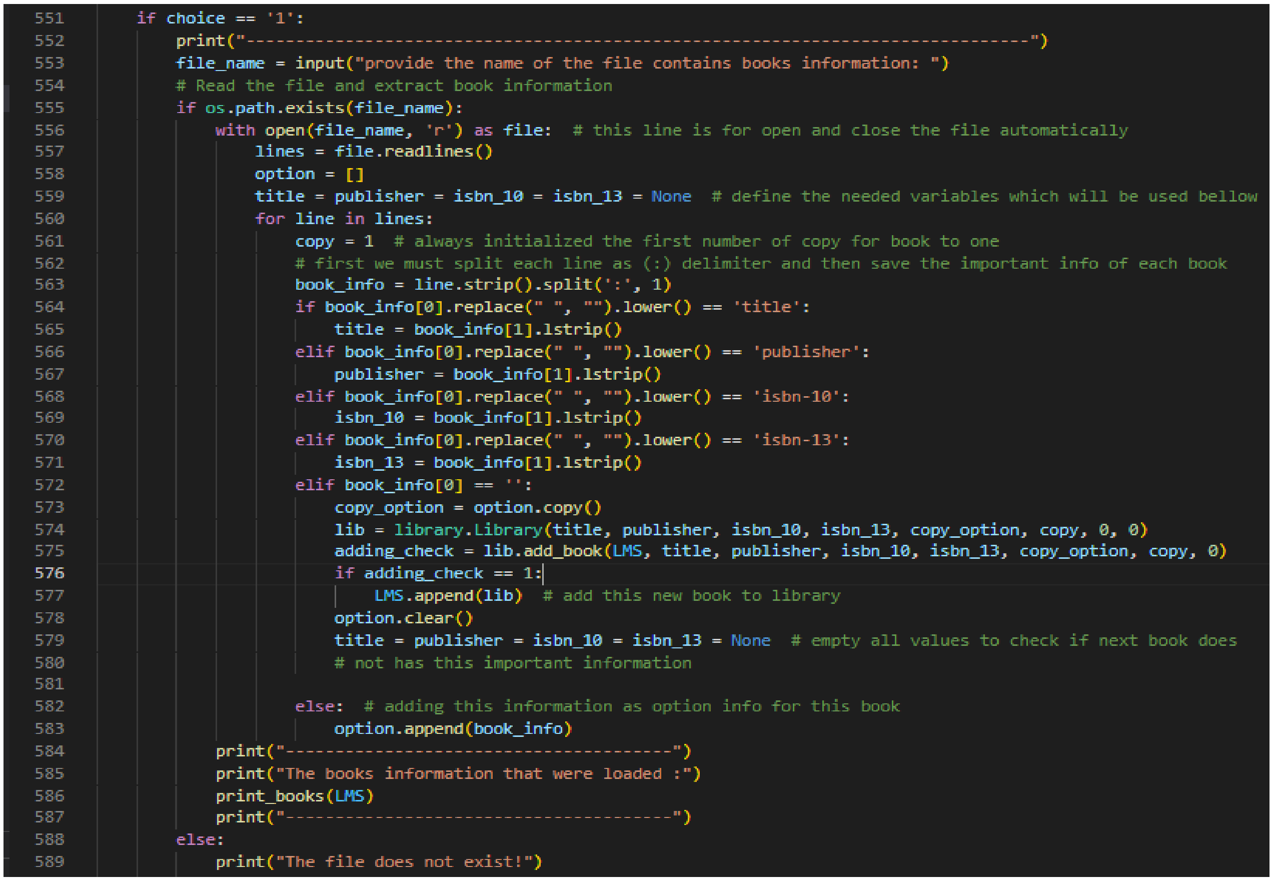
- **Option (5) : Removing books from the LMS:-**
  The fifth enhancement allows the deletion of books from the LMS. Only archived books can be deleted, and the LMS will prompt the user for confirmation before proceeding with the deletion.

- **Option (6) : Generating reports about the books available in the LMS :-**
  The sixth enhancement focuses on generating comprehensive reports about the books available in the LMS. This option will provide the following information on the screen:
  - Total number of books in the LMS
  - Number of unique books in the LMS
  - Number of archived books in the LMS
  - Number of books in the LMS published after a specific year
  - Book distribution by publisher
  - Book distribution by year

- **Option (7) : Exit : -**
  The final enhancement allows users to exit the LMS. Upon selecting this option, the LMS will terminate, and all book data will be saved to an LMS file. The next time the LMS is launched, it will load the data from this file.

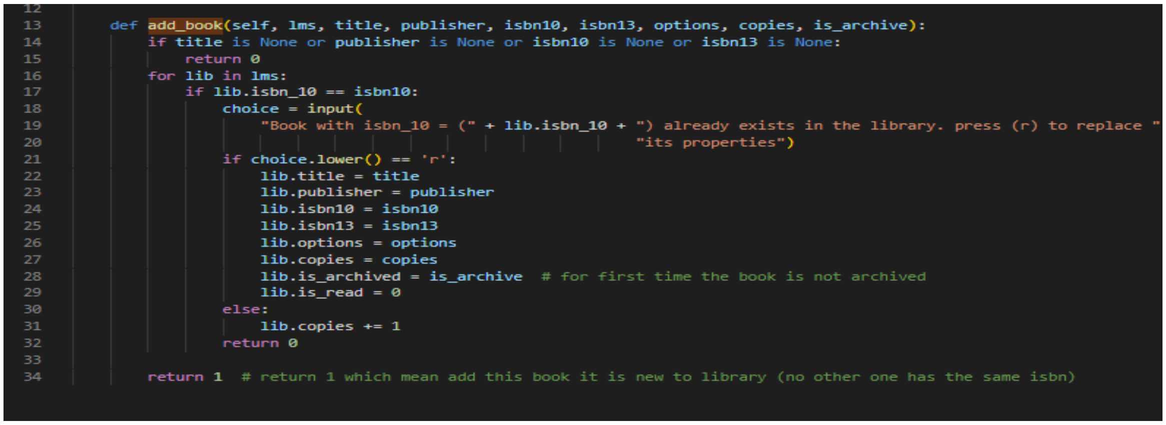
✞ Task1:

choosing 1st option … when choosing option1 , the storing data in right position starts in the moment of reading the file then we call the add book , there is an exception handling if the file does not exist ,



```python
if choice == '1':
    print("--------------------------------------------------------------------------------")
    file_name = input("provide the name of the file contains books information: ")
    # Read the file and extract book information
    if os.path.exists(file_name):
        with open(file_name, 'r') as file:  # this line is for open and close the file automatically
            lines = file.readlines()
            option = []
            title = publisher = isbn_10 = isbn_13 = None  # define the needed variables which will be used bellow
            for line in lines:
                copy = 1  # always initialized the first number of copy for book to one
                # first we must split each line as (:) delimiter and then save the important info of each book
                book_info = line.strip().split(':', 1)
                if book_info[0].replace(" ", "").lower() == 'title':
                    title = book_info[1].lstrip()
                elif book_info[0].replace(" ", "").lower() == 'publisher':
                    publisher = book_info[1].lstrip()
                elif book_info[0].replace(" ", "").lower() == 'isbn-10':
                    isbn_10 = book_info[1].lstrip()
                elif book_info[0].replace(" ", "").lower() == 'isbn-13':
                    isbn_13 = book_info[1].lstrip()
                elif book_info[0] == '':
                    copy_option = option.copy()
                    lib = library.Library(title, publisher, isbn_10, isbn_13, copy_option, copy, 0, 0)
                    adding_check = lib.add_book(LMS, title, publisher, isbn_10, isbn_13, copy_option, copy, 0)
                    if adding_check == 1:
                        LMS.append(lib)  # add this new book to library
                    option.clear()
                    title = publisher = isbn_10 = isbn_13 = None  # empty all values to check if next book does
                    # not has this important information

                else:  # adding this information as option info for this book
                    option.append(book_info)
        print("---------------------------------------")
        print("The books information that were loaded :")
        print_books(LMS)
        print("---------------------------------------")
    else:
        print("The file does not exist!")
```

