National University of Computer and Emerging Sciences

MINIPROJECT

FOR

DATABASE SYSTEMS

Student Name	Muhammad Ibraheem Noor
Roll No.	21F-9068
Section	BSCS 4th (A)
Lab Instructor(s)	Ms. Saba Naseem
Semester	Spring 2022

FAST School of Computing

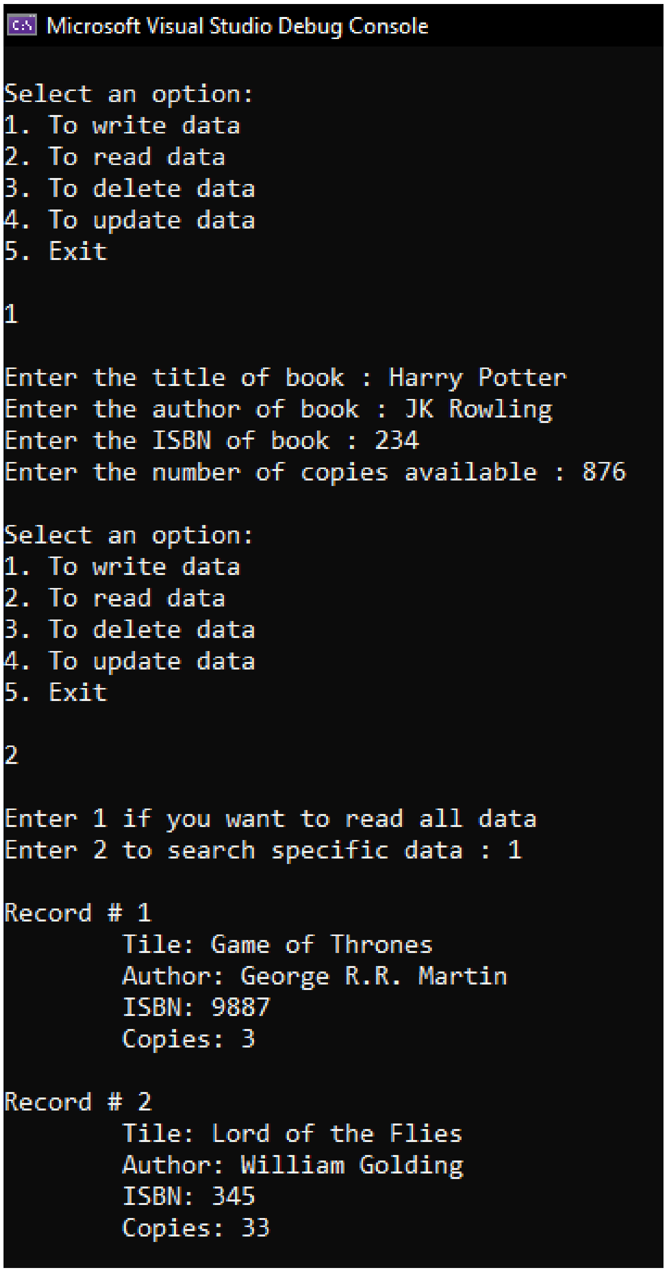
MINIPROJECT: DATABASE

CODE:

