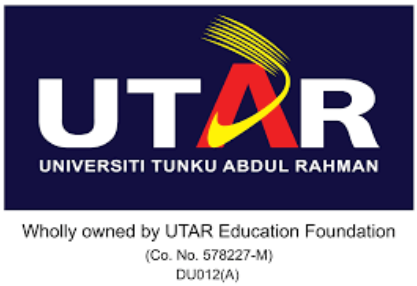
UNIVERSITI TUNKU ABDUL RAHMAN

ACADEMIC YEAR 2022/2023



UCCD 1004 PROGRAMMING CONCEPTS AND PRACTICES

ASSIGNMENT 2

Group 39			
Name	**ID**	**Programme**	**UTAR Email**
DESMOND HO JIA SHEN	2105034	CS	desmondhjs@1utar.my
ONG YI SHENG	2103887	CS	ongyisheng0309@1utar.my
TAN KAI JUN	2206494	CS	kaijuntan423@1utar.my

Task Division

	Name	Modules	Description	*A2 Contribution (Overall, %)	
1.	DESMOND HO JIA SHEN	ADD	Adding books into library system	1	
		DELETE	Deleting books from library system	2	
		SEARCH	Searching books from library system	3	
2.	ONG YI SHENG	PENALTY	Counting the differences of date that student return and giving penalty for late return	1	
		PAYMENT	Multiple payment ways system and receipt	2	
		SENSOR	Booking seat system at library for student	3	
3.	TAN KAI JUN	CHECK IN	Borrowed books from library system	1	
		VIEW	Search and view books list and student's borrowed list in library system	2	
		CHECK OUT	Return borrowed books from library system	3	

* Depends on the evaluation of the markers as well;

Objectives

- The objective of this C++ library management system is to provide an efficient and user-friendly interface for managing library resources. The system should allow administrators to easily add, delete and update books, as well as manage member accounts and lending records. The system should also provide a user interface for members to browse and request books, as well as track their lending history. In addition, the system should have a secure login system to protect the confidentiality of user information. The ultimate goal of this library system is to promote access to educational resources and encourage lifelong learning.

Pseudocode

```
FUNCTION showOptions(string& ADMIN_name)
{
	system clear screen;
	print("Welcome back Admin", ADMIN_name, "What would you like to do today ? ");
	print("options");//decorative design
	print("1.Search Books");
	print("2.View Books");
	print("3.Add Books");
	print("4.Delete Books");
	print("5.Logout");
}

FUNCTION view()
{
	system clear screen;
	print("VIEW BOOK LIST");//decorative design
}

int main()
{
	char login;
	int noADMIN = 0, noBOOKS = 0;//initialize counter to 0
	bool loginSuccessful = false;
	string ADMIN_name, NAME_check;
	string string ADMIN_pass, PASS_check;
	string BOOK_name, BOOK_check;
	string BOOKadd, BOOKdel;
	ifstream inFile;
	ofstream outFile;
	ofstream temp;

	do
```

```
{
        print("Welcome to AMBATUBUS Library System");
        print("Login");
        print("1.Administrator");
        print("2.Member");
        print("Please choose login method: ");
        input(login);

        if (login not equal to '1' and '2')
        {
                print("Please enter a number according to the menu!");
        }
} while (login not equal to '1' and '2');

if (login equal to '1')
{
        while (not loginSuccessful)
        {
                inFile open("ADMIN.txt");
                if (inFile open fail())
                {
                        print("Error opening file");
                        exit(1);
                }
                print("Username: ");
                input(ADMIN_name);
                print("Password: ");
                input(ADMIN_pass);

                while (get line NAME_check from inFile and get line PASS_check from
inFile)
                {
                        noADMIN++;
```

```
                    if (ADMIN_name equal to NAME_check and ADMIN_pass equal to
PASS_check)
                    {
                        loginSuccessful equal to true;
                        break;
                    }
                    ignore inFile;
                }
                if (noADMIN equal to 0 or ADMIN_name not equal to NAME_check or
ADMIN_pass not equal to PASS_check)
                {
                    print("Invalid username or password!");
                    loginSuccessful equal to false;
                }
                close inFile;
            }
            char options;
            do
            {
                showOptions(ADMIN_name);
                print("Please enter options: ");
                input(options);

                if (options not equal to '1', '2', '3', '4', '5')
                {
                    print("Please choose an available options!");
                    system(pause);
                }

                else if (options == '1')
                {
                    open inFile("BookList.txt");
                    if (inFile.is_open is false)
                    {
```

```
                print("Error opening file");
                system(pause);
        }
        char choice;
        do
        {
                system clear screen;
                print("SEARCH BOOKS");//decorative design
                bool foundBOOK = false;
                print("What book would you like to search for?");
                input(BOOK_name);
                seek inFile for the beginning;
                while (inFile get line for BOOK_check)
                {
                        if (BOOK_name equal to BOOK_check)
                        {
                                print("The book is registered in the library
system");

                                foundBOOK becomes true;
                                break;
                        }
                }
                if (not foundBOOK)
                {
                        print("The book is not registered in the library
system");
                }
                print("Continue Searching? (y/n)");
                input(choice);

                while (choice not equal to 'y' or 'n')
                {
                        print("Please enter y or n: ");
                        input(choice);
```

```
                }
                if (choice equal to 'n')
                {
                        break;
                }
        } while (choice equal to 'y');
        close inFile;
}
else if (options equal to '2')
{
        call view_books function;
        inFile open("BookList.txt");
        if (not inFile open())
        {
                print("Error opening file");
                system(pause);
        }
        string line;
        while (not inFile end of line())
        {
                int x = 0;
                while (getline line from inFile)
                {
                        print(x + 1. line);
                        x++;
                }
        }
        close inFile;
        print("Press enter to return to Options menu");
        system(pause);
}
else if (options == '3')
{
        char choice;
```

```
		do
		{
			system clear screen;
			print("ADD BOOKS");//decorative design
			open outFile("BookList.txt", ios_base::app);
			if (not outFile open())
			{
				print("Error opening file");
				system(pause);
			}
			print("What books would you like to add into the system: ");
			input(BOOKadd);

			append BOOKadd into outFile;
			print("Book successfully added into library");
			close outFile();

			print("Continue Adding? (y/n)");
			input(choice);

			while (choice not equal to 'y' or 'n')
			{
				print("Please enter y or n: ");
				input(choice);
			}
			if (choice equal to 'n')
			{
				break;
			}
		} while (choice equal to 'y');
}
else if (options == '4')
{
		char choice;
```

```
do
{
	system clear screen;
	print("DELETE BOOKS");//decorative design
	open inFile("BookList.txt");
	open temp("temp.txt");
	string line;
	string bookToDelete;
	print("Please enter the name of the book you want to delete:
");
	input(bookToDelete);
	bool found = false;
	while (getline(inFile, line))
	{
		if (bookToDelete != line)
		{
			temp << line << endl;
		}
		else
		{
			found = true; //set found to true when book
is found
		}
	}
	close inFile();
	close temp();
	remove("BookList.txt");
	rename("temp.txt", "BookList.txt");
	if (found)
	{
		print(bookToDelete, "has been deleted.");
	}
	else
	{
```

```
                                        print(bookToDelete, "was not found in Book List.");
                                    }
                                    print("Continue deleting? (y/n):");
                                    input(choice);
                                    while (choice not equal to 'y' or 'n')
                                    {
                                        print("Please enter y or n: ");
                                        input(choice);
                                    }
                                    if (choice equal to 'n')
                                    {
                                        break;
                                    }
                                } while (choice equal to 'y');
                            }
                        }while (options not equal to '5');
                        print("Goodbye and have a nice day!");
                        print("Looking forward to your next visit admin", ADMIN_name);
                        print(":D"):
                    }
                    return 0;
}

FUNCTION member():
   WHILE true:
      CALL user_mainmenu()
   ENDWHILE
END FUNCTION

FUNCTION is_alpha(c):
   RETURN (c is between 'A' and 'Z') OR (c is between 'a' and 'z')
END FUNCTION

FUNCTION is_full_book_name(book_name):
```

```
    OPEN_FILE "books.txt" for reading as file
    DECLARE line as string

    IF file is open:
        WHILE there are lines to read from file:
            READ line from file
            FIND book_name in line, store in pos
            IF book_name is found AND (book_name is followed by a comma OR end of the line):
                CLOSE file
                RETURN true
        ENDWHILE
        CLOSE file
    ELSE:
        PRINT "Unable to open file."

    RETURN false
END FUNCTION

FUNCTION is_valid_student_id(student_id):
    OPEN_FILE "students.txt" for reading as file
    DECLARE line as string

    IF file is open:
        WHILE there are lines to read from file:
            READ line from file
            FIND first and second commas in line, store positions in pos and next_pos
            EXTRACT student ID from line, store in id
            IF extracted ID matches input student_id:
                CLOSE file
                RETURN true
        ENDWHILE
        CLOSE file
    ELSE:
        PRINT ("Unable to open file.")
```

```
    RETURN false
END FUNCTION




FUNCTION user_mainmenu():
    DECLARE choice, validInput, diff_days

    WHILE NOT validInput:
        CLEAR console
        SET console color
        Call the function "add_new_books"
        Call the function "delete_books"
        DISPLAY main menu
        PROMPT user for choice

        IF choice is valid integer:
            SWITCH choice:
                CASE 1:
                    CALL view_books()
                    validInput = true
                CASE 2:
                    CALL check_in()
                    validInput = true
                CASE 3:
                    CALL check_out()
                    validInput = true
                CASE 4:
                    CALL view_student_books()
                    validInput = true
                CASE 5:
```

```
            CALL penalty()
            CALL payment()
            validInput = true
        CASE 6:
            CALL sensor()
            validInput = true
        CASE 7:
            DISPLAY exit message
            EXIT program
    ELSE:
        CALL handle_invalid_input()
    ENDIF
  ENDWHILE
END FUNCTION

FUNCTION handle_invalid_input():
  DISPLAY invalid input message
  PAUSE system
  CLEAR input buffer
  IGNORE remaining characters
END FUNCTION

FUNCTION return_menu():
  PAUSE system
  CLEAR input buffer
  IGNORE remaining characters
END FUNCTION

FUNCTION check_in():
  DECLARE choice
  CLEAR console
  DISPLAY check-in menu
  PROMPT user for choice
```

```
    SWITCH choice:
        CASE 1:
            CALL borrow_book()
        CASE 2:
            CALL user_mainmenu()
        DEFAULT:
            CALL handle_invalid_input()
    ENDSWITCH
END FUNCTION

FUNCTION check_out():
    DECLARE choice
    CLEAR console
    DISPLAY check-out menu
    PROMPT user for choice

    SWITCH choice:
        CASE 1:
            CALL return_book()
        CASE 2:
            CALL user_mainmenu()
        DEFAULT:
            CALL handle_invalid_input()
    ENDSWITCH
END FUNCTION

FUNCTION view_student_books():
    DECLARE student_id
    PROMPT user for student_id

    OPEN_FILE "students.txt" for reading as students_file
    DECLARE line, found

    IF students_file is open:
```

```
    WHILE there are lines to read from students_file:
        READ line from students_file
        FIND student_id in line, store in id_pos
        IF student_id is found with correct format:
            found = true
            EXTRACT student_name
            DISPLAY student_name

            FIND positions for book_start and book_end

            IF no borrowed books:
                DISPLAY "Didn't borrow any books"
            ELSE:
                DECLARE book_number
                WHILE book_end != -1:
                    EXTRACT book_title
                    DISPLAY book_title
                    UPDATE book_start, book_end, and book_number
                ENDWHILE
                EXTRACT last_book_title
                DISPLAY last_book_title
            ENDIF
            BREAK
    ENDWHILE
    CLOSE students_file
ELSE:
    DISPLAY "Unable to open file."
ENDIF

IF NOT found:
    DISPLAY "Invalid student ID."
ENDIF

DISPLAY "Press any key to continue..."
```

```
    WAIT for user input
END FUNCTION




FUNCTION update_student_record_borrow(student_id, book_name):
    OPEN_FILE "students.txt" for reading as file
    DECLARE updated_records as string
    DECLARE line as string
    DECLARE found as boolean, SET found = false

    IF file is open:
        WHILE there are lines to read from file:
            READ line from file
            FIND position of student_id in line, store in pos
            IF pos != -1:
                SET found = true
                APPEND book_name to line
            ENDIF
            APPEND line to updated_records with newline character
        ENDWHILE
        CLOSE file
    ELSE:
        PRINT ("Unable to open file.")
    ENDIF

    IF found:
        OPEN_FILE "students.txt" for writing as file
        IF file is open:
            WRITE updated_records to file
            CLOSE file
```

```
        ELSE:
            PRINT ("Unable to open file.")
        ENDIF
    ELSE:
        PRINT ("Invalid student ID. Please try again.")
    ENDIF
END FUNCTION

FUNCTION update_student_record_return(student_id, book_name):
    OPEN_FILE "students.txt" for reading as file
    DECLARE updated_records as string
    DECLARE line as string
    DECLARE found as boolean, SET found = false

    IF file is open:
        WHILE there are lines to read from file:
            READ line from file
            FIND position of student_id in line, store in pos
            IF pos != -1:
                SET found = true
                FIND position of book_name in line, store in book_pos
                IF book_pos != -1:
                    REMOVE book_name from line
                ENDIF
            ENDIF
            APPEND line to updated_records with newline character
        ENDWHILE
        CLOSE file
    ELSE:
        PRINT ("Unable to open file.")
    ENDIF

    IF found:
        OPEN_FILE "students.txt" for writing as file
```

```
        IF file is open:
            WRITE updated_records to file
            CLOSE file
        ELSE:
            PRINT ("Unable to open file.")
        ENDIF
    ELSE:
        PRINT ("Invalid student ID. Please try again.")
    ENDIF
END FUNCTION




FUNCTION view_books():
    DECLARE choice as integer

    CLEAR_SCREEN
    DISPLAY menu header
    PRINT ("1. Search book by name")
    PRINT ("2. View all books")
    PRINT ("3. Exit")
    PRINT ("Enter your choice: ")
    READ choice

    SWITCH choice:
        CASE 1:
            CALL search_book_by_name()
            BREAK
        CASE 2:
            CALL view_all_books()
            BREAK
        CASE 3:
            BREAK
```

```
        DEFAULT:
            CALL handle_invalid_input()
    ENDSWITCH
END FUNCTION

FUNCTION view_all_books():
    OPEN_FILE "books.txt" for reading as file
    DECLARE line as string

    IF file is open:
        WHILE there are lines to read from file:
            READ line from file
            FIND first comma in line, store in first_comma

            EXTRACT book_title from line, starting at 0 to first_comma
            EXTRACT status from line, starting at first_comma + 2

            IF status = "Available"
                PRINT ("Book title: ", book_title)
                PRINT ("Status: ", status)
                PRINT empty line
        ENDWHILE
        CLOSE file
    ELSE:
        PRINT ("Unable to open file.")
    ENDIF
    CALL return_menu()
END FUNCTION


FUNCTION add_new_books()
    books_file = open file "books.txt" in read mode
    booklist_file = open file "BookList.txt" in read mode
    string = line1, line 2 and updated books
```

```
    found = false

    if books_file is open
        read line1 from books_file
        while line1 is not empty
            add line1 and new line character to updated_books
            read next line1 from books_file
        close books_file

    if booklist_file is open
        read line2 from booklist_file
        while line2 is not empty
            found = false
            books_file = open file "books.txt" in read mode
            if books_file is open
                read line1 from books_file
                while line1 is not empty
                    if line2 is a substring of line1
                        found = true
                        break
                    read next line1 from books_file
                close books_file
            if found is false
                add line2, comma and "Available" to a new line in updated_books
            read next line2 from booklist_file
        close booklist_file

    books_file_out = open file "books.txt" in write mode
    if books_file_out is open
        write updated_books to books_file_out
        close books_file_out
    else
        print ("Unable to open file 'books.txt'")
end function
```

```
FUNCTION delete_books()
    books_file = open file "books.txt" in read mode
    booklist_file = open file "BookList.txt" in read mode
    string = line1, line 2 and updated books
    found = false

    if books_file is open
        read line1 from books_file
        while line1 is not empty
            found = false
            booklist_file = open file "BookList.txt" in read mode
            if booklist_file is open
                read line2 from booklist_file
                while line2 is not empty
                    if line2 is a substring of line1
                        found = true
                        break
                    read next line2 from booklist_file
                close booklist_file
            if found is true
                add line1 and new line character to updated_books
            read next line1 from books_file
        close books_file

    books_file_out = open file "books.txt" in write mode
    if books_file_out is open
        write updated_books to books_file_out
        close books_file_out
    else
        print ("Unable to open file 'books.txt'")
end function
```

```
FUNCTION update_book_status(book_name, new_status):
    OPEN_FILE "books.txt" for reading as file
    DECLARE updated_books as string
    DECLARE line as string
    DECLARE found as boolean, set to false
    DECLARE already_in_desired_status as boolean, set to false

    IF file is open:
        WHILE there are lines to read from file:
            READ line from file
            FIND book_name in line, store in pos
            IF book_name is found:
                SET found to true
                FIND new_status in line, store in status_pos
                IF new_status is found:
                    SET already_in_desired_status to true
                ELSE:
                    DETERMINE old_status based on new_status
                    FIND old_status in line, store in status_pos
                    REPLACE old_status with new_status in line
            APPEND updated line to updated_books
        ENDWHILE
        CLOSE file
    ELSE:
        PRINT ("Unable to open file.")

    IF book is found:
        IF book is not already in desired status:
            OPEN_FILE "books.txt" for writing as file
            IF file is open:
                WRITE updated_books to file
                CLOSE file
                PRINT success message
            ELSE:
```

```
            PRINT ("Unable to open file.")
        ELSE:
            PRINT action rejected message
            RETURN to main menu
    ELSE:
        PRINT ("Book not found.")
END FUNCTION

FUNCTION search_book_by_name():
    DECLARE first_letter as character
    PRINT ("Enter the first letter of the book you want to search: ")
    READ first_letter with input validation

    OPEN_FILE "books.txt" for reading as file
    DECLARE line as string
    DECLARE found as boolean, set to false

    IF file is open:
        WHILE there are lines to read from file:
            READ line from file
            IF first letter of line matches user input:
                EXTRACT book_title, book_author, and status from line
                PRINT book information
                SET found to true
        ENDWHILE
        CLOSE file
    ELSE:
        PRINT ("Unable to open file.")

    IF no book is found:
        PRINT ("No books found with the given first letter.")
    CALL return_menu()
END FUNCTION
```

```
void payment()
{
   declare pin[7] as char;
   declare payment_way, online_way as char;
   declare loop_payment equal 1 as char;
   declare loop_credit equal 1 as char;
   declare loop_online equal 1 as char;
   declare loop_tng equal 1 as char;
   declare loop_promo equal false as bool;
   declare card_num[17] as char;
   declare expiry_date, cvv as char;
   declare card_name, online_display, username_online, password_online as string;
   declare tng as string;
   declare real_promo[3] as string;
   declare check_promo as string;
   declare no_promo equal 1 as int;

   while (loop_payment equal 1)
   {
      print("Please select your payment method!");
      print("****************************************");
      print("1. Credit Card / Debit Card ");
      print("2. Online Payment ");
      print("3. Touch N Go Online Pay ");
      print("4. Cash Service ");
      print("5. Promo Code  ");
      input(payment_way);

      if (payment_way equal 1)
      {
         while (loop_credit equal  1) // loop for credit
         {
            print("Card Details");
            print("---------------");
            print("Card Number (16 digit) : ");
            input(card_num);
            input(ignore());
            print("Expiry Date : ");
            input(expiry_date);
            input(ignore());
            print("CVV : ");
            input(cvv);
            input(ignore());
            print("Name On Card : ");
            input(card_name); // available for space & blank

            if (number of digit of(card_num) equal 16 and isdigit(expiry_date) and isdigit(cvv)) // only
digit is available in card_num,expiry_date,cvv
            {
               back to the loop_credit; // exit the loop
            }
            else // if non-digit is filled go in this statement
            {
```

```
            print("Somethings wrong found in Card Details");
            print("Please enter the Card details again!");
            back to the loop_credit; // back to loop
        }


    }
    print("You had succesfully paid the payment. Thank you !");
    system(PAUSE);

    exit the loop_payment; // exit payment loop
}

else if (payment_way equal 2) //Online Payment
{
    system(CLEAR SCREEN);
    while (loop_online equal 1) //Loop for online payment
    {
        print("Please select bank : ");
        print("1. Public Bank ");
        print("2. Maybank ");
        print("3. CIMB Bank ");
        print("4. Am Bank ");
        input(online_way);

        if (online_way equal 1) //Public bank
        {
            online_display equal("Public Bank");
            back to the loop_online;
        }

        else if (online_way equal 2)// Maybank
        {
            online_display equal("May Bank");
            back to the loop_online;

        }
        else if (online_way equal 3) // Cimb Bank
        {
            online_display equal("Cimb Bank");
            back to the loop_online;

        }
        else if (online_way equal 4) //  Am bank
        {
            online_display equal("Am Bank");
            back to the loop_online;

        }

        else
        {
            print("Error! Please answer the question with number.");
            exit the loop_online;
        }
```

```
        print("Welcome to "(online_display)" !");
        print("-----------------------------------------------");
        input(ignore);
        print("Username:  ");
        input(username_online);
        print("Password:");
        input(password_online);
        input(ignore);

        print("Login Successfully!");
        print("-------------------------------");
        print((username_online) << ", you successfully transfer the money through online !");
        print("Thanks for using " << (online_display) << " as your services!");
        print("The Recipt will printed after the payment is done! ");
        print(".");
        print(".");
        print(".");
        print(".");
        print(".");
        print(".");
        print(".");
        print(".");
        print(".");
        print(".");
        print(".");
        print(".");
        print(".");
        print("Click enter to proceed the progress.....");
        system(PAUSE);



    }
    exit the loop_payment;
}

else if (payment_way equal 3)//Touch N Go Online Pay
{

    while (loop_tng equal 1)
    {
        print(" Please filled in the phone number: ");
        input(tng);
        print("Succesfully login ! ");
        print("Please enter your 6-digit PIN number");
        input(pin);
        if (number of digit of(pin) equal 6)    //only accept 6- digit of TNG PIN
        {
            print("Thanks for using Touch N Go as ur services !");
            print("You had succesfully transfer the money to the receiver!");
            print("The Recipt will printed after the payment is done! ");
            exit the loop_tng;

        }
```

```
            else
            {
               print("Error ! Re-enter your phone number with correct PIN .");
               back to the loop_tng;
            }

         }


         exit the loop_payment;
      }
      else if (payment_way equal 4)//Cash Service
      {
         print("Please move to the counter for further payment.");
         print("Thank you for using our services.");
         print("The Recipt will be printed out in awhile..... ");
         system((PAUSE);

         exit the loop_payment;
      }
      else if (payment_way equal 5)//  Promo Code
      {
         while (not equal loop_promo)
         {
            print("Please enter the secret promo code for free payment: ");
            input(check_promo);
            ifstream promofile;
            promofile.open("promo.txt");


            for (int i equal 0 while i less than 3 while i++)
            {
               getline(promofile, real_promo[i]);

               if (real_promo[i] equal check_promo) // checking the promocode user type same with the
list or not
               {
                  print("Congratulation ! You had earn a free payment ! ");
                  print("The Recipt will be printed out in awhile..... ");
                  system(PAUSE);
                  loop_promo equal true;
                  break;


               }
            }
            if (not equal loop_promo) // back loop
            {
               print("Please fill in again ! ");
            }

         }
         exit the loop_payment;
```

```
    }

    else
    {
      print("Error !Please enter a valid answer.");
      back to the loop_payment;
    }
  }

}


void recipt()
{
  system(COLOUR E4); //decoration
  system(CLEAR SCREEN);
  print ("
**********************************************************************************
******************** ");
  print("  _____                              _      ______         _        _   ");
  print(" |  __  |                            | |    |  __  |        (_)       | |  ");
  print(" | |__) |  __ _  _   _   _ __ ___    ___  _ __  | |_   | |__) |  ___   ___  _  _ __  | |_ ");
  print(" |  ___/   / _` | | | | | | '_ ` _ |  / _ | | '_ |  | __|  | _  /  / _ |  / __| | | | '_ |  | __|");
  print(" | |     | (_| | | |_| | | | | | | | |  __/ | | | | | |_    | | | |  __/ | (__  | | | |_) | | |_ ");
  print(" |_|      |__,_| |__,| |_| |_| |_| |___| |_| |_||_|  __|    |_| |_|  |___|  |__| |_| |.__  |__|");
  print("              __/ |                                          | |       ");
  print("             |___/                                           |_|       ");
  print("
**********************************************************************************
******************** ");
  print(" **                                                  **");
  print(" **                                                  **");
  print(" **                                                  **");
  print(" **                                                  **");
  print(" **                                                  **");
  print(" **                                                  **");
  print(" **                                                  **");
  print(" **                                                  **");
  print(" **                    Total Price : $ 1                     **");
  print(" **                                                  **");
  print(" **                                                  **");
  print(" **                                                  **");
  print(" **                                                  **");
  print(" **                                                  **");
  print(" **                                                  **");
  print(" **                                                  **");
  print(" **                                                  **");
  print(" **                                                  **");
  print(" **                                                  **");
  print(" **                                                  **");
  print(" **                                                  **");
  print(" **                                                  **");
```

```
   print("
**********************************************************************************
********************* ");




}
void penalty()
{
   declare date_days as int;
   declare diff_days as int;
   declare loop_date = false;
   declare date, current_date as string;
   declare days, month, year as int;
   declare          max_days equal 0 as int;
   declare current_days, current_month, current_year as int;
   declare pena_price as int;

   while (loop_date equal false)
   {
      system(CLEARSCREEN);

print("**********************************************************************************
*****************************************");
      print("  ______                    _   _             ______            _                  ");
      print(" |  __  |                  | | | |           / _____|          | |                 ");
      print(" | |__) |   ___   _ __     __ _ | | | |_    _   _    | (___    _   _    ___   | |_    ___    _ __ ___   ");
      print(" |  ___/   / _ | | '_  | / _` | | | | __| | | | | |  |___ | | | | / __| | __|  / _ | | '_ ` _ | ")
      print(" | |      |  __/ | | | | | (_| | | | | |_  | |_| |  _____) | | |_| | |__ | | |_  |  __/ | | | | | |");
      print(" |_|       |____| |_| |_|  |__,_| |_|  |__|  |__, |  |______/  | __, | |___/ |__|   |____| |_| |_| |_|");
      print("                                              __/ |            __/ |                    ");
      print("                                             |___/            |___/                      ");

print("**********************************************************************************
*****************************************");

      print("Please enter the date of you lended (DD/MM/YYYY) : ");
      input(date);
      print(date);
      print("Please enter the current date (DD/MM/YYYY) :");
      input(current_date);


      if ((date.length() equal 10 and date[2] equal '/' and date[5] equal '/') and (current_date.length()
equal 10 and current_date[2] equal '/' and current_date[5] equal '/'))
      {
         days equal first digit and second digit of (date);// take the 1st and 2nd word inside the date to
make days
         month equal forth digit and fifth digit of(date);// take the 4th and 5th word inside the date to
make days
         year equal sixth digit till tenth digit of(date);// take the 7th,8th,9th and 10th word inside the
date to make days
```

```
        current_days equal first digit and second digit of(current_date); // take the 1st and 2nd word inside the current_date to make days
        current_month equal forth digit and fifth digit of(current_date);// take the 4th and 5th word inside the current_date to make days
        current_year equal sixth digit till tenth digit of(current_date);// take the 7th,8th,9th and 10th word inside the current_date to make days
        if ((days less equal than 31 and days more equal than 1 and month more equal than 1 and month less equal than 12 and year more than 0) and (current_days less equal than 31 and current_days more equal than 1 and current_month more equal than 1 and current_month less equal than 12 and current_year more than 0))
        {

          if ((month equal 1 or month equal 3 or month equal 5 or month equal 7 or month equal 8 or month equal 10 or month equal 12) and (current_month equal 1 or current_month equal 3 or current_month equal 5 or current_month equal 7 or current_month equal 8 or current_month equal 10 or current_month equal 12))
            {
              max_days equal 31;
              exit the loop_date;

            }

            else if (month equal 2 and current_month equal 2)
            {
              max_days equal 28;
              exit the loop_date;

            }

            else if ((month equal 4 or month equal 6 or month equal 9 or month equal 11) and (current_month equal 4 or current_month equal 6 or current_month equal 9 or current_month equal 11))
            {
              max_days equal 30;
              exit the loop_date;

            }


            exit the loop_date;
            // counting the difference of day with the date that user proviced depending on the month
            date_days equal days + (month - 1) * 31 + (year - 1) * 12 * 31;
            current_days equal current_days + (current_month - 1) * 31 + (current_year - 1) * 12 * 31;
            diff_days equal current_days - date_days;

        }
        else
        {
          print("Please give a valid date!");
           system(PAUSE);
           back to loop_date; // back to loop


        }
```

```
        }
        else
        {
           print("Please enter again the date with right format ! (DD/MM/YYYY)");
           system(PAUSE);
           back to loop_date; // back to loop


        }


    }
    print("Request Accepted !");
    print("The calculation is proceeding to calculate the funds.");
    system(PAUSE);
    print("Difference in days: ")(diff_days);
    if (diff_days more equal than 7)
    {
        diff_days equal diff_days * 1;
    }
    else if (diff_days less eqaul than 0)
    {
        print("Error");

    }
    else if (diff_days more than 0 and diff_days less than 7) // no peanlty if the day is less than 7
    {
        print("Your books have no penalty since the limited day is 7 days.");

    }
    system(PAUSE);
    pena_recipt(diff_days);

}




void pena_recipt(int& diff_days)
{



    system(COLOUR E4); // decoration
    system(CLEARSCREEN);
    print("
**********************************************************************************
******************* ");
    print("  _____                                  _      ______          _         _   ");
    print(" |  __ |                                | |    | __ |          (_)       | |  ");
    print(" | |__) |  __ _  _  _  _ __ ___    ___  _ __  | |_   | |__) | ___    ___  _  _ __   | |_ ");
    print(" |  ___/  / _` | | | | | '_ ` _ |  / _ | | '_ |  | __|  | _  /  / _ |  / __| | | '_ |  | __|");
    print(" | |     | (_| | | |_| | | | | | | | __/ | | | | |_   | | | |  __/ | (__  | | | |_) | | |_ ");
    print(" |_|      |__,_| |__,| |_| |_| |_| |___| |_| |_||_|   __|    |_| |_|  |___|  |__| |_| |.__  |__|");
    print("                 __/ |                                                    | |        ");
    print("                |___/                                                     |_|        ");
```

```
   print("
*******************************************************************************
******************* ");
   print(" **                                                        **");
   print(" **                                                        **");
   print(" **                                                        **");
   print(" **                                                        **");
   print(" **                                                        **");
   print(" **                                                        **");
   print(" **                                                        **");
   print(" **                                                        **");
   print(" **               Total Price : $ "(diff_days)"                        **");
   print(" **                                                        **");
   print(" **                                                        **");
   print(" **                                                        **");
   print(" **                                                        **");
   print(" **                                                        **");
   print(" **                                                        **");
   print(" **                                                        **");
   print(" **                                                        **");
   print(" **                                                        **");
   print(" **                                                        **");
   print(" **                                                        **");
   print(" **                                                        **");
   print(" **                                                        **");
   print(" **                                                        **");
   print("
*******************************************************************************
******************* ");


   system(PAUSE);
}


void sensor()
{

   declare seat[24] equal { 0 } as int; // Array to keep track of the occupancy status of parking spaces,
0 = empty, 1 = occupied
   declare availableseat equal 24 as int; // The total number of available parking spaces
   declare chosenseat as char; // The parking space chosen by the user
   declare loop_seat equal 0 as int;
   // Main loop for the parking guidance system
   while (true)
   {
      system(CLEARSCREEN);
      // Display the current status of the parking spaces

print("*****************************************************************************
******************* ");
      print("   _____                          _____           _                ");
      print("  / ____|                        / ____|         | |               ");
```

```
        print(" | (____    ___  _ __   ___   ___   _ __    | (____   _   _  ____  | |_   ___  _ __ ___  ");
        print("  |___ |   / _ | | '_ | / __| / _ | | '__|    |___ | | | | | / __| | __| / _ | | '_ ` _ | ");
        print("  _____) | | __/ | | | | |__ | | (_) | | |       _____) | | |_| | |__ | | |_  | __/ | | | | || |");
        print(" |______/   |___| |_| |_| |___/  |___/  |_|      |______/  | __, | |___/ |__|  |___|  |_| |_||_|");
        print("                                                          __/ |                          ");
        print("                                                         |___/                           ");

print("**************************************************************************************** ");




        print("Current seat status: \n\n");

        print(" ==================================");
        print(" 0 = empty, 1 = occupied");
        print(" ==================================");

        print("
=============================================================================
============");
        print(" |               |               |               |               |");
        print(" |               |               |               |               |");
        print(" |       A       |       B       |       C       |       D       |");
        print(" |       "(seat[0])"       |       "(seat[1])"       |       "(seat[2])"       |       "(seat[3]) "
| ");
        print(" |               |               |               |               |");
        print(" |               |               |               |               |");
        print("
=============================================================================
============");
        print(" |               |               |               |               |");
        print(" |               |               |               |               |");
        print(" |       E       |       F       |       G       |       H       |");
        print(" |       "(seat[4])"       |       "(seat[5])"       |       "(seat[6]) "       |       "(seat[7])"
| ");
        print(" |               |               |               |               |");
        print(" |               |               |               |               |");
        print("
=============================================================================
============");




        print("
=============================================================================
============");
        print(" |               |               |               |               |");
        print(" |               |               |               |               |");
        print(" |       I       |       J       |       K       |       L       |");
        print(" |       "(seat[8])"       |       "(seat[9)"       |       "(seat[10])"       |       "(seat[11])"
| ");
        print(" |               |               |               |               |");
```