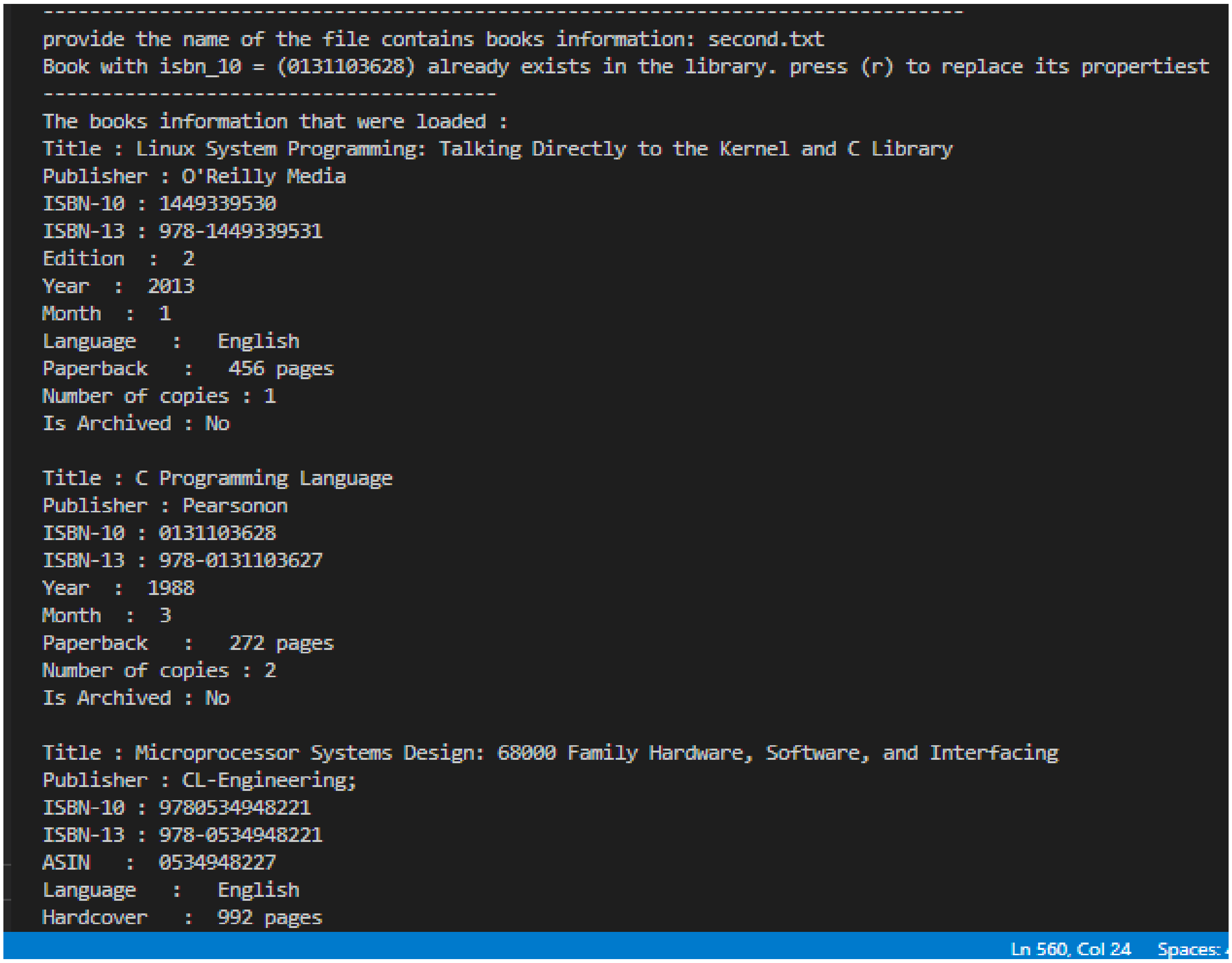
add book functionality… allows for the addition of new books to the LMS. It checks for existing books with the same ISBN-10, provides options to replace or add copies, and updates the book's information accordingly



```python
def add_book(self, lms, title, publisher, isbn10, isbn13, options, copies, is_archive):
    if title is None or publisher is None or isbn10 is None or isbn13 is None:
        return 0
    for lib in lms:
        if lib.isbn_10 == isbn10:
            choice = input(
                "Book with isbn_10 = (" + lib.isbn_10 + ") already exists in the library. press (r) to replace "
                                                        "its properties")
            if choice.lower() == 'r':
                lib.title = title
                lib.publisher = publisher
                lib.isbn10 = isbn10
                lib.isbn13 = isbn13
                lib.options = options
                lib.copies = copies
                lib.is_archived = is_archive  # for first time the book is not archived
                lib.is_read = 0
            else:
                lib.copies += 1
            return 0

    return 1  # return 1 which mean add this book it is new to library (no other one has the same isbn)
```

. **For example**:

As you see here , we added a two file that contains a same book , so the program asked if you want to replace it or considerate another copy , watch number of copies in “C programming language” book when I chose to make it a copy

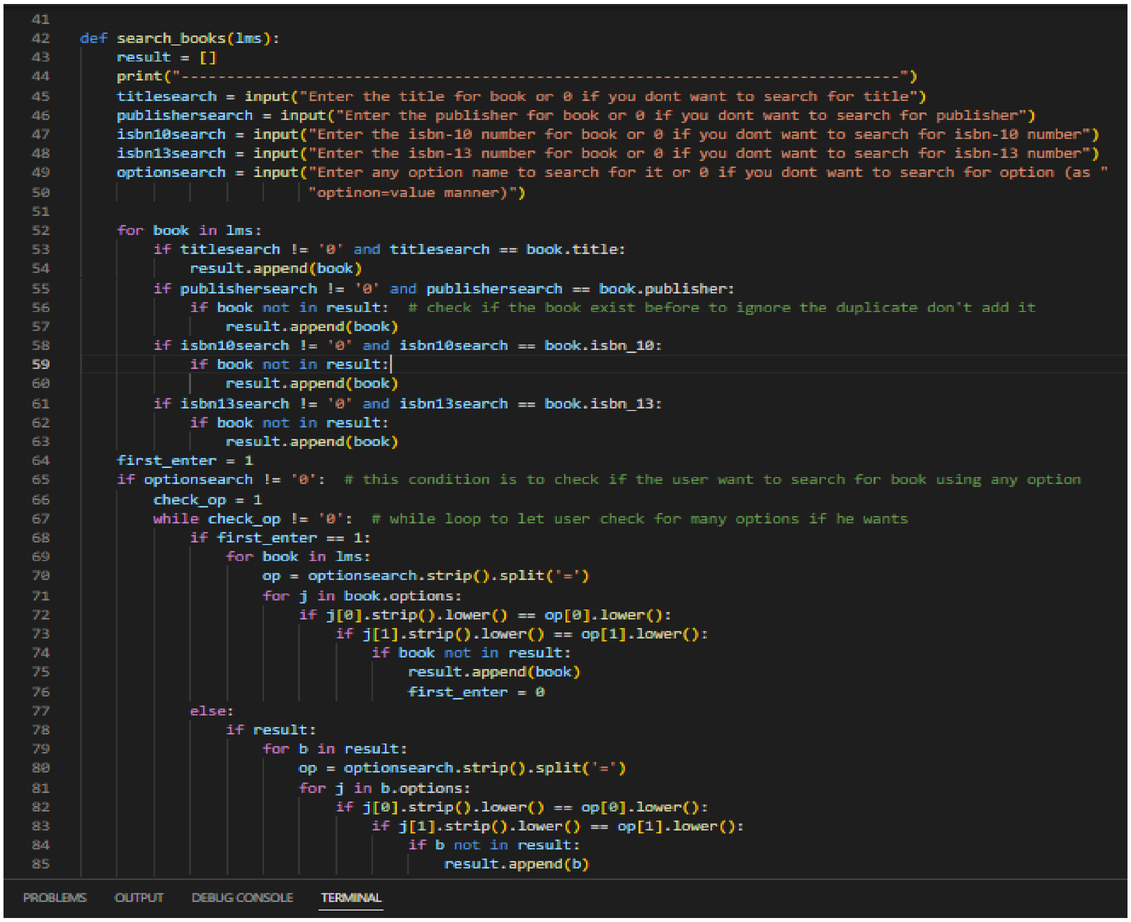


```
--------------------------------------------------------------------------------
provide the name of the file contains books information: second.txt
Book with isbn_10 = (0131103628) already exists in the library. press (r) to replace its propertiest
---------------------------------------
The books information that were loaded :
Title : Linux System Programming: Talking Directly to the Kernel and C Library
Publisher : O'Reilly Media
ISBN-10 : 1449339530
ISBN-13 : 978-1449339531
Edition  :  2
Year  :  2013
Month  :  1
Language   :   English
Paperback   :   456 pages
Number of copies : 1
Is Archived : No

Title : C Programming Language
Publisher : Pearsonon
ISBN-10 : 0131103628
ISBN-13 : 978-0131103627
Year  :  1988
Month  :  3
Paperback   :   272 pages
Number of copies : 2
Is Archived : No

Title : Microprocessor Systems Design: 68000 Family Hardware, Software, and Interfacing
Publisher : CL-Engineering;
ISBN-10 : 9780534948221
ISBN-13 : 978-0534948221
ASIN   :  0534948227
Language   :   English
Hardcover   :  992 pages
```