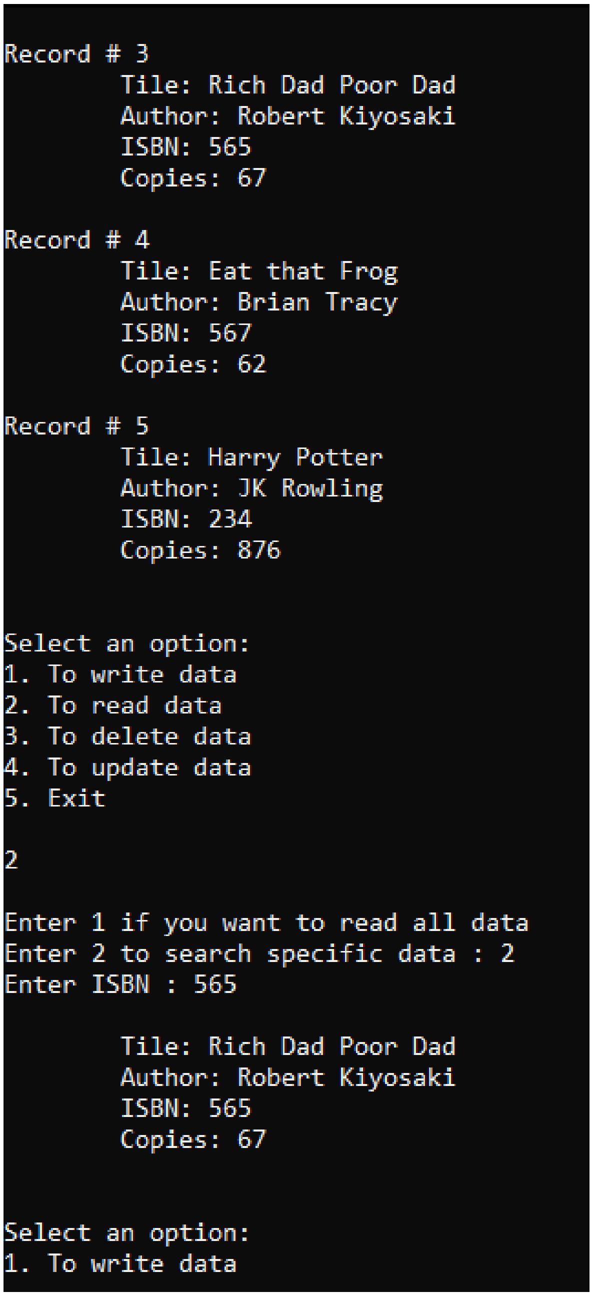
```
#include<iostream>
#include<string>
#include<fstream>
using namespace std;
struct node
	//title, author,ISBN, and number of copies available
{
	string title, author;
	int ISBN, num_copy;
	struct node* next;
	struct node* prev;
	node()
	{
		next = NULL;
		prev = NULL;
	}
};
class double_llist
{
public:
	node* head;
	node* current_ptr;

	double_llist()
	{
		head = NULL;
		current_ptr = NULL; //constructor for my class
	}
	~double_llist()
	{
		current_ptr = head;
		while (current_ptr != NULL)
		{
			node* next = current_ptr->next;
			delete current_ptr;
			current_ptr = next;

		}
	}
	//1
	void insert_at_begin(int ISBN, int num_copy, string title, string author) // Adding a new node
	{
        node* temp = new node;
        temp->ISBN = ISBN;
        temp->num_copy = num_copy;
        temp->title = title;
        temp->author = author;
        temp->next = NULL;
        temp->prev = NULL;
        if (isempty())
        {
            head = current_ptr = temp;
        }
        else
        {
            current_ptr->next = temp;
            temp->prev = current_ptr;
            current_ptr = current_ptr->next;
        }
    }
    //4
    void delete_at_begin() // deleting the first node
    {
        node* temp = head;
        temp = head;
        head = temp->next;
        temp->next->prev = NULL;
        delete temp;
    }
    //6
    void delete_at_end() // deleting the last node
    {
        node* temp = head;
        temp = head;
        while (temp->next != NULL)
        {
            temp = temp->next;
        }
        temp->prev->next = NULL;
        current_ptr = temp->prev;
        delete temp;
    }
    void delete_at_middle(int ISBN)
    {
        node* temp = head;
        temp = head;
        while (temp->ISBN != ISBN)
        {
            temp = temp->next;
        }
        temp->prev->next = temp->next;
        temp->next->prev = temp->prev;
        delete temp;
    }
    void display_dlist() // displaying list
    {
        node* current_pt;
        current_pt = head;
        int i = 0;
        while (current_pt != NULL)
        {
            i++;
            cout << "\nRecord # " << i << "\n\tTile: " <<
current_pt->title << "\n\tAuthor: " << current_pt->author << "\n\tISBN: "
<< current_pt->ISBN << "\n\tCopies: " << current_pt->num_copy << endl;
            current_pt = current_pt->next;
        }
        cout << endl;
    }
    void search_ISBN(int ISBN)
    {
        node* current_pt;
        current_pt = head;
        while (current_pt != NULL)
        {
            if (current_pt->ISBN == ISBN)
            {
                cout << "\n\tTile: " << current_pt->title <<
"\n\tAuthor: " << current_pt->author << "\n\tISBN: " << current_pt->ISBN
<< "\n\tCopies: " << current_pt->num_copy << endl;
            }
            current_pt = current_pt->next;
        }
        cout << endl;
    }
    void update_ISBN(int ISBN)
    {
        int choice;
        node* current_pt;
        current_pt = head;
        while (current_pt != NULL)
        {
            if (current_pt->ISBN == ISBN)
            {
                cout << "\nPrevious data : \n";
                cout << "\n\tTile: " << current_pt->title <<
"\n\tAuthor: " << current_pt->author << "\n\tISBN: " << current_pt->ISBN
<< "\n\tCopies: " << current_pt->num_copy << endl;
```

```
                cout << "\nTo update title press 1\nTo update 
author name press 2\nTo update ISBN press 3\nTo update number of copies 
press 4\n";
                cin >> choice;
                if (choice == 1)
                {
                    string title;
                    cout << "Enter the new title : ";
                    cin.ignore();
                    getline(cin, title);
                    current_pt->title = title;
                }
                else if (choice == 2)
                {
                    string author;
                    cout << "Enter the new author : ";
                    cin >> author;
                    current_pt->author = author;
                }
                else if (choice == 3)
                {
                    int ISBN;
                    cout << "Enter the new ISBN : ";
                    cin >> ISBN;
                    current_pt->ISBN = ISBN;
                }
                else if (choice == 4)
                {
                    int num_copy;
                    cout << "Enter the new number of copies : ";
                    cin >> num_copy;
                    current_pt->num_copy = num_copy;
                }
            }
            current_pt = current_pt->next;
        }
        cout << endl;
    }
    void deletion(int ISBN)
    {
        node* current_pt;
        node* temp = new node;
        current_pt = head;
        while (current_pt != NULL)
        {
            if (current_pt->ISBN == ISBN)
            {
                cout << "\n\tTile: " << current_pt->title <<
"\n\tAuthor: " << current_pt->author << "\n\tISBN: " << current_pt->ISBN
<< "\n\tCopies: " << current_pt->num_copy << endl;
                cout << "\tRecord deleted\n";
                temp = current_pt;
            }
            current_pt = current_pt->next;
        }
        if (temp != NULL)
        {
            if (temp->next == NULL)
            {
                delete_at_end();
            }
            else if (temp->prev == NULL)
            {
                delete_at_begin();
            }
            else
            {
                delete_at_middle(ISBN);
            }
        }
        cout << endl;
    }
    bool isempty() // Checking if the list is empty
    {
        return (head == NULL);
    }

};
class fileHandling
{
    fstream file;