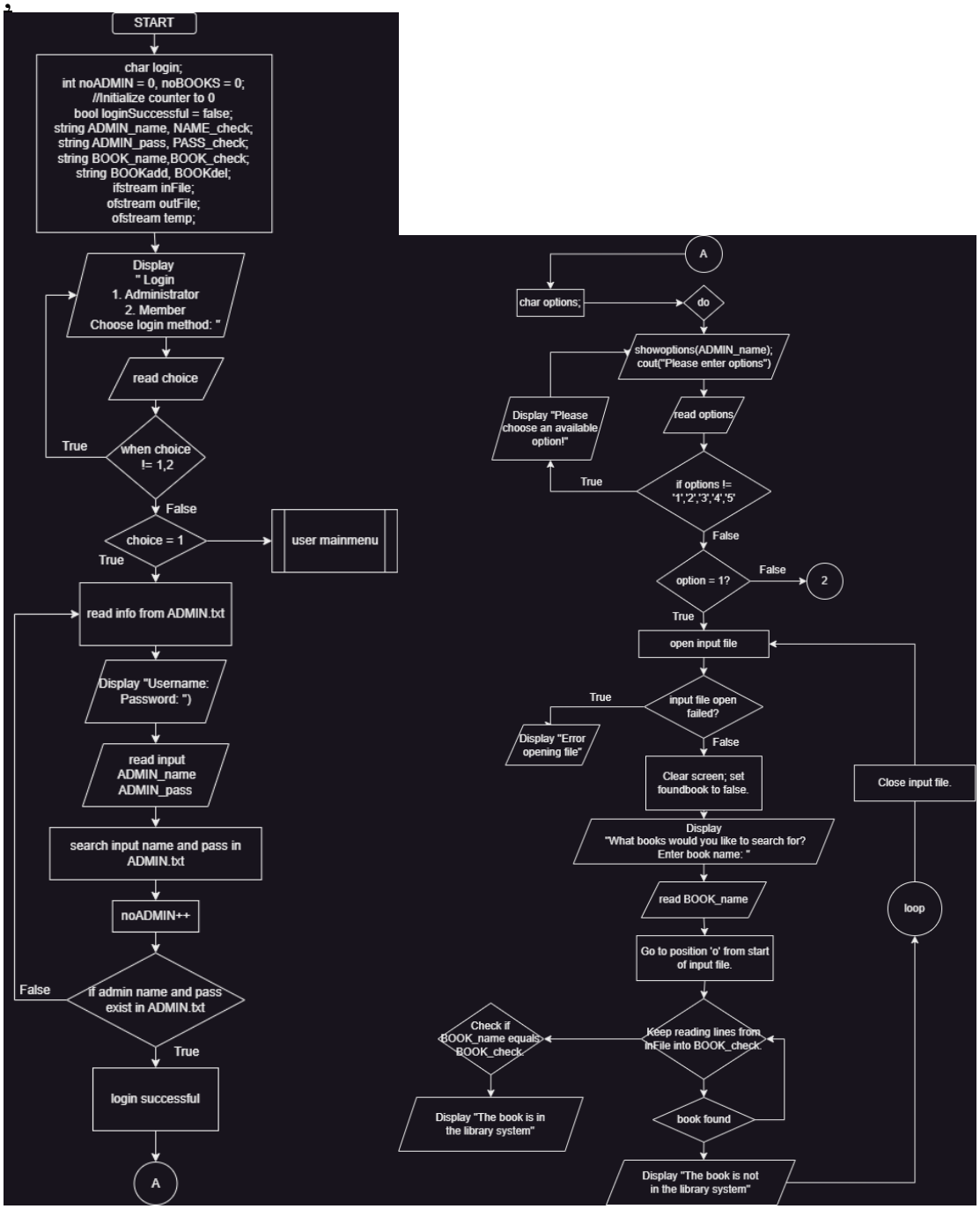
```
        print(" |                 |                 |                 |                 |");
        print("
==========================================================================
============");
        print(" |                 |                 |                 |                 |");
        print(" |                 |                 |                 |                 |");
        print(" |        M        |        N        |        O        |        P        |");
        print(" |        "(seat[12]) "        |        "(seat[13])"        |        "(seat[14])"        |
"(seat[15])"        | ");
        print(" |                 |                 |                 |                 |");
        print(" |                 |                 |                 |                 |");
        print("
==========================================================================
============");


        print("
==========================================================================
============");
        print(" |                 |                 |                 |                 |");
        print(" |                 |                 |                 |                 |");
        print(" |        Q        |        R        |        S        |        T        |");
        print(" |        "(seat[16]) "        |        "(seat[17])"        |        "(seat[18]) "        |
"(seat[19]) "        | ");
        print(" |                 |                 |                 |                 |");
        print(" |                 |                 |                 |                 |");
        print("
==========================================================================
============");
        print(" |                 |                 |                 |                 |");
        print(" |                 |                 |                 |                 |");
        print(" |        U        |        V        |        W        |        X        |");
        print(" |        "(seat[20]) "        |        "(seat[21]) "        |        "(seat[22]) "        |
"(seat[23]) "        | ");
        print(" |                 |                 |                 |                 |");
        print(" |                 |                 |                 |                 |");
        print("
==========================================================================
============");



        // Prompt the user to choose a seat space
        print("Please enter the number of the seat space you would like to sit (A-X)(Z to exit): ");

       input(chosenseat);
        while (loop_seat equal 0)
        {
           if (isdigit(chosenseat))
           {
              print("Please enter (A-X) !");
              system(PAUSE);
              back to loop_seat;
              break;
```

```
        }
        else if (chosenseat equal 'Z' or chosenseat equal 'z')
        {
           member();
           break;


        }
        else
        {
           chosenseat equal toupper(chosenseat);
           declare spaceindex as int  equal chosenseat - 'A';

           // Check if the chosen space is already occupied
           if (seat[spaceindex] equal 1)
           {
              print("Sorry, that space is already occupied. Please choose another space.");
              system(PAUSE);
              break;
           }


           else
           {
              // Mark the chosen space as occupied and update the available spaces count
              seat[spaceindex] equal 1;
              availableseat--;

              // Display a message indicating the chosen space and the number of available spaces
              print("You have register in seat ")(spaceindex)(". There are ")(availableseat)(" spaces
available.");
              system(PAUSE);
              break;

              // Check if all parking spaces are now occupied
              if (availableseat equal 0) {
                 print("All seat spaces are now occupied. Please come back later.");
                 system(PAUSE);
                 break;

              }
           }
           exit the loop_seat;
        }

     }
  }


}
```

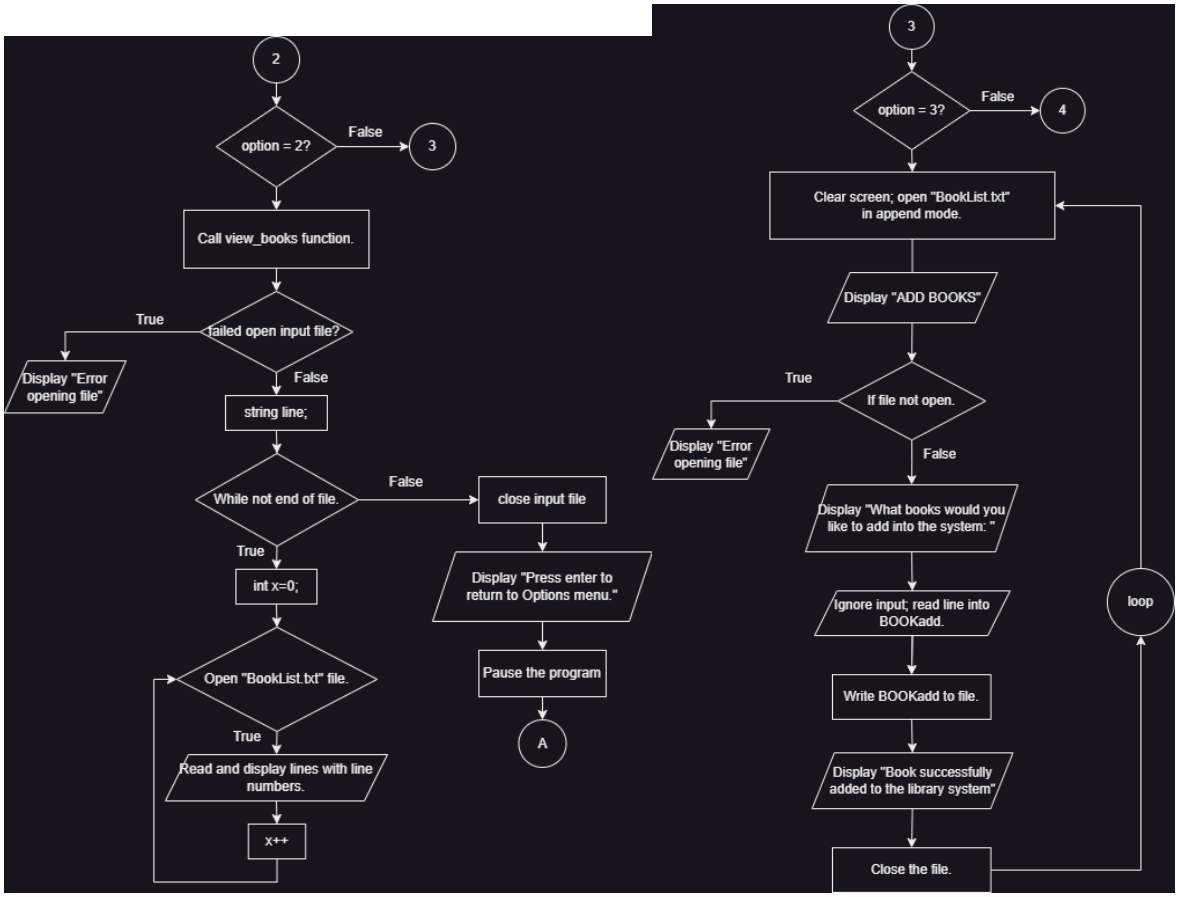
Flowchart

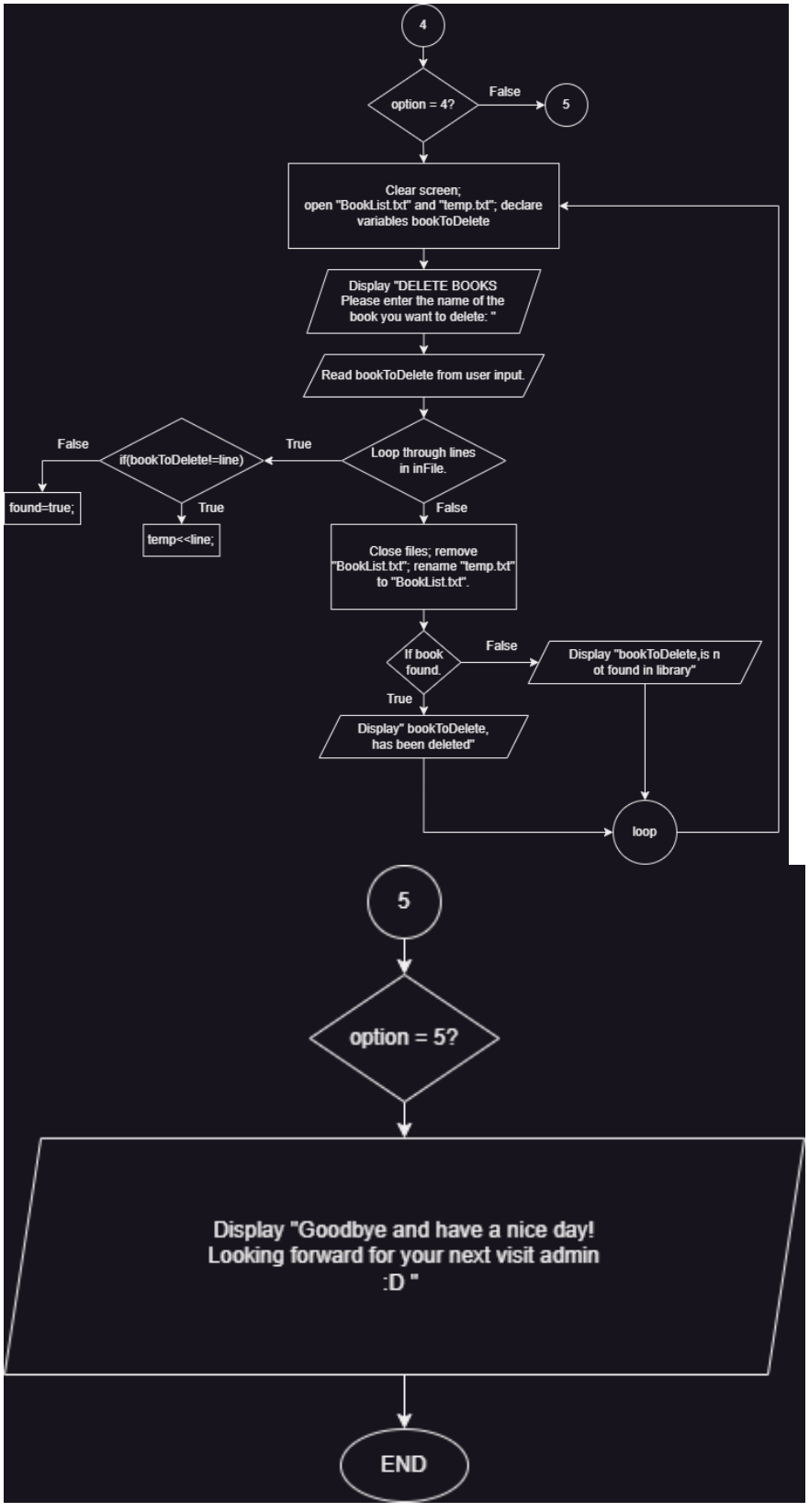
void member()void showOptions(string&);