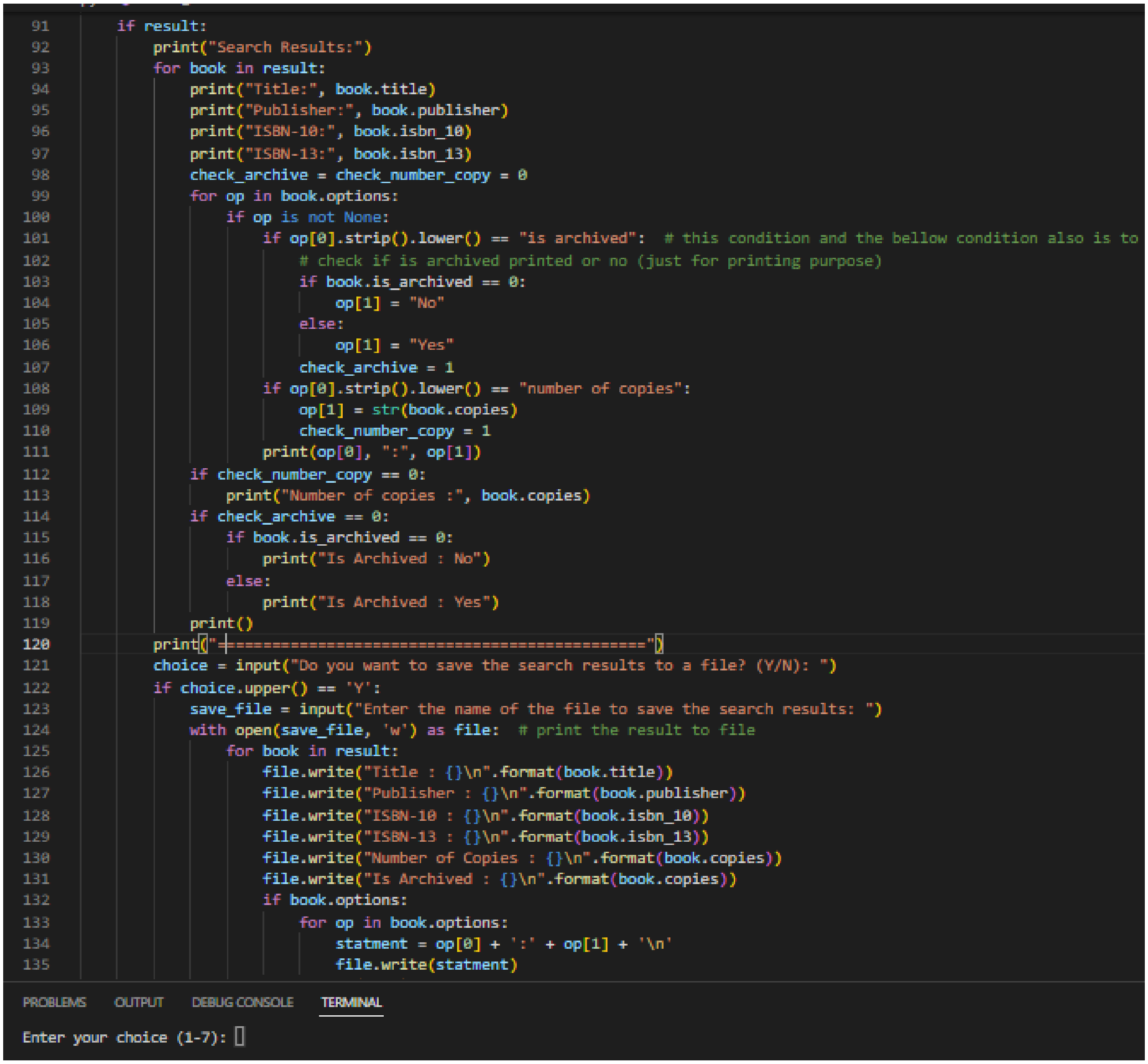
✞ **Task2:**

Search functionality :

The search_books function allows users to search for books based on different criteria such as title, publisher, ISBN numbers, and specific options. It iterates over a list of books and checks for matches, adding them to a result list. If the user specifies additional options, the function further filters the results. The function then displays the search results, including book details and any specified options. Users have the option to save the results to a file. If no matches are found, a message indicating so is displayed. Overall, the function provides a flexible search functionality for managing and retrieving book information.



```python

def search_books(lms):
    result = []
    print("--------------------------------------------------------------------------------")
    titlesearch = input("Enter the title for book or 0 if you dont want to search for title")
    publishersearch = input("Enter the publisher for book or 0 if you dont want to search for publisher")
    isbn10search = input("Enter the isbn-10 number for book or 0 if you dont want to search for isbn-10 number")
    isbn13search = input("Enter the isbn-13 number for book or 0 if you dont want to search for isbn-13 number")
    optionsearch = input("Enter any option name to search for it or 0 if you dont want to search for option (as "
                         "optinon=value manner)")

    for book in lms:
        if titlesearch != '0' and titlesearch == book.title:
            result.append(book)
        if publishersearch != '0' and publishersearch == book.publisher:
            if book not in result:  # check if the book exist before to ignore the duplicate don't add it
                result.append(book)
        if isbn10search != '0' and isbn10search == book.isbn_10:
            if book not in result:
                result.append(book)
        if isbn13search != '0' and isbn13search == book.isbn_13:
            if book not in result:
                result.append(book)
    first_enter = 1
    if optionsearch != '0':  # this condition is to check if the user want to search for book using any option
        check_op = 1
        while check_op != '0':  # while loop to let user check for many options if he wants
            if first_enter == 1:
                for book in lms:
                    op = optionsearch.strip().split('=')
                    for j in book.options:
                        if j[0].strip().lower() == op[0].lower():
                            if j[1].strip().lower() == op[1].lower():
                                if book not in result:
                                    result.append(book)
                                    first_enter = 0
            else:
                if result:
                    for b in result:
                        op = optionsearch.strip().split('=')
                        for j in b.options:
                            if j[0].strip().lower() == op[0].lower():
                                if j[1].strip().lower() == op[1].lower():
                                    if b not in result:
                                        result.append(b)
```



```
    if result:
        print("Search Results:")
        for book in result:
            print("Title:", book.title)
            print("Publisher:", book.publisher)
            print("ISBN-10:", book.isbn_10)
            print("ISBN-13:", book.isbn_13)
            check_archive = check_number_copy = 0
            for op in book.options:
                if op is not None:
                    if op[0].strip().lower() == "is archived":  # this condition and the bellow condition also is to
                        # check if is archived printed or no (just for printing purpose)
                        if book.is_archived == 0:
                            op[1] = "No"
                        else:
                            op[1] = "Yes"
                        check_archive = 1
                    if op[0].strip().lower() == "number of copies":
                        op[1] = str(book.copies)
                        check_number_copy = 1
                    print(op[0], ":", op[1])
            if check_number_copy == 0:
                print("Number of copies :", book.copies)
            if check_archive == 0:
                if book.is_archived == 0:
                    print("Is Archived : No")
                else:
                    print("Is Archived : Yes")
            print()
        print("==============================================")
        choice = input("Do you want to save the search results to a file? (Y/N): ")
        if choice.upper() == 'Y':
            save_file = input("Enter the name of the file to save the search results: ")
            with open(save_file, 'w') as file:  # print the result to file
                for book in result:
                    file.write("Title : {}\n".format(book.title))
                    file.write("Publisher : {}\n".format(book.publisher))
                    file.write("ISBN-10 : {}\n".format(book.isbn_10))
                    file.write("ISBN-13 : {}\n".format(book.isbn_13))
                    file.write("Number of Copies : {}\n".format(book.copies))
                    file.write("Is Archived : {}\n".format(book.copies))
                    if book.options:
                        for op in book.options:
                            statment = op[0] + ':' + op[1] + '\n'
                            file.write(statment)
```

