DATA STRUCTURES IN C

A PROJECT REPORT ON

BOOKSTORE MANAGEMENT SYSTEM

SUBMITTED BY

Name of the students	PRN
Parth Nikam	20070123120
Jude Praneet Maria	20070123105
Pratham Yogesh Talekar	20070123113
Satvik Sasidhar Sattenapalli	20070123089

SYMBIOSIS INSTITUTE OF TECHNOLOGY

A CONSTITUENT OF SYMBIOSIS INTERNATIONAL (DEEMED UNIVERSITY)

Pune - 412115

2021-22

Aim:

The aim of this project is to emulate a system comprising of the functions of a working bookstore management set-up in a console application environment.

Program Objectives :

- Create a basic yet navigable user interface for the system.
- Implement “add items to inventory/stock.”
- Implement “View list of items in inventory/stock.”
- Implement “Highest Rated Book in the inventory.”
- Start-up and Exit Procedures.

Program Details :

- Number of User Defined Functions : Nine
- Constructs Used : Switch Case and If Constructs.
- Libraries Used : stdio and string
- Data Structures Used : 2 (Including Read and Unread)
- Predefined Functions Used : fprintf, scanf, fopen, fclose, fread, fwrite, strcpy, strcmp.

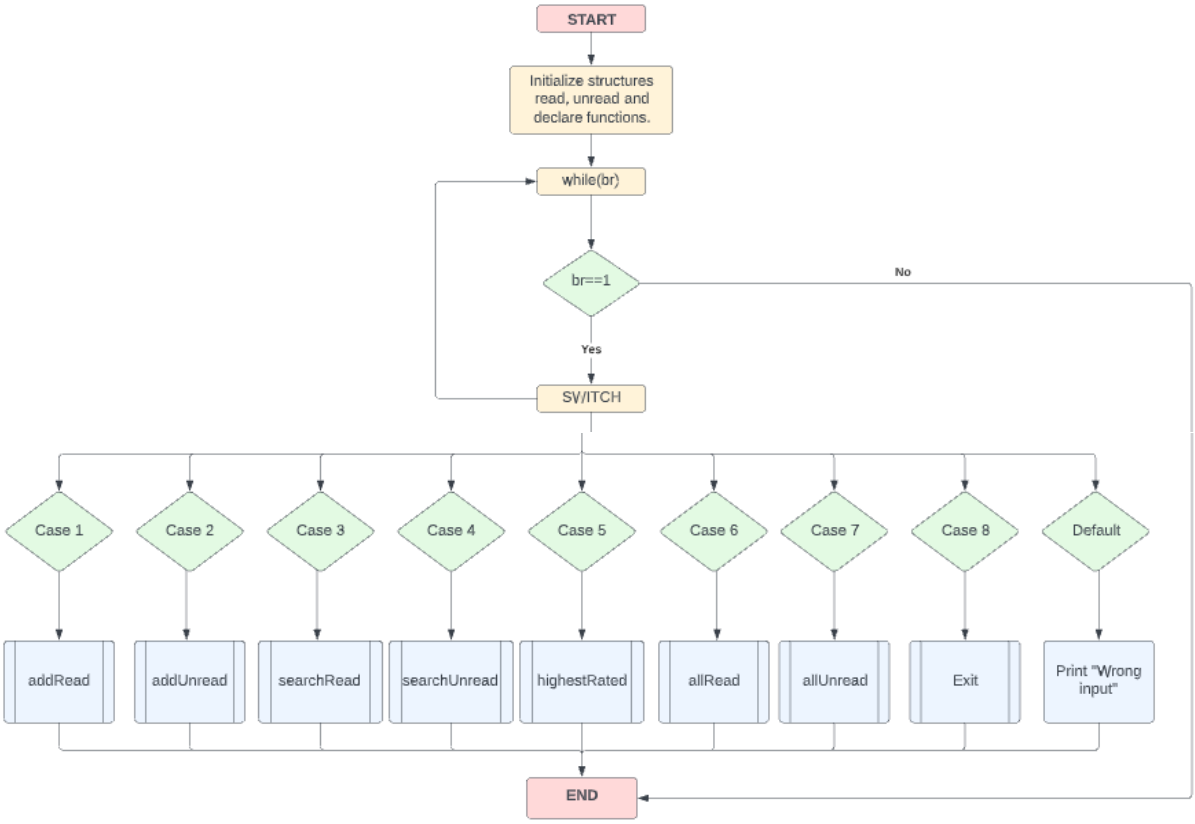
Introduction

The mini-project "Book Store management system project in C" is a console application using the C programming language. *The aim of this project is to simulate the working i.e., all the functions of a real world bookstore – hence the name 'Bookstore Management System' through the mode of a console application*. The code for this program is designed to perform operations such as Inserting a new book name/author/rating to a read/unread list and perform the delete operation for the same, in addition to this, the program also facilitates the viewing of all of the items in the list of read/unread books.

This particular application of Data Structures (in C) was specifically chosen by our group considering current trend data and other related info. Post the severity of the Coronavirus Pandemic, several old retail practices have suffered a considerable pitfall in terms of sales figures and even to the point of being closed. In an article by VOX media, the percentage of imminent threat to closing of *independent bookstores* in terms of being shut down was estimated to an *aggregate of twenty*. This is despite the functioning of several renowned charitable foundations.[1] Hence, we decided to build a console application system wherein, bookstores can add the data pertaining to the stock of books in-store or in their respective inventories.