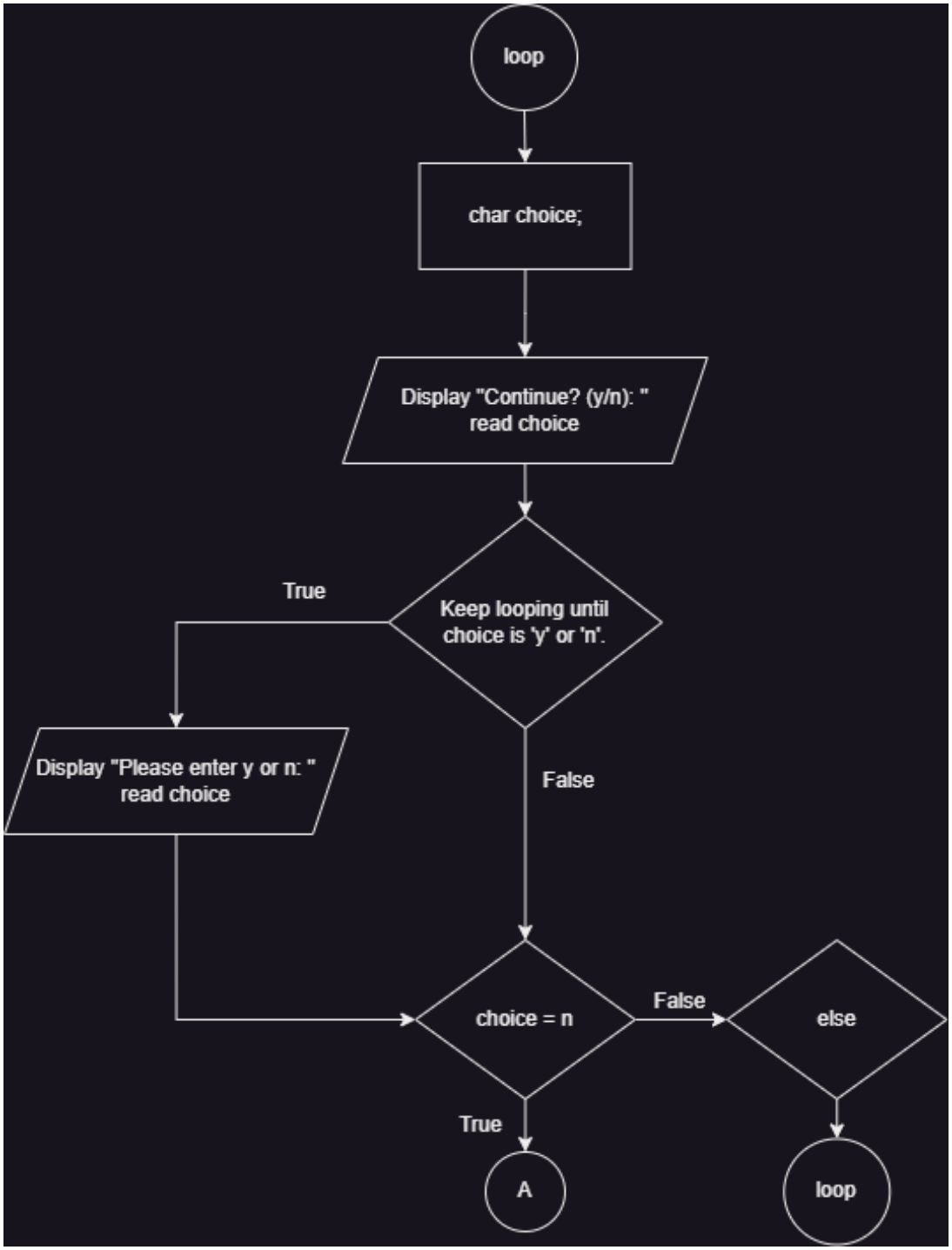
;



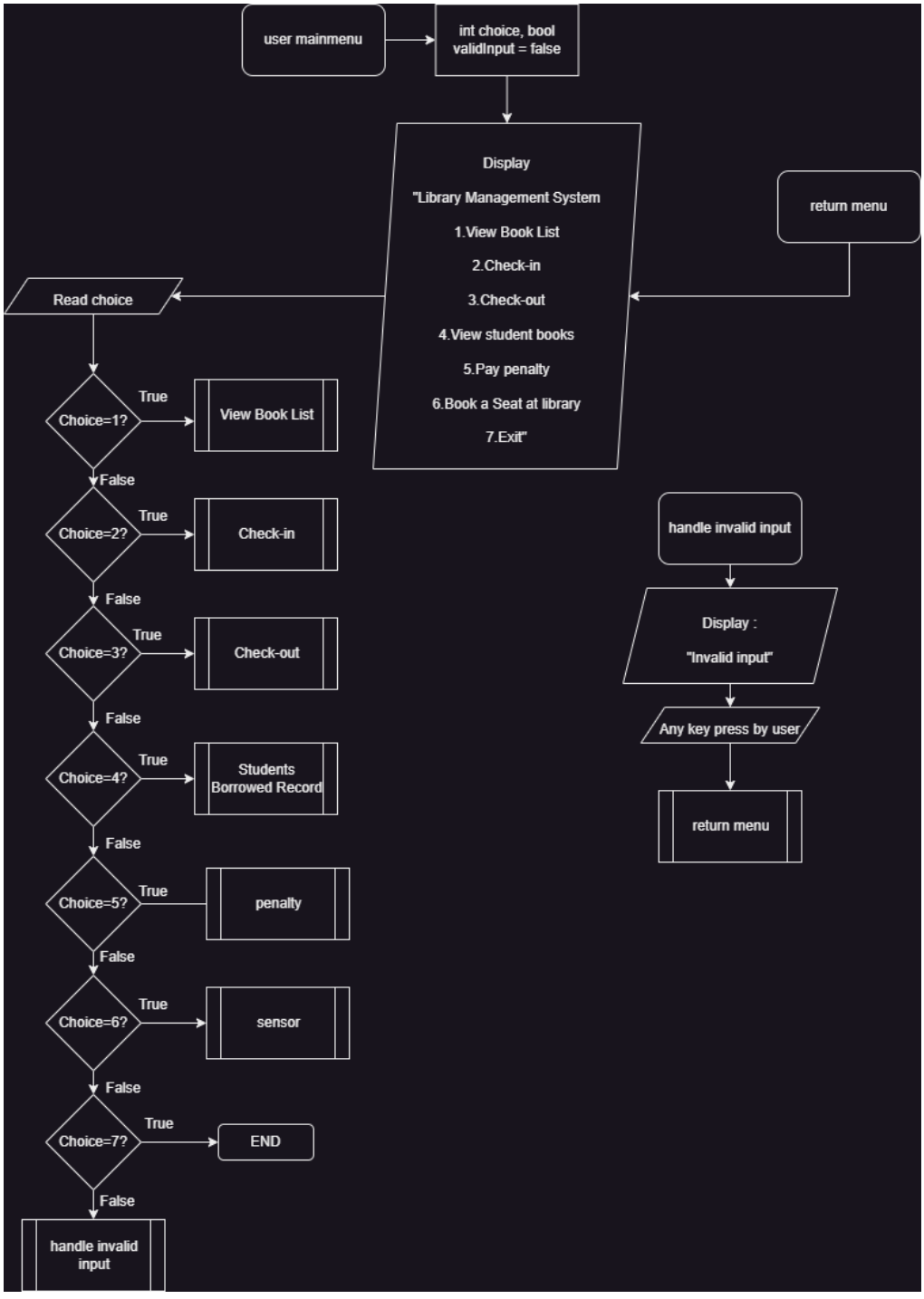
void view ();


2
option = 2?
False
3
Call view_books function.
True
failed open input file?
Display "Error opening file"
False
string line;
While not end of file.
False
close input file
True
int x=0;
Display "Press enter to return to Options menu."
Open "BookList.txt" file.
Pause the program
True
A
Read and display lines with line numbers.
x++
3
option = 3?
False
4
Clear screen; open "BookList.txt" in append mode.
Display "ADD BOOKS"
True
If file not open.
Display "Error opening file"
False
Display "What books would you like to add into the system: "
Ignore input; read line into BOOKadd.
loop
Write BOOKadd to file.
Display "Book successfully added to the library system"
Close the file.

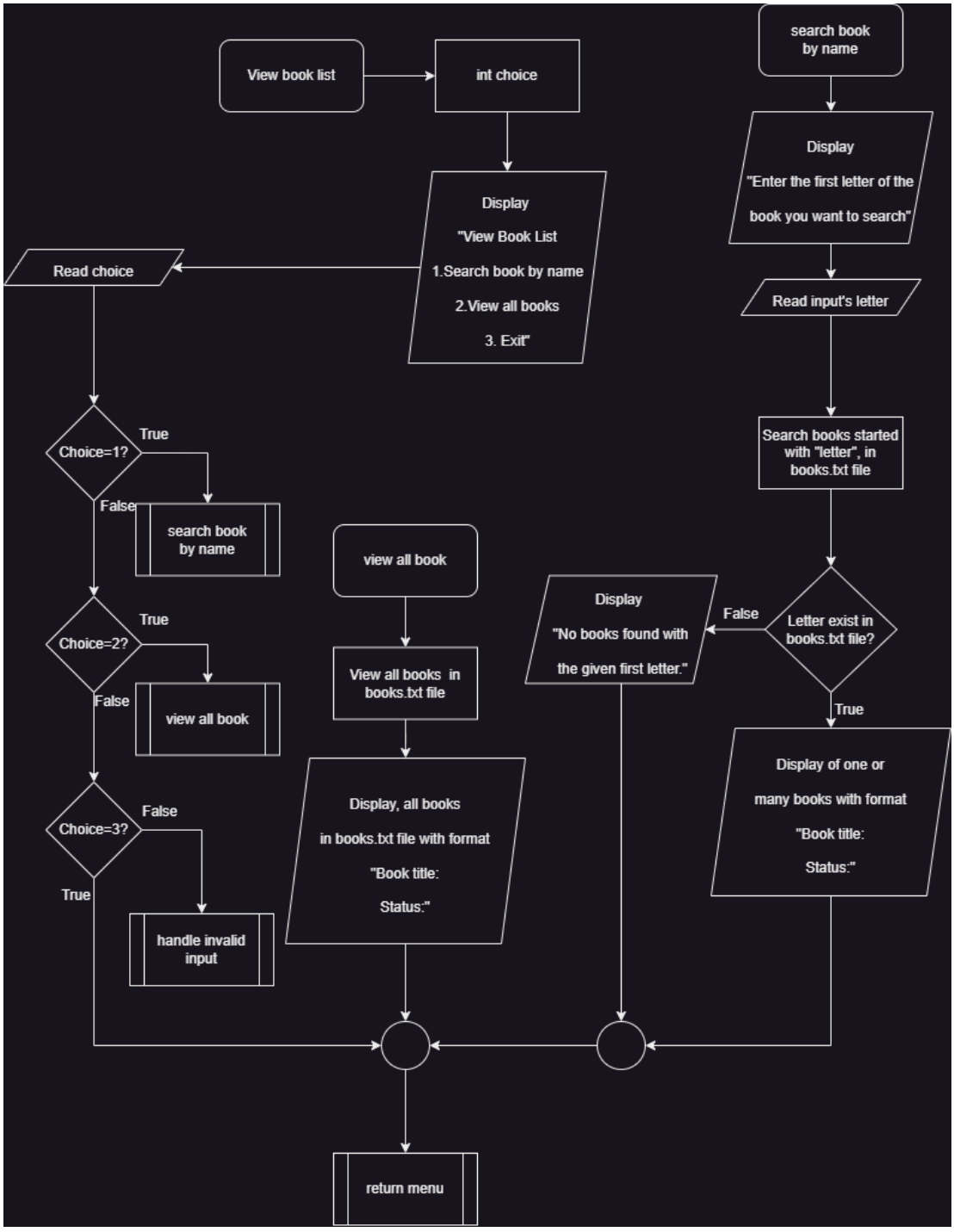

4
option = 4?
False
5
Clear screen;
open "BookList.txt" and "temp.txt"; declare
variables bookToDelete
Display "DELETE BOOKS
Please enter the name of the
book you want to delete: "
Read bookToDelete from user input.
Loop through lines
in inFile.
True
if(bookToDelete!=line)
False
found=true;
True
temp<<line;
False
Close files; remove
"BookList.txt"; rename "temp.txt"
to "BookList.txt".
If book
found.
False
Display "bookToDelete,is n
ot found in library"
True
Display" bookToDelete,
has been deleted"
loop
5
option = 5?
Display "Goodbye and have a nice day!
Looking forward for your next visit admin
:D "
END


loop
char choice;
Display "Continue? (y/n): "
read choice
Keep looping until
choice is 'y' or 'n'.
True
False
Display "Please enter y or n: "
read choice
choice = n
False
else
True
A
loop

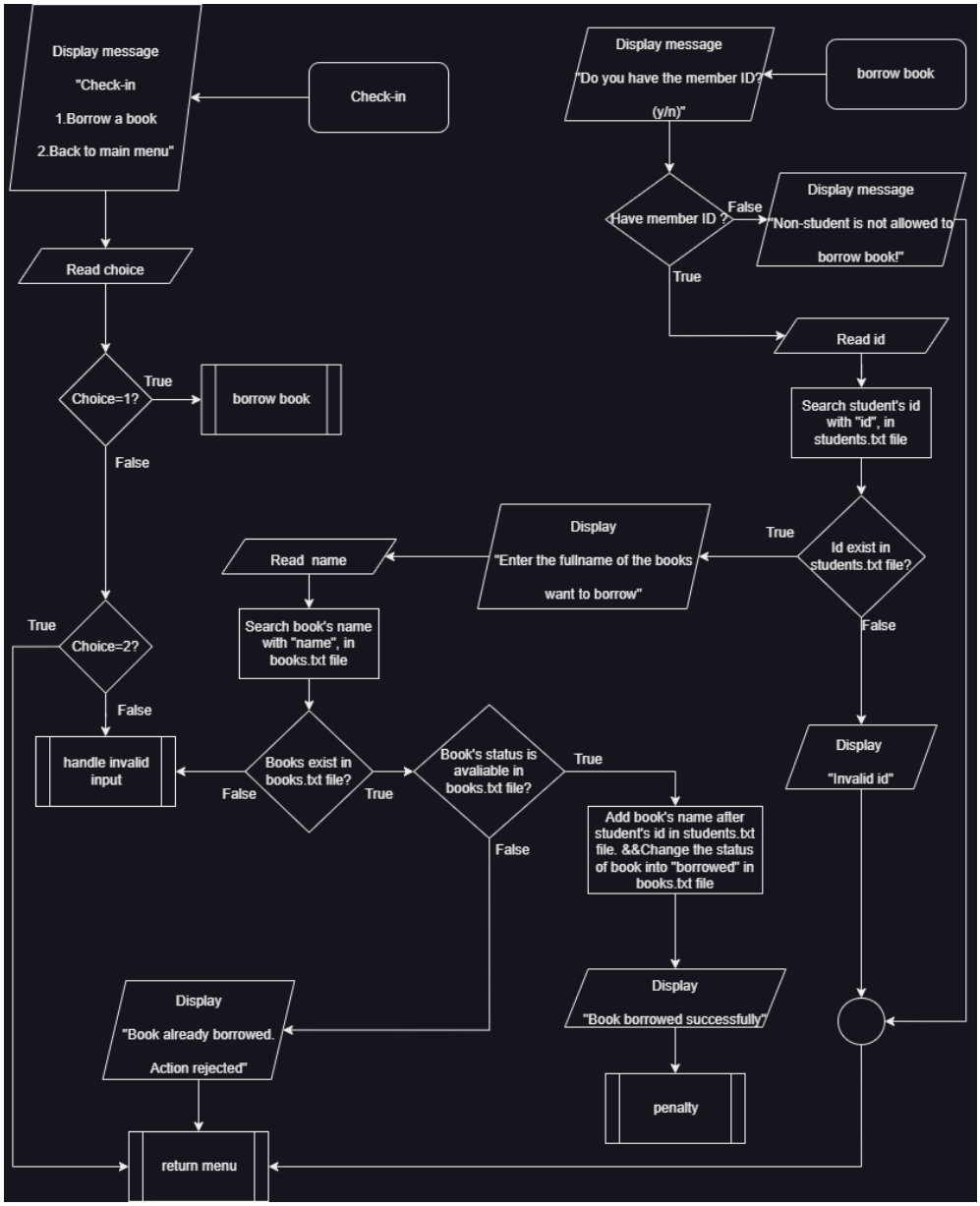
void user_mainmenu(); & void handle_invalid_input();



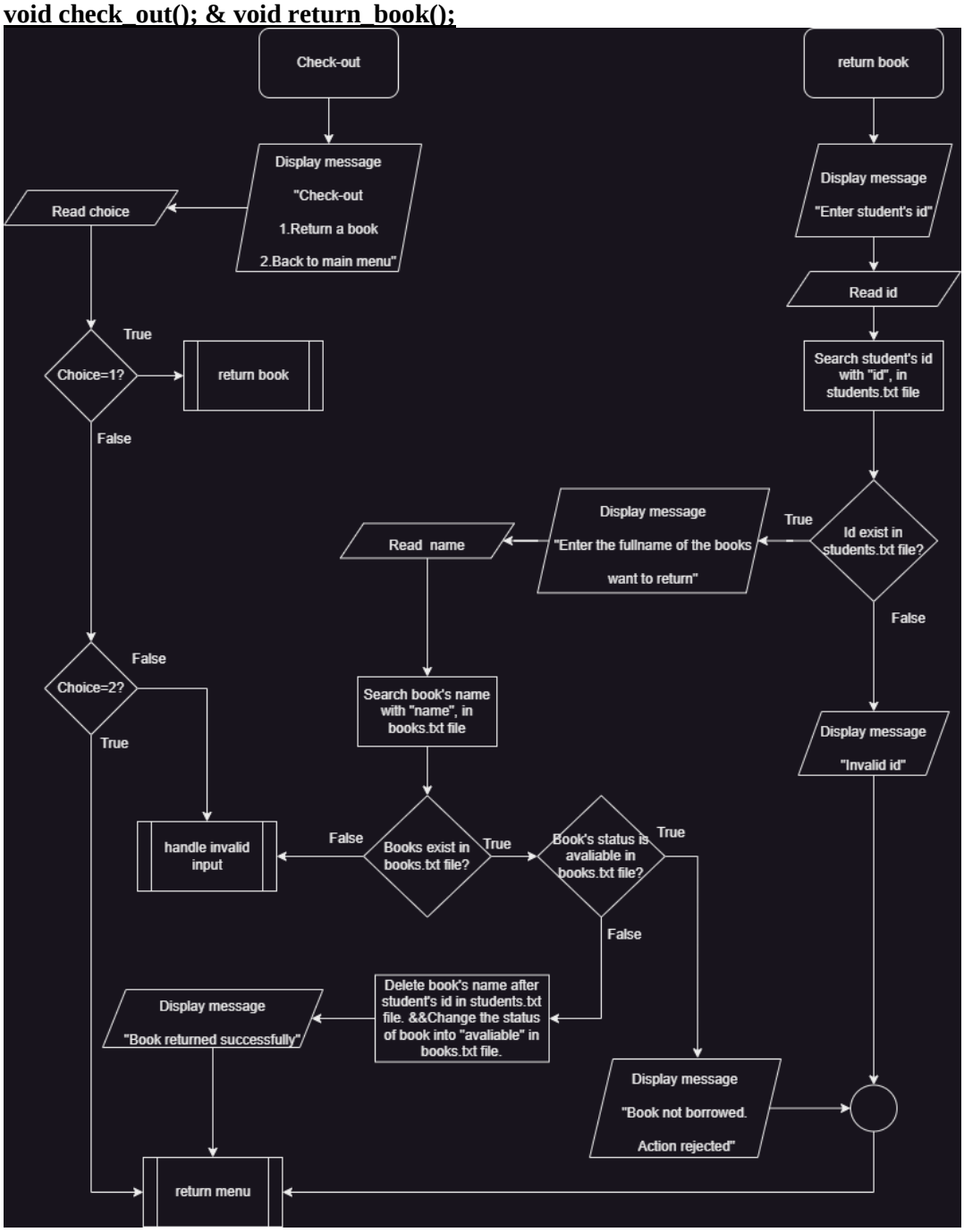
void view_books(); & void view_all_books();&void search_book_by_name();



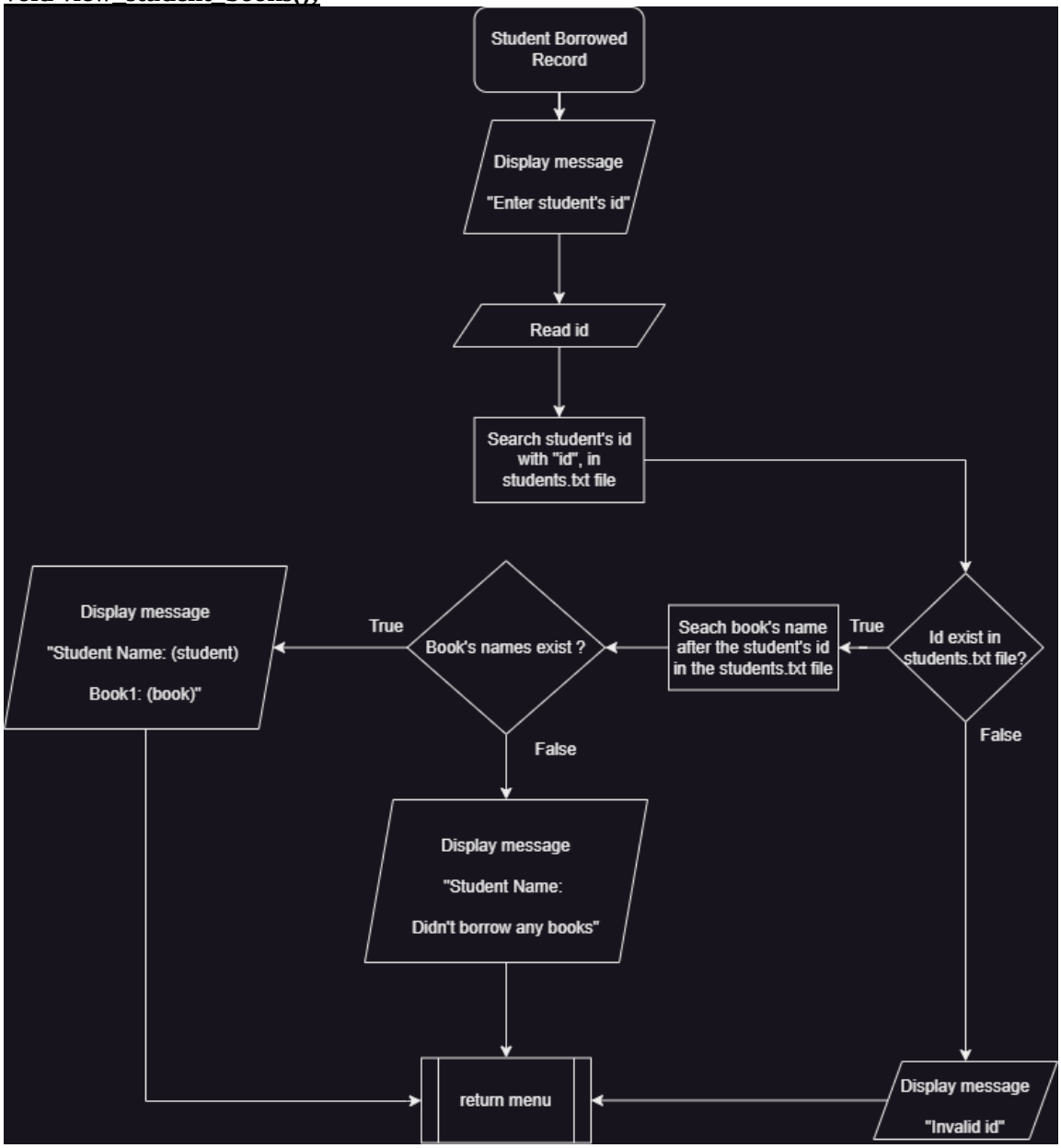
void check_in(); & void borrow_book();
& void update_book_status();