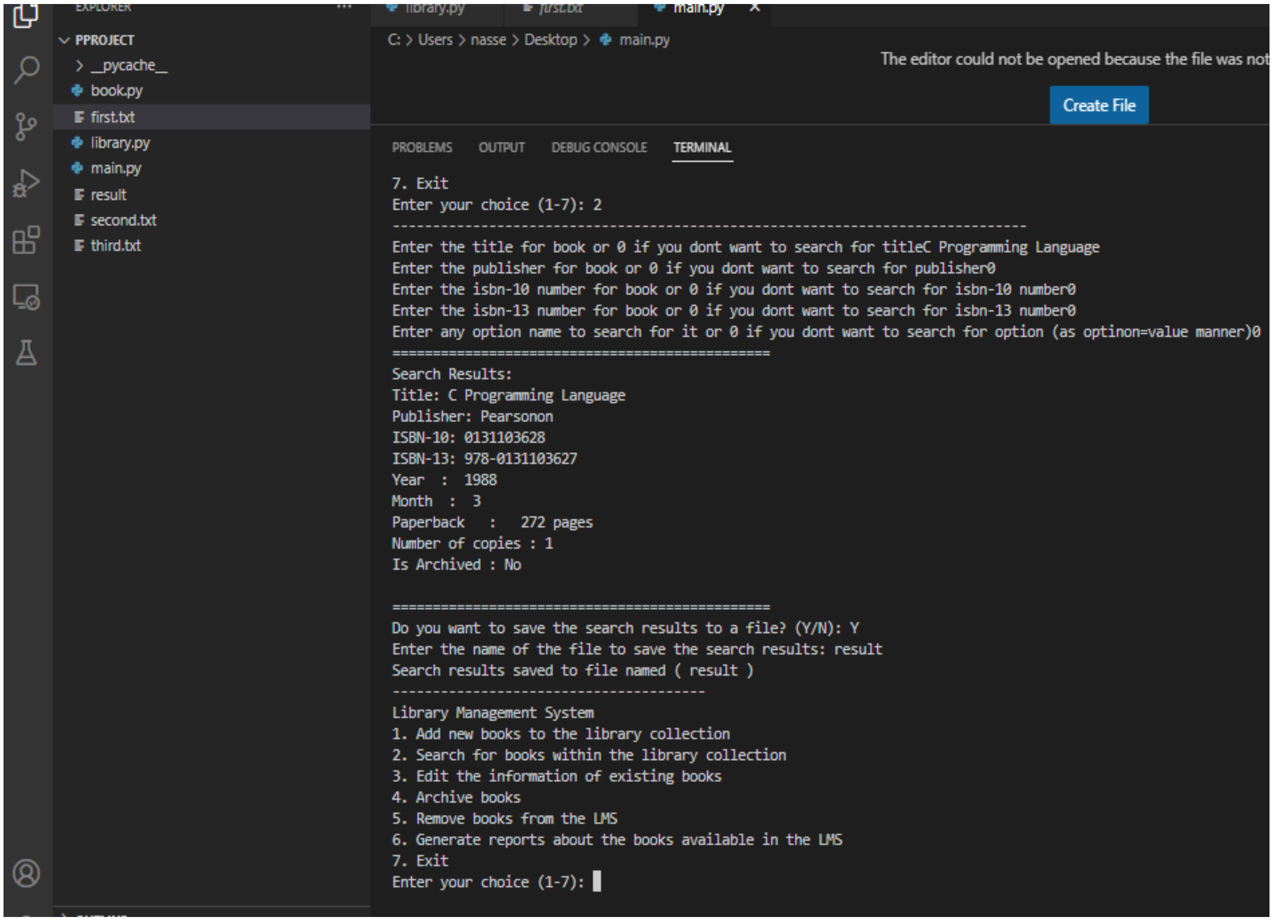
```
Enter your choice (1-7): 
```

Saving in file :

To write the search results to a file in the search_books function, the user is prompted for a file name to save the results. The function opens the file in write mode and uses a with statement to handle file operations automatically. It iterates over each book in the result list and writes its details, including title, publisher, ISBN numbers, number of copies, and any options, to the file. Each piece of information is formatted and written on separate lines. Once all books are processed, the file is closed. A message is then displayed, confirming that the search results have been saved to the file.

For example :

After make sure that you passed option 1 and there are books in library , you can search for it according to its info , and after that you can make the result appear in terminal or in a new file named by user . , as you see here I chose to search by title and save in file

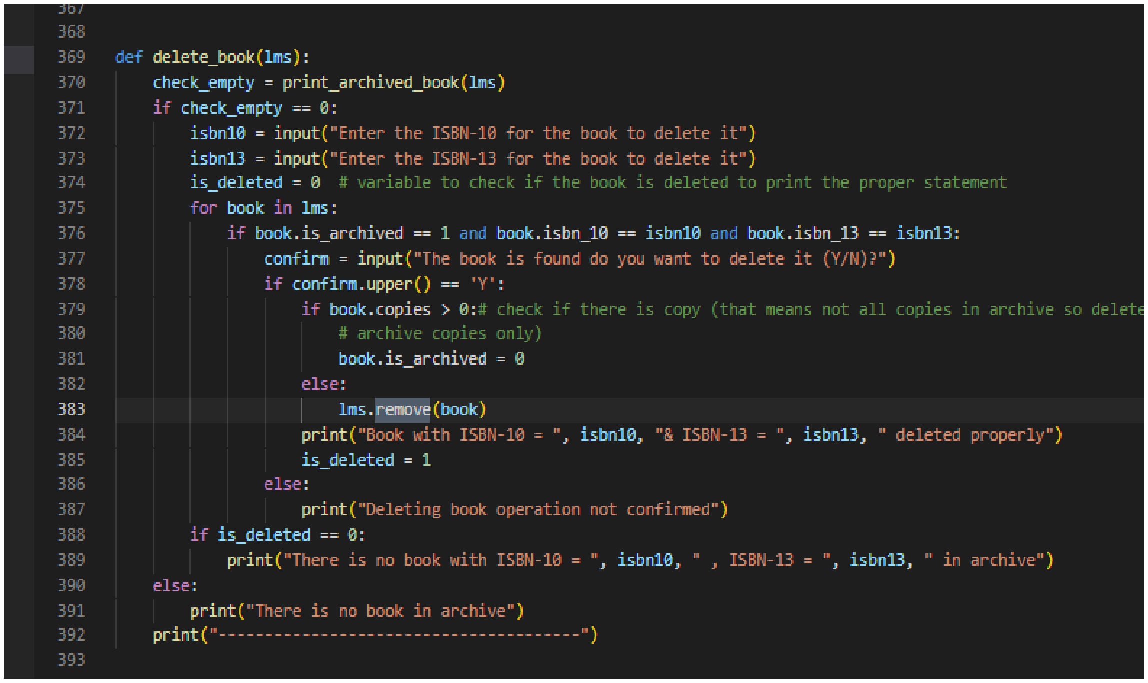


```
7. Exit
Enter your choice (1-7): 2
--------------------------------------------------------------------------------
Enter the title for book or 0 if you dont want to search for titleC Programming Language
Enter the publisher for book or 0 if you dont want to search for publisher0
Enter the isbn-10 number for book or 0 if you dont want to search for isbn-10 number0
Enter the isbn-13 number for book or 0 if you dont want to search for isbn-13 number0
Enter any option name to search for it or 0 if you dont want to search for option (as optinon=value manner)0
================================================
Search Results:
Title: C Programming Language
Publisher: Pearsonon
ISBN-10: 0131103628
ISBN-13: 978-0131103627
Year  :  1988
Month  :  3
Paperback   :   272 pages
Number of copies : 1
Is Archived : No

================================================
Do you want to save the search results to a file? (Y/N): Y
Enter the name of the file to save the search results: result
Search results saved to file named ( result )
---------------------------------------
Library Management System
1. Add new books to the library collection
2. Search for books within the library collection
3. Edit the information of existing books
4. Archive books
5. Remove books from the LMS
6. Generate reports about the books available in the LMS
7. Exit
Enter your choice (1-7): 
```

