DATA FILE HANDLING IN C++

File

- A file is a stream of bytes stored on some secondary storage devices.
- **Text file:** A text file stores information in readable and printable form. Each line of text is terminated with an **EOL** End of Line cha racter.
- **Binary file:** A binary file contains information in the non-readable form i.e. in the same format in which it is held in memory.

File Stream

- **Stream:** A stream is a general term used to name flow of data. Different streams are used to represent different kinds of data flow.
- There are three file I/O classes used for file read / write operations.
  - **ifstream** - can be used for read operations.
  - **ofstream** - can be used for write operations.
  - **fstream** - can be used for both read & write operations.
- **fstream.h**:
- This header file includes the definitions for the stream classes ifstream, ofstream and fstream. In C++ **file input output** facilities implemented through fstream.h header file.
- It contain predefines set of operation for handling file related input and output, fstream class ties a file to the program for input and output operation.
- A file can be opened using:
  - **By the constructor method**. This will use default streams for file input or output. This method is preferred when file is opened in input or output mode only.
    Example : **ofstream file"student.dat"); or ifstream file"student.dat");**
  - **By the open member function** of the stream. It will preferred when file is opened in various modes i.e ios::in, ios::out, ios::app, ios::ate etc.
    e.g **fstream file;**
    file.open "book.dat", ios::in | ios::out | ios::binary);

File modes:

- **ios::out** It open file in output mode i.e write mode and place the file pointer in beginning, if file already exist it will overwrite the file.
- **ios::in** It open file in input moderead mode and permit reading from the file.
- **ios::app** It open the file in write mode, and place file pointer at the end of file i.e to add new contents and retains previous contents. If file does not exist it will create a new file.
- **ios::ate** It open the file in write or read mode, and place file pointer at the end of file i.e input/ output operations can performed anywhere in the file.
- **ios::trunc** It truncates the existing file (empties the file.
- **ios::nocreate** If file does not exist this file mode ensures that no file is created and open fails.
- **ios::noreplace** If file does not exist, a new file gets created but if the file already exists, the open(fails.
- **ios::binary** Opens a file in binary mode.

eof: This function determines the end-of-file by returning truenon -zero) for end of file otherwise returning falsezero).

close: This function terminates the connection between the file and stream associated with it.
Stream_object.close; e.g file.clos e;

Text File functions:

Char I/O :

- **get** – read a single character from text file and store in a buffer.
  e.g **file.getch;**
- **put** - writing a single character in textfile e.g. **file.putch;**
- **getline** - read a line of text from text file store in a buffer.
  e.g **file.getlines,80);**

- We can also use **file>>ch** for reading and **file<<ch** writing in text file. But >> operator does not accept white spaces.

Binary file functions:

- **read - ** read a block of binary data or reads a fixed number of bytes from the specified stream and store in a buffer.
  Syntax : Stream_object.read(char *& Object, sizeofObject;
  e.g file.read(char *&s, sizeofs);
- **write – ** write a block of binary data or writes fixed number of bytes from a specific memory location to the specified stream.
  Syntax : Stream_object.writechar *)& Object, sizeofObject;
  e.g file.writechar *)&s, sizeofs);

Note:
Both functions take two arguments.
- The first is the address of variable, and the second is the length of that variable in bytes. The address of variable must be type cast to type char*pointer to character type
- The data written to a file using write) can only be read accurately using read( ).

File Pointer: The file pointer indicates the position in the file at which the next input/output is to occur.

Moving the file pointer in a file for various operations viz modification, deletion , searching etc. Following functions are used:

seekg(: It places the file pointer to the specified position in input mode of file.
e.g file.seekg(p,ios::beg); or file.seekg -p,ios::end, or file.seekg(p,ios::cur
i.e to move to **p** byte position from beginning, end or current position.

seekp: It places the file pointer to the specified position in output mode of file.
e.g file.seekpp,ios::beg); or file.seekp -p,ios::end, or file.seekpp,ios::cur
i.e to move to **p** byte position from beginning, end or current position.

tellg(: This function returns the current working position of the file pointer in the input mode.
e.g int p=file.tellg(;

tellp: This function returns the current working position of the file pointer in the output mode.
e.f int p=file.tellp;

Steps To Create A File

- Declare an object of the desired file stream class(ifstream, ofstream, or fstream
- Open the required file to be processed using constructor or open function.
- Process the file.
- Close the file stream using the object of file stream.