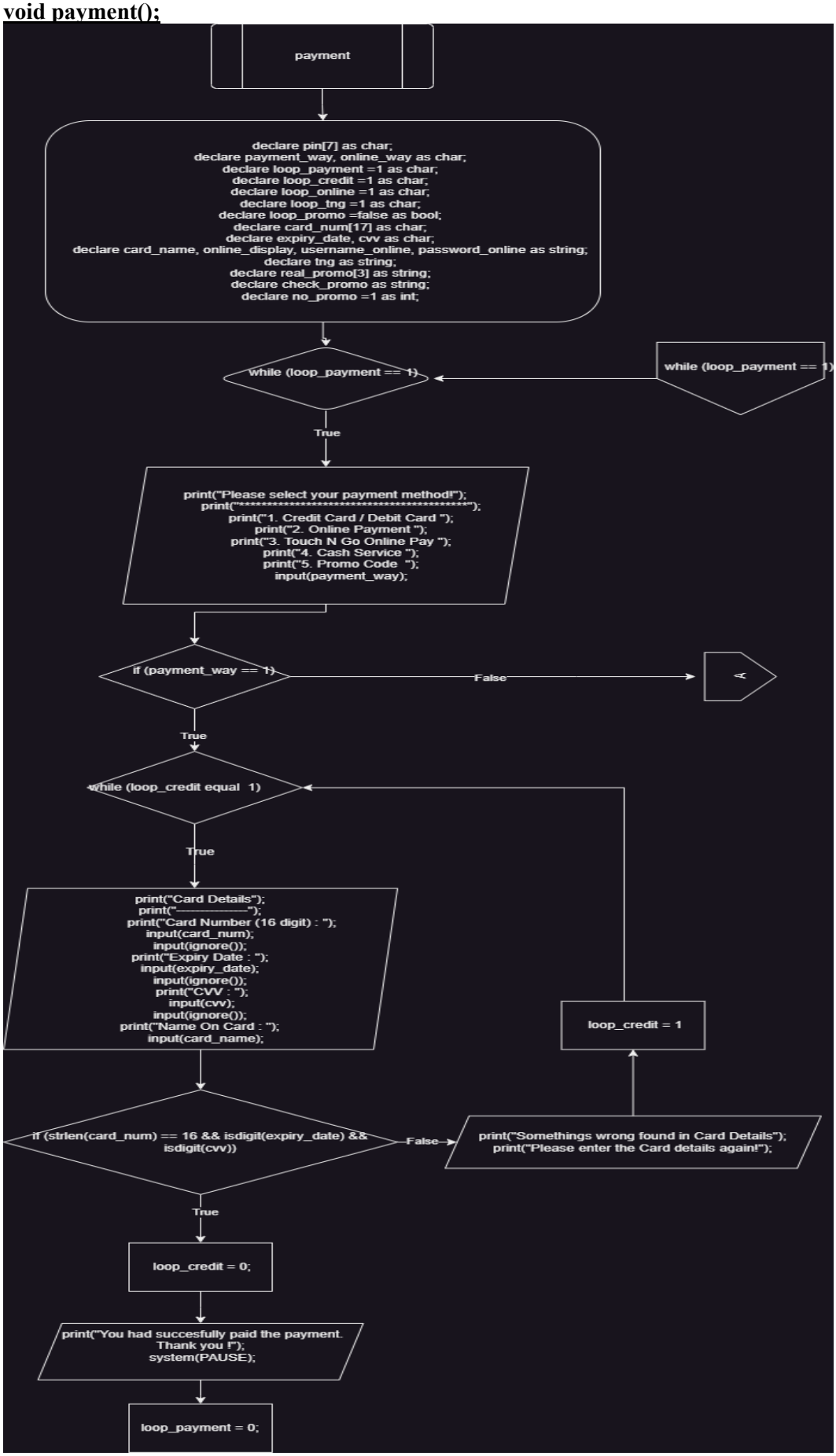
void check_out(); & void return_book();

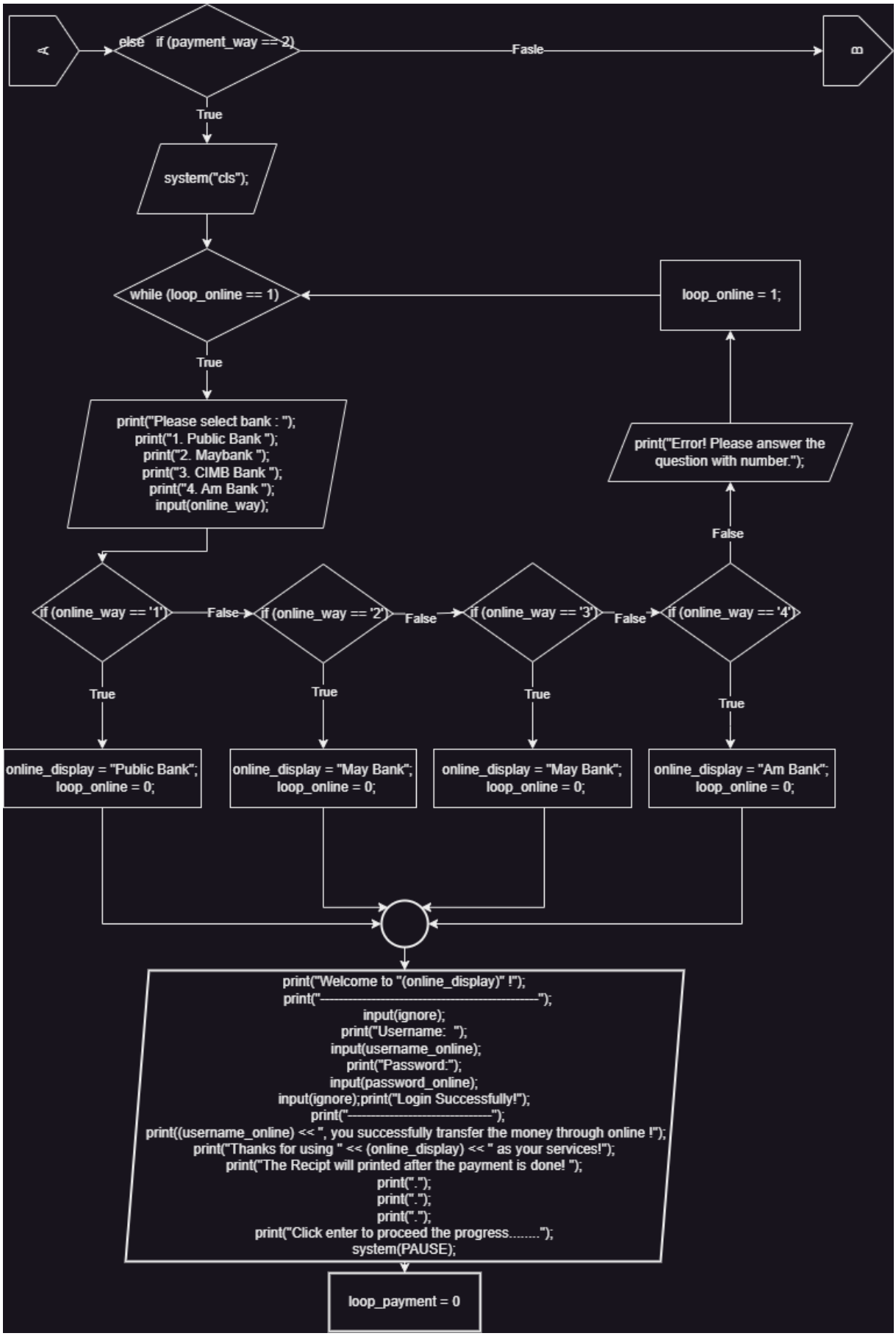


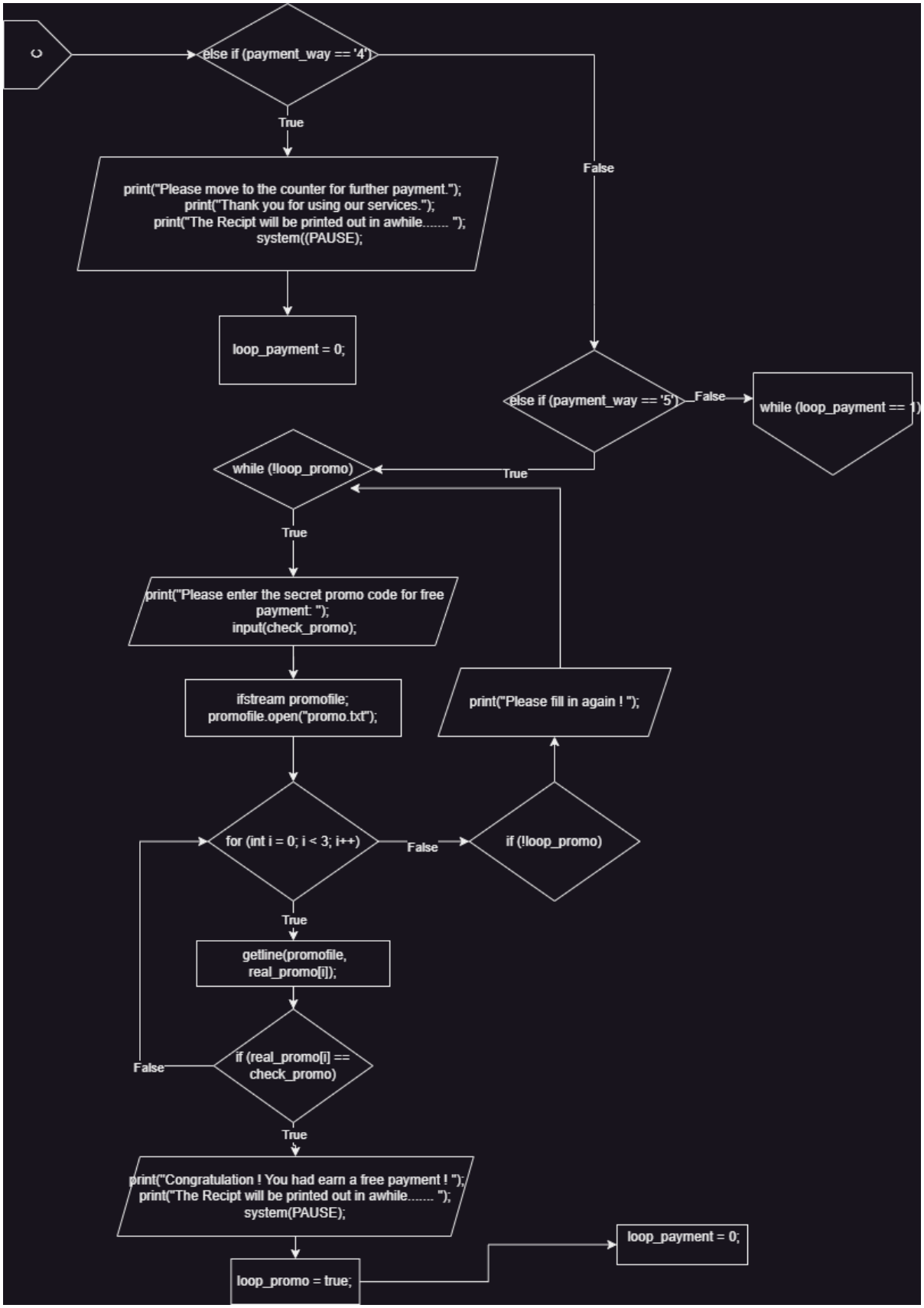
void view_student_books();



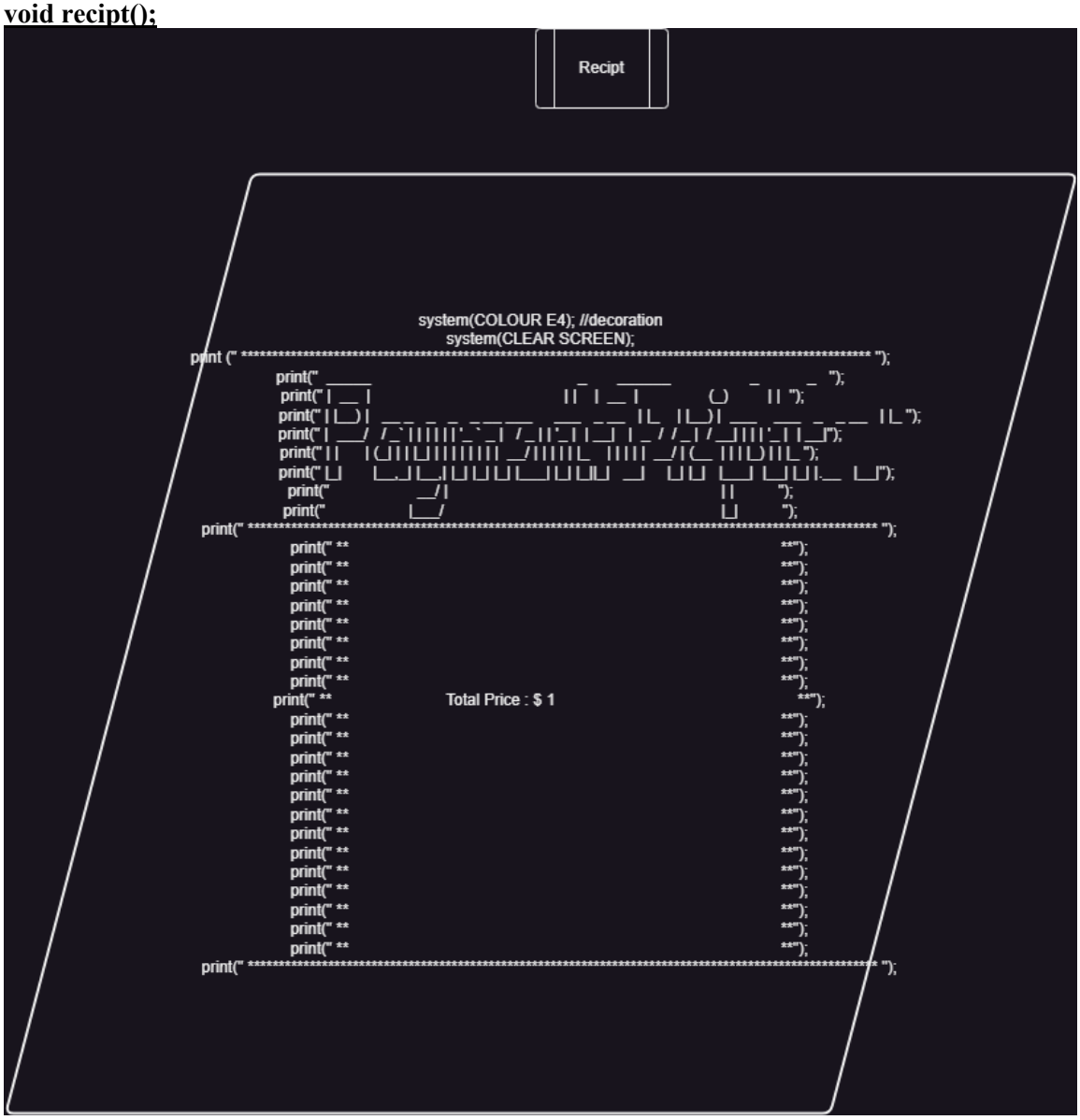
void payment();



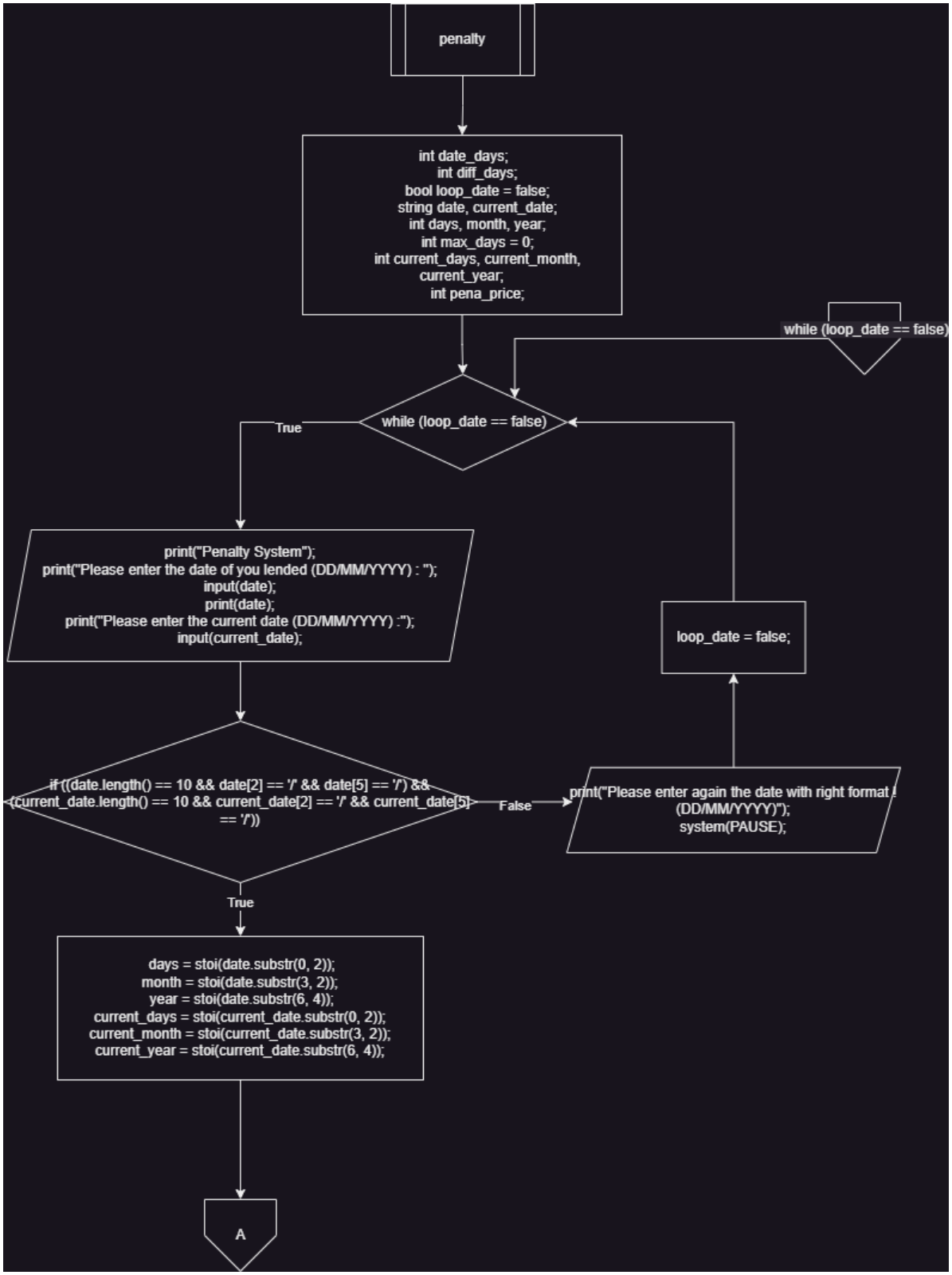

A
else if (payment_way == 2)
Fasle
B
True
system("cls");
while (loop_online == 1)
loop_online = 1;
True
print("Please select bank : ");
print("1. Public Bank ");
print("2. Maybank ");
print("3. CIMB Bank ");
print("4. Am Bank ");
input(online_way);
print("Error! Please answer the question with number.");
False
if (online_way == '1')
False
if (online_way == '2')
False
if (online_way == '3')
False
if (online_way == '4')
True
True
True
True
online_display = "Public Bank";
loop_online = 0;
online_display = "May Bank";
loop_online = 0;
online_display = "May Bank";
loop_online = 0;
online_display = "Am Bank";
loop_online = 0;
print("Welcome to "(online_display)" !");
print("----------");
input(ignore);
print("Username: ");
input(username_online);
print("Password:");
input(password_online);
input(ignore);print("Login Successfully!");
print("----------");
print((username_online) << ", you successfully transfer the money through online !");
print("Thanks for using " << (online_display) << " as your services!");
print("The Recipt will printed after the payment is done! ");
print(".");
print(".");
print(".");
print("Click enter to proceed the progress.....");
system(PAUSE);
loop_payment = 0

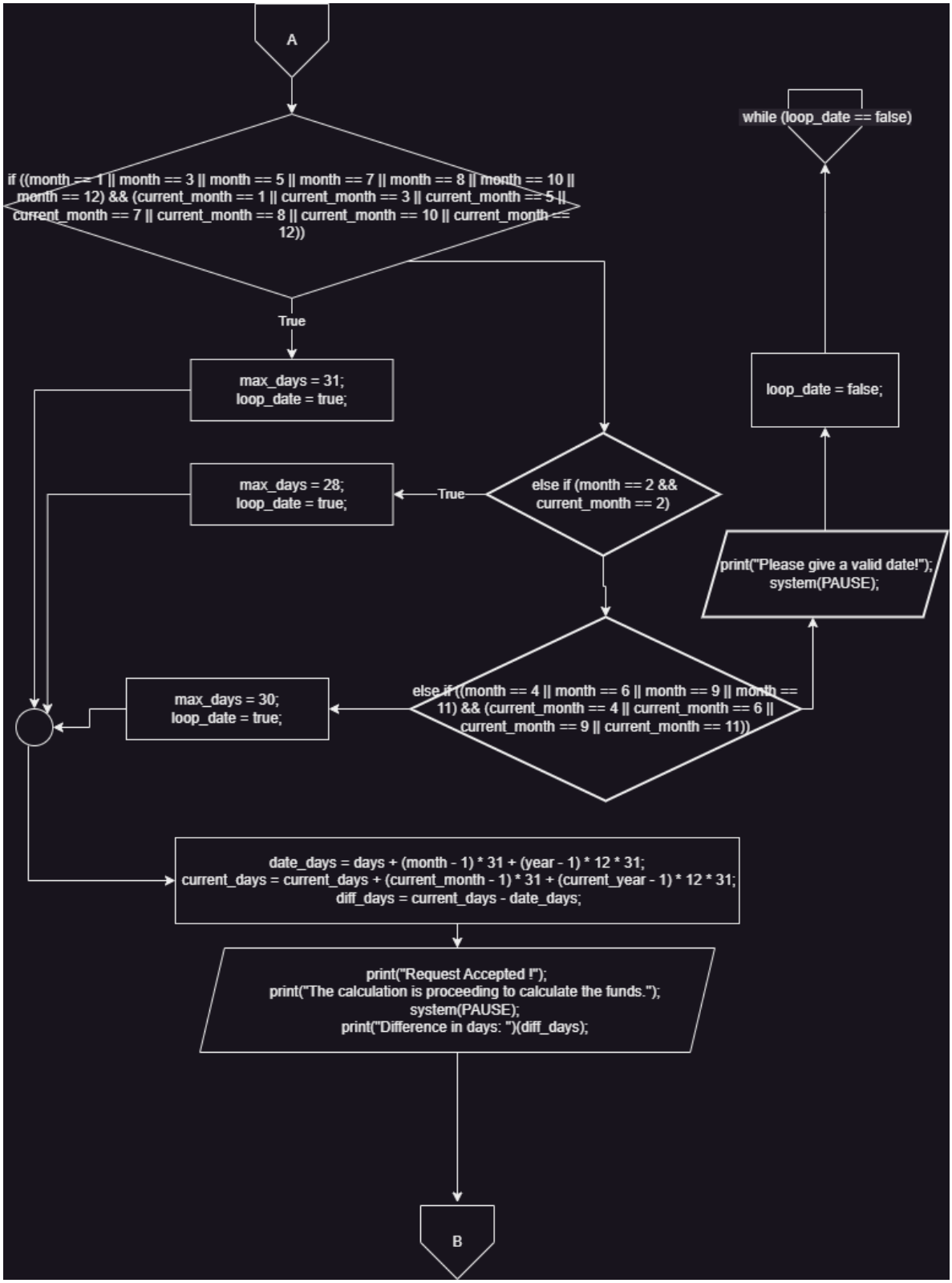

else if (payment_way == '4')
True
False
print("Please move to the counter for further payment.");
print("Thank you for using our services.");
print("The Recipt will be printed out in awhile..... ");
system((PAUSE);
loop_payment = 0;
else if (payment_way == '5')
False
while (loop_payment == 1)
while (!loop_promo)
True
True
print("Please enter the secret promo code for free payment: ");
input(check_promo);
ifstream promofile;
promofile.open("promo.txt");
print("Please fill in again ! ");
for (int i = 0; i < 3; i++)
False
if (!loop_promo)
True
getline(promofile, real_promo[i]);
if (real_promo[i] == check_promo)
False
True
print("Congratulation ! You had earn a free payment ! ");
print("The Recipt will be printed out in awhile..... ");
system(PAUSE);
loop_promo = true;
loop_payment = 0;