Therefore, the system proposed in the project is that of the same functionality. In this program, we utilize techniques such as file handling to store data in .txt files. This stored data is further extracted from the same .txt files by the use of file pointers, file based command (like fread and fwrite) and the accessed using structs. In this way, the code of this program functions. On the front end side of things, a switch case set-up has been designed for ease of operation wherein, each case pertains to each distinct functionality of this system and there are ten cases. In each case methods for the dedicated task has been called (with/without inputs) thus fulfilling the motive of that particular case. In this manner, the whole program depicts the functionalities of a working set-up thus being one itself.

Flowchart



Algorithm

```
Start
 Declare and define functions
 Define struct read and unread
 Set variables br to 1 and choice = 0
 while (br)
  case 1 :
        call addRead()
        break

  case 2 :
        call addUnread()
        break

  case 3 :
        call search()
        break

  case 4 :
        call searchUnread()
        break

  case 5 :
        call highestRated()
        break
  case 6 :
        call allRead()
        break
  case 7:
        call allUnread()
        break
  default
  END
```

Algorithms for Functions

search()

Start

Define char sk[30]

Declare file pointer

Declare structure to store the data

Open file in read mode

If file does not exists, print error

While loop to read till end of the file

if input given to search within the file

print searching and print name,author,genre,rating

else

print no book found

Close file

End

addUnread()

Start

Open file in append mode

If outfile is null

print error opening file

Declare structure

print enter the details of book you want to read

print name

print author

print genre

Declare structure to write

print contents to file written succesfully

else

print error

Close file

End

search()

Start

Define char sk[30]

Declare file pointer

Declare structure to store the data

Open file in read mode

If file does not exists, print error

While loop to read till end of the file

if input given to search within the file

print searching and print name,author and genre

else

print no book found

Close file

Highest()

Start

Set highest to 0

set max character to 30

Declare structure

Open file in read mode

If inf = null

print error opening file

While loop to read till end of the file

If input stars is greater than highest, input stars = highest stars

set bookname = inp.name

Close file

print highest rated book

print rating

End

Allread()

Start

Declare file pointer

Open file (read.txt) in read mode

Declare structure to read data (from read)

While loop to read till end of file

print name, author, rating .....

End

Allunread()

Start

Declare file pointer

Define char filename [100]

Open file (unread.txt) in read mode

Declare structure to read data (from unread)

While loop to read till end of file

Print name,author and genre.