▯

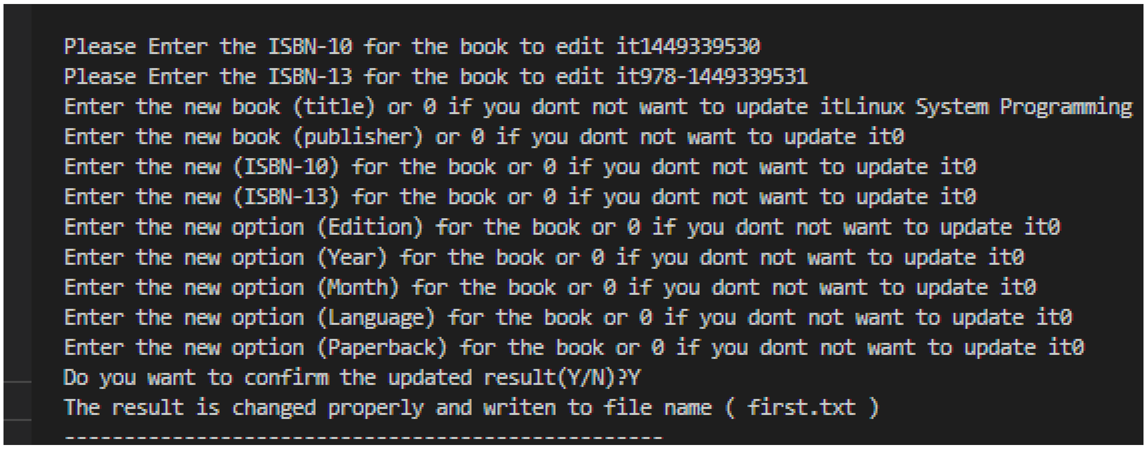
✞ Task3:

The delete_book function enables the deletion of a book from the library management system's archive (lms). It first displays the list of books in the archive using the print_archived_book function. If there are books in the archive, the user is prompted to enter the ISBN-10 and ISBN-13 of the book to be deleted. The function searches for a matching book in the archive and asks for confirmation before proceeding with deletion. If confirmed, the book is either unarchived (if there are remaining copies) or completely removed from the system. Messages are displayed to confirm the deletion or indicate if no matching book is found or if the deletion is not confirmed. Finally, a line of dashes separates the output.



```python
def delete_book(lms):
    check_empty = print_archived_book(lms)
    if check_empty == 0:
        isbn10 = input("Enter the ISBN-10 for the book to delete it")
        isbn13 = input("Enter the ISBN-13 for the book to delete it")
        is_deleted = 0  # variable to check if the book is deleted to print the proper statement
        for book in lms:
            if book.is_archived == 1 and book.isbn_10 == isbn10 and book.isbn_13 == isbn13:
                confirm = input("The book is found do you want to delete it (Y/N)?")
                if confirm.upper() == 'Y':
                    if book.copies > 0:# check if there is copy (that means not all copies in archive so delete
                        # archive copies only)
                        book.is_archived = 0
                    else:
                        lms.remove(book)
                    print("Book with ISBN-10 = ", isbn10, "& ISBN-13 = ", isbn13, " deleted properly")
                    is_deleted = 1
                else:
                    print("Deleting book operation not confirmed")
        if is_deleted == 0:
            print("There is no book with ISBN-10 = ", isbn10, " , ISBN-13 = ", isbn13, " in archive")
    else:
        print("There is no book in archive")
    print("---------------------------------------")
```

For example :

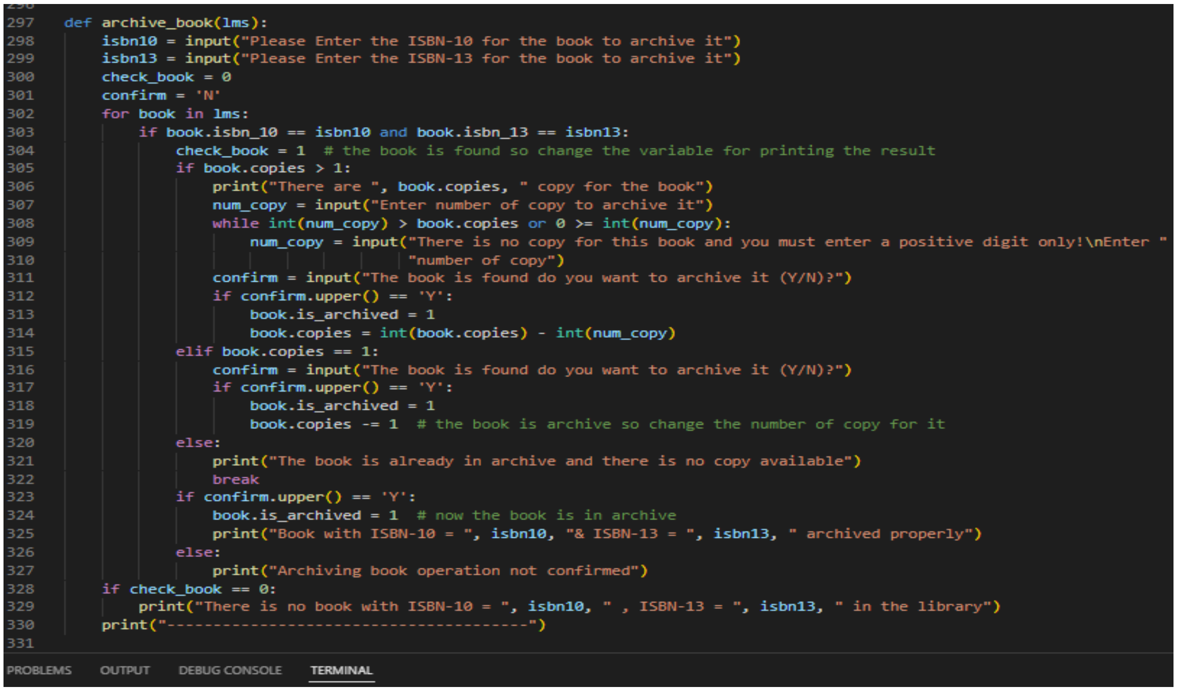


```
Please Enter the ISBN-10 for the book to edit it1449339530
Please Enter the ISBN-13 for the book to edit it978-1449339531
Enter the new book (title) or 0 if you dont not want to update itLinux System Programming
Enter the new book (publisher) or 0 if you dont not want to update it0
Enter the new (ISBN-10) for the book or 0 if you dont not want to update it0
Enter the new (ISBN-13) for the book or 0 if you dont not want to update it0
Enter the new option (Edition) for the book or 0 if you dont not want to update it0
Enter the new option (Year) for the book or 0 if you dont not want to update it0
Enter the new option (Month) for the book or 0 if you dont not want to update it0
Enter the new option (Language) for the book or 0 if you dont not want to update it0
Enter the new option (Paperback) for the book or 0 if you dont not want to update it0
Do you want to confirm the updated result(Y/N)?Y
The result is changed properly and writen to file name ( first.txt )
--------------------------------------------------
```