    string title, author;
    int num_copy, ISBN;
    double_llist obj_ll;
public:
    fileHandling()
    {
        file.open("lib_data.txt", ios::in);
        if (!file)
        {
            return;
        }
        while (!file.eof())
        {
            node* temp = new node;
            getline(file, temp->title);
            if (temp->title == "")
            {
                break;
            }
            getline(file, temp->author);
            file >> temp->ISBN;
            file >> temp->num_copy;
            file.ignore();
            file.ignore();
            if (obj_ll.isempty())
            {
                obj_ll.head = obj_ll.current_ptr = temp;
            }
            else
            {
                obj_ll.current_ptr->next = temp;
                temp->prev = obj_ll.current_ptr;
                obj_ll.current_ptr = obj_ll.current_ptr->next;
            }
        }
        file.close();
    }
    void write()
    {
        file.open("lib_data.txt", ios::app);
        if (!file)
        {
            cout << "File creation failed" << endl;
        }
        else
        {
            //title, author,ISBN, and number of copies available
            cout << "Enter the title of book : ";
            getline(cin, title);
            cout << "Enter the author of book : ";
            getline(cin, author);
            cout << "Enter the ISBN of book : ";
            cin >> ISBN;
            cout << "Enter the number of copies available : ";
            cin >> num_copy;
            file << title << endl << author << endl << ISBN << endl
<< num_copy << endl << endl;    //Writing to file
            obj_ll.insert_at_begin(ISBN, num_copy, title, author);
            file.close();
        }
    }
    void read()
    {
        int choice = 0;
        cout << "Enter 1 if you want to read all data\nEnter 2 to
search specific data : ";
        cin >> choice;

        while (choice != 1 && choice != 2)
        {
            cout << "Input again\nEnter 1 if you want to read all
data\nEnter 2 to search specific data\nINPUT : ";
            cin >> choice;
        }
        if (choice == 1)
        {
            obj_ll.display_dlist();
        }
        else if (choice == 2)
        {
            node* tempo = new node;
            tempo = obj_ll.head;
            cout << "Enter ISBN : ";
            cin >> ISBN;
            while (tempo != NULL)
            {
                if (tempo->ISBN == ISBN)
                {
                    obj_ll.search_ISBN(ISBN);
                    break;
                }
                else if (tempo->ISBN != ISBN && tempo->next ==
NULL)
                {
                    cout << "\n\tINVALID ISBN\n";
                }
                tempo = tempo->next;
            }

        }
    }
    void update()
    {
        node* tempo = new node;
        tempo = obj_ll.head;
        node* current_pt = new node;
        cout << "Enter the ISBN number of book: ";
        cin >> ISBN;
        while (tempo != NULL)
        {
            if (tempo->ISBN == ISBN)
            {
                obj_ll.update_ISBN(ISBN);
                file.open("lib_data.txt", ios::out);
                current_pt = obj_ll.head;
                while (current_pt != NULL)
                {
                    file << current_pt->title << endl << 
current_pt->author << endl << current_pt->ISBN << endl << current_pt-
>num_copy << endl << endl;    //Writing to file
                    current_pt = current_pt->next;
                }
                break;
            }
            else if (tempo->ISBN != ISBN && tempo->next == NULL)
            {
                cout << "\n\tINVALID ISBN\n";
                return;
            }
            tempo = tempo->next;
        }

        file.close();
    }
    void delete_data()
    {
        node* tempo = new node;
        tempo = obj_ll.head;
        node* current_pt = new node;
        file.open("lib_data.txt", ios::out);
        cout << "Enter the ISBN number of book to delete: ";
        cin >> ISBN;
        while (tempo != NULL)
        {
            if (tempo->ISBN == ISBN)
            {
                obj_ll.deletion(ISBN);
                current_pt = obj_ll.head;
                while (current_pt != NULL)
                {
                    file << current_pt->title << endl << 
current_pt->author << endl << current_pt->ISBN << endl << current_pt-
>num_copy << endl << endl;    //Writing to file
                    current_pt = current_pt->next;
                }
                break;
            }
            else if (tempo->ISBN != ISBN && tempo->next == NULL)
            {
                cout << "\n\tINVALID ISBN\n";
                return;
            }
            tempo = tempo->next;
        }

        file.close();
    }
};
int main()
{
    fileHandling obj;
    int opt=0;
    while (opt != 5)
    {
        cout << "\nSelect an option: \n";
        cout << "1. To write data \n";
        cout << "2. To read data \n";
        cout << "3. To delete data \n";
        cout << "4. To update data \n";
        cout << "5. Exit \n\n";
        cin >> opt;
        cout << endl;
            switch (opt)
            {
                case 1:
                    cin.ignore();
                    obj.write();
                    break;
                case 2:
                    obj.read();
                    break;
                case 3:
                    obj.delete_data();
                    break;
                case 4:
                    obj.update();
                    break;
                case 5:
                    break;
                default:
                    cout << "Select right option\n\n";
            }
    }
}
```