General program structure used for creating a Text File

<u>To create a text file using strings I/O</u>

```
#include<fstream.h>    //header file for file operations
void main(
{
char s[80], ch;
ofstream file“myfile.txt”;     //open myfile.txt in default output mode
do
{
cout<<” n enter line of text”;
gets(s;  //standard input
file<<s;  // write in a file myfile.txt
cout<<” n more input y/n”;
```

cin>>ch;
}whilech!='n'||ch!='N';
file.close;
} **//end of main**

To create a text file using characters I/O

#include<fstream.h>   **//header file for file operations**
void main(
{
char ch;
ofstream file"myfile.txt";   **//open myfile.txt in default output mode**
do{
ch=getche;
if (ch==13) //check if character is enter key
cout<<' n';
else
file<<ch; // write a character in text file 'myfile.txt '
} whilech!=27); // check for escape key
file.close;
} **//end of main**

Text files in input mode:

To read content of 'myfile.txt' and display it on monitor.
#include<fstream.h>   **//header file for file operations**
void main(
{
char ch;
ifstream file"myfile.txt";   **//open myfile.txt in default input mode**
whilefile
{
file.getch) // read a character from text file 'myfile.txt'
cout<<ch; // write a character in text file 'myfile.txt '
}
file.close;
} **//end of main**

2 Marks Questions:

Write a function in a C++ to read the content of a text file "DELHI.TXT" and display all those lines on screen, which are either starting with 'D' or starting with 'M'. [CBSE 2012]

```
void DispDorM(
{
        ifstream File"DELHI.TXT"
        char str[80];
        whileFile.getlinestr,80)
        {
                ifstr[0] = ='D' || str[0] = ='M'
                        cout<<str<<endl;
        }
        File.close; //Ignore
}
```

Write a function in a C++ to count the number of lowercase alphabets present in a text file "BOOK.txt".

```
int countalpha
{       ifstream Fin("BOOK.txt";
        char ch;
        int count=0;
        while!Fin.eof
                {Fin.getch);
```

```
                    if islowerch)
                              count++;
                    }
          Fin.close;
          return count;
}
```

Function to calculate the average word size of a text file.

```
     void calculate
     {    fstream File;
          File.open(“book.txt”,ios::in);
          char a[20];
          char ch;
          int i=0,sum=0,n=0;
          whileFile
          {     File.getch;
               a[i]=ch;
               i++;
               ifch==’ ‘ || ch== ‘.’||char==’,’ch==’          t’||ch==’    n’
                    {i --;  sum=sum +i;
                    i=0;  N++;
               }
          }
          cout<<”average word size is “<<sum/n);
          }
```

Assume a text file “coordinate.txt” is already created. Using this file create a C++ function to count the number of words having first character capital.

```
int countword(
{         ifstream Fin(“BOOK.txt”;
          char ch[25];
          int count=0;
          while!Fin.eof
                    {Fin>>ch;
                    if (isupperch[0]
                              count++;
                    }
          Fin.close;
          return count;
}
```

Function to count number of lines from a text files a line can have maximum 70 characters or ends at ‘.’

```
int countword(
{         ifstream Fin(“BOOK.txt”;
          char ch[70];
          int count=0;
          if (!Fin)
          {         cout<<”Error opening file!” ;
                    exit0);
          }
          while1)
          {Fin.getlinech,70,'.';
                    if (Fin.eof
                              break;
                    count++;
          }
          Fin.close;
          return count;
          }
```

2/3 Marks Practice Questions

1. Write a function in C++ to count the number of uppercase alphabets present in a text file "BOOK.txt"
2. Write a function in C++ to count the number of alphabets present in a text file "BOOK.txt"
3. Write a function in C++ to count the number of digits present in a text file "BOOK.txt"
4. Write a function in C++ to count the number of white spaces present in a text file "BOOK.txt"
5. Write a function in C++ to count the number of vowels present in a text file "BOOK.txt"
6. Assume a text file "Test.txt" is already created. Using this file, write a function to create three files "LOWER.TXT" which contains all the lowercase vowels and "UPPER.TXT" which contains all the uppercase vowels and "DIGIT.TXT" which contains all digits.