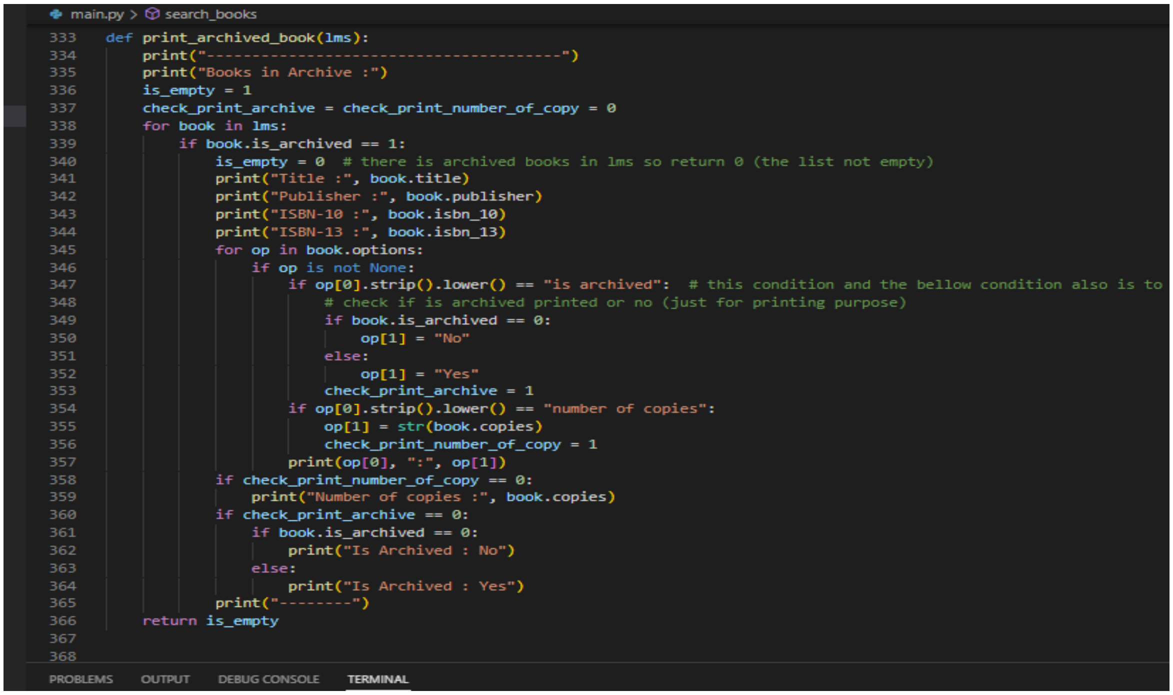
File:

✞ Task4:

In summary, the archive_book function facilitates archiving a book, while the print_archived_book function displays the details of archived books. These functions provide functionality to manage and retrieve information about archived books in the library.



```python
def archive_book(lms):
    isbn10 = input("Please Enter the ISBN-10 for the book to archive it")
    isbn13 = input("Please Enter the ISBN-13 for the book to archive it")
    check_book = 0
    confirm = 'N'
    for book in lms:
        if book.isbn_10 == isbn10 and book.isbn_13 == isbn13:
            check_book = 1  # the book is found so change the variable for printing the result
            if book.copies > 1:
                print("There are ", book.copies, " copy for the book")
                num_copy = input("Enter number of copy to archive it")
                while int(num_copy) > book.copies or 0 >= int(num_copy):
                    num_copy = input("There is no copy for this book and you must enter a positive digit only!\nEnter "
                                     "number of copy")
                confirm = input("The book is found do you want to archive it (Y/N)?")
                if confirm.upper() == 'Y':
                    book.is_archived = 1
                    book.copies = int(book.copies) - int(num_copy)
            elif book.copies == 1:
                confirm = input("The book is found do you want to archive it (Y/N)?")
                if confirm.upper() == 'Y':
                    book.is_archived = 1
                    book.copies -= 1  # the book is archive so change the number of copy for it
            else:
                print("The book is already in archive and there is no copy available")
                break
            if confirm.upper() == 'Y':
                book.is_archived = 1  # now the book is in archive
                print("Book with ISBN-10 = ", isbn10, "& ISBN-13 = ", isbn13, " archived properly")
            else:
                print("Archiving book operation not confirmed")
    if check_book == 0:
        print("There is no book with ISBN-10 = ", isbn10, " , ISBN-13 = ", isbn13, " in the library")
    print("--------------------------------------")
```

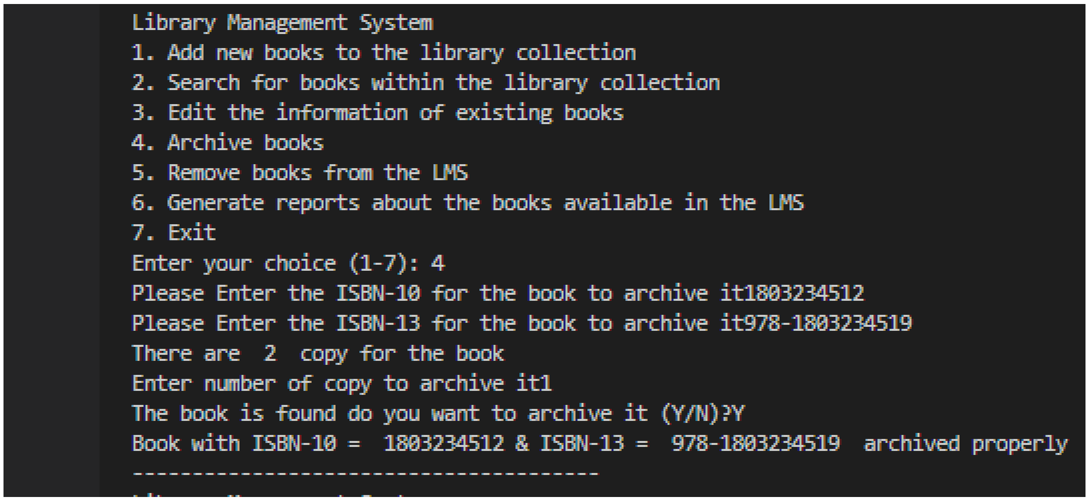


```python
def print_archived_book(lms):
    print("--------------------------------------")
    print("Books in Archive :")
    is_empty = 1
    check_print_archive = check_print_number_of_copy = 0
    for book in lms:
        if book.is_archived == 1:
            is_empty = 0  # there is archived books in lms so return 0 (the list not empty)
            print("Title :", book.title)
            print("Publisher :", book.publisher)
            print("ISBN-10 :", book.isbn_10)
            print("ISBN-13 :", book.isbn_13)
            for op in book.options:
                if op is not None:
                    if op[0].strip().lower() == "is archived":  # this condition and the bellow condition also is to
                        # check if is archived printed or no (just for printing purpose)
                        if book.is_archived == 0:
                            op[1] = "No"
                        else:
                            op[1] = "Yes"
                        check_print_archive = 1
                    if op[0].strip().lower() == "number of copies":
                        op[1] = str(book.copies)
                        check_print_number_of_copy = 1
                    print(op[0], ":", op[1])
            if check_print_number_of_copy == 0:
                print("Number of copies :", book.copies)
            if check_print_archive == 0:
                if book.is_archived == 0:
                    print("Is Archived : No")
                else:
                    print("Is Archived : Yes")
            print("--------")
    return is_empty
```

□ File:

For example :