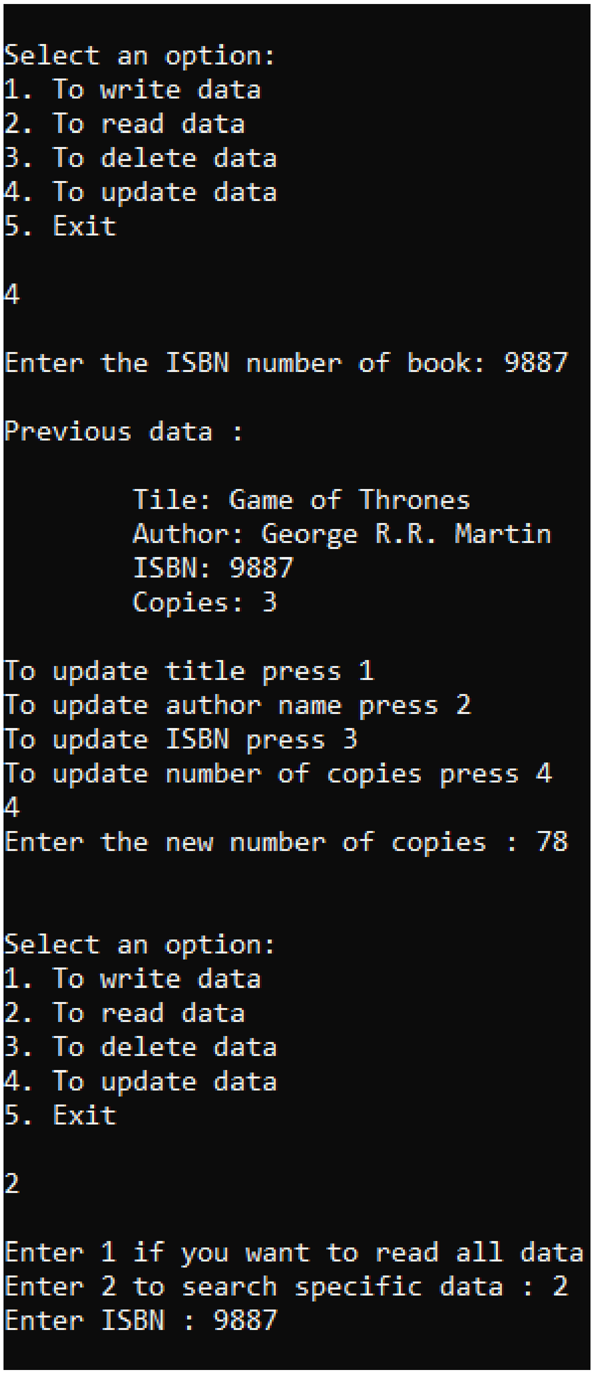
OUTPUT:



