LAKSHMI NARAIN COLLEGE OF TECHNOLOGY, MCA BHOPAL

**BOOKSHOP**

**MANAGEMENT SYSTEM**

**PROJECT**

****

|  |  |
| --- | --- |
| **Guided by**  **T&P C C++ IT Trainer**  **Shri. RAM**  **LOWANSHI** | **Name : Hemant Kumar Vishwakarma**  **Class: MCA 2nd (SEM)**  **Enroll. No. 0103CA201055** |

**TABLE OF CONTANT**

|  |  |
| --- | --- |
| **1** | **PROJECT INTRODUCTION** |
| **2** | **MAIN FEATURES** |
| **3** | **PROGRAMMING CONCEPT** |
| **4** | **STRUCTURE** |
| **5** | **LOOPING** |
| **6** | **FILE HANDLING** |
| **7** | **HARDWARE REQUIREMENTS** |
| **8** | **SOFTWARE REQUIREMENTS** |

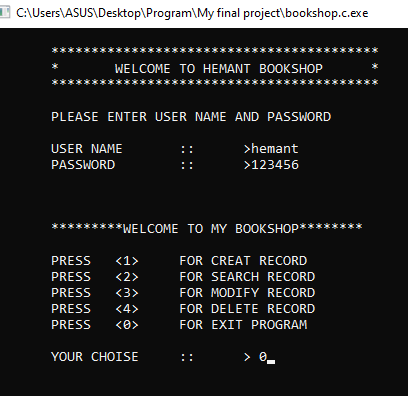
**PROJECT INTRODUCTION**

The purpose of this software is to manage the books in the book store.It includes the Stock Management. We developed this software to maintain records of sales. This project developed using C programming as file handling. The books store owner sells the book to the buyer. All the name of the books, sell date, customer name, barcode of the book and mobile number of the customer is noted down in the software along with rate. In the present system the user has to do all work manually. And it takes time to search records and modify them.In case of books store owner Record Management, the registers need to be updated from time to time as information (like Phone No., Address) changes frequently. When a customer buys a book from the store the bill contains the name of the book purchased, rate of book. For example, if any customer wants to purchase any book from the shop, then first of all the customer just chooses the stream of the book, then he/she can see the more than one type of books there and then he/she can choose the specific book from there.In present book store management system, In each process Customer Records store performing the tasks. In the proposed system all records are saved in the file.

**MAIN FEATURES** **:**

Listed below are the main features of this project:

**1.** **CREATE RECORDS**: This feature allows bookshop management system applications to create book records. Information such as the date of purchase, customer's name, customer's mobile number, book name, book barcode number and book rate are stored in the file.



**2.** **SEARCH RECORD** : This feature allows search book records from the Bookstore Management System app. it can help search the record by date or barcode.

**4.** **MODIFY RECORD**: This feature allows the Bookstore Management System app to modify the information provided while creating records. It can edit the date of purchase, customer name, book name, book rate. All modified data is then stored in a file.

**5.** **DELETE RECORD** : This feature allows the delete records from record file of bookshop management. For this the barcode number of the book should be provided, and a confirmation from the user is asked before deleting the book record.

**PROGRAMMING CONCEPT**

**1.** **ARRAY**

* An array is a collection of data items, all of the same type, accessed using a common name.
* A one-dimensional array is like a list;  A two dimensional array is like a table;  The C language places no limits on the number of dimensions in an array, though specific implementations may.
* Some texts refer to one-dimensional arrays as ***vectors***, two-dimensional arrays as ***matrices***, and use the general term ***arrays*** when the number of dimensions is unspecified or unimportant.

**2.** **DECLARING ARRAY**

* To declare an array in C, a programmer specifies the type of the elements and the number of elements required by an array as follows −
* type arrayName [ arraySize ];

**STRUCTURE**

Arrays allow to define type of variables that can hold several data items of the same kind. Similarly **structure** is another user defined data type available in C that allows to combine data items of different kinds.