General program structure used for operating a Binary File

Program to create a binary file 'student.dat' using structure.

```
#include<fstream.h>
struct student
{
char name[15];
float percent;
};
void main(
{
          ofstream fout;
          char ch;
          fout.open("student.dat", ios::out | ios:: binary;
          clrscr;
          student s;
          if!fout
          {
                    cout<<"File can't be opened";
                    break;
          }
          do
          {
          cout<<" n enter name of student";
          gets(s;
          cout<<" n enter persentage";
          cin>>percent;
          fout.writechar *&s,sizeofs);  // writing a record in a student.dat file
          cout<<" n more record y/n";
          cin>>ch;
          }whilech!='n' || ch!='N';
          fout.close;
}
```

Program to read a binary file 'student.dat' display records on monitor.

```
#include<fstream.h>
struct student
{
char name[15];
float percent;
};
void main(
{
        ifstream fin;
        student s;
        fin.open("student.dat",ios::in | ios:: binary;
        fin.read(char *) &s, sizeofstudent;  //read a record from file 'student.dat'
        whilefile
        {
        cout<<s.name;
        cout<<" n has the percent: "<<s.percent;
        fin.read(char *) &s, sizeofstudent;
        }
        fin.close;
}
```

Binary file using Objects and other file operations:

Consider the following class declaration then write c++ function for following file operations viz create_file, read_file, add new records, modify record, delete a record, search for a record.

```
#include<iostream.h>
class student
{
        int rno;
        char name[30];
        int age;
public:
        void input
        {
                cout<<" n enter roll no";
                cin>>rno;
                cout<<" n enter name ";
                gets(name;
                cout<<" n enter age";
                cin>>age;
        }
        void output
        {
                cout<< " n roll no:"<<rno;
                cout<< " n name :"<<name;
                cout<< " n age:"<<age;
        }
        int getrno( { return rno;}
};
void create_file
{
        ofstream fout;
        char ch;
        fout.open("student", ios::out | ios:: binary;
        clrscr;
        student s;
        if!fout
        {cout<<"File can't be opened";
                break;
        }
        do
        {
        s.input;
        cout<<" n more record y/n";
        cin>>ch;
        }whilech!='n' || ch!='N';
        fout.close;
}
void read_file
{
        ifstream fin;
        student s;
        fin.open"st udent.dat",ios::in | ios:: binary;
        fin.read(char *) &s,sizeofstudent;
        whilefile
        {
        s.output;
        cout<< " n";
        fin.read(char *) & s,sizeofstudent;
        }
        fin.close;
}
void modify_record
{         student s;
        fstream file;
        file.open("student.dat",ios::in|ios::out|ios::ate|ios::binary;
        int r,pos=-1;
        cout<<" n enter the rollo no of student whom data to be modified";
        cin>>r;
        file.readchar *)&s,sizeofs;
        whilefile
        {
                if r==s.getrno(
                {
                cout<<" n record is ";
                s.output;
                pos =file.tellg   -sizes;
                break;
                }
        file.readchar *)&s,sizeofs;
        }
        ifpos> -1)
        {
        cout<< " n enter new     record";
        s.input;
        file.seekp(pos,ios::beg;
        file.writechar *)&s,sizeofs;
        cout<< " n record modified successfully";
        }

        else
        cout<< " n record not exist";
}
```

```
void delete_record
{
fstream file“student.dat”, ios::in|ios::binary;
fstream newfile“newstu.dat”,ios::out|ios::binary;
student s;
cout<<” n enter the rollno no of student whom record to be deleted”;
cin>>r;
file.readchar *)&s,sizeofs;
whilefile
        {
                if r!=s.getrno(
                {
                newfile.writechar *&s,sizeofs;
                }
        file.readchar *)&s,sizeofs;
        }
file.close;
newfile.close;
}
void search_record
{
student s;
        fstream file;
        file.open(“student.dat”,ios::in|os::binary;
        int r,flag=-1;
        cout<<” n enter the rollo no of student whom record to be searched”;
        cin>>r;
        file.readchar *)&s,sizeofs;
        whilefile
        {
                if r==s.getrno(
                {
                cout<<” n record is “;
                s.output;
                flag=1;
                break;
                }
        file.readchar *)&s,sizeofs;
        }
        ifflag==1)
        cout<< “ n search successfull”;
        else
        cout<< “ n search unsuccessfull”;
        file.close;
}
```