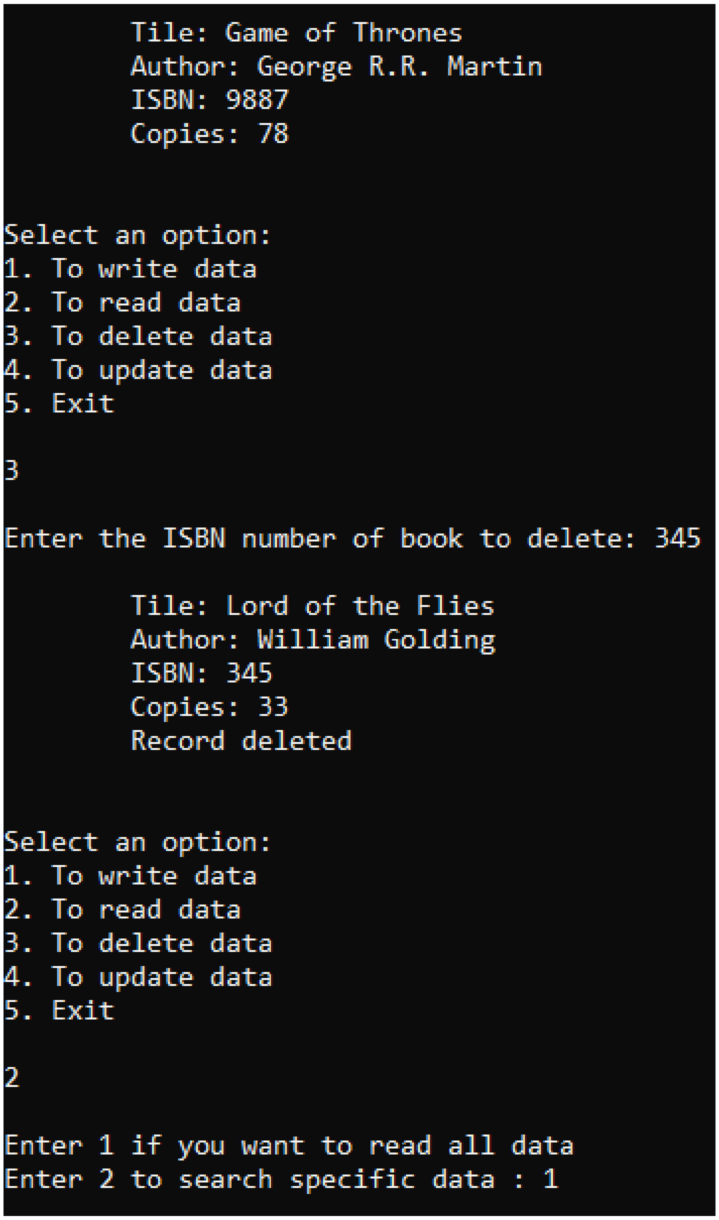
```
Select an option:
1. To write data
2. To read data
3. To delete data
4. To update data
5. Exit

1

Enter the title of book : Harry Potter
Enter the author of book : JK Rowling
Enter the ISBN of book : 234
Enter the number of copies available : 876

Select an option:
1. To write data
2. To read data
3. To delete data
4. To update data
5. Exit

2

Enter 1 if you want to read all data
Enter 2 to search specific data : 1

Record # 1
        Tile: Game of Thrones
        Author: George R.R. Martin
        ISBN: 9887
        Copies: 3

Record # 2
        Tile: Lord of the Flies
        Author: William Golding
        ISBN: 345
        Copies: 33
```