Structures are used to represent a record. Suppose you want to keep track of your books in a library. You might want to track the following attributes about each book −

* Title
* Author
* Subject
* Book ID

Defining a Structure

To define a structure, you must use the **struct** statement. The struct statement defines a new data type, with more than one member. The format of the struct statement is as follows −

struct [structure tag] {

member definition;

member definition;

...

member definition;

} [one or more structure variables];

The **structure tag** is optional and each member definition is a normal variable definition, such as int i; or float f; or any other valid variable definition. At the end of the structure's definition, before the final semicolon, you can specify one or more structure variables but it is optional. Here is the way you would declare the Book structure −

struct Books {

char title[50];

char author[50];

char subject[100];

int book\_id;

} book;

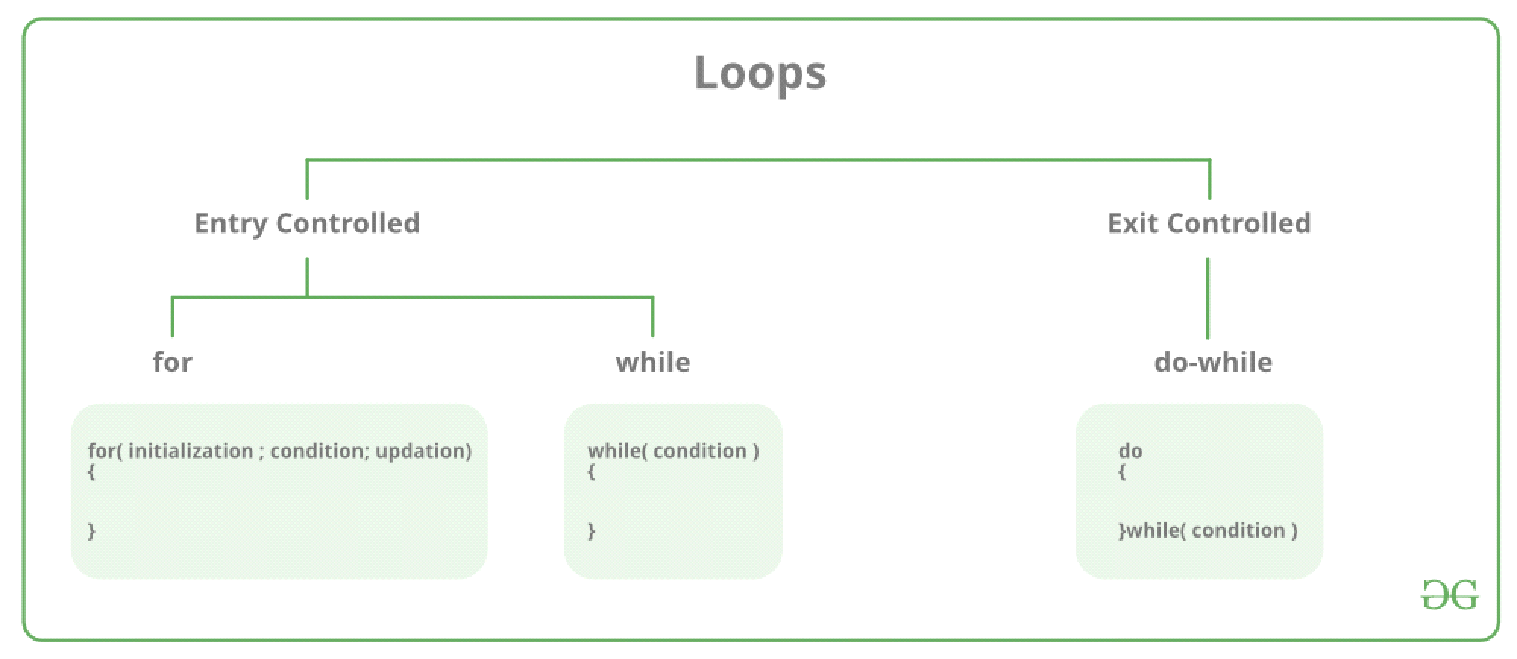
**LOOPING**

In Loop, the statement needs to be written only once and the loop will be executed 10 times as shown below. In computer programming, a loop is a sequence of instructions that is repeated until a certain condition is reached.

* An operation is done, such as getting an item of data and changing it, and then some condition is checked such as whether a counter has reached a prescribed number.
* **Counter not Reached:**If the counter has not reached the desired number, the next instruction in the sequence returns to the first instruction in the sequence and repeat it.
* **Counter reached:** If the condition has been reached, the next instruction “falls through” to the next sequential instruction or branches outside the loop.

**There are mainly two types of loops:**

* **Entry Controlled loops**: In this type of loops the test condition is tested before entering the loop body. **For Loop** and **While Loop** are entry controlled loops.
* **Exit Controlled Loops**: In this type of loops the test condition is tested or evaluated at the end of loop body. Therefore, the loop body will execute atleast once, irrespective of whether the test condition is true or false. **do – while loop** is exit controlled loop.