End

Code

```
#include <stdio.h>
#include <string.h>

/*
   Functions:

   add to read (done)
   add to unread (done)
   seaarch using name (done)
   find highest rated (done)
   add form unread to read
   delete from read (done)
   delete form unread (done)
   view all read
   view all unread

    */

//all functions declared here
void addRead();
void addUnread();
void search(char sk[30]);
void searchUnread(char sk[30]);
void deleteUnread();
int highestRated();
void deleteRead(char sk[30]);
//void addUnrRea();
void allRead();
void allUnread();
void addUnrRea(char sk[30]);

//struct for books that have been read
//includes name, author, rating, and genre
struct read{

    char name[30];
    char author[30];
    char genre[20];
    int stars;

};
```

```
//struct for books that have not been read
//includes name, author and genre

struct unread{

    char name[30];
    char author[30];
    char genre[20];

};

//main function, ie, the menu of the program
//the user can choose from the following options

int main(){

    int br=1, choice=0;

    while(br){

        printf("\n===============MENU==============\n\n");
        printf("(1) Add to READ \n(2) Add to UNREAD \n(3) Search READ\n(4)
Search UNREAD\n(5) Find highest rated \n(6) Add from unread to read \n(7)
View all read\n(8) View All unread\n(9) EXIT\n\nEnter your choice: ");

        scanf("%d",&choice);

        switch (choice)
        {
        case 1:
            addRead();
            break;

        case 2:
            addUnread();
            break;

        case 3:
            char key[30];
            printf("\nEnter the name of the book you want to find: ");
            scanf("%s",key);
            search(key);
            break;
```

```
        case 4:
            char key0[30];
            printf("\nEnter the name of the book you want to find1: ");
            scanf("%s",key);
            searchUnread(key);
            break;

        case 5:
            int top=0;
            top=highestRated();
            printf("%d",top);
            break;

        case 6:
            char key4[30];
            printf("Mark book as read\n");
            printf("Enter name of book: ");
            scanf("%s",key4);
            addUnrRea(key4);
            break;

        case 7:
            printf("\nAll Read\n");
            allRead();
            break;

        case 8:
            printf("All Unread\n");
            allUnread();
            break;

        case 9:
            printf("\n**** G O O D B Y E ****\n\n");
            br=0;
            break;

        default:
            printf("Invalid choice. Please try again");
            break;
        }
    }
}
```

```
void searchUnread(char sk[30]){

    //file pointer
    FILE *inf;

    //struct to store the data (inp being input)
    struct read inp;

    //open file, read.txt in read mode
    inf = fopen ("unread.txt", "r");

    //if function for if file exists or not
    if (inf == NULL) {
        fprintf(stderr, "\nError to open the file\n");
        //exit (1);
    }

    //while loop to read the file
    //fread if file read, input is stored in inp, size of read, file name

    //pointer to a block of memory, size in bytes of each element to be read
    //number of elements, each one with a size of size bytes
    //pointer to a FILE object that specifies an input stream
    while(fread(&inp, sizeof(struct unread), 1, inf)){

    //if statement to compare strings, ie, input given and to see if it exists in the file
      if(!strcmp(sk,inp.name)){
          printf("\nSearching...\n");
          printf("NAME: %s\nAUTHOR: %s\nGENRE:
%s\n",inp.name,inp.author,inp.genre);
          break;
          exit(1);
      }
      else{
            printf("\nSearching...\n");