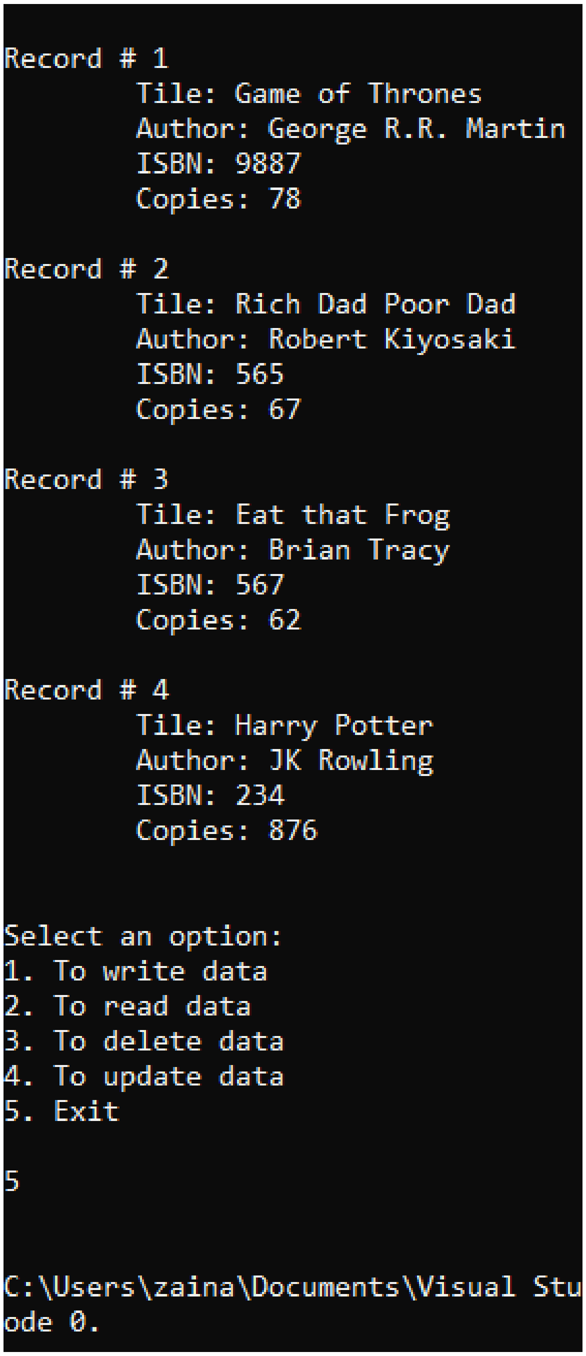
```
Record # 3
        Tile: Rich Dad Poor Dad
        Author: Robert Kiyosaki
        ISBN: 565
        Copies: 67

Record # 4
        Tile: Eat that Frog
        Author: Brian Tracy
        ISBN: 567
        Copies: 62

Record # 5
        Tile: Harry Potter
        Author: JK Rowling
        ISBN: 234
        Copies: 876


Select an option:
1. To write data
2. To read data
3. To delete data
4. To update data
5. Exit

2

Enter 1 if you want to read all data
Enter 2 to search specific data : 2
Enter ISBN : 565

        Tile: Rich Dad Poor Dad
        Author: Robert Kiyosaki
        ISBN: 565
        Copies: 67


Select an option:
1. To write data
```