**1. FOR LOOP**

A for loop is a repetition control structure which allows us to write a loop that is executed a specific number of times. The loop enables us to perform n number of steps together in one line.  
**Syntax:**

for (initialization expr; test expr; update expr)

{

// body of the loop

// statements we want to execute

}

In for loop, a loop variable is used to control the loop. First initialize this loop variable to some value, then check whether this variable is less than or greater than counter value. If statement is true, then loop body is executed and loop variable gets updated . Steps are repeated till exit condition comes.

* **Initialization Expression**: In this expression we have to initialize the loop counter to some value. for example: int i=1;
* **Test Expression**: In this expression we have to test the condition. If the condition evaluates to true then we will execute the body of loop and go to update expression otherwise we will exit from the for loop. For example: i <= 10;
* **Update Expression**: After executing loop body this expression increments/decrements the loop variable by some value. for example: i++;

**2. DO WHILE LOOP**

In do while loops also the loop execution is terminated on the basis of test condition. The main difference between do while loop and while loop is in do while loop the condition is tested at the end of loop body, i.e do while loop is exit controlled whereas the other two loops are entry controlled loops.  
**Note**: In do while loop the loop body will execute at least once irrespective of test condition.

**Syntax**:

**initialization expression;**

do

{

// statements

**update\_expression;**

} while (**test\_expression**);

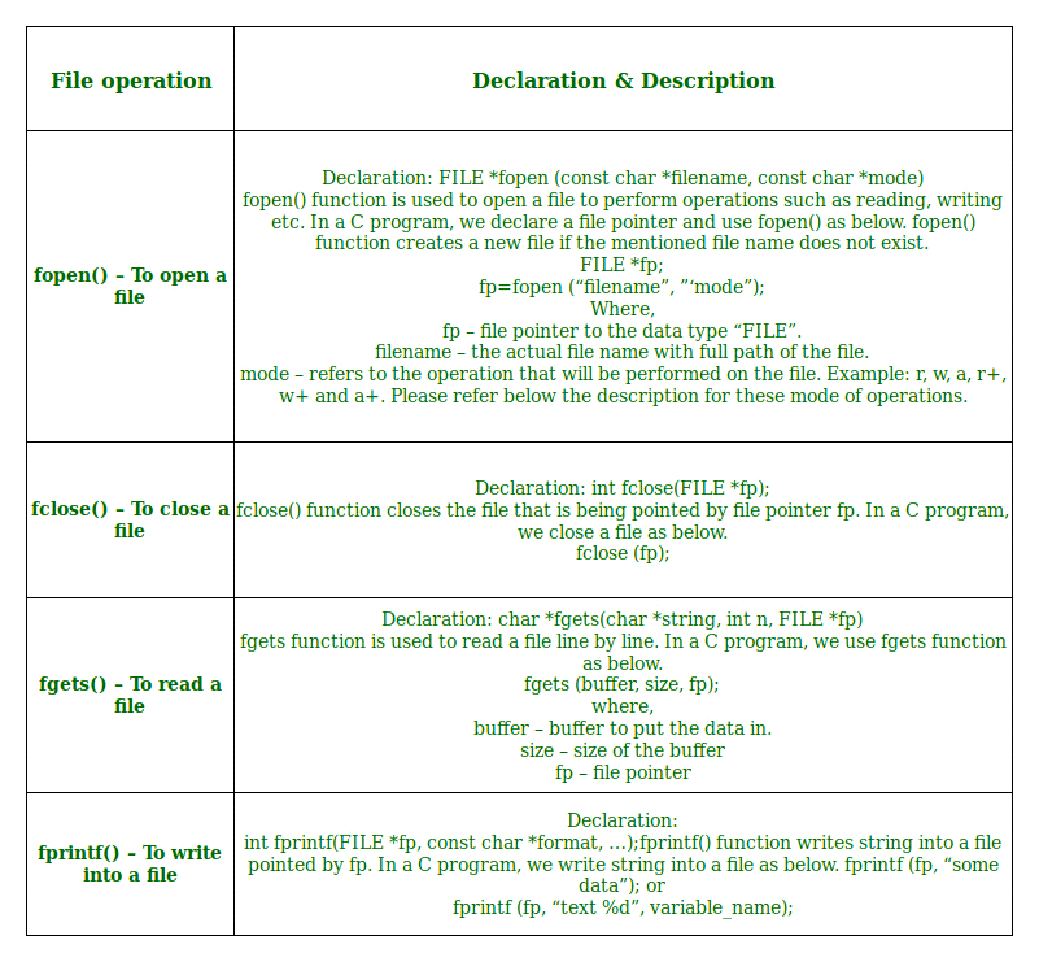
**Note**: Notice the semi – colon(“;”) in the end of loop.