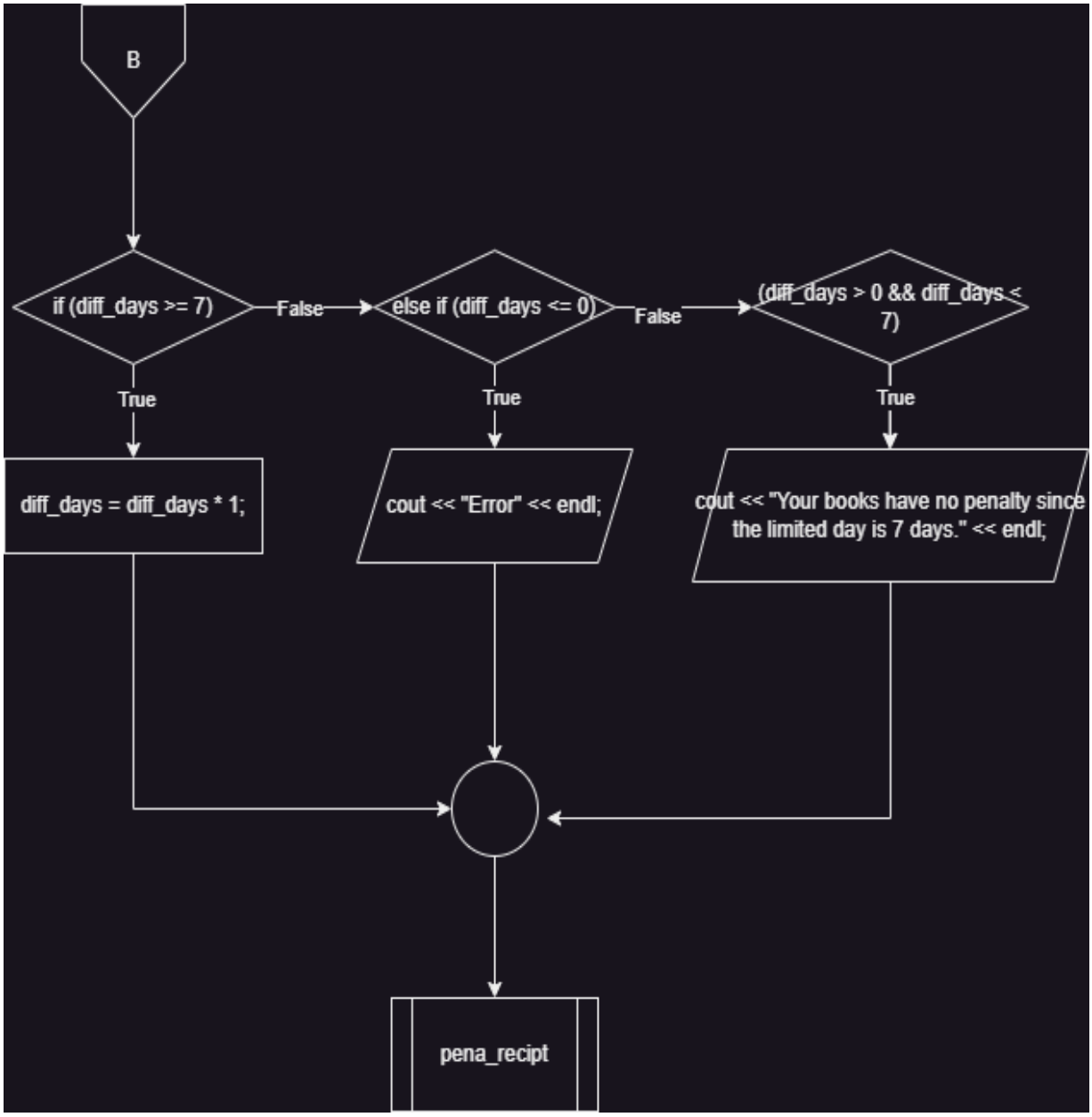
void recipt();


Recipt
system(COLOUR E4); //decoration
system(CLEAR SCREEN);
print (" ********** ");
print(" _____ _ _____ _ _ ");
print("| __ | | | __ | (_) | | ");
print("| |__) | ___ ___ ___ ___ | |_ | |__) | ___ ___ _ _ __ | |_ ");
print("| ___/ / _ \| '_ \ / _ \ / _ \| __| | _ / / _ \ / __| '_ \| __|");
print("| | | (_) | | | | (_) | __/ | | | |_ | | \ \ (_) | (__| |_) | |_ ");
print("|_| ___/|_| |_|___/ ___| ___| __| |_| ____/ ___| .__/ __|");
print(" | | ");
print(" |_| ");
print(" ********** ");
print(" ** **");
print(" ** **");
print(" ** **");
print(" ** **");
print(" ** **");
print(" ** **");
print(" ** **");
print(" ** **");
print(" ** **");
print(" ** Total Price : $ 1 **");
print(" ** **");
print(" ** **");
print(" ** **");
print(" ** **");
print(" ** **");
print(" ** **");
print(" ** **");
print(" ** **");
print(" ** **");
print(" ** **");
print(" ** **");
print(" ** **");
print(" ********** ");

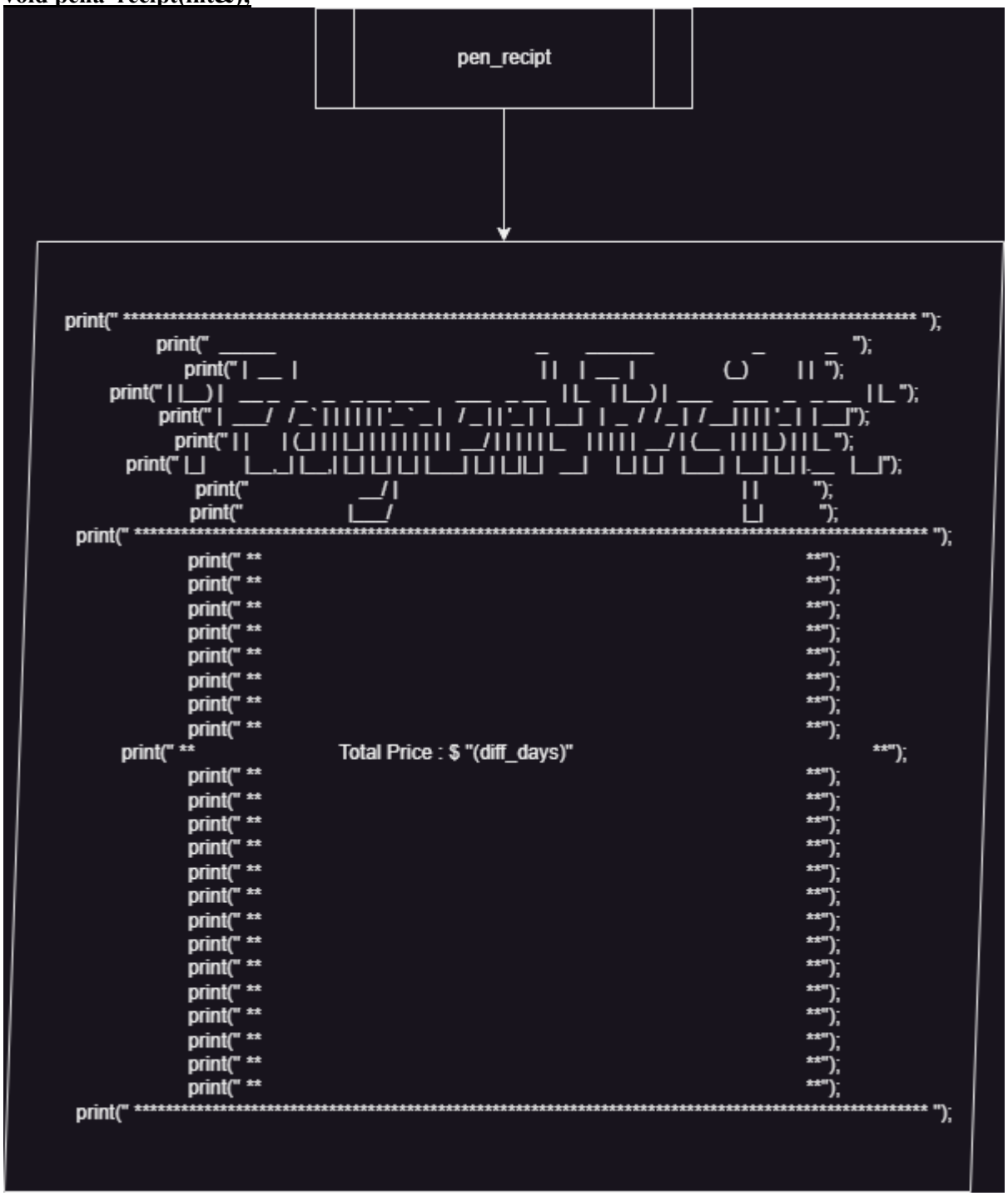
void penalty();



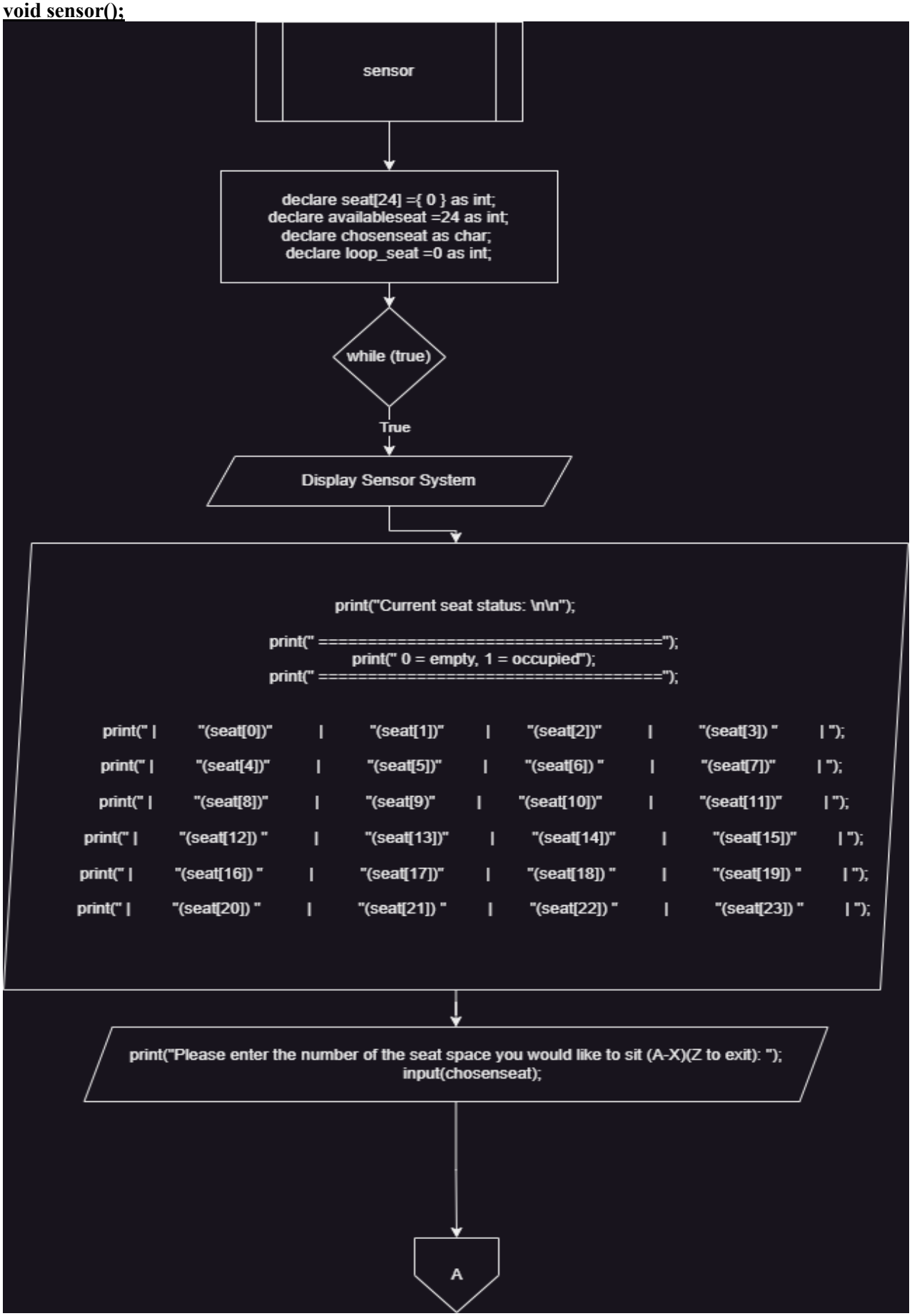

A
while (loop_date == false)
if ((month == 1 || month == 3 || month == 5 || month == 7 || month == 8 || month == 10 || month == 12) && (current_month == 1 || current_month == 3 || current_month == 5 || current_month == 7 || current_month == 8 || current_month == 10 || current_month == 12))
True
max_days = 31;
loop_date = true;
loop_date = false;
max_days = 28;
loop_date = true;
True
else if (month == 2 && current_month == 2)
print("Please give a valid date!");
system(PAUSE);
else if ((month == 4 || month == 6 || month == 9 || month == 11) && (current_month == 4 || current_month == 6 || current_month == 9 || current_month == 11))
max_days = 30;
loop_date = true;
date_days = days + (month - 1) * 31 + (year - 1) * 12 * 31;
current_days = current_days + (current_month - 1) * 31 + (current_year - 1) * 12 * 31;
diff_days = current_days - date_days;
print("Request Accepted !");
print("The calculation is proceeding to calculate the funds.");
system(PAUSE);
print("Difference in days: ")(diff_days);
B

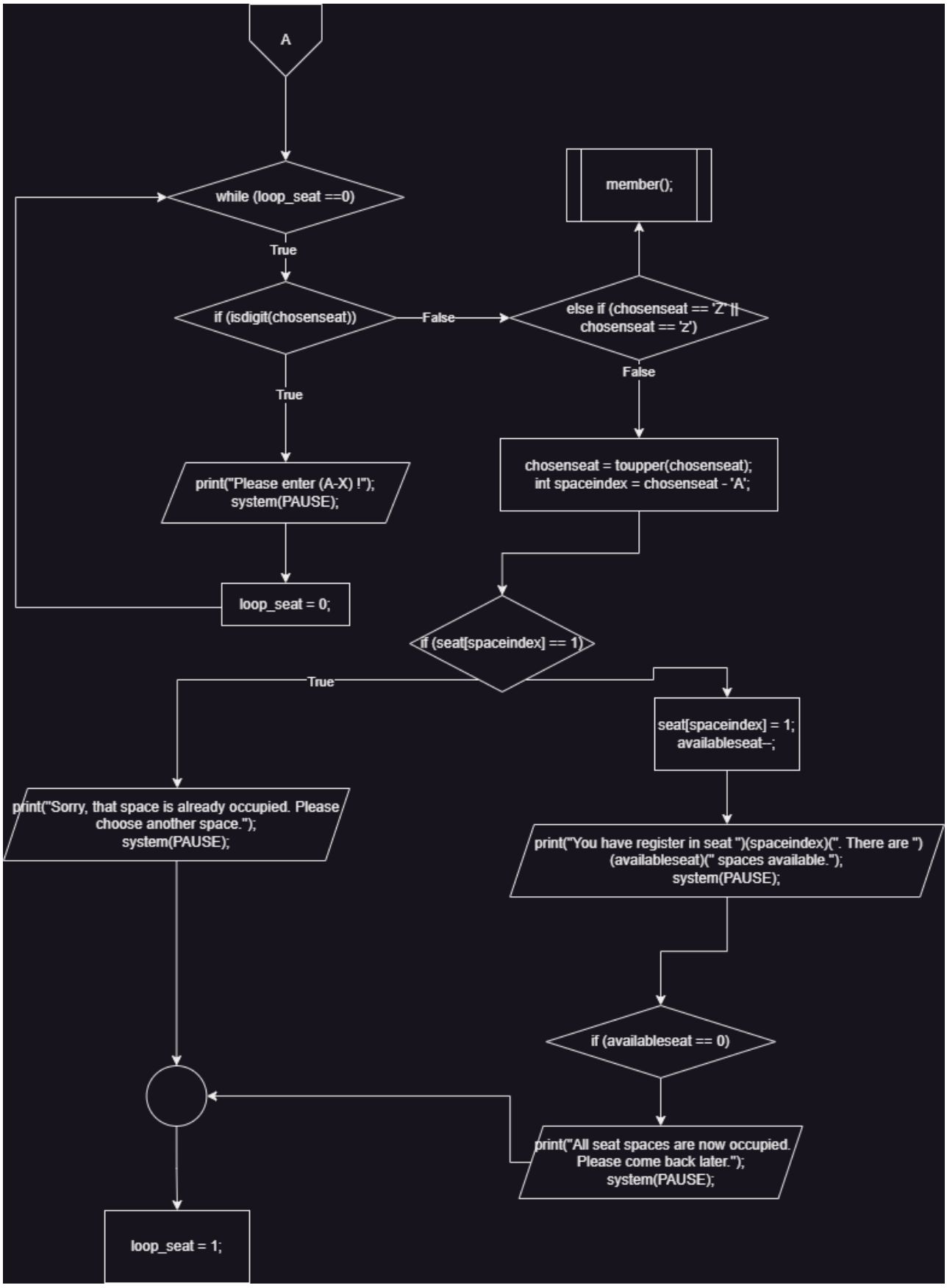

B
if (diff_days >= 7)
False
else if (diff_days <= 0)
False
(diff_days > 0 && diff_days < 7)
True
True
True
diff_days = diff_days * 1;
cout << "Error" << endl;
cout << "Your books have no penalty since the limited day is 7 days." << endl;
pena_recipt

void pena_recipt(int&);

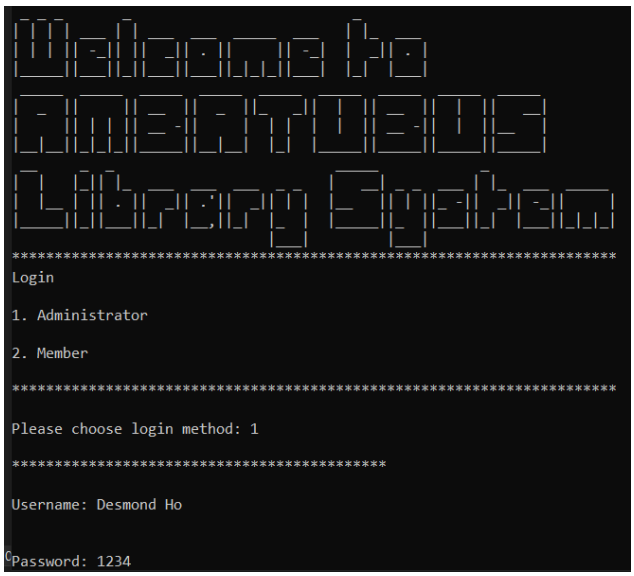


void sensor();

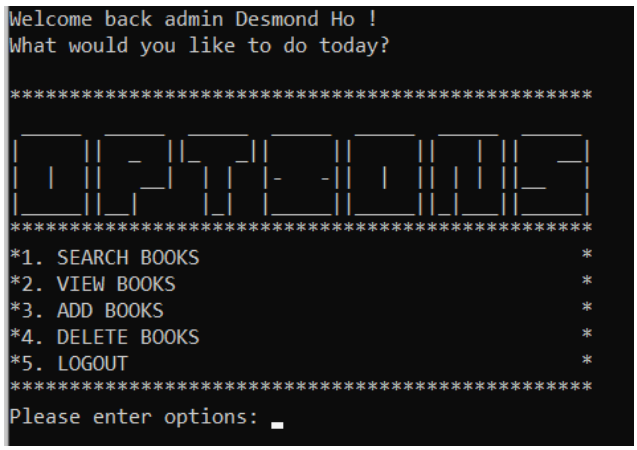

sensor
declare seat[24] ={ 0 } as int;
declare availableseat =24 as int;
declare chosenseat as char;
declare loop_seat =0 as int;
while (true)
True
Display Sensor System
print("Current seat status: \n\n");
print(" ==================================");
print(" 0 = empty, 1 = occupied");
print(" ==================================");
print(" | "(seat[0])" | "(seat[1])" | "(seat[2])" | "(seat[3]) " | ");
print(" | "(seat[4])" | "(seat[5])" | "(seat[6]) " | "(seat[7])" | ");
print(" | "(seat[8])" | "(seat[9)" | "(seat[10])" | "(seat[11])" | ");
print(" | "(seat[12]) " | "(seat[13])" | "(seat[14])" | "(seat[15])" | ");
print(" | "(seat[16]) " | "(seat[17])" | "(seat[18]) " | "(seat[19]) " | ");
print(" | "(seat[20]) " | "(seat[21]) " | "(seat[22]) " | "(seat[23]) " | ");
print("Please enter the number of the seat space you would like to sit (A-X)(Z to exit): ");
input(chosenseat);
A


A
while (loop_seat ==0)
True
if (isdigit(chosenseat))
False
else if (chosenseat == 'Z' || chosenseat == 'z')
member();
True
False
print("Please enter (A-X) !");
system(PAUSE);
chosenseat = toupper(chosenseat);
int spaceindex = chosenseat - 'A';
loop_seat = 0;
if (seat[spaceindex] == 1)
True
seat[spaceindex] = 1;
availableseat--;
print("Sorry, that space is already occupied. Please choose another space.");
system(PAUSE);
print("You have register in seat ")(spaceindex)(". There are ")(availableseat)(" spaces available.");
system(PAUSE);
if (availableseat == 0)
print("All seat spaces are now occupied. Please come back later.");
system(PAUSE);
loop_seat = 1;