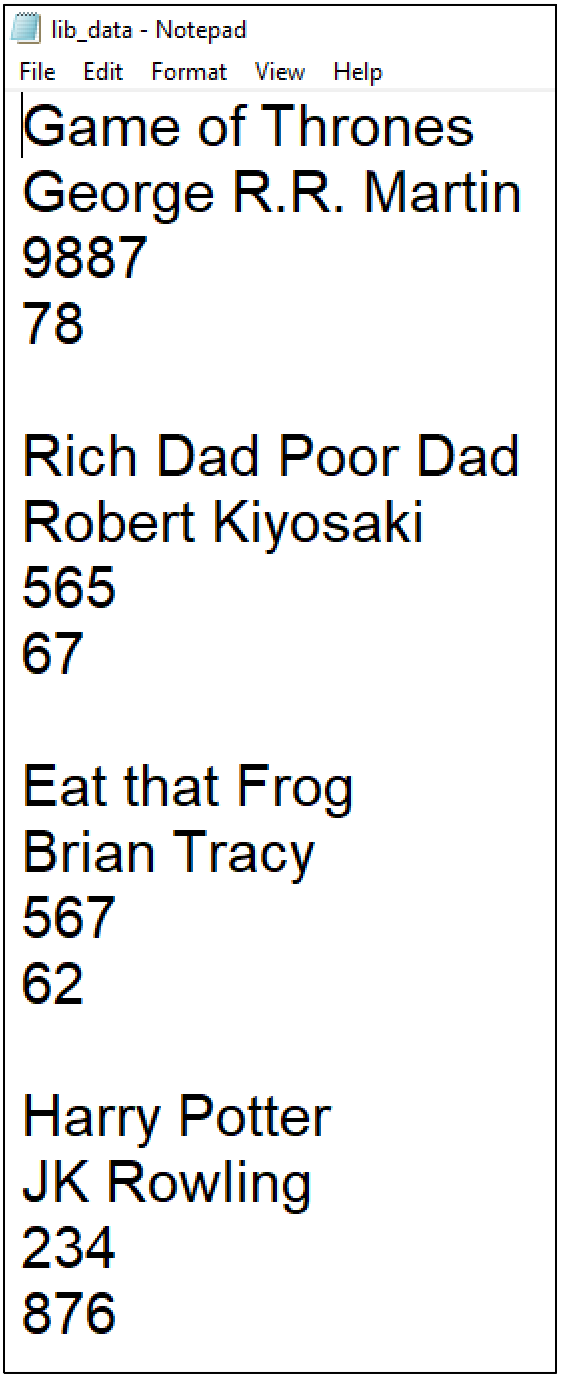
```
Select an option:
1. To write data
2. To read data
3. To delete data
4. To update data
5. Exit

4

Enter the ISBN number of book: 9887

Previous data :

        Tile: Game of Thrones
        Author: George R.R. Martin
        ISBN: 9887
        Copies: 3

To update title press 1
To update author name press 2
To update ISBN press 3
To update number of copies press 4
4
Enter the new number of copies : 78


Select an option:
1. To write data
2. To read data
3. To delete data
4. To update data
5. Exit

2

Enter 1 if you want to read all data
Enter 2 to search specific data : 2
Enter ISBN : 9887
```



```
        Tile: Game of Thrones
        Author: George R.R. Martin
        ISBN: 9887
        Copies: 78


Select an option:
1. To write data
2. To read data
3. To delete data
4. To update data
5. Exit

3

Enter the ISBN number of book to delete: 345

        Tile: Lord of the Flies
        Author: William Golding
        ISBN: 345
        Copies: 33
        Record deleted


Select an option:
1. To write data
2. To read data
3. To delete data
4. To update data
5. Exit

2

Enter 1 if you want to read all data
Enter 2 to search specific data : 1
```



```
Record # 1
        Tile: Game of Thrones
        Author: George R.R. Martin
        ISBN: 9887
        Copies: 78

Record # 2
        Tile: Rich Dad Poor Dad
        Author: Robert Kiyosaki
        ISBN: 565
        Copies: 67

Record # 3
        Tile: Eat that Frog
        Author: Brian Tracy
        ISBN: 567
        Copies: 62

Record # 4
        Tile: Harry Potter
        Author: JK Rowling
        ISBN: 234
        Copies: 876


Select an option:
1. To write data
2. To read data
3. To delete data
4. To update data
5. Exit

5


C:\Users\zaina\Documents\Visual Stu
ode 0.
```



```
Game of Thrones
George R.R. Martin
9887
78

Rich Dad Poor Dad
Robert Kiyosaki
565
67

Eat that Frog
Brian Tracy
567
62

Harry Potter
JK Rowling
234
876
```

Notepad File