1 Mark Questions

1. **Observe the program segment carefully and answer the question that follows:**

```
class stock
{
        int Ino, Qty; Char Item[20];
public:
        void Enter { cin>>Ino; get  s(Item; cin>>Qty;}
        void issueint Q { Qty+=0;}
        void Purchaseint Q {Qty  -=Q;}
        int GetIno { return Ino;}
};
void PurchaseItemint Pino, int PQty
{       fstream File;
```

```
          File.open(“stock.dat”, ios::binary|ios::in|ios::out;
          Stock s;
          int success=0;
          while success= = 0 && File.read(char *&s,sizeofs)
          {
                    IfPino= = ss.GetIno(
                    {
                              s.PurchasePQty;
                              ________________________    // statement 1
                              ________________________    // statement 2
                              Success++;
                    }
          }
if (success = =1)
          cout<< “Purchase Updated”<<endl;
else
          cout<< “Wrong Item No”<<endl;
File.close ;
}
```

Ans.1. i Statement 1 to position the file pointer to the appropriate place so that the data updation is done for the required item.

File.seekpFile.tellg(-sizeofstock;
OR
File.seekp -sizeofstock,ios::cur;

ii Staement 2 to perform write operation so that the updation is done in the binary file.
File.writechar *)&s, sizeofs); OR File.writechar *)&s, sizeofstock;

3 Marks Question

2. **Write a function in c++ to search for details Phoneno and Calls) of those Phones which have more than 800 calls from binary file “phones.dat”. Assuming that this binary file contains records/ objects of class Phone, which is defined below.**

class Phone **CBSE 2012**

```
{
          Char Phoneno[10]; int  Calls;
public:
          void Get {gets(Phoneno); cin>>Calls;}
          void Billing { cout<<Phoneno<< “#”<<Calls<<endl;}
          int GetCalls( {return Calls;}
};
```

Ans 2 : void Search(

```
{
          Phone P;
          fstream fin;
          fin.open( “Phone.dat”, ios::binary| ios::in);
          whilefin.read(char *&P, sizeofP
          {
                    ifp.GetCalls( >800
                              p.Billing;
          }
          Fin.close; //ignore
}};
```

3. **Write a function in C++ to add new objects at the bottom of a binary file "STUDENT.DAT", assuming the binary file is containing the objects of the following class.**

```
class STUD
{int Rno;
char Name[20];
public:
void Enter
{cin>>Rno;gets(Name;}
void Display{cout<<Rno<<Name<<endl;}
};
```

Ans.3. void searchbook(int bookno)

```
{ifstream ifile"BOOK.DAT",ios::in|ios::binary;
 if!ifile
        {cout<<"could not open BOOK.DAT file"; exit -1);}
else
        {BOOK b; int found=0;

                whileifile.read(char *&b, sizeofb)
                {ifb.RBno(==bookno
                        {b.Display; found=1; break;}
                }
        if! found)
                cout<<"record is not found ";
        ifile.close;
        }
}
```

4. **Given a binary file PHONE.DAT, containing records of the following class type**

```
 class Phonlist
{
char name[20];
char address[30];
char areacode[5];
char Phoneno[15];
public:
void Register
void Show;
void CheckCodechar AC[]
{return(strcmpareacode,AC;
}};
```

Write a function TRANSFER) in C++, that would copy all those records which are having areacode as "DEL" from PHONE.DAT to PHONBACK.DAT.

Ans 4. void TRANSFER

```
{
        fstream File1,File2;
        Phonelist P;
        File1.open("PHONE.DAT", ios::binary|ios::in);
        File2.open("PHONEBACK.DAT", ios::binary|ios::OUT)
        whileFile1.read(char *)&P, sizeofP
        {               if p.CheckCode "DEL"
                        File2.writechar *)&P,sizeofP;           }
       File1.close;
       File2.close;
}
```