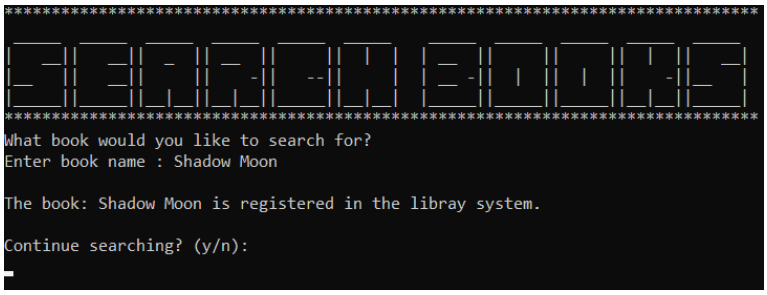
Test Cases



Login Screen

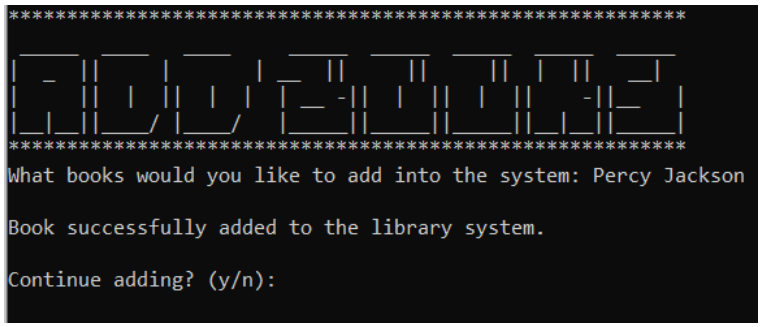


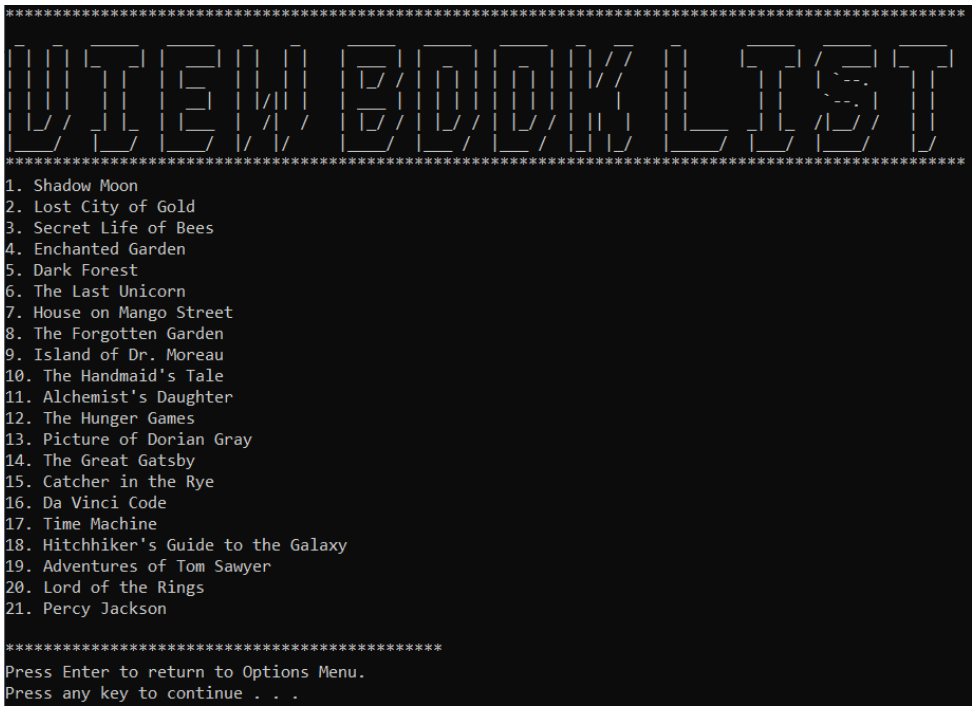
Options menu



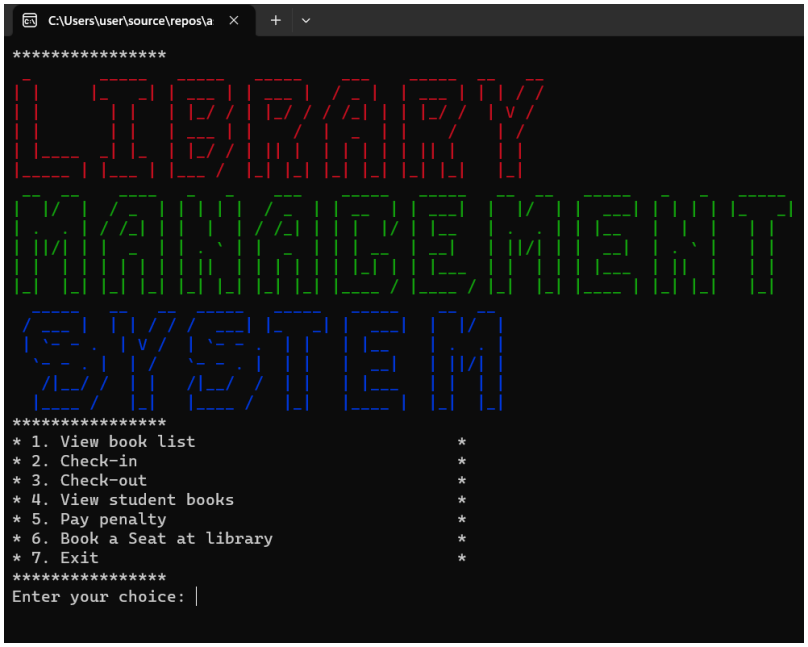
Search book function

Add book function

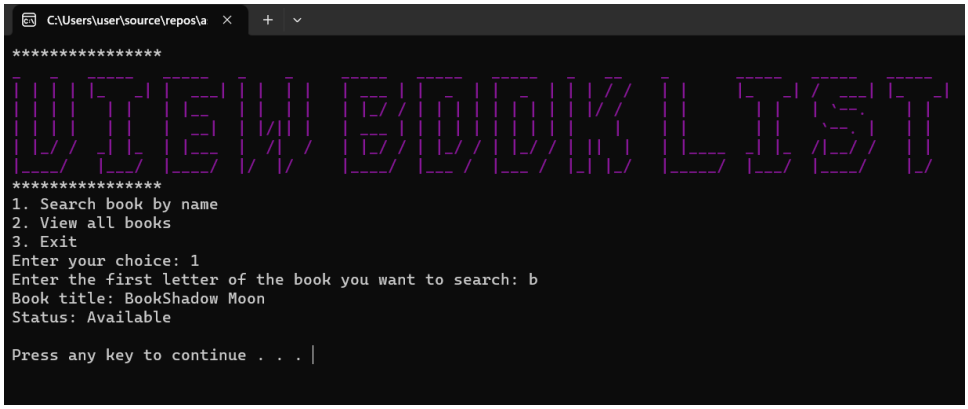




View book list function

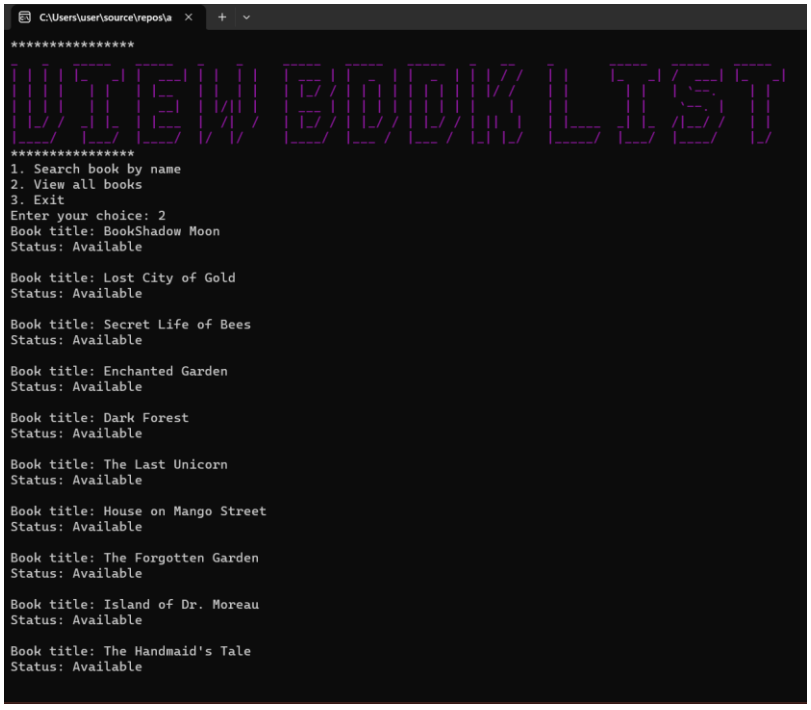


View user main menu function



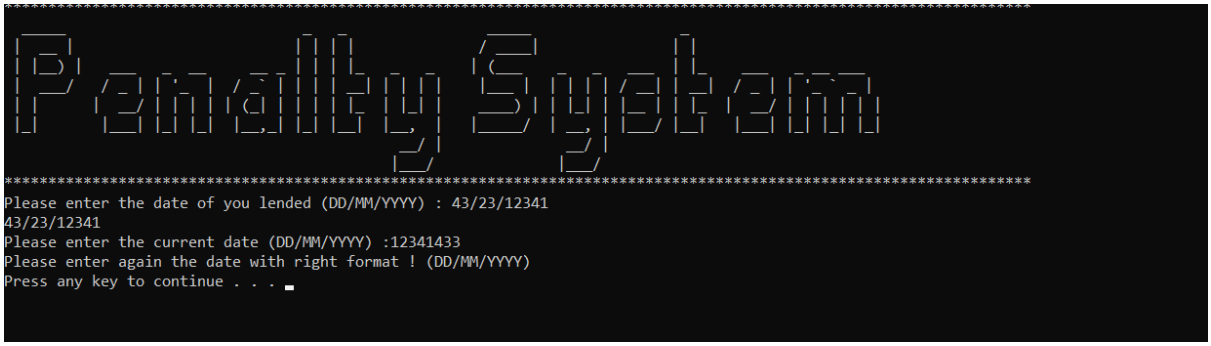
Search book and book’s status from booklist function

View book list function



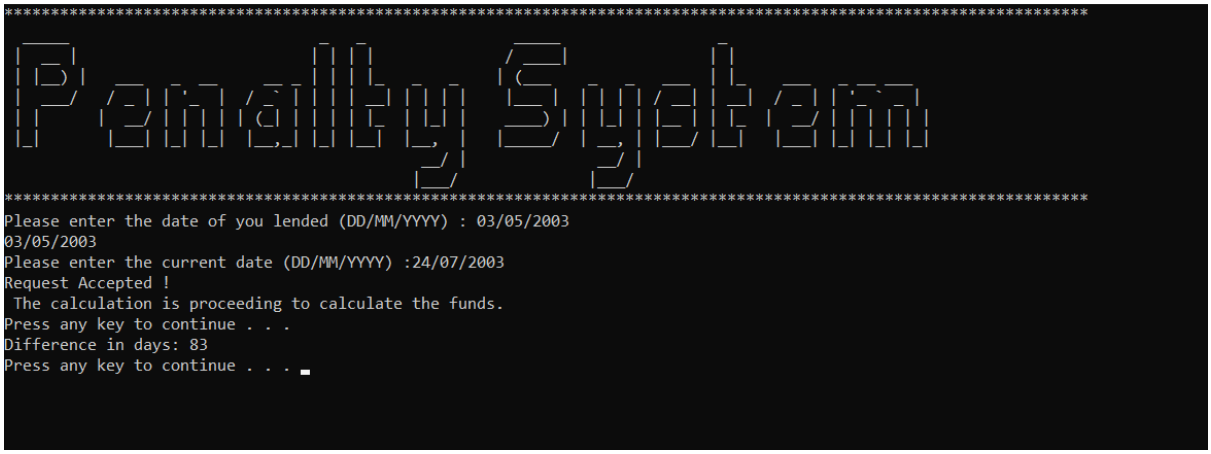
View all books from book list function

View student's borrowed book function



Not accept the wrong date format

Not accept non-valid date



Able to count the difference o date when the user giving correct date

Payment Receipt of penalty system.(Price are depending on the day of difference)

```
CHECK-IN
****
1. Borrow a book
2. Go back to main menu
Enter your choice: 1
Do you have the member ID? (y/n) : y
Enter your student ID: S001
Enter the full name of the book you want to borrow: Shadow Moon
Book borrowed successfully!
You had succesfully borrow the book <<Shadow Moon>>.
The system will start turning to the payment service........
Press any key to continue . . .
Please select your payment method!
****************************************

1. Credit Card / Debit Card
2. Online Payment
3. Touch N Go Online Pay
4. Cash Service
5. Promo Code
1
Card Details
---------------
Card Number (16 digit) : 134
Expiry Date : 123
CVV : Name On Card :
Somethings wrong found in Card Details
Please enter the Card details again!
Card Details
---------------
Card Number (16 digit) :
```

Different Payment way and when the user type in wrong data in credit card

```
Please select bank :
1. Public Bank
2. Maybank
3. CIMB Bank
4. Am Bank
1
Welcome to Public Bank !
-----------------------------------------------
Username:  ong ys bla
Password:123123123

Login Successfully!
------------------------------
ong ys bla, you successfully transfer the money through online !
Thanks for using Public Bank as your services!
The Recipt will printed after the payment is done!
.
.
.
.
.
.
.
.
.
.
.
.
Click enter to proceed the progress........
Press any key to continue . . .
```

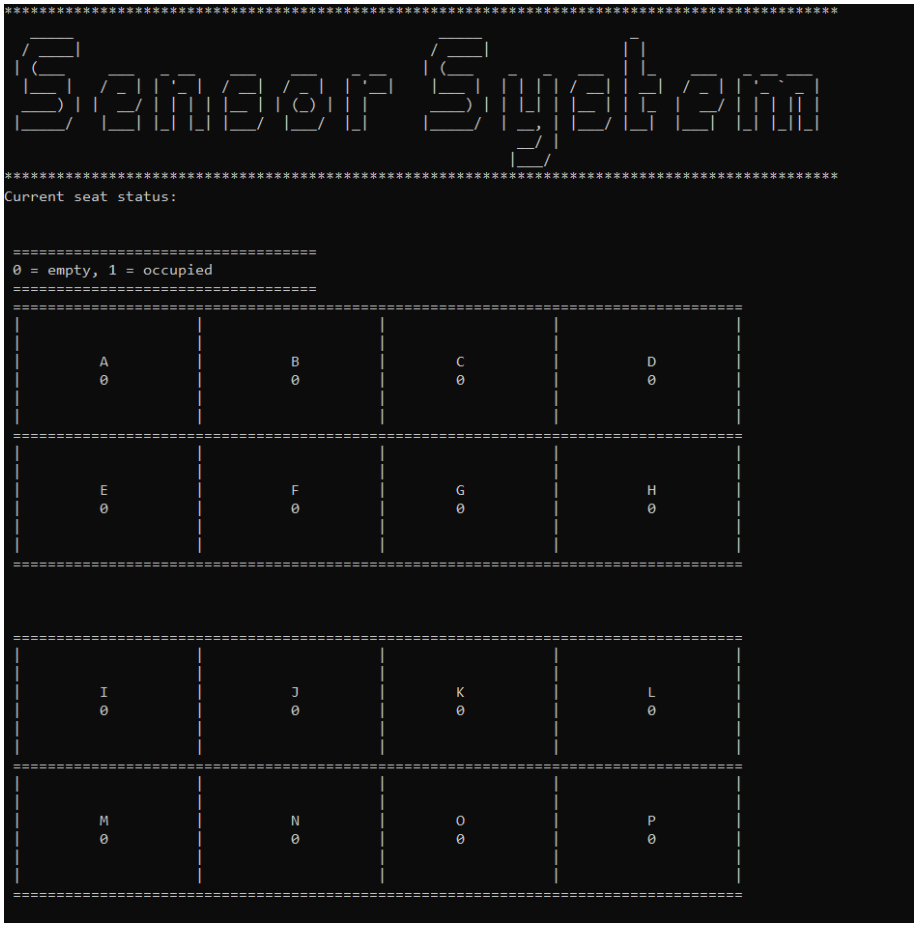
Online Payment and different bank is provided

```
CHECK-IN
****
1. Borrow a book
2. Go back to main menu
Enter your choice: 1
Do you have the member ID? (y/n) : y
Enter your student ID: S001
Enter the full name of the book you want to borrow: Shadow Moon
Book borrowed successfully!
You had succesfully borrow the book <<Shadow Moon>>.
The system will start turning to the payment service........
Press any key to continue . . .
Please select your payment method!
****************************************

1. Credit Card / Debit Card
2. Online Payment
3. Touch N Go Online Pay
4. Cash Service
5. Promo Code
3
 Please filled in the phone number:
01111998088
Succesfully login !
Please enter your 6-digit PIN number
123132413
Error ! Re-enter your phone number with correct PIN .
 Please filled in the phone number:
01111998088
Succesfully login !
Please enter your 6-digit PIN number
123456
Thanks for using Touch N Go as ur services !
You had succesfully transfer the money to the receiver!
The Recipt will printed after the payment is done!
```

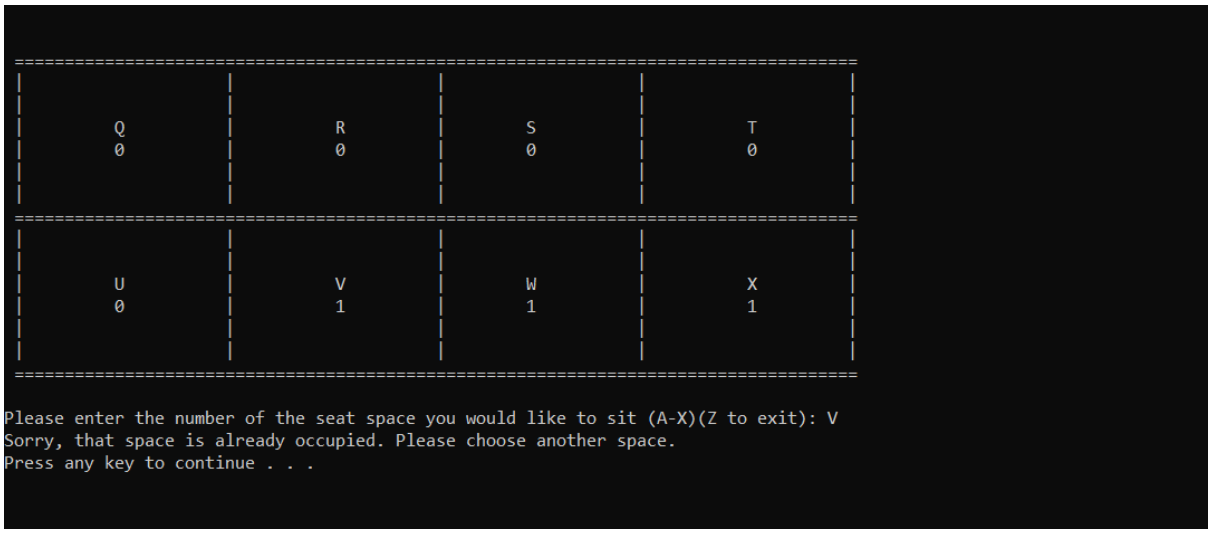
Touch N Go payment and when user type in wrong data

```
CHECK-IN
****
1. Borrow a book
2. Go back to main menu
Enter your choice: 1
Do you have the member ID? (y/n) : y
Enter your student ID: S001
Enter the full name of the book you want to borrow: Shadow Moon
Book borrowed successfully!
You had succesfully borrow the book <<Shadow Moon>>.
The system will start turning to the payment service........
Press any key to continue . . .
Please select your payment method!
****************************************

1. Credit Card / Debit Card
2. Online Payment
3. Touch N Go Online Pay
4. Cash Service
5. Promo Code
5
Please enter the secret promo code for free payment: 123
Please fill in again !
Please enter the secret promo code for free payment: 123
Please fill in again !
Please enter the secret promo code for free payment: qwe
Please fill in again !
Please enter the secret promo code for free payment: MROYS
Congratulation ! You had earn a free payment !
The Recipt will be printed out in awhile.......
Press any key to continue . . .
```

Promo code and when user type in wrong data



Sensor system let student book their seat in library



If the place is occupied (1) the system will tell user that already occupied . and unable student to seat there.