            printf("No book found\n");
            break;
            exit(1);
      }
    }
```

```
    //close file
    fclose (inf);

}

//function for to display all books that have been read
void allRead(){

    //file pointer
    FILE *inf;

    //open file read.txt in read mode
    inf=fopen("read.txt","r");

    //read data from unread in the format of read struct and print it till end of file
(eof)
    struct read r;

    while(fread(&r,sizeof(struct read),1,inf)){
            printf("\nNAME: %s\nAUTHOR: %s\nGENRE: %s\nRATING:
%d\n",r.name,r.author,r.genre,r.stars);
        }
}


//function for to display all books that have not been read
void allUnread(){

        //file pointer
        FILE *inf;
        //char filename[100], c;
        //open file unread.txt in read mode
        inf=fopen("unread.txt","r");

        //read data from unread in the format of unread struct and print it till end of
file (eof)
        struct unread r;

        while(fread(&r,sizeof(struct unread),1,inf)){
            printf("\nNAME: %s\nAUTHOR: %s\nGENRE:
%s\n\n",r.name,r.author,r.genre);
        }
}
```

```
//function to search for a book that has been read
void search(char sk[30]){

    //file pointer
    FILE *inf;

    //struct to store the data (inp being input)
    struct read inp;

    //open file, read.txt in read mode
    inf = fopen ("read.txt", "r");

    //if function for if file exists or not
    if (inf == NULL) {
        fprintf(stderr, "\nError to open the file\n");
        //exit (1);
    }

    //while loop to read the file
    //fread if file read, input is stored in inp, size of read, file name

    //pointer to a block of memory, size in bytes of each element to be read
    //number of elements, each one with a size of size bytes
    //pointer to a FILE object that specifies an input stream
    while(fread(&inp, sizeof(struct read), 1, inf)){

    //if statement to compare strings, ie, input given and to see if it exists in the file
      if(!strcmp(sk,inp.name)){
          printf("\nSearching...\n");
          printf("NAME: %s\nAUTHOR: %s\nGENRE: %s\nRATING:
%d\n",inp.name,inp.author,inp.genre,inp.stars);
          break;
          exit(1);
      }
      else{
            printf("\nSearching...\n");
            printf("No book found\n");
            break;
            exit(1);
      }
    }
```

```
    //close file
    fclose (inf);

}

void deleteUnread(){
    FILE *inp;
    FILE *inp1;
    struct unread s;

    int j,name,found=0;

    inp = fopen("unread.txt","r");
    inp1 = fopen("unread1.txt","w");
    printf("enter name of book to delete: ");
    scanf("%s",name);

    while(fread(&s,sizeof(struct unread),1,inp)){
        if(s.name==name){
            found=1;
        }
        else{
            fwrite(&s,sizeof(struct unread),1,inp1);
        }

    }
    fclose(inp);
    fclose(inp1);
    if(found){
        inp1 = fopen("unread1.txt","r");
        inp = fopen("unread.txt","w");

        while(fread(&s,sizeof(struct unread),1,inp1)){
            fwrite(&s,sizeof(struct unread),1,inp);
        }
    fclose(inp);
    fclose(inp1);
    }
    return;
}

void addRead(){
```

```
    FILE *outfile;

   // open file for writing
   outfile = fopen ("read.txt", "a");
   if (outfile == NULL)
   {
      fprintf(stderr, "\nError opening file\n");
      //exit (1);
   }

   struct read a;

   printf("\nEnter the details of the book you just read in the format below\n");
   printf("\nName: "); scanf("%s",&a.name);
   printf("\nAuthor: "); scanf("%s",&a.author);
   printf("\nGenre: "); scanf("%s", &a.genre);
   printf("\nRating (Out of 5): "); scanf("%d",&a.stars);

   // write struct to file
   fwrite(&a, sizeof(struct read) ,1 ,outfile);

   if(fwrite != 0){
