Chapter-6

DATA FILE HANDLING

Q1. Define File .

Ans.A file is a bunch of bytes stored on some storage devices.

Q2. How are binary file differ from text files in c++?

Ans When a file is opened in text mode, various character translations may take place, such as the conversion of carriage return and linefeed sequences into newlines. However, no such character translations occur in files opened in binary mode. Any file can be opened in either text or binary mode. The only difference is the occurrence of character translations in text mode.

Q3. What is a stream? Name the streams generally used for file I/O.

Ans A stream is a sequence of bytes. Or in other words, a stream is a flow of bytes into or out of a program. Generally three streams are used for file I/O. These are:

(i) Ifstream: It is derived from istream class and it is used to associate a file with an input buffer so that the file can be read from.

(ii) Ofstream: It is derived from ostream class and it is used to associate a file with an output buffer so that the file can be written from.

(iii) fstream: It is derived from iostream class and it is used to associate a file with a buffer so that the file can be read from as well as written onto.

Q4. Differentiate between ifstream class and ofstream class.

Ans The ifstream class is an input stream class, which provides input operations for file.
The ofstream class is an output stream class, which provides output operations for file.

Q5. Differentiate between functions read() and write().

Ans The read() lets one read a unit of information from a file and a write() lets one write a unit of information to a file.

Q6. Differentiate between getline() and getc() functions.

Ans getc() function can read one character at a time .On the other hand , getline() can read a line of text of specified size.

getc() is defined in stdio.h however getline() is defined in iostream.h

Q7. Name two member functions of ofstream class.

Ans seekp() , tellp()

Q8. Distinguish between ios::out and ios::app.

Ans The ios::out is the default file mode of ofstream. With the mode of the file does not exist, it gets created but if the file exists then its existing contents get deleted. On the other hand ios::app is also an output mode , which creates the file if it does not exist but if the file exists then its existing contents are retained and new information is appended to it.

Q9. What is the difference between the functioning of ios::ate and ios::app file modes.

Ans Both ios::ate and ios::app place the file pointer at the end of the file just opened. The difference between two is that ios::app lets you add data to the end of the file only, while the ios::ate mode when opened with ofstream allows you to write data anywhere in the file, even over old data.

Q10. What is the need and usage of read() and write() functions when there are get() and put() functions for I/O.

Ans The get () and put () function perform I/O byte by byte .On the other hand, read () and write () functions lets you read and write structures and objects in one go without creating the need for I/O for individual constituent fields.

Q11. What is the difference between get() and read()?

Ans The read() declared under iostream.h extracts a given number of characters into an array. The get() declared under iostream.h either extracts the next character or EOF or extracts characters into a char* until eof or delimiter encountered or specified(len-1) bytes have been read.

Q12. Write a C++ program, which reads one line at a time from the disk file TEST.TXT, and displays it to a monitor. Your program has to read all the contents of the file. Assume the length of the line does not exceed 70 characters.

Ans

```
#include<fstream.h>

void main()
{
        char str[80];
        ifstream fin("Test.txt");
        if (!fin)
        {       cout<<"Error in the opening";
                return –1;
        }
        while(fin)
        {       fin.getline(str,70);
                cout<<str<<endl;
        }
        fin.close();
}
```

Q13. Write a function to count the number of blanks present in text file named "PARA.TXT";

Ans

```
        void countblanks()
        {
        char ch;
        int count=0;
        ifstream fin("PARA.TXT",ios::in);
        while(!fin.eof())
        {       fin.get(ch)
                if(fin.eof())
                break;
                if(ch==' ')
                count++;                        }
        fin.close();
        cout<<count;                            }
```

Q14. Write a Program to write and read a structure using write() and read() function using a binary file

```
#include<fstream.h>

#include<string.h>
#include<conio.h>
struct customer
{
char name[51];
float balance;          };
```

```
void main()
{
clrscr();
customer savac;
strcpy(savac.name,"Tina Marshall");
savac.balance=21310.75;
ofstream fout;
fout.open("Saving" , ios::out|ios::binary);
if(!fout)
{          cout<<"File cant be opened \n";
return 1;          }
fout.write(char *)&savac ,sizeof(customer));
fout.close();
ifstream fin;
fin.open("Saving",ios::in|ios::binary);
fin.read(cahr *)&savac,sizeof(customer));
cout<<savac.name;
cout<<"has the balance Rs"<<savac.balance<<"\n";
fin.close();                }
```

Ans :- Tina Marshall has the balance Rs 21310.75

Q14. Write a program for reading and writing class objects.

```
#include<fstream.h>
#include<conio.h>
class Student
{
char name[40];
char grade;
float marks;
public:
  void getdata(void);
  void display(void);
};
void Student::getdat(void)
{
char ch;
cin.get(ch);
cout<<"Enter name";  cin.getline(name,40);
cout<<"Enter grade";  cin>>grade;
cout<<"Enter marks"; cin>>marks;
cout<<"\n";
}
void Student::display(void)
{
cout<<"Name:"<<name<<"\t"
<<"Grade:"<<grade<<"\t"
<<"Marks:"<<marks<<"\t"<<"\n";
}
void main()
```

```
{
clrscr();
Student arts[3];
fstream filin;
        filin.open("Stu.dat",ios::in|ios::out);
        if(!filin)
        {
        cout<<"Cannot open file \n";
        return 1;
        }
cout<<"Enter details for 3 students \n";
for(int i=0;i<=3;i++)
{
arts[i].getdata();
filin.write(char *) &arts[i],sizeof(arts[i]));
}
filin.seekg(0);
cout<<"the contents of stu.dat are shown below
for(int i=0;i<=3;i++)
{
filin.read(char *) &arts[i],sizeof(arts[i]));
arts[i].display();
}
filin.close();
}
```

Chapter -8

POINTERS

Pointers :

- Pointer is a variable that holds a memory address of another variable.
- It supports dynamic allocation routines.
- It can improve the efficiency of certain routines.

C++ Memory Map :

- Program Code : It holds the compiled code of the program.
- Global Variables : They remain in the memory as long as program continues.
- Stack : It is used for holding return addresses at function calls, arguments passed to the functions, local variables for functions. It also stores the current state of the CPU.
- Heap : It is a region of free memory from which chunks of memory are allocated via DMA functions.

Static Memory Allocation : The amount of memory to be allocated is known in advance and it allocated during compilation, it is referred to as Static Memory Allocation.

Eg. Int a; // This will allocate 2 bytes for a during compilation.

Dynamic Memory Allocation : The amount of memory to be allocated is not known beforehand rather it is required to allocated as and when required during runtime, it is referred to as dynamic memory allocation.

C++ offers two operator for DMA – new and delete.