Source code

```
#include<iostream>
#include<iomanip>
#include<cmath>
#include<cctype>
#include<cstring>
#include<string>
#include<fstream>
using namespace std;



void member();


void view();
void view_books();
void view_all_books();
void search_book_by_name();
void borrow_book();
void return_book();
void update_book_status(const string& book_name, const string& new_status);
void user_mainmenu();
void view_student_books();
void check_in();
void check_out();
void handle_invalid_input();
void add_new_books();
void delete_books();
void return_menu();
void payment();
void recipt();
void penalty();
void pena_recipt(int&);
void sensor();
```

```
void showOptions(string&);

int main()
{
   char login;
   int noADMIN = 0, noBOOKS = 0; //Initialize counter to 0
   bool loginSuccessful = false;
   string ADMIN_name, NAME_check;
   string ADMIN_pass, PASS_check;
   string BOOK_name, BOOK_check;
   string BOOKadd, BOOKdel;
   ifstream inFile;
   ofstream outFile;
   ofstream temp;

   do
   {
      cout <<
"*********************************************************************" << endl;
      cout << " _ _ _       _                       _                          " << endl;
      cout << "| | | | ___ | | ___  ___  _____  ___   | |_  ___                 " << endl;
      cout << "| | | || -_|| ||  _|| . ||     || -_|  |  _|| . |                " << endl;
      cout << "|_____||___||_||___||___||_|_|_||___|  |_|  |___|                 " << endl;
      cout << " _____  _____  _____  _____  _____  _____  _____  _____         " << endl;
      cout << "|  _  ||     || __  ||  _  ||_   _|| |  || __  ||  |  ||   __|       " << endl;
      cout << "|     || | | || __ -||     |  | |  | |  || __ -||  |  ||__   |       " << endl;
      cout << "|__|__||_|_|_||_____||__|__|  |_|  |_____||_____||_____||_____|       " << endl;
      cout << " __     _  _                      _____           _              " << endl;
      cout << "|  |   |_|| |_  ___  ___  ___  _ _  |   __| _ _  ___ | |_  ___  _____ " << endl;
      cout << "|  |__ | || . ||  _|| .'||  _|| | |  |__   || | ||_ -||  _|| -_|     |" << endl;
      cout << "|_____||_||___||_|  |__,||_|  |_  |  |_____||_  ||___||_|  |___||_|_|_|" << endl;
      cout << "                              |___|         |___|                   " << endl;
      cout <<
"*********************************************************************" << endl;
```

```
        cout <<
           "Login\n\n";
        cout <<
           "1. Administrator\n\n";
        cout <<
           "2. Member\n\n";
        cout <<
           "**********************************************************************\n"
<< endl;
        cout <<
           "Please choose login method: ";
        cin >> login;
        cout << endl;

        if (login != '1' && login != '2')
        {
           cout << "-----------------------------------------------------" << endl;
           cout << "Please enter a required number according to the menu!" << endl;
           cout << "-----------------------------------------------------" << endl;
        }
    } while (login != '1' && login != '2'); //To loop again if there is error

    if (login == '1')
    {
        while (!loginSuccessful)
        {
           inFile.open("ADMIN.txt"); //Call Admin List file out
           if (inFile.fail())
           {
              cout << "Error opening file" << endl;
              exit(1);
           }
```

```
        cout <<
            "*****************************************\n";
        cout << "\nUsername: ";
        cin.clear();
        getline(cin >> std::ws, ADMIN_name);//discard whitespace char before reading input
        cout << endl;
        cout << "\nPassword: ";
        getline(cin >> std::ws, ADMIN_pass);
        cout << endl;
        cout << "\n*****************************************\n";



        while (getline(inFile, NAME_check) && inFile >> PASS_check)//check every line of the
ADMIN.txt with input
        {
            noADMIN++;
            if (ADMIN_name == NAME_check && ADMIN_pass == PASS_check)
            {
                loginSuccessful = true;
                break; //Exit the while loop if correct info
            }
            inFile.ignore(); //discard any leftover characters in the input buffer
        }
        if (noADMIN == 0 || ADMIN_name != NAME_check || ADMIN_pass != PASS_check)
        {
            cout << "Invalid username or password!" << endl;
            loginSuccessful = false;
        }
        inFile.close(); //Close txt file
    }

    char options;
    do
    {
```

```
        showOptions(ADMIN_name);

        cout << "Please enter options: ";
        cin >> options;
        cout << endl;

        if (options != '1' && options != '2' && options != '3' && options != '4' && options != '5')
        {
           cout << "Please chooose an available options!" << endl;
           system("pause");
        }

        else if (options == '1')
        {
           inFile.open("BookList.txt");
           if (!inFile.is_open())
           {
              cout << "Error opening file" << endl;
              system("pause");
           }
           char choice;
           do
           {
              system("cls");
              cout <<
"*********************************************************************************"
<< endl;
              cout << " _____  _____  _____  _____  _____  _____    _____  _____  _____  _____
_____ " << endl;
              cout << "|  __||  __|| _  || __  ||     ||  |  |  |__  ||     ||     ||  |  ||  __|" << endl;
              cout << "|__   ||  __||     ||    -||   --||     |  |__ -||  |  ||  |  ||    -||__   |" << endl;
              cout << "|_____||_____||__|__||__|__||_____||__|__|  |_____||_____||_____||__|__||_____|"
<< endl;
              cout <<
"*********************************************************************************"
<< endl;
```

```
            bool foundBOOK = false;
            cout << "What book would you like to search for?" << endl;
            cout << "Enter book name : ";
            cin.ignore();//discard the newline character left in the input buffer
            getline(cin >> std::ws, BOOK_name);
            cout << endl;

            inFile.seekg(0, ios::beg); //reset file pointer to beginning of file
            while (getline(inFile, BOOK_check))//check every books registered in the system
            {
               if (BOOK_name == BOOK_check)
               {
                  cout << "The book: " << BOOK_name << " is registered in the libray system." <<
endl;
                  foundBOOK = true;
                  break;
               }
            }
            if (!foundBOOK)//return false if there is no such books in system
            {
               cout << "The book: " << BOOK_name << " is not registered in the library system." <<
endl;
            }

            cout << "\nContinue searching? (y/n): " << endl;//prompt for continue search or no
            cin >> choice;

            while (tolower(choice) != 'y' && tolower(choice) != 'n')//to validate value
            {
               cout << "Please enter y or n: " << endl;
               cin >> choice;
            }

            if (tolower(choice) == 'n')
```

```
        {
            break;
        }

    } while (tolower(choice) == 'y');//loop if yes
    inFile.close();
}
else if (options == '2')
{
    view(); //call function for view books options
    inFile.open("BookList.txt");
    if (!inFile.is_open())
    {
        cout << "Error opening file" << endl;
        system("pause");
    }
    string line;
    while (!inFile.eof())
    {
        int x = 0;
        while (getline(inFile, line))
        {
            cout << x + 1 << ". " << line << endl;
            x++;
        }
    }
    inFile.close();
    cout << "\n**********************************************" << endl;
    cout << "Press Enter to return to Options Menu. " << endl;
    system("pause");
}
else if (options == '3')
{
    char choice;
```

```
        do
        {
            system("cls");
            cout << "************************************************************" <<
endl;
            cout << " _____  ____   ____    _____  _____  _____  _____  _____ " << endl;
            cout << "| _  ||   ||   |  |__  ||    ||    || |  ||   __|" << endl;
            cout << "|    || ||| ||  |__ -|| |  || |  ||   -||__   |" << endl;
            cout << "|__|__||____/ |____/   |_____||_____||_____||__|__||_____|" << endl;
            cout << "************************************************************" <<
endl;
            outFile.open("BookList.txt", ios_base::app);
            if (!outFile.is_open())
            {
                cout << "Error: failed to open file for appending" << endl;
                system("pause");
            }
            cout << "What books would you like to add into the system: ";
            cin.ignore();
            getline(cin, BOOKadd);
            cout << endl;

            outFile << BOOKadd << endl;
            cout << "Book successfully added to the library system. " << endl;
            outFile.close();

            cout << "\nContinue adding? (y/n): " << endl;//prompt for continue search or no
            cin >> choice;

            while (tolower(choice) != 'y' && tolower(choice) != 'n')//to validate value
            {
                cout << "Please enter y or n: " << endl;
                cin >> choice;
            }
```

```
            if (tolower(choice) == 'n')
            {
               break;
            }
         } while (tolower(choice) == 'y');
      }
      else if (options == '4')
      {
         char choice;
         do
         {
            system("cls");
            cout <<
"*********************************************************************************"
<< endl;
            cout << " ____   _____  __    _____  _____  _____   _____  _____  _____  _____  _____
" << endl;
            cout << "|   ||  __|| |  |  __||_  _||  __| |__  ||    ||    || | ||  __|" << endl;
            cout << "| |||  __||  |__ |  __| ||| |  __| |__ -|| |  || |  ||   -||__  |" << endl;
            cout << "|____/ |_____||_____||_____| |_| |_____| |_____||_____||_____||__|__||_____|"
<< endl;
            cout <<
"*********************************************************************************"
<< endl;
            inFile.open("BookList.txt");
            temp.open("temp.txt");
            string line;
            string bookToDelete;
            cout << "Please enter the name of the book you want to delete: ";
            getline(cin >> ws, bookToDelete);
            bool found = false;
            while (getline(inFile, line))
            {
               if (bookToDelete != line)
               {
```

```
        temp << line << endl;
    }
    else
    {
        found = true; // set found to true when book is found
    }

}
inFile.close();
temp.close();
remove("BookList.txt");
rename("temp.txt", "BookList.txt");

if (found)
{
    cout << bookToDelete << " has been deleted." << endl;
}
else
{
    cout << bookToDelete << " was not found in the book list." << endl;
}
cout << "\nContinue deleting? (y/n): " << endl;//prompt for continue search or no
cin >> choice;

while (tolower(choice) != 'y' && tolower(choice) != 'n')//to validate value
{
    cout << "Please enter y or n: " << endl;
    cin >> choice;
}

if (tolower(choice) == 'n')
{
    break;
}
```

```
            } while (tolower(choice) == 'y');
        }
    } while (options != '5');
    cout << "Goodbye and have a nice day!" << endl;
    cout << "Looking forward for your next visit admin " << ADMIN_name << "." << endl;
    cout << ":D " << endl;
  }
  if (login == '2')
  {
    member();

  }


  return 0;
}

void member()// Function to display the main menu for the user and handle their actions
{
  while (true)    // Keep running the main menu until the user chooses to exit
  {
    user_mainmenu();
  }
}

bool is_alpha(char c)// Function to check if a character is an alphabetic character (A-Z or a-z)
{
  return ((c >= 'A' && c <= 'Z') || (c >= 'a' && c <= 'z'));
}

bool is_full_book_name(const string& book_name) {// Function to check if the given book name is a full book name present in the "books.txt" file
```

```
    ifstream file("books.txt");// Open the "books.txt" file for reading
    string line;

    if (file.is_open()) { // Check if the file is open
        while (getline(file, line)) {        // Read the file line by line
            int pos = line.find(book_name);            // Search for the book name in the current line
            if (pos != -1 && (line[book_name.size()] == ',' || line[book_name.size()] == '\0')) {            // If the book name is found and is followed by a comma or end of the line, return true
                file.close();
                return true;
            }
        }
        file.close();        // Close the file after processing
    }
    else {
        cout << "Unable to open file." << endl;         // If the file could not be opened, display an error message
    }

    return false;     // If the book name is not found in the file, return false
}

bool is_valid_student_id(const string& student_id) {
    ifstream file("students.txt"); // Open the file "students.txt" for reading
    string line; // Create a string variable to store each line from the file

    if (file.is_open()) { // Check if the file was successfully opened
        while (getline(file, line)) { // Read each line from the file
            int pos = line.find(','); // Find the first comma in the line
            int next_pos = line.find(',', pos + 1); // Find the next comma in the line
            string id = line.substr(pos + 1, next_pos - pos - 1); // Extract the student ID

            if (id == student_id) { // Check if the extracted ID matches the input student ID
                file.close(); // Close the file
```

```
            return true; // Return true if a match was found
          }
        }
        file.close(); // Close the file
    }
    else {
        cout << "Unable to open file." << endl; // Output error message if file could not be opened
    }

    return false; // Return false if the end of the file is reached or if the file could not be opened
}

void user_mainmenu() {
    int choice; // Initialize variable to store the user's menu choice
    bool validInput = false; // Initialize a flag to determine if the user has entered a valid input
    int diff_days = 0; // Initialize a variable to store the difference between two dates

    while (!validInput) { // Repeat until a valid input is entered
        system("cls"); // Clear the console
        system("Color 07"); // Set the console color
        add_new_books();
        delete_books();
        cout << "****************          " << endl;
        cout << "\033[31m"; // Change text color to red
        cout << " _      ______  ______  ______   ____   ______  __   __ " << endl;
        cout << "| |    |_   _| | ____| | ____|  / _ |  | ____| | | / / " << endl;
        cout << "| |      | |   | |_/ / | |_/ / / /_| | | |_/ /  | V /  " << endl;
        cout << "| |      | |   | ____| |   /  |  _  | |   /    | /   " << endl;
        cout << "| |_____  _| |_  | |_/ / | || |  | | | | | || |    | |   " << endl;
        cout << "|______| |____| |____/  |_| |_| |_| |_| |_| |_|   |_|   " << endl;
        cout << "\033[0m"; // Reset text color
        cout << "\033[32m"; // Change text color to green
        cout << " __  __   _____  _  _   ____   ______  ______  __  __  ______  _  _  ______  " << endl;
        cout << "|  |/  |  / _  | | | | |  / _  | | __  | | ___|  |  |/  | |  ___| | | | | |_   _| " << endl;
```

```
cout << "| .  . | / /_| | | | | / /_| | | | |/ | |__  | .  . | | |__  | | |  | |  " << endl;
cout << "| ||/| | | _ | | . `| | _ | | | __ | __|  | ||/| | | __| | . `|  | |  " << endl;
cout << "| | | | | | | | || | | | | | |_| | | |___ | | | | |___ | || |  | |  " << endl;
cout << "|_|  |_| |_| |_| |_| |_| |_| |_| |_| |_____ / |_____ / |_|  |_| |_____ | |_| |_|  |_|  " << endl;
cout << "\033[0m"; // Reset text color
cout << "\033[34m"; // Change text color to blue
cout << "  _____   __   __ _____   _____   _____    __ __  " << endl;
cout << " / ___ | | | / / /  ___|| _   _|| ___|  | |  |/  | " << endl;
cout << " | `- - .  | V /  | `- - .  | |  | | __   | .  . | " << endl;
cout << "  `- - . | | /   `- - . | | |  | __|  | ||/| | " << endl;
cout << "   /|__/ / | |   /|__/  /  | |  | |___   | | | " << endl;
cout << "  |_____ /   |_|   |_____ /   |_|   |_____ |  |_|  |_| " << endl;
cout << "\033[0m"; // Reset text color
cout << "*****************          " << endl;
cout << "* 1. View book list                    *        " << endl;
cout << "* 2. Check-in                          *        " << endl;
cout << "* 3. Check-out                         *        " << endl;
cout << "* 4. View student books                *        " << endl;
cout << "* 5. Pay penalty                       *        " << endl;
cout << "* 6. Book a Seat at library            *        " << endl;
cout << "* 7. Exit                              *        " << endl;
cout << "*****************          " << endl;
cout << "Enter your choice: ";// Prompt the user for a menu choice

if (cin >> choice) { // If the user enters a valid integer
    switch (choice) {
    case 1:
        view_books(); // Call the "view_books" function
        validInput = true;
        break;
    case 2:
        check_in(); // Call the "check_in" function
        validInput = true;
        break;
```

```
            case 3:
                check_out(); // Call the "check_out" function
                validInput = true;
                break;
            case 4:
                view_student_books(); // Call the "view_student_books" function
                validInput = true;
                break;

            case 5:
                cin.ignore();
                penalty(); // Call the "penalty" function
                payment(); // Call the "payment" function
                validInput = true;
                break;

            case 6:
                sensor(); // Call the "sensor" function
                validInput = true;
                break;

            case 7:
                cout << "Thank you for using the Library Management System!" << endl;
                exit(0); // Exit the program
                break;
            }
        }
        else {
            handle_invalid_input(); // Call the "handle_invalid_input" function
        }
    }
}
```

```
void handle_invalid_input() {// pausing the system, clearing the input buffer, and ignoring any remaining characters.
    cout << "Invalid input!" << endl;
    system("pause");
    cin.clear();
    cin.ignore(INT_MAX, '\n');
}

void return_menu() {// clearing the input buffer, and ignoring any remaining characters.
    system("pause");
    cin.clear();
    cin.ignore(INT_MAX, '\n');
}

void check_in() {// This function displays the check-in menu, allowing the user to borrow a book
// or return to the main menu. It also handles invalid input using the handle_invalid_input function.
    int choice;
    system("cls");
    cout << "CHECK-IN" << endl;
    cout << "****" << endl;
    cout << "1. Borrow a book" << endl;
    cout << "2. Go back to main menu" << endl;
    cout << "Enter your choice: ";
    cin >> choice;

    switch (choice) {
    case 1:
        borrow_book();
        break;
    case 2:
        user_mainmenu();
        break;
    default:
        handle_invalid_input();
```

```
    }
}

void check_out() {
    // This function displays the check-out menu, allowing the user to return a book
// or return to the main menu. It also handles invalid input using the handle_invalid_input function.
    int choice;
    system("cls");
    cout << "*****************" << endl;
    cout << "Check-out" << endl;
    cout << "*****************" << endl;
    cout << "1. Return a book" << endl;
    cout << "2. Go back to main menu" << endl;
    cout << "Enter your choice: ";
    cin >> choice;

    switch (choice) {
    case 1:
        return_book();
        break;
    case 2:
        user_mainmenu();
        break;
    default:
        handle_invalid_input();
    }
}

void view_student_books() {
    // This function displays the list of borrowed books for a specific student using their ID.
// It reads the student data from a file and displays the student's name and the titles of the borrowed books.
    string student_id;
    cout << "Enter the student's ID: ";
```

```
    cin.ignore();
    getline(cin, student_id);

    ifstream students_file("students.txt");
    string line;
    bool found = false;

    if (students_file.is_open()) {
        while (getline(students_file, line)) {
            int id_pos = line.find(student_id);
            if (id_pos != -1 && id_pos > 0 && line[id_pos - 1] == ',' && (line[id_pos + student_id.size()]
== ',' || line[id_pos + student_id.size()] == '\0')) {
                found = true;
                string student_name = line.substr(0, id_pos - 1);

                cout << "Student Name: " << student_name << endl;

                int book_start = id_pos + student_id.size() + 1;
                int book_end = line.find(',', book_start);

                if (book_start >= line.size()) {
                    cout << "Didn't borrow any books" << endl;
                }
                else {
                    int book_number = 1;
                    while (book_end != -1) {
                        string book_title = line.substr(book_start, book_end - book_start);
                        cout << "Book " << book_number << ": " << book_title << endl;
                        book_start = book_end + 1;
                        book_end = line.find(',', book_start);
                        book_number++;
                    }
                    string last_book_title = line.substr(book_start);
                    cout << "Book " << book_number << ": " << last_book_title << endl;
```

```
            }
            break;
        }
    }
    students_file.close();
}
else {
    cout << "Unable to open file." << endl;
}

if (!found) {
    cout << "Invalid student ID." << endl;
}

system("pause");
}

void update_student_record_borrow(const string& student_id, const string& book_name) {
    ifstream file("students.txt");
    string updated_records; // Stores the updated records of the students
    string line;
    bool found = false;

    // Read the student records file
    if (file.is_open()) {
        while (getline(file, line)) {
            int pos = line.find(student_id);
            // If the student ID is found, add the book name to the record
            if (pos != -1) {
                found = true;
                line += "," + book_name;
            }
            updated_records += line + '\n';
        }
```

```
        file.close();
    }
    else {
        cout << "Unable to open file." << endl;
    }

    // If the student ID is found, update the records file
    if (found) {
        ofstream file("students.txt");
        if (file.is_open()) {
            file << updated_records;
            file.close();
        }
        else {
            cout << "Unable to open file." << endl;
        }
    }
    else {
        cout << "Invalid student ID. Please try again." << endl;
    }
}

void update_student_record_return(const string& student_id, const string& book_name) { // It removes the book name from the student's record in the "students.txt" file.
    ifstream file("students.txt");
    string updated_records;// Stores the updated records of the students
    string line;
    bool found = false;

    // Read the student records file
    if (file.is_open()) {
        while (getline(file, line)) {
            int pos = line.find(student_id);
            if (pos != -1) {
```

```
                found = true;
                int book_pos = line.find(book_name);
                // If the book name is found, remove it from the record
                if (book_pos != -1) {
                    line.erase(book_pos - 1, book_name.length() + 1);
                }
            }
            updated_records += line + '\n';
        }
        file.close();
    }
    else {
        cout << "Unable to open file." << endl;
    }

    // If the student ID is found, update the records file
    if (found) {
        ofstream file("students.txt");
        if (file.is_open()) {
            file << updated_records;
            file.close();
        }
        else {
            cout << "Unable to open file." << endl;
        }
    }
    else {
        cout << "Invalid student ID. Please try again." << endl;
    }
}
```