      printf("\nContents to file written successfully!\n");
   }
   else{
      printf("Error writing to file!\n");
   }

   // close file
   fclose (outfile);

   return;

}

void addUnread(){

       FILE *outfile;

   // open file for writing
   outfile = fopen ("unread.txt", "a");
   if (outfile == NULL)
   {
```

```
        fprintf(stderr, "\nError opening file\n");
        //exit (1);
    }


    struct unread a;

    printf("\nEnter the details of the book you want to read\n");
    printf("\nName:"); scanf("%s",&a.name);
    printf("\nAuthor:"); scanf("%s",&a.author);
    printf("\nGenre: "); scanf("%s", &a.genre);


    // write struct to file
    fwrite(&a, sizeof(struct unread) ,1 ,outfile);

    if(fwrite != 0){
        printf("\nContents to file written successfully!\n");
    }
    else{
        printf("Error writing file!\n");
    }

    // close file
    fclose (outfile);

    return;

}

int highestRated(){

    FILE *inf;
    struct read inp;

    int highest=0;
    char bookname[30];

    inf = fopen ("read.txt", "r");
    if (inf == NULL) {
        fprintf(stderr, "\nError to open the file\n");
        //exit (1);
    }
```

```
    while(fread(&inp, sizeof(struct read), 1, inf)){
      //printf ("stars = %d\n", inp.stars);

      if(inp.stars>highest){

          highest=inp.stars;

          strcpy(bookname,inp.name);

      }
    }
  fclose (inf);

  printf("\nHighest Rated Book: %s\n",bookname);
  printf("Rating: ");
  return highest;

}

void deleteRead(char sk[30]){
    FILE *inf;
    struct read inp;
    struct read readbooks[100];
    int counter=0;

    inf = fopen ("read.txt", "r");
    if (inf == NULL) {
        fprintf(stderr, "\nError to open the file\n");
        //exit (1);
    }
    while(fread(&inp, sizeof(struct read), 1, inf)){

      strcpy(readbooks[counter].name,inp.name);
      strcpy(readbooks[counter].author,inp.author);
      strcpy(readbooks[counter].genre,inp.genre);
      readbooks[counter].stars=inp.stars;

      counter++;
    }

  fclose (inf);

  for(int i=0;i<counter;i++){
```

```
      printf("Deleting the following data:\n");
      printf("NAME: %s\nAUTHOR: %s\nGENRE: %s\nRATING:%d\n",
readbooks[i].name, readbooks[i].author, readbooks[i].genre,readbooks[i].stars);
      exit(1);
   }

   FILE *outfile;

   // open file for writing
   outfile = fopen ("read.txt", "w");
   if (outfile == NULL)
   {
       fprintf(stderr, "\nError opening file\n");
       //exit (1);
   }

  for(int i=0;i<counter;i++){
      // write struct to file

   if(strcmp(sk,readbooks[i].name)){
   fwrite(&readbooks[i], sizeof(struct read) ,1 ,outfile);
   }
   // close file
   }

    fclose (outfile);
   return;
}

void addUnrRea(char sk[30]){

   struct read a;

   FILE *inf;
   struct unread inp;

   inf = fopen ("unread.txt", "r");
   if (inf == NULL) {
       fprintf(stderr, "\nError to open the file\n");
       //exit (1);
   }
   while(fread(&inp, sizeof(struct unread), 1, inf)){
     printf ("name = %s\n", inp.name);
```

```
    if(!strcmp(sk,inp.name)){

      strcpy(a.name,inp.name);
      strcpy(a.author,inp.author);
      strcpy(a.genre,inp.genre);
      printf("Rating (out of 5)\n"); scanf("%d",&a.stars);

        break;
        exit(1);
    }
  fclose (inf);
}


   FILE *outfile;

   // open file for writing
   outfile = fopen ("read.txt", "a");
   if (outfile == NULL)
   {
      fprintf(stderr, "\nError opening file\n");
      //exit (1);
   }


   // write struct to file
   fwrite(&a, sizeof(struct read) ,1 ,outfile);

   if(fwrite != 0){
      printf("Book marked as read!\n");
      exit(1);
   }
   else{
      printf("error writing file !\n");
   }

   // close file
   fclose (outfile);


   deleteRead(a.name);
   return;
}
```