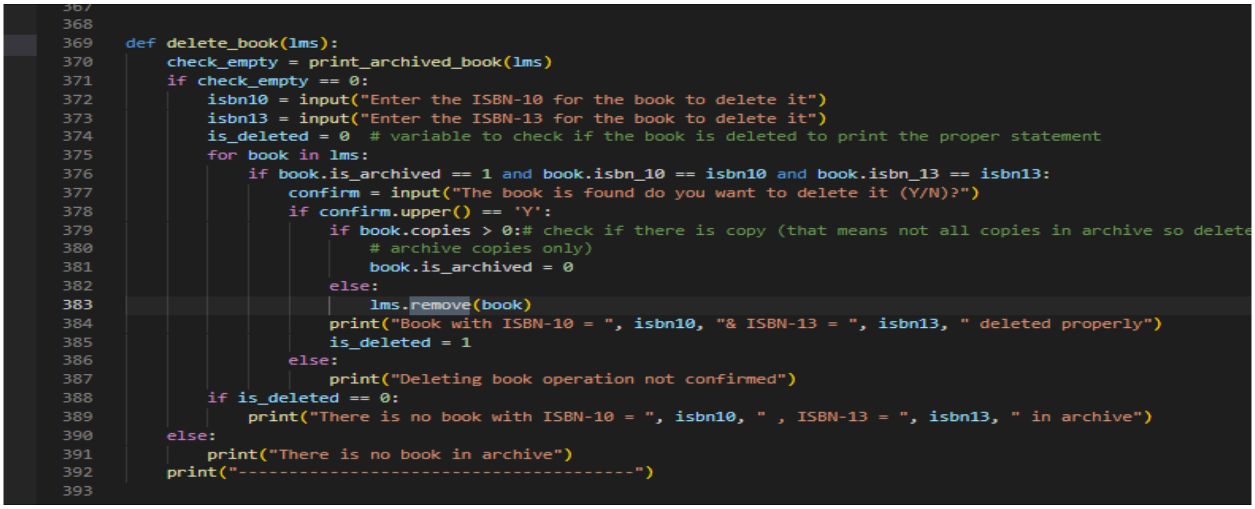
We archived a book that has 2 copies , confirmed it has been found and archived successfully



```
Library Management System
1. Add new books to the library collection
2. Search for books within the library collection
3. Edit the information of existing books
4. Archive books
5. Remove books from the LMS
6. Generate reports about the books available in the LMS
7. Exit
Enter your choice (1-7): 4
Please Enter the ISBN-10 for the book to archive it1803234512
Please Enter the ISBN-13 for the book to archive it978-1803234519
There are  2  copy for the book
Enter number of copy to archive it1
The book is found do you want to archive it (Y/N)?Y
Book with ISBN-10 =  1803234512 & ISBN-13 =  978-1803234519  archived properly
--------------------------------------
```

✞ Task5:

The delete_book function enables the deletion of a book from the library management system's archive (lms). It first displays the list of books in the archive using the print_archived_book function. If there are books in the archive, the user is prompted to enter the ISBN-10 and ISBN-13 of the book to be deleted. The function searches for a matching book in the archive and asks for confirmation before proceeding with deletion. If confirmed, the book is either unarchived (if there are remaining copies) or completely removed from the system. Messages are displayed to confirm the deletion or indicate if no matching book is found or if the deletion is not confirmed. Finally, a line of dashes separates the output.

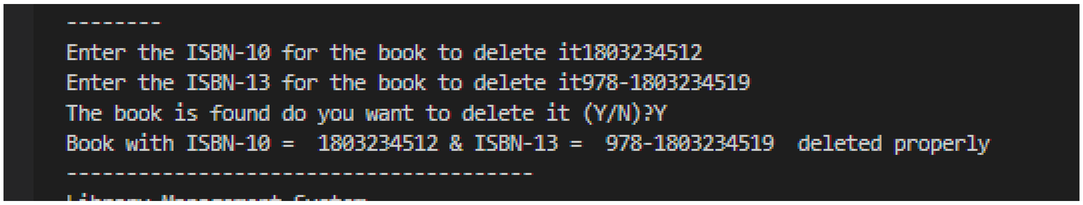


```python

def delete_book(lms):
    check_empty = print_archived_book(lms)
    if check_empty == 0:
        isbn10 = input("Enter the ISBN-10 for the book to delete it")
        isbn13 = input("Enter the ISBN-13 for the book to delete it")
        is_deleted = 0  # variable to check if the book is deleted to print the proper statement
        for book in lms:
            if book.is_archived == 1 and book.isbn_10 == isbn10 and book.isbn_13 == isbn13:
                confirm = input("The book is found do you want to delete it (Y/N)?")
                if confirm.upper() == 'Y':
                    if book.copies > 0:# check if there is copy (that means not all copies in archive so delete
                        # archive copies only)
                        book.is_archived = 0
                    else:
                        lms.remove(book)
                    print("Book with ISBN-10 = ", isbn10, "& ISBN-13 = ", isbn13, " deleted properly")
                    is_deleted = 1
                else:
                    print("Deleting book operation not confirmed")
        if is_deleted == 0:
            print("There is no book with ISBN-10 = ", isbn10, " , ISBN-13 = ", isbn13, " in archive")
    else:
        print("There is no book in archive")
    print("--------------------------------------")

```

File:

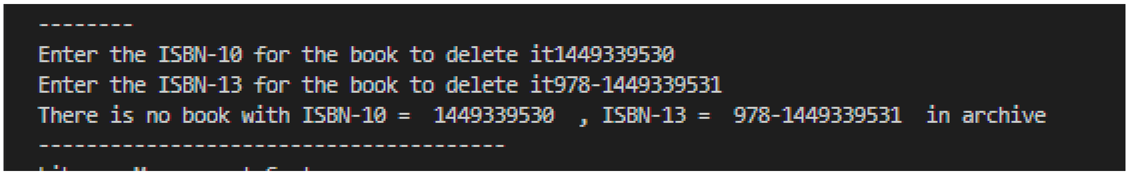
For example :

1 – if the book is archived :



```
--------
Enter the ISBN-10 for the book to delete it1803234512
Enter the ISBN-13 for the book to delete it978-1803234519
The book is found do you want to delete it (Y/N)?Y
Book with ISBN-10 =  1803234512 & ISBN-13 =  978-1803234519  deleted properly
--------------------------------------
```

2- if the book is not archived :



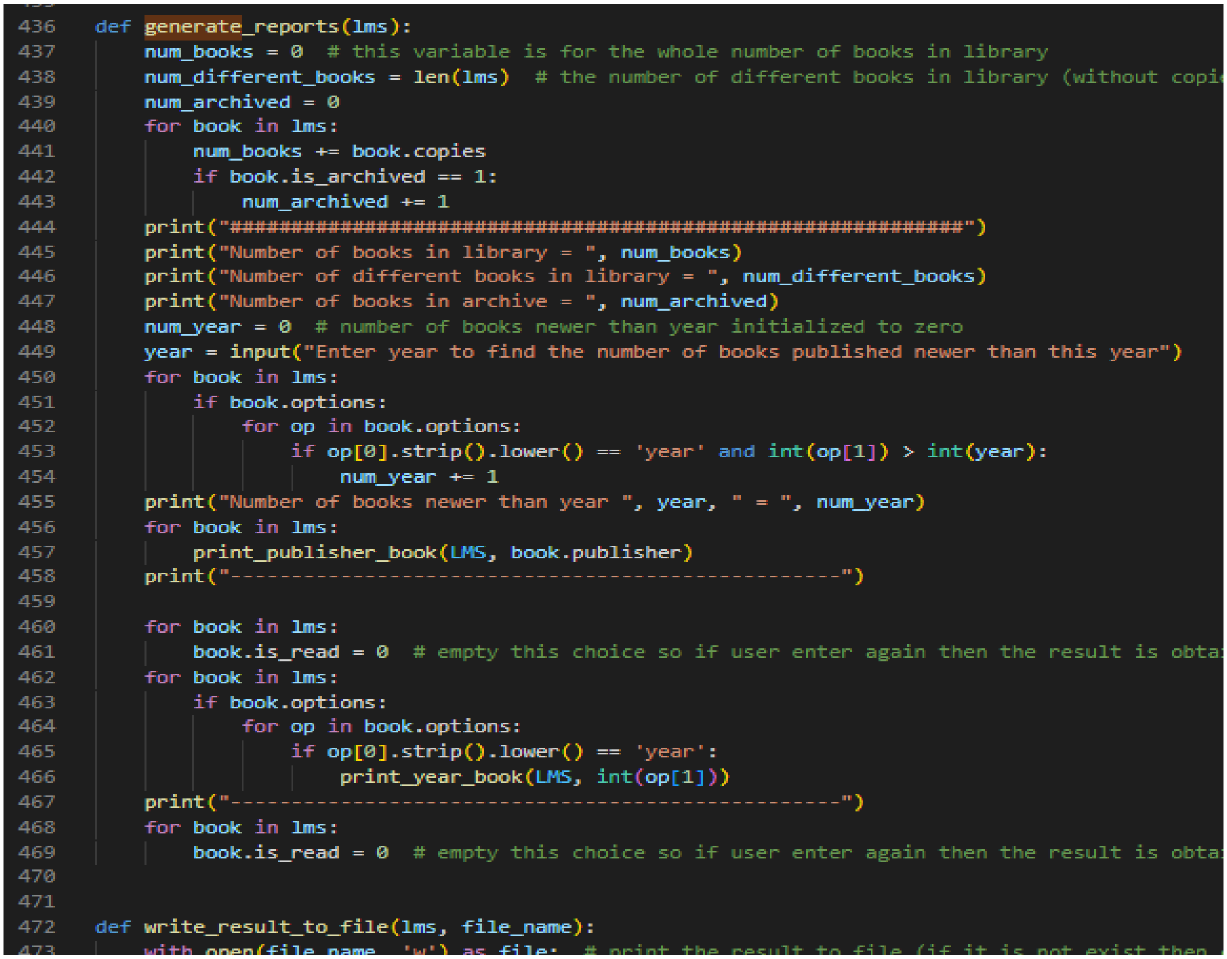
```
--------
Enter the ISBN-10 for the book to delete it1449339530
Enter the ISBN-13 for the book to delete it978-1449339531
There is no book with ISBN-10 =  1449339530  , ISBN-13 =  978-1449339531  in archive
--------------------------------------
```