void view_books() {// This function displays the view books menu, allowing the user to search a book by name, view all books, or exit.

```
    int choice;
    system("cls");
    cout << "*****************          " << endl;
    cout << "\033[35m";
        cout << "_   _   _____   _____   _   _    _____   _____   _____   _   __   _      _____   _____
_____  " << endl;
        cout << "| | | | |_   _| |  ___| | | | |  | ___ | | _ | | _ | | | / /  | |    |_   _| /  ___| |_   _| " << endl;
        cout << "| | | |   | |   | |__   | | | |  | |_/ / | | | | | | | |/ /   | |      | |   | `--.    | |   " << endl;
        cout << "| | | |   | |   |  __|  | |/| |  | ___ | | | | | | | | |    | |      | |    `--. |   | |   " << endl;
        cout << "| |_/ /  _| |_  | |___  | /| /  | |_/ / | |_/ / | |_/ / | || |  | |____  _| |_  /|__/ /   | |   " << endl;
        cout << "|_____/   |___/  |_____/  |/ |/    |_____/  |____/  |____/  |_| |_/   |_____/  |____/  |_____/    |_/   "
<< endl;
    cout << "\033[0m";
    cout << "*****************          " << endl;
    cout << "1. Search book by name" << endl;
    cout << "2. View all books" << endl;
    cout << "3. Exit" << endl;
    cout << "Enter your choice: ";
    cin >> choice;

    switch (choice) {
    case 1:
        search_book_by_name();
        break;
    case 2:
        view_all_books();
        break;
    case 3:
        break;
    default:
        handle_invalid_input();
    }
}
```

```
void view_all_books() { // This function reads the "books.txt" file and displays the details of all books.
    ifstream file("books.txt");
    string line;

    // Open the file
    if (file.is_open()) {
        while (getline(file, line)) {
            int first_comma = line.find(',');

            string book_title = line.substr(0, first_comma);
            string status = line.substr(first_comma + 2);

            if (status == "Available") {
                cout << "Book title: " << book_title << endl;
                cout << "Status: " << status << endl;
                cout << endl;
            }
        }
        file.close();
    }
    else {
        cout << "Unable to open file." << endl;
    }
    return_menu();
}

void borrow_book() {

    char loop_ID = 1;
    int loop_brw = 1;
    char ans_ID;
    double price;
```

```
while (loop_ID == 1)
{

    cout << "Do you have the member ID? (y/n) : ";
    cin >> ans_ID;
    ans_ID = tolower(ans_ID); // changing the ans_ID to lowercase



    if (ans_ID == 'y') // no need define Y and y cause ans only y
    {
        string student_id;
        cout << "Enter your student ID: ";
        cin.ignore();
        getline(cin, student_id);
        if (is_valid_student_id(student_id)) {
            string book_name;
            cout << "Enter the full name of the book you want to borrow: ";
            getline(cin, book_name);

            if (is_full_book_name(book_name)) {
                update_book_status(book_name, "Borrowed");
                update_student_record_borrow(student_id, book_name);
                cout << "You had succesfully borrow the book <<" << book_name << ">>. " << endl;
                cout << "The system will start turning to the payment service....." << endl;
                system("pause");
                payment();
                cout << "Successfully paid ! " << endl;
                recipt();
            }
            else {
                cout << "Invalid input. Please enter the full name of the book." << endl;
            }

        }
```

```
            else {
                cout << "Invalid student ID." << endl;
            }
            price = 1; // declare the price
            loop_ID = 0; // back to loop
        }

        else if (ans_ID == 'n')
        {
            cout << " Invalid student ID. Non-student is not allowed to borrow book!" << endl;
            loop_ID = 0; // back to loop
        }

        else
        {
            cout << "Error. Please answer in (y/n)" << endl;

        }
    }



    system("pause");
    user_mainmenu();
}

void return_book() {
    int diff_days = 0;
    string student_id;
    char choice_return;

    cout << "Enter your student ID: ";
    cin.ignore();
    getline(cin, student_id);
```

```
    if (is_valid_student_id(student_id)) {
        string book_name;
        cout << "Enter the full name of the book you want to return: ";
        getline(cin, book_name);
        int date_days;
        int diff_days;
        bool loop_date = false;
        string date, current_date;
        int days, month, year;
        int max_days = 0;
        int current_days, current_month, current_year;
        int pena_price;

        while (loop_date == false)
        {
            system("cls");
            cout << "Please enter the date of you lended (DD/MM/YYYY) : ";
            getline(cin, date);
            cout << date << endl;
            cout << "Please enter the current date (DD/MM/YYYY) :";
            getline(cin, current_date);
            if ((date.length() == 10 && date[2] == '/' && date[5] == '/') && (current_date.length() == 10
&& current_date[2] == '/' && current_date[5] == '/'))
            {
                days = stoi(date.substr(0, 2)); // take the 1st and 2nd word inside the date to make days
                month = stoi(date.substr(3, 2));// take the 4th and 5th word inside the date to make month
                year = stoi(date.substr(6, 4));// take the 7th,8th,9th and 10th inside the date to make year
                current_days = stoi(current_date.substr(0, 2)); // take the 1st and 2nd word inside the
current_date to make days
                current_month = stoi(current_date.substr(3, 2));// take the 4th and 5th word inside the
current_date to make month
                current_year = stoi(current_date.substr(6, 4));// take the 7th,8th,9th and 10th inside the
current_date to make year
```

```
        if ((days <= 31 && days >= 1 && month >= 1 && month <= 12 && year > 0) &&
(current_days <= 31 && current_days >= 1 && current_month >= 1 && current_month <= 12 &&
current_year > 0)) // let user only can type in valid date

        {


          if ((month == 1 || month == 3 || month == 5 || month == 7 || month == 8 || month == 10 ||
month == 12) && (current_month == 1 || current_month == 3 || current_month == 5 || current_month
== 7 || current_month == 8 || current_month == 10 || current_month == 12))

          {

            max_days = 31;

            loop_date = true;


          }


          else if (month == 2 && current_month == 2) // February only 28 day

          {

            max_days = 28;

            loop_date = true;


          }


          else if ((month == 4 || month == 6 || month == 9 || month == 11) && (current_month == 4
|| current_month == 6 || current_month == 9 || current_month == 11))

          {

            max_days = 30;

            loop_date = true;


          }


          loop_date = true;

          // counting the different of days depending on user type in date

          date_days = days + (month - 1) * 31 + (year - 1) * 12 * 31;

          current_days = current_days + (current_month - 1) * 31 + (current_year - 1) * 12 * 31;

          diff_days = current_days - date_days;


        }
```

```
            else
            {
                cout << "Please give a valid date!" << endl;
                system("pause");
                loop_date = false; // back to the loop



            }

        }
        else
        {
            cout << "Please enter again the date with right format ! (DD/MM/YYYY) " << endl;
            system("pause");
            loop_date = false; // back to the loop



        }



    }
    cout << "Difference in days: " << diff_days << endl;




    cout << "Please pay the bills if your borrowed day more than 7 at the penalty part in main menu!" << endl;
    cout << "(y - proceed to return book / n - exit to main menu) : " << endl;
    cin >> choice_return;
    choice_return = tolower(choice_return); // lowercase the choice_return
    if (choice_return == 'y')
    {
        if (is_full_book_name(book_name)) {
            update_book_status(book_name, "Available");
```

```
                update_student_record_return(student_id, book_name);
            }
            else {
                cout << "Invalid input. Please enter the full name of the book." << endl;
            }
        }
        else if (choice_return == 'n')
        {
            cout << "The system is exiting to main menu....." << endl;
            system("pause");

        }

    }
    else {
        cout << "Invalid student ID." << endl;
    }
    cout << "Press Enter to continue...";
    cin.get();
    user_mainmenu();
}

void add_new_books() {
    ifstream books_file("books.txt");
    ifstream booklist_file("BookList.txt");
    string line1, line2;
    string updated_books; // To store the updated book records
    bool found; // Flag to check if the book is found in the books.txt file

    // Read the books.txt file line by line
    if (books_file.is_open()) {
        while (getline(books_file, line1)) {
            updated_books += line1 + '\n';
        }
```

```
    books_file.close();
}
else {
    cout << "Unable to open file 'books.txt'" << endl;
    return;
}

// Read the BookList.txt file line by line
if (booklist_file.is_open()) {
    while (getline(booklist_file, line2)) {
        found = false;
        // Check if the book from the BookList.txt file is already in the books.txt file
        ifstream books_file("books.txt");
        if (books_file.is_open()) {
            while (getline(books_file, line1)) {
                int pos = line1.find(line2);
                if (pos != -1) {
                    found = true;
                    break;
                }
            }
            books_file.close();
        }
        else {
            cout << "Unable to open file 'books.txt'" << endl;
            return;
        }

        // If the book is not found in the books.txt file, add it to the updated_books string
        if (!found) {
            updated_books += line2 + ", Available\n";
        }
    }
    booklist_file.close();
```

```
    }
    else {
        cout << "Unable to open file 'BookList.txt'" << endl;
        return;
    }

    // Write the updated_books string to the books.txt file
    ofstream books_file_out("books.txt");
    if (books_file_out.is_open()) {
        books_file_out << updated_books;
        books_file_out.close();
    }
    else {
        cout << "Unable to open file 'books.txt'" << endl;
    }
}

void delete_books() {
    ifstream books_file("books.txt");
    ifstream booklist_file("BookList.txt");
    string line1, line2;
    string updated_books; // To store the updated book records
    bool found; // Flag to check if the book is found in the BookList.txt file

    // Read the books.txt file line by line
    if (books_file.is_open()) {
        while (getline(books_file, line1)) {
            found = false;
            // Check if the book from the books.txt file is in the BookList.txt file
            ifstream booklist_file("BookList.txt");
            if (booklist_file.is_open()) {
                while (getline(booklist_file, line2)) {
                    int pos = line1.find(line2);
                    if (pos != -1) {
```

```
                found = true;
                break;
            }
        }
        booklist_file.close();
    }
    else {
        cout << "Unable to open file 'BookList.txt'" << endl;
        return;
    }

    // If the book is found in the BookList.txt file, add it to the updated_books string
    if (found) {
        updated_books += line1 + '\n';
    }
  }
  books_file.close();
}
else {
  cout << "Unable to open file 'books.txt'" << endl;
  return;
}

// Write the updated_books string to the books.txt file
ofstream books_file_out("books.txt");
if (books_file_out.is_open()) {
  books_file_out << updated_books;
  books_file_out.close();
}
else {
  cout << "Unable to open file 'books.txt'" << endl;
}
}
```

```
void update_book_status(const string& book_name, const string& new_status) {
    ifstream file("books.txt"); // Open the books.txt file for reading
    string updated_books; // To store the updated book records
    string line; // To read each line from the file
    bool found = false; // Flag to check if the book is found in the file
    bool already_in_desired_status = false; // Flag to check if the book is already in the desired status

    // Read the file line by line
    if (file.is_open()) {
        while (getline(file, line)) {
            int pos = line.find(book_name); // Check if the book_name is found in the current line
            if (pos != -1) {
                found = true; // If found, set the flag to true
                int status_pos = line.find(new_status); // Check if the new_status is already in the line
                if (status_pos != -1) {
                    already_in_desired_status = true; // If found, set the flag to true
                }
                else {
                    string old_status = (new_status == "Available") ? "Borrowed" : "Available"; // Determine the old_status based on the new_status
                    status_pos = line.find(old_status); // Find the position of the old_status in the line
                    if (status_pos != -1) {
                        line.replace(status_pos, old_status.length(), new_status); // Replace old_status with new_status in the line
                    }
                }
            }
            updated_books += line + '\n'; // Append the updated line to the updated_books string
        }
        file.close(); // Close the file
    }
    else {
        cout << "Unable to open file." << endl;
```

```
    }


    // If the book is found
    if (found) {
        // If the book is not already in the desired status
        if (!already_in_desired_status) {
            ofstream file("books.txt"); // Open the books.txt file for writing
            if (file.is_open()) {
                file << updated_books; // Write the updated_books string to the file
                file.close(); // Close the file
                cout << "Book " << ((new_status == "Available") ? "returned" : "borrowed") << "
successfully!" << endl;
            }
            else {
                cout << "Unable to open file." << endl;
            }
        }
        else {
            cout << "Book is already " << ((new_status == "Available") ? "available" : "borrowed") << ".
Action rejected." << endl;
            system("pause");
            user_mainmenu(); // Return to the main menu
        }
    }
    else {
        cout << "Book not found." << endl;
    }
}


void search_book_by_name() {
    char first_letter;
    cout << "Enter the first letter of the book you want to search: ";
    cin.ignore();
```

```
    // Validate the user input to ensure it's a single alphabet
    while (true) {
        cin >> first_letter;
        if (is_alpha(first_letter) && cin.peek() == '\n') {
            break;
        }
        else {
            cout << "Please enter a single alphabet: ";
            cin.clear();
            cin.ignore(INT_MAX, '\n');
        }
    }

    ifstream file("books.txt"); // Open the books.txt file for reading
    string line; // To read each line from the file
    bool found = false; // Flag to check if any book is found with the given first letter

    // Read the file line by line
    if (file.is_open()) {
        while (getline(file, line)) {
            // Check if the first letter of the line (book title) matches the user input
            if (toupper(line[0]) == toupper(first_letter)) {
                int title = line.find(","); // Find the position of the first comma (separating the title and
author)
                string book_title = line.substr(0, title); // Extract the book title
                string book_status = line.substr(title + 2); // Extract the book status

                // Print the book information
                cout << "Book title: " << book_title << endl;
                cout << "Status: " << book_status << endl << endl;
                found = true; // Set the flag to true as a book is found
            }
        }
        file.close(); // Close the file
```

```
    }
    else {
        cout << "Unable to open file." << endl;
    }

    // If no book is found with the given first letter
    if (!found) {
        cout << "No books found with the given first letter." << endl;
    }
    return_menu(); // Call the return_menu() function to display the menu again
}

void payment()
{
    char pin[7];
    char payment_way, online_way;
    char loop_payment = 1;
    char loop_credit = 1;
    char loop_online = 1;
    char loop_tng = 1;
    bool loop_promo = false;
    char card_num[17];
    char expiry_date, cvv;
    string card_name, online_display, username_online, password_online;
    string tng;
    string real_promo[3];
    string check_promo;
    int no_promo = 1;

    while (loop_payment == 1)
    {
        cout << "Please select your payment method!" << endl;
        cout << "****************************************\n" << endl;
```

```
cout << "1. Credit Card / Debit Card " << endl;
cout << "2. Online Payment " << endl;
cout << "3. Touch N Go Online Pay " << endl;
cout << "4. Cash Service " << endl;
cout << "5. Promo Code  " << endl;
cin >> payment_way;

if (payment_way == '1') //Credit Card / Debit Card
{
   while (loop_credit == 1) // loop for credit
   {
      cout << "Card Details" << endl;
      cout << "----------" << endl;
      cout << "Card Number (16 digit) : ";
      cin >> card_num;
      cin.ignore();
      cout << "Expiry Date : ";
      cin >> expiry_date;
      cin.ignore();
      cout << "CVV : ";
      cin >> cvv;
      cin.ignore();
      cout << "Name On Card : ";
      getline(cin, card_name); // available for space & blank

      if (strlen(card_num) == 16 && isdigit(expiry_date) && isdigit(cvv)) // only digit is
available in card_num,expiry_date,cvv
      {
         loop_credit = 0; // exit the loop
      }
      else // if non-digit is filled go in this statement
      {
         cout << "Somethings wrong found in Card Details" << endl;
         cout << "Please enter the Card details again!" << endl;
```

```
            loop_credit = 1; // back to loop
        }


    }
    cout << "You had succesfully paid the payment. Thank you !" << endl;
    system("pause");

    loop_payment = 0; // exit payment loop
}

else if (payment_way == '2') //Online Payment
{
    system("cls");
    while (loop_online == 1) //Loop for online payment
    {
        cout << "Please select bank : " << endl;
        cout << "1. Public Bank " << endl;
        cout << "2. Maybank " << endl;
        cout << "3. CIMB Bank " << endl;
        cout << "4. Am Bank " << endl;
        cin >> online_way;

        if (online_way == '1') //Public bank
        {
            online_display = "Public Bank";
            loop_online = 0;
        }

        else if (online_way == '2')// Maybank
        {
            online_display = "May Bank";
            loop_online = 0;
```

```
        }
        else if (online_way == '3') // Cimb Bank
        {
            online_display = "Cimb Bank";
            loop_online = 0;

        }
        else if (online_way == '4') //  Am bank
        {
            online_display = "Am Bank";
            loop_online = 0;

        }

        else
        {
            cout << "Error! Please answer the question with number." << endl;
            loop_online = 1;
        }

        cout << "Welcome to " << online_display << " !" << endl;
        cout << "-----------------------------------------------" << endl;
        cin.ignore();
        cout << "Username:  ";
        getline(cin, username_online);
        cout << "Password:";
        getline(cin, password_online);
        cin.ignore();

        cout << "Login Successfully!" << endl;
        cout << "-------------------------------" << endl;
        cout << username_online << ", you successfully transfer the money through online !" <<
endl;
        cout << "Thanks for using " << online_display << " as your services!" << endl;
```

```
        cout << "The Recipt will printed after the payment is done! " << endl;
        cout << "." << endl;
        cout << "." << endl;
        cout << "." << endl;
        cout << "." << endl;
        cout << "." << endl;
        cout << "." << endl;
        cout << "." << endl;
        cout << "." << endl;
        cout << "." << endl;
        cout << "." << endl;
        cout << "." << endl;
        cout << "." << endl;
        cout << "." << endl;
        cout << "Click enter to proceed the progress....." << endl;
        system("pause");



    }
    loop_payment = 0;
}

else if (payment_way == '3')//Touch N Go Online Pay
{

    while (loop_tng == 1)
    {
        cout << " Please filled in the phone number: " << endl;
        cin >> tng;
        cout << "Succesfully login ! " << endl;
        cout << "Please enter your 6-digit PIN number" << endl;
        cin >> pin;
        if (strlen(pin) == 6)    //only accept 6- digit of TNG PIN
```

```
        {
            cout << "Thanks for using Touch N Go as ur services !" << endl;
            cout << "You had succesfully transfer the money to the receiver!" << endl;
            cout << "The Recipt will printed after the payment is done! " << endl;
            loop_tng = 0;

        }

        else
        {
            cout << "Error ! Re-enter your phone number with correct PIN ." << endl;
            loop_tng = 1;
        }

    }


    loop_payment = 0;
}
else if (payment_way == '4')//Cash Service
{
    cout << "Please move to the counter for further payment." << endl;
    cout << "Thank you for using our services." << endl;
    cout << "The Recipt will be printed out in awhile..... " << endl;
    system("pause");

    loop_payment = 0;
}
else if (payment_way == '5')//  Promo Code
{
    while (!loop_promo)
    {
        cout << "Please enter the secret promo code for free payment: ";
        cin >> check_promo;
```

```
            ifstream promofile;
            promofile.open("promo.txt");



            for (int i = 0; i < 3; i++)
            {
               getline(promofile, real_promo[i]);

               if (real_promo[i] == check_promo) // checking the promocode user type same with the
list or not
               {
                  cout << "Congratulation ! You had earn a free payment ! " << endl;
                  cout << "The Recipt will be printed out in awhile..... " << endl;
                  system("pause");
                  loop_promo = true;
                  break;



               }
            }
            if (!loop_promo) // back loop
            {
               cout << "Please fill in again ! " << endl;
            }

         }
         loop_payment = 0;

      }

      else
      {
         cout << "Error !Please enter a valid answer." << endl;
         loop_payment = 1;
```

```
        }
    }
}
void recipt()
{
    system("Color E4"); //decoration
    system("cls");
    cout << "
********************************************************************************
********************* " << endl;
    cout << "  _____                                  _      _______              _          _   " << endl;
    cout << " |  __  |                                | |    |  ___  |            (_)        | |  " << endl;
    cout << " | |__) |   __ _   _   _   _ __ ___     ___   _ __   | |_    | |__) | ___    ___   _   _ __   | |_ " <<
endl;
    cout << " |  ___/   / _` | | | | | '_ ` _ |   / _ | | '_ | | __|   | _  /  / _ |  / __| | | | '_ | | __|" << endl;
    cout << " | |      | (_| | |_| | | | | | | |  __/ | | | | | |_    | | | | (__  | | | |_) | | | |_ " << endl;
    cout << " |_|       |__,_| |__,_| |_| |_| |_| |___| |_| |_| |_|   |___|  |___| |_| |_| |.__   |__|" << endl;
    cout << "                    __/ |                                                    | |          " << endl;
    cout << "                   |___/                                                     |_|          " << endl;
    cout << "
********************************************************************************
********************* " << endl;
    cout << " **                                                                        **" << endl;
    cout << " **                                                                        **" << endl;
    cout << " **                                                                        **" << endl;
    cout << " **                                                                        **" << endl;
    cout << " **                                                                        **" << endl;
    cout << " **                                                                        **" << endl;
    cout << " **                                                                        **" << endl;
    cout << " **                                                                        **" << endl;
    cout << " **                        Total Price : $ 1                                   **" << endl;
    cout << " **                                                                        **" << endl;
    cout << " **                                                                        **" << endl;
```

```
    cout << " **                                                                **" << endl;
    cout << " **                                                                **" << endl;
    cout << " **                                                                **" << endl;
    cout << " **                                                                **" << endl;
    cout << " **                                                                **" << endl;
    cout << " **                                                                **" << endl;
    cout << " **                                                                **" << endl;
    cout << " **                                                                **" << endl;
    cout << " **                                                                **" << endl;
    cout << " **                                                                **" << endl;
    cout << "
********************************************************************************
********************* " << endl;




}

void penalty()
{
    int date_days;
    int diff_days;
    bool loop_date = false;
    string date, current_date;
    int days, month, year;
    int     max_days = 0;
    int current_days, current_month, current_year;
    int pena_price;

    while (loop_date == false)
    {
        system("cls");
```

```
	cout << "*****************************************************************************************************************************" << endl;
	cout << "  _____                    _  _            _____            _                  " << endl;
	cout << " |  __ |                  | | | |          / ____|          | |                 " << endl;
	cout << " | |__) |  ___   _ __    __ _ | | | |_  _  _  _  | (___    _  _  ___  | |_   ___   _ __ ___  " << endl;
	cout << " |  ___/  / _ | | '_  |  / _` | | | | __| | | | |  |___ | | | | / __| | __| / _ | | '_ ` _ | " << endl;
	cout << " | |     | __/ | | | | | (_| | | | | |_  | |_| |  ____) | | |_| | |__ | | |_ | __/ | | | | | |" << endl;
	cout << " |_|      |___| |_| |_|  |__,_| |_|  |__|  |__, |  |_____/  | __, | |___/ |___|  |___| |_| |_| |_|" << endl;
	cout << "                                              __/ |           __/ |                      " << endl;
	cout << "                                             |___/           |___/                       " << endl;
	cout << "*****************************************************************************************************************************" << endl;


	cout << "Please enter the date of you lended (DD/MM/YYYY) : ";
	getline(cin, date);
	cout << "Please enter the current date (DD/MM/YYYY) :";
	getline(cin, current_date);



	if ((date.length() == 10 && date[2] == '/' && date[5] == '/') && (current_date.length() == 10 && current_date[2] == '/' && current_date[5] == '/'))
	{
		days = stoi(date.substr(0, 2));// take the 1st and 2nd word inside the date to make days
		month = stoi(date.substr(3, 2));// take the 3th and 4th word inside the date to make days
		year = stoi(date.substr(6, 4));// take the 6th,7th,8th and 10th word inside the date to make days
		current_days = stoi(current_date.substr(0, 2)); // take the 1st and 2nd word inside the current_date to make days
		current_month = stoi(current_date.substr(3, 2));// take the 3th and 4th word inside the current_date to make days
		current_year = stoi(current_date.substr(6, 4));// take the 6th,7th,8th and 10th word inside the current_date to make days
		if ((days <= 31 && days >= 1 && month >= 1 && month <= 12 && year > 0) && (current_days <= 31 && current_days >= 1 && current_month >= 1 && current_month <= 12 && current_year > 0))
```

```
{

    if ((month == 1 || month == 3 || month == 5 || month == 7 || month == 8 || month == 10 ||
month == 12) && (current_month == 1 || current_month == 3 || current_month == 5 || current_month
== 7 || current_month == 8 || current_month == 10 || current_month == 12))
    {
       max_days = 31;
       loop_date = true;

    }

    else if (month == 2 && current_month == 2)
    {
       max_days = 28;
       loop_date = true;

    }

    else if ((month == 4 || month == 6 || month == 9 || month == 11) && (current_month == 4 ||
current_month == 6 || current_month == 9 || current_month == 11))
    {
       max_days = 30;
       loop_date = true;

    }



    loop_date = true;
    // counting the difference of day with the date that user proviced depending on the month
    date_days = days + (month - 1) * 31 + (year - 1) * 12 * 31;
    current_days = current_days + (current_month - 1) * 31 + (current_year - 1) * 12 * 31;
    diff_days = current_days - date_days;

}
else
```

```
        {
            cout << "Please give a valid date!" << endl;
            system("pause");
            loop_date = false; // back to loop



        }

    }
    else
    {
        cout << "Please enter again the date with right format ! (DD/MM/YYYY) " << endl;
        system("pause");
        loop_date = false; // back to the loop



    }



}
cout << "Request Accepted !" << endl;
cout << " The calculation is proceeding to calculate the funds." << endl;
system("pause");
cout << "Difference in days: " << diff_days << endl;
if (diff_days >= 7)
{
    diff_days = diff_days * 1;
}
else if (diff_days <= 0)
{
    cout << "Error" << endl;

}
else if (diff_days > 0 && diff_days < 7) // no peanlty if the day is less than 7
```

```
    {
        cout << "Your books have no penalty since the limited day is 7 days." << endl;

    }
    system("pause");
    pena_recipt(diff_days);

}

void pena_recipt(int& diff_days)
{



    system("Color E4"); // decoration
    system("cls");
    cout << "
********************************************************************************
********************* " << endl;
    cout << "  _____                                  _      _______               _          _   " << endl;
    cout << " |  __  |                                ||    |  __  |             (_)        ||  " << endl;
    cout << " | |__) |   __ _   _   _   _ __ ___     ___   _ __   ||_    ||__) | ___    ___   _   _ __   ||_ " <<
endl;
    cout << " |  ___/   / _` | | | | | '_ ` _ |   / _ | | '_ | |  __|   |  _  /  / _ |  / __| | | | '_ | |  __|" << endl;
    cout << " | |      | (_| | |_| | | | | | | | __/ | | | | |_    | | | |  __/ | (__  | | | |_) | | | |_ " << endl;
    cout << " |_|       |__,_| |__,_| |_| |_| |_| |___| |_| |_||_|   __|     |_| |_|   |____|  |___| |_| |_.__   |__|" << endl;
    cout << "                    __/ |                                                  | |          " << endl;
    cout << "                   |___/                                                   |_|          " << endl;
    cout << "
********************************************************************************
********************* " << endl;
    cout << " **                                                                      **" << endl;
    cout << " **                                                                      **" << endl;
    cout << " **                                                                      **" << endl;
    cout << " **                                                                      **" << endl;
    cout << " **                                                                      **" << endl;
    cout << " **                                                                      **" << endl;
```

```
    cout << " **                                                          **" << endl;
    cout << " **                                                          **" << endl;
    cout << " **                    Total Price : $ " << diff_days << "
**" << endl;
    cout << " **                                                          **" << endl;
    cout << " **                                                          **" << endl;
    cout << " **                                                          **" << endl;
    cout << " **                                                          **" << endl;
    cout << " **                                                          **" << endl;
    cout << " **                                                          **" << endl;
    cout << " **                                                          **" << endl;
    cout << " **                                                          **" << endl;
    cout << " **                                                          **" << endl;
    cout << " **                                                          **" << endl;
    cout << " **                                                          **" << endl;
    cout << " **                                                          **" << endl;
    cout << "
*********************************************************************************
********************* " << endl;




    system("pause");
}


void sensor()
{


    int seat[24] = { 0 }; // Array to keep track of the occupancy status of parking spaces, 0 = empty, 1 =
occupied
    int availableseat = 24; // The total number of available parking spaces
    char chosenseat; // The parking space chosen by the user
    int loop_seat = 0;
    // Main loop for the parking guidance system
```

```
   while (true)
   {
      system("cls");
      // Display the current status of the parking spaces
      cout <<
"*********************************************************************************************
*************** " << endl;
      cout << "   _____                                 _____              _                  " << endl;
      cout << "  / ____|                               / ____|            | |                 " << endl;
      cout << " | (___    ___   _ __    ___    ___    _ __    | (___    _  _  ___  | |_   ___   _ __ ___  " <<
endl;
      cout << "  |___ |  / _ | | '_ |  / __|  / _ | | '__|   |___ |  | | | | / __| | __|  / _ | | '_ ` _ | " << endl;
      cout << "  ____) | | __/ | | | | |__ | | (_) | | |      ____) | | |_| | |__ | | |_  | __/ | | | || |" << endl;
      cout << " |_____/   |___| |_| |_| |___/  |___/  |_|     |_____/  | __, | |___/ |___|  |___|  |_| |_||_|" << endl;
      cout << "                                              __/ |                        " << endl;
      cout << "                                             |___/                         " << endl;
      cout <<
"*********************************************************************************************
*************** " << endl;




      cout << "Current seat status: \n\n" << endl;


      cout << " ==================================" << endl;
      cout << " 0 = empty, 1 = occupied" << endl;
      cout << " ==================================" << endl;


      cout << "
=========================================================================
============" << endl;
      cout << " |               |               |               |               |" << endl;
      cout << " |               |               |               |               |" << endl;
      cout << " |       A       |       B       |       C       |       D       |" << endl;
```

```
	cout << " |	" << seat[0] << "	|	" << seat[1] << "	|	" << seat[2] << "
|	" << seat[3] << "	| " << endl;

	cout << " |		|		|		|		|" << endl;

	cout << " |		|		|		|		|" << endl;

	cout << "
==========================================================================
============" << endl;

	cout << " |		|		|		|		|" << endl;

	cout << " |		|		|		|		|" << endl;

	cout << " |	E	|	F	|	G	|	H	|" << endl;

	cout << " |	" << seat[4] << "	|	" << seat[5] << "	|	" << seat[6] << "
|	" << seat[7] << "	| " << endl;

	cout << " |		|		|		|		|" << endl;

	cout << " |		|		|		|		|" << endl;

	cout << "
==========================================================================
============" << endl;


	cout << endl;

	cout << endl;

	cout << endl;


	cout << "
==========================================================================
============" << endl;

	cout << " |		|		|		|		|" << endl;

	cout << " |		|		|		|		|" << endl;

	cout << " |	I	|	J	|	K	|	L	|" << endl;

	cout << " |	" << seat[8] << "	|	" << seat[9] << "	|	" << seat[10] << "
|	" << seat[11] << "	| " << endl;

	cout << " |		|		|		|		|" << endl;

	cout << " |		|		|		|		|" << endl;

	cout << "
==========================================================================
============" << endl;

	cout << " |		|		|		|		|" << endl;

	cout << " |		|		|		|		|" << endl;

	cout << " |	M	|	N	|	O	|	P	|" << endl;
```

```
	cout << " |	" << seat[12] << "	|	" << seat[13] << "	|	" << seat[14] << "
|	" << seat[15] << "	| " << endl;

	cout << " |		|		|		|		|" << endl;

	cout << " |		|		|		|		|" << endl;

	cout << "
==========================================================================
============" << endl;


	cout << endl;

	cout << endl;

	cout << endl;

	cout << "
==========================================================================
============" << endl;

	cout << " |		|		|		|		|" << endl;

	cout << " |		|		|		|		|" << endl;

	cout << " |	Q	|	R	|	S	|	T	|" << endl;

	cout << " |	" << seat[16] << "	|	" << seat[17] << "	|	" << seat[18] << "
|	" << seat[19] << "	| " << endl;

	cout << " |		|		|		|		|" << endl;

	cout << " |		|		|		|		|" << endl;

	cout << "
==========================================================================
============" << endl;

	cout << " |		|		|		|		|" << endl;

	cout << " |		|		|		|		|" << endl;

	cout << " |	U	|	V	|	W	|	X	|" << endl;

	cout << " |	" << seat[20] << "	|	" << seat[21] << "	|	" << seat[22] << "
|	" << seat[23] << "	| " << endl;

	cout << " |		|		|		|		|" << endl;

	cout << " |		|		|		|		|" << endl;

	cout << "
==========================================================================
============" << endl;



	cout << endl;
```

```
// Prompt the user to choose a seat space
cout << "Please enter the number of the seat space you would like to sit (A-X)(Z to exit): ";

cin >> chosenseat;
while (loop_seat == 0)
{
   if (isdigit(chosenseat))
   {
      cout << "Please enter (A-X) !" << endl;
      system("pause");
      loop_seat = 0;
      break;
   }
   else if (chosenseat == 'Z' || chosenseat == 'z')
   {
      member();
      break;



   }
   else
   {
      chosenseat = toupper(chosenseat);
      int spaceindex = chosenseat - 'A';

      // Check if the chosen space is already occupied
      if (seat[spaceindex] == 1)
      {
         cout << "Sorry, that space is already occupied. Please choose another space." << endl;
         system("pause");
         break;
      }
```

```
            else
            {
                // Mark the chosen space as occupied and update the available spaces count
                seat[spaceindex] = 1;
                availableseat--;

                // Display a message indicating the chosen space and the number of available spaces
                cout << "You have register in seat " << spaceindex << ". There are " << availableseat <<
" spaces available." << endl;
                system("pause");
                break;

                // Check if all parking spaces are now occupied
                if (availableseat == 0) {
                    cout << "All seat spaces are now occupied. Please come back later." << endl;
                    system("pause");
                    break;

                }
            }
            loop_seat = 1;
        }

    }
  }



}

void showOptions(string& ADMIN_name)
{
   system("cls");
   cout << "Welcome back admin " << ADMIN_name << " !\nWhat would you like to do today?\n"
<< endl;
```

```
    cout << "************************************************" << endl;
    cout << " _____  _____  _____  _____  _____  _____  _____ " << endl;
    cout << "|     ||  _  ||_   _||     ||     ||  ||  __|" << endl;
    cout << "|  |  ||   __|  ||  |-   -||  |  ||  |  ||__   |" << endl;
    cout << "|_____||__|     |_|  |_____||_____||_|___||_____|" << endl;
    cout << "************************************************" << endl;
    cout << "*1. SEARCH BOOKS                          *" << endl;
    cout << "*2. VIEW BOOKS                           *" << endl;
    cout << "*3. ADD BOOKS                            *" << endl;
    cout << "*4. DELETE BOOKS                         *" << endl;
    cout << "*5. LOGOUT                               *" << endl;
    cout << "************************************************" << endl;
}

void view()
{
    system("cls");
    cout <<
"*****************************************************************************************
*********************" << endl;
    cout << " _   _   _____   _____   _   _    _____   _____   _____   _   __   _      _____   _____
_____  " << endl;
    cout << "|  |  |  |_   _|  |  ___|  |  |  |  ___|  |  _  |  |  _  |  |  |  / /  |  |    |_   _|  / ___|  |_   _| " << endl;
    cout << "|  |  |  |  | |    |  |__   |  |  |  |_   |  |  |  |  |  |  |  | / /   |  |      |  |  `--.    |  |   " << endl;
    cout << "|  |  |  |  | |    |   __|  |  |  |  _|  |  |  |  |  |  |  |  |/ /    |  |      |  |    `--. |  |   " << endl;
    cout << "|  |_/ /  _| |_   |  |___   \  /   |  |_  |  |_/ /  |  |_/ /  |  |\ \   |  |____  _| |_  /\__/ /   |  |   " << endl;
    cout << "|_____/   |___/   |_____|   \/    |_____| |_____/  |_____/  |_|  \_\  |_____/  |_____| |_____/    |_|   " <<
endl;
    cout <<
"*****************************************************************************************
*********************" << endl;
}
```