CODE

```c
#include <stdio.h>
#include <string.h>

/*

   Functions:

   add to read (done)
   add to unread (done)
   seaarch using name (done)
   find highest rated (done)
   add form unread to read
   delete from read (done)
   delete form unread (done)
   view all read
   view all unread


    */


//all functions declared here
void addRead();
void addUnread();
void search(char sk[30]);
void searchUnread(char sk[30]);
void deleteUnread();
int highestRated();
void deleteRead(char sk[30]);
//void addUnrRea();
void allRead();
void allUnread();
void addUnrRea(char sk[30]);


//struct for books that have been read
//includes name, author, rating, and genre
struct read{

    char name[30];
    char author[30];
    char genre[20];
    int stars;

};

//struct for books that have not been read
//includes name, author and genre
```

```c

struct unread{

    char name[30];
    char author[30];
    char genre[20];

};



//main function, ie, the menu of the program
//the user can choose from the following options

int main(){

    int br=1, choice=0;

    while(br){

        printf("\n===============MENU==============\n\n");
        printf("(1) Add to READ \n(2) Add to UNREAD \n(3) Search READ\n(4) Search UNREAD\n(5) Find highest rated \n(6) Add from unread to read \n(7) View all
            read\n(8) View All unread\n(9) EXIT\n\nEnter your choice: ");

        scanf("%d",&choice);

        switch (choice)
        {
        case 1:
            addRead();
            break;

        case 2:
            addUnread();
            break;

        case 3:
            char key[30];
            printf("\nEnter the name of the book you want to find: ");
            scanf("%s",key);
            search(key);
            break;

        case 4:
            char key0[30];
            printf("\nEnter the name of the book you want to find1: ");
            scanf("%s",key);
            searchUnread(key);
```

```c
        case 5:
            int top=0;
            top=highestRated();
            printf("%d",top);
            break;

        case 6:
            char key4[30];
            printf("Mark book as read\n");
            printf("Enter name of book: ");
            scanf("%s",key4);
            addUnrRea(key4);
            break;

        case 7:
            printf("\nAll Read\n");
            allRead();
            break;

        case 8:
            printf("All Unread\n");
            allUnread();
            break;

        case 9:
            printf("\n**** G O O D B Y E ****\n\n");
            br=0;
            break;

        default:
            printf("Invalid choice. Please try again");
            break;
        }
    }
}

void searchUnread(char sk[30]){

    //file pointer
    FILE *inf;

    //struct to store the data (inp being input)
    struct read inp;

    //open file, read.txt in read mode
    inf = fopen ("unread.txt", "r");

    //if function for if file exists or not
```

```c
    if (inf == NULL) {
        fprintf(stderr, "\nError to open the file\n");
        //exit (1);
    }

    //while loop to read the file
    //fread if file read, input is stored in inp, size of read, file name

    //pointer to a block of memory, size in bytes of each element to be read
    //number of elements, each one with a size of size bytes
    //pointer to a FILE object that specifies an input stream
    while(fread(&inp, sizeof(struct unread), 1, inf)){

    //if statement to compare strings, ie, input given and to see if it exists in the file
      if(!strcmp(sk,inp.name)){
          printf("\nSearching...\n");
          printf("NAME: %s\nAUTHOR: %s\nGENRE: %s\n",inp.name,inp.author,inp.genre);
          break;
          exit(1);
      }
      else{
          printf("\nSearching...\n");
          printf("No book found\n");
          break;
          exit(1);
      }
    }

    //close file
    fclose (inf);

}

//function for to display all books that have been read
void allRead(){

    //file pointer
    FILE *inf;

    //open file read.txt in read mode
    inf=fopen("read.txt","r");

    //read data from unread in the format of read struct and print it till end of file (eof)
    struct read r;

    while(fread(&r,sizeof(struct read),1,inf)){