✞ Task6:

The generate_reports function calculates and prints various reports based on the library management system (lms). It determines the total number of books, the number of different books (excluding copies), the number of archived books, and the number of books published newer than a specified year. It also calls other functions (print_publisher_book and print_year_book) to print additional reports. After generating the reports, the function clears the is_read attribute of each book.

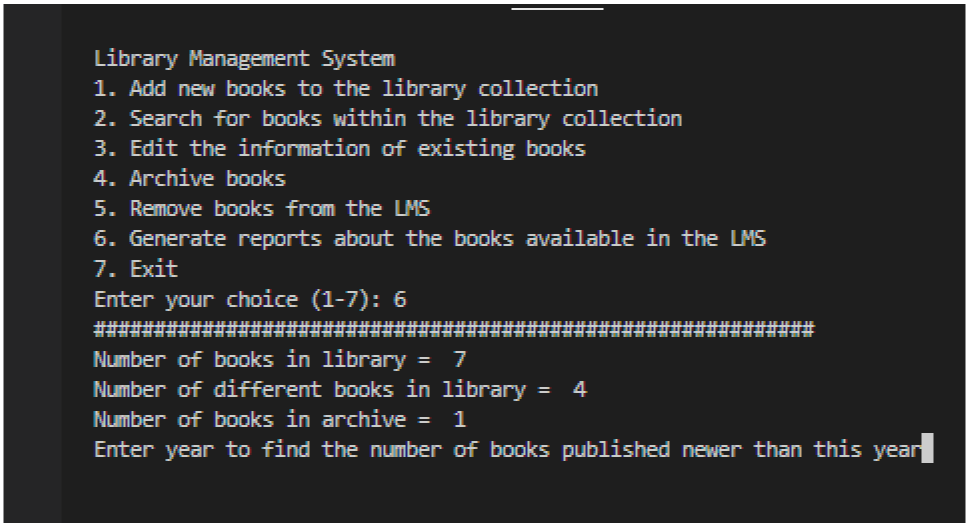
The write_result_to_file function writes the information from the library management system (lms) to a specified file. It iterates over each book and writes its title, publisher, ISBN numbers, number of copies, archiving status, and any additional options to the file.

File:



```python
def generate_reports(lms):
    num_books = 0  # this variable is for the whole number of books in library
    num_different_books = len(lms)  # the number of different books in library (without copie
    num_archived = 0
    for book in lms:
        num_books += book.copies
        if book.is_archived == 1:
            num_archived += 1
    print("############################################################")
    print("Number of books in library = ", num_books)
    print("Number of different books in library = ", num_different_books)
    print("Number of books in archive = ", num_archived)
    num_year = 0  # number of books newer than year initialized to zero
    year = input("Enter year to find the number of books published newer than this year")
    for book in lms:
        if book.options:
            for op in book.options:
                if op[0].strip().lower() == 'year' and int(op[1]) > int(year):
                    num_year += 1
    print("Number of books newer than year ", year, " = ", num_year)
    for book in lms:
        print_publisher_book(LMS, book.publisher)
    print("--------------------------------------------------")

    for book in lms:
        book.is_read = 0  # empty this choice so if user enter again then the result is obta
    for book in lms:
        if book.options:
            for op in book.options:
                if op[0].strip().lower() == 'year':
                    print_year_book(LMS, int(op[1]))
    print("--------------------------------------------------")
    for book in lms:
        book.is_read = 0  # empty this choice so if user enter again then the result is obta


def write_result_to_file(lms, file_name):
    with open(file_name, 'w') as file:  # print the result to file (if it is not exist then
```

For example :

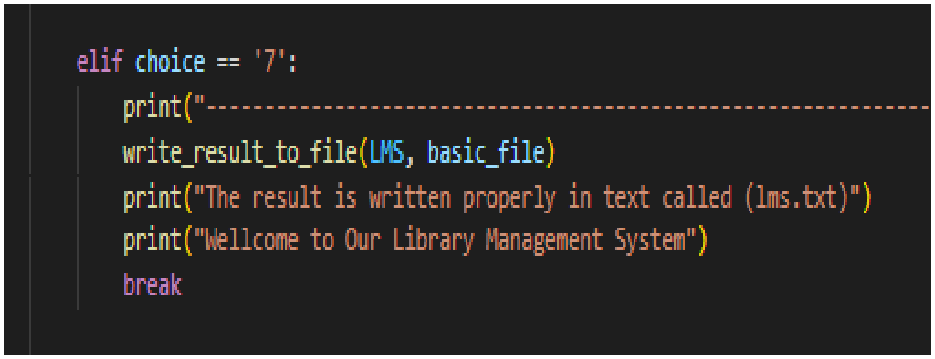


```
Library Management System
1. Add new books to the library collection
2. Search for books within the library collection
3. Edit the information of existing books
4. Archive books
5. Remove books from the LMS
6. Generate reports about the books available in the LMS
7. Exit
Enter your choice (1-7): 6
############################################################
Number of books in library =  7
Number of different books in library =  4
Number of books in archive =  1
Enter year to find the number of books published newer than this year
```

File:

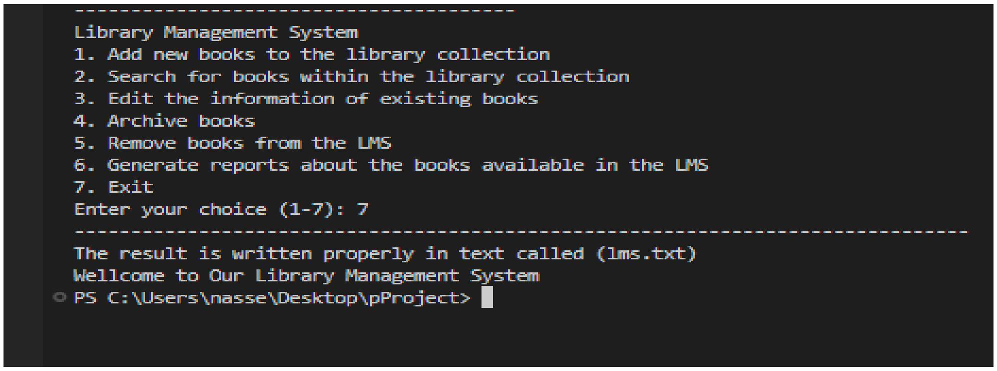
✞ Task7:

In this code snippet, when the user selects option 7 in the menu, the program calls the write_result_to_file function to write the result to a file. Afterward, a message confirms that the result has been successfully written. The program then displays a welcome message to the Library Management System and terminates.



```
elif choice == '7':
    print("-------------------------------------------------------------
    write_result_to_file(LMS, basic_file)
    print("The result is written properly in text called (lms.txt)")
    print("Wellcome to Our Library Management System")
    break
```

For example :



```
----------------------------------------
Library Management System
1. Add new books to the library collection
2. Search for books within the library collection
3. Edit the information of existing books
4. Archive books
5. Remove books from the LMS
6. Generate reports about the books available in the LMS
7. Exit
Enter your choice (1-7): 7
--------------------------------------------------------------------------------
The result is written properly in text called (lms.txt)
Wellcome to Our Library Management System
PS C:\Users\nasse\Desktop\pProject>
```