**FILE HANDLING**

So far the operations using C program are done on a prompt / terminal which is not stored anywhere. But in the software industry, most of the programs are written to store the information fetched from the program. One such way is to store the fetched information in a file. Different operations that can be performed on a file are:

* Creation of a new file (**fopen with attributes as “a” or “a+” or “w” or “w++”)**
* Opening an existing file (**fopen**)
* Reading from file (**fscanf or fgets**)
* Writing to a file (**fprintf or fputs**)
* Moving to a specific location in a file (**fseek, rewind**)
* Closing a file (**fclose**)

The text in the brackets denotes the functions used for performing those operations.

**1. FUNCTIONS IN FILE OPERATIONS**  


**2. OPENING OR CREATING FILE**

For opening a file, fopen function is used with the required access modes. Some of the commonly used file access modes are mentioned below.

**3. FILE OPENING MODES IN C**

* **“r” –** Searches file. If the file is opened successfully fopen( ) loads it into memory and sets up a pointer which points to the first character in it. If the file cannot be opened fopen( ) returns NULL.
* **“w” –** Searches file. If the file exists, its contents are overwritten. If the file doesn’t exist, a new file is created. Returns NULL, if unable to open file.
* **“a” –** Searches file. If the file is opened successfully fopen( ) loads it into memory and sets up a pointer that points to the last character in it. If the file doesn’t exist, a new file is created. Returns NULL, if unable to open file.
* **“r+” –** Searches file. If is opened successfully fopen( ) loads it into memory and sets up a pointer which points to the first character in it. Returns NULL, if unable to open the file.
* **“w+” –** Searches file. If the file exists, its contents are overwritten. If the file doesn’t exist a new file is created. Returns NULL, if unable to open file.
* **“a+” –** Searches file. If the file is opened successfully fopen( ) loads it into memory and sets up a pointer which points to the last character in it. If the file doesn’t exist, a new file is created. Returns NULL, if unable to open file.

As given above, if you want to perform operations on a binary file, then you have to append ‘b’ at the last. For example, instead of “w”, you have to use “wb”, instead of “a+” you have to use “a+b”. For performing the operations on the file, a special pointer called File pointer is used which is declared as

FILE \*filePointer;

So, the file can be opened as

filePointer = fopen(“fileName.txt”, “w”)

The second parameter can be changed to contain all the attributes listed in the above table.

**4. READING FROM A FILE**   
 The file read operations can be performed using functions fscanf or fgets. Both the functions performed the same operations as that of scanf and gets but with an additional parameter, the file pointer. So, it depends on you if you want to read the file line by line or character by character.

And the code snippet for reading a file is as:

FILE \* filePointer;

filePointer = fopen(“fileName.txt”, “r”);

fread(void \*file pointer, size\_t size, size\_t nmemb, FILE \*stream)

**5. WRITING A FILE**

The file write operations can be perfomed by the functions fprintf and fputs with similarities to read operations. The snippet for writing to a file is as :

FILE \*filePointer ;

filePointer = fopen(“fileName.txt”, “w”);

size\_t fwrite (const void \*filePointer, size\_t size, size\_t nmemb, FILE \*stream)

**6. CLOSING A FILE**  
 After every successful fie operations, you must always close a file. For closing a file, you have to use fclose function. The snippet for closing a file is given as :

* FILE \*filePointer ;
* filePointer= fopen(“fileName.txt”, “w”);
* ---------- Some file Operations -------

fclose(filePointer)

**HARDWARE REQUIREMENTS**

• PIV 2.8 GHz Processor and Above

• RAM 512MB and Above

• HDD 20 GB Hard Disk Space and Above

**SOFTWARE REQUIREMENTS**

• WINDOWS OS (XP / 2000 / 200 Server / 2003 Server)

• C and C++ Compiler Code Block and Dev C++.