          printf("\nNAME: %s\nAUTHOR: %s\nGENRE: %s\nRATING: %d\n",r.name,r.author,r.genre,r.stars);
        }
```

```c
//function for to display all books that have not been read
void allUnread(){

        //file pointer
        FILE *inf;
        //char filename[100], c;
        //open file unread.txt in read mode
        inf=fopen("unread.txt","r");

        //read data from unread in the format of unread struct and print it till end of file (eof)
        struct unread r;

        while(fread(&r,sizeof(struct unread),1,inf)){
            printf("\nNAME: %s\nAUTHOR: %s\nGENRE: %s\n\n",r.name,r.author,r.genre);
        }
}


//function to search for a book that has been read
void search(char sk[30]){

    //file pointer
    FILE *inf;

    //struct to store the data (inp being input)
    struct read inp;

    //open file, read.txt in read mode
    inf = fopen ("read.txt", "r");

    //if function for if file exists or not
    if (inf == NULL) {
        fprintf(stderr, "\nError to open the file\n");
        //exit (1);
    }

    //while loop to read the file
    //fread if file read, input is stored in inp, size of read, file name

    //pointer to a block of memory, size in bytes of each element to be read
    //number of elements, each one with a size of size bytes
    //pointer to a FILE object that specifies an input stream
    while(fread(&inp, sizeof(struct read), 1, inf)){

    //if statement to compare strings, ie, input given and to see if it exists in the file
      if(!strcmp(sk,inp.name)){
          printf("\nSearching...\n");
```

```c
            break;
            exit(1);
      }
      else{
            printf("\nSearching...\n");
            printf("No book found\n");
            break;
            exit(1);
      }
    }

    //close file
    fclose (inf);

}

void deleteUnread(){
    FILE *inp;
    FILE *inp1;
    struct unread s;

    int j,name,found=0;

    inp = fopen("unread.txt","r");
    inp1 = fopen("unread1.txt","w");
    printf("enter name of book to delete: ");
    scanf("%s",name);

    while(fread(&s,sizeof(struct unread),1,inp)){
        if(s.name==name){
            found=1;
        }
        else{
            fwrite(&s,sizeof(struct unread),1,inp1);
        }

    }
    fclose(inp);
    fclose(inp1);
    if(found){
        inp1 = fopen("unread1.txt","r");
        inp = fopen("unread.txt","w");

        while(fread(&s,sizeof(struct unread),1,inp1)){
            fwrite(&s,sizeof(struct unread),1,inp);
        }
    fclose(inp);
    fclose(inp1);
```

```c
void addRead(){

     FILE *outfile;

    // open file for writing
    outfile = fopen ("read.txt", "a");
    if (outfile == NULL)
    {
        fprintf(stderr, "\nError opening file\n");
        //exit (1);
    }

    struct read a;

    printf("\nEnter the details of the book you just read in the format below\n");
    printf("\nName: "); scanf("%s",&a.name);
    printf("\nAuthor: "); scanf("%s",&a.author);
    printf("\nGenre: "); scanf("%s", &a.genre);
    printf("\nRating (Out of 5): "); scanf("%d",&a.stars);

    // write struct to file
    fwrite(&a, sizeof(struct read) ,1 ,outfile);

    if(fwrite != 0){
        printf("\nContents to file written successfully!\n");
    }
    else{
        printf("Error writing to file!\n");
    }

    // close file
    fclose (outfile);

    return;

}

void addUnread(){

         FILE *outfile;

    // open file for writing
    outfile = fopen ("unread.txt", "a");
    if (outfile == NULL)
    {
        fprintf(stderr, "\nError opening file\n");
        //exit (1);
```

```c
    printf("\nEnter the details of the book you want to read\n");
    printf("\nName:"); scanf("%s",&a.name);
    printf("\nAuthor:"); scanf("%s",&a.author);
    printf("\nGenre: "); scanf("%s", &a.genre);


    // write struct to file
    fwrite(&a, sizeof(struct unread) ,1 ,outfile);

    if(fwrite != 0){
        printf("\nContents to file written successfully!\n");
    }
    else{
        printf("Error writing file!\n");
    }

    // close file
    fclose (outfile);

    return;

}

int highestRated(){

    FILE *inf;
    struct read inp;

    int highest=0;
    char bookname[30];

    inf = fopen ("read.txt", "r");
    if (inf == NULL) {
        fprintf(stderr, "\nError to open the file\n");
        //exit (1);
    }
    while(fread(&inp, sizeof(struct read), 1, inf)){
      //printf ("stars = %d\n", inp.stars);

      if(inp.stars>highest){

          highest=inp.stars;

          strcpy(bookname,inp.name);

      }
    }
```

```c
    printf("\nHighest Rated Book: %s\n",bookname);
    printf("Rating: ");
    return highest;

}

void deleteRead(char sk[30]){
    FILE *inf;
    struct read inp;
    struct read readbooks[100];
    int counter=0;

    inf = fopen ("read.txt", "r");
    if (inf == NULL) {
        fprintf(stderr, "\nError to open the file\n");
        //exit (1);
    }
    while(fread(&inp, sizeof(struct read), 1, inf)){

      strcpy(readbooks[counter].name,inp.name);
      strcpy(readbooks[counter].author,inp.author);
      strcpy(readbooks[counter].genre,inp.genre);
      readbooks[counter].stars=inp.stars;

      counter++;
    }

   fclose (inf);

   for(int i=0;i<counter;i++){
       printf("Deleting the following data:\n");
       printf("NAME: %s\nAUTHOR: %s\nGENRE: %s\nRATING:%d\n", readbooks[i].name, readbooks[i].author, readbooks[i].genre,readbooks[i].stars);
       exit(1);
   }

    FILE *outfile;

    // open file for writing
    outfile = fopen ("read.txt", "w");
    if (outfile == NULL)
    {
        fprintf(stderr, "\nError opening file\n");
        //exit (1);
    }

   for(int i=0;i<counter;i++){
       // write struct to file
```

```c
    if(strcmp(sk,readbooks[i].name)){
    fwrite(&readbooks[i], sizeof(struct read) ,1 ,outfile);
    }
    // close file
   }

     fclose (outfile);
    return;
}

void addUnrRea(char sk[30]){

    struct read a;

    FILE *inf;
    struct unread inp;

    inf = fopen ("unread.txt", "r");
    if (inf == NULL) {
        fprintf(stderr, "\nError to open the file\n");
        //exit (1);
    }
    while(fread(&inp, sizeof(struct unread), 1, inf)){
      printf ("name = %s\n", inp.name);
      if(!strcmp(sk,inp.name)){

        strcpy(a.name,inp.name);
        strcpy(a.author,inp.author);
        strcpy(a.genre,inp.genre);
        printf("Rating (out of 5)\n"); scanf("%d",&a.stars);

          break;
          exit(1);
      }
   fclose (inf);
}


    FILE *outfile;

    // open file for writing
    outfile = fopen ("read.txt", "a");
    if (outfile == NULL)
    {
        fprintf(stderr, "\nError opening file\n");
        //exit (1);
    }
```

```c
    }
    while(fread(&inp, sizeof(struct unread), 1, inf)){
      printf ("name = %s\n", inp.name);
      if(!strcmp(sk,inp.name)){

        strcpy(a.name,inp.name);
        strcpy(a.author,inp.author);
        strcpy(a.genre,inp.genre);
        printf("Rating (out of 5)\n"); scanf("%d",&a.stars);

          break;
          exit(1);
      }
   fclose (inf);
}


    FILE *outfile;

    // open file for writing
    outfile = fopen ("read.txt", "a");
    if (outfile == NULL)
    {
        fprintf(stderr, "\nError opening file\n");
        //exit (1);
    }


    // write struct to file
    fwrite(&a, sizeof(struct read) ,1 ,outfile);

    if(fwrite != 0){
        printf("Book marked as read!\n");
        exit(1);
    }
    else{
        printf("error writing file !\n");
    }

    // close file
    fclose (outfile);


    deleteRead(a.name);
    return;
}
```

Output

```
================MENU===============

(1) Add to READ
(2) Add to UNREAD
(3) Search READ
(4) Search UNREAD
(5) Find highest rated
(6) Add from unread to read
(7) View all read
(8) View All unread
(9) EXIT
```

Project Learnings

- How to write a console application
- How to manage data in .txt files and extract data from them using file pointers
- How to work with file-based commands like fread, fwrite.
- Working with constructs like switch, if and several loops
- Managing several functions and their execution in one single project.

Conclusion

This project has been a profound for me and my team-mates in every manner. From learning how to build a console application to using it to code our formulated idea, we have learnt several new lessons in C. I would like to take this opportunity to thank our professor, Dr Prabhat Thakur for giving us this opportunity to code such a project and put ourselves to this test.

References

1. How bookstores are weathering the Pandemic – VOX Media