Binary Files

Opening a binary file in OUTPUT mode (For writing on file)

At the time of opening the file in output mode the earlier content (if existing) gets deleted.

```
Syntax:
fstream <FileObject>;
<FileObject>.open(<FileName>,ios::binary|ios::out);
```

```
Examples:
fstream  FIL;
FIL.open("STUDENT.DAT",ios::binary|ios::out);
OR
char Fname[20];
cout<<"FileName:";gets(Fname);
fstream Fil;
Fil.open(Fname,ios::binary|ios::out);
```

Opening a binary file in INPUT mode (For reading from file).

```
Syntax:
fstream <FileObject>;
<FileObject>.open(<FileName>,ios::binary|ios::in);
```

```
Examples:
fstream  FIL;
FIL.open("STUDENT.DAT",ios::binary|ios::in);
OR
char Fname[20];
cout<<"Enter Filename:";gets(Fname);
fstream Fil;
Fil.open(Fname,ios::binary|ios::in);
```

Opening a binary file in APPEND mode (For writing on file)

At the time of opening the file in append mode the earlier content (if existing) is not over written and file write pointer moves to the end of file for adding new records at the bottom of file.

```
Syntax:
fstream <FileObject>;
<FileObject>.open(<FileName>,ios::binary|ios::app);
```

```
Examples:
fstream  FIL;
FIL.open("STUDENT.DAT",ios::binary|ios::app);
OR
char Fname[20];
cout<<"Filename:";gets(Fname);
fstream Fil;
Fil.open(Fname,ios::binary|ios::app);
```

Binary files can be created in c++ with the help of variables of struct type data or objects of class.

```
struct Stock
{
  int INo;
  char Item[20];
  float Qty;
};
```

```
class Stock
{
  int INo;
  char Item[20];float Qty;
public:
  void Input()
  {cin>>INo;gets(Item);cin>>Qty;}
  void Output()
  {cout<<INo<<':'<<Item<<':'<<Qty;}
  int Rino(){return Ino;}
};
```

Creation of a binary file with a record

```
void Create(char Fname[])
{
  fstream Fil; Stock S;
  Fil.open(Fname,ios::binary|ios::out);
  cout<<"Item No?";cin>>S.INo;
  cout<<"Item   ?";gets(S.Item);
  cout<<"Qty    ?";cin>>S.Qty;
  Fil.write((char*)&S,sizeof(S));
  Fil.close();
}
```

```
void Create(char Fname[])
{
  fstream Fil; Stock S;
  Fil.open(Fname,ios::binary|ios::out);
  S.Input();


  Fil.write((char*)&S,sizeof(S));
  Fil.close();
}
```

Creation of a binary file with multiple records

```
void Create(char Fname[])
{
  fstream Fil; Stock S;
  Fil.open(Fname,ios::binary|ios::out);
  char Choice;
  do
  {
    cout<<"Item No?";cin>>S.INo;
    cout<<"Item   ?";gets(S.Item);
    cout<<"Qty    ?";cin>>S.Qty;
    Fil.write((char*)&S,sizeof(S));
    cout<<"More(Y/N)?";cin>>Choice;
  }
  while (Choice=='Y');
  Fil.close();
}
```

```
void Create(char Fname[])
{
  fstream Fil; Stock S;
  Fil.open(Fname,ios::binary|ios::out);
  char Choice;
  do
  {
    S.Input();


    Fil.write((char*)&S,sizeof(S));
    cout<<"More(Y/N)?";cin>>Choice;
  }
  while (Choice=='Y');
  Fil.close();
}
```

Displaying a record from a binary file

```
void Display(char Fname[])
{
  fstream Fil; Stock S;
  Fil.open(Fname,ios::binary|ios::in);
  Fil.read((char*)&S,sizeof(S));
  cout<<"ItemNo:"<<S.INo<<endl;
  cout<<S.Item<<":Qty:"<<S.Qty<<endl;
  Fil.close();
}
```

```
void Display(char Fname[])
{
  fstream Fil; Stock S;
  Fil.open(Fname,ios::binary|ios::in);
  Fil.read((char*)&S,sizeof(S));
  S.Output();

  Fil.close();
}
```

Displaying the entire content (all records) from a binary file

```
void Display(char Fname[])
{
  fstream Fil; Stock S;
  Fil.open(Fname,ios::binary|ios::in);
  while(Fil.read((char*)&S,sizeof(S)))
  {
   cout<<"ItemNo:"<<S.INo<<endl;
   cout<<S.Item<<":Qty:"<<S.Qty<<endl;
  }
  Fil.close();
}
```

```
void Display(char Fname[])
{
  fstream Fil; Stock S;
  Fil.open(Fname,ios::binary|ios::in);
  while (Fil.read((char*)&S,sizeof(S)))
    S.Output();


  Fil.close();
}
```

Displaying selected range of records from a binary file

(Record having a particular range of Item Numbers)

```
void Display(char Fname[],int NOF,int NOT)
{       //NOF-Item No From & NOT-Item No To
  fstream Fil; Stock S;
  Fil.open(Fname,ios::binary|ios::in);
  while(Fil.read((char*)&S,sizeof(S)))
  {
   if (S.INo>=NOF && S.INo<=NOT)
   {
    cout<<"ItemNo:"<<S.INo<<endl;
    cout<<S.Item<<":Qty:"<<S.Qty<<endl;
   }
  }
  Fil.close();
}
```

```
void Display(char Fname[],int NOF,int NOT)
{       //NOF-Item No From & NOT-Item No To
  fstream Fil; Stock S;
  Fil.open(Fname,ios::binary|ios::in);

  while (Fil.read((char*)&S,sizeof(S)))
    if (S.Rino()>=NOF && S.Rino()<=NOT)
      S.Output();




  Fil.close();
}
```

Searching for a record from a binary file

```
void Display(char Fname[])
{
  fstream Fil; Stock S;
  int InoS,Found=0;
  Fil.open(Fname,ios::binary|ios::in);
  cout<<"Enter ItemNo to search:";
  cin>>InoS;
  while(Fil.read((char*)&S,sizeof(S)))
  {
   if(S.INo==InoS)
   {
     cout<<"ItemNo:"<<S.INo<<endl;
     cout<<S.Item<<":Qty:"<<S.Qty<<endl;
     Found++;
   }
  }
  if (Found==0)
   cout<<"Record Not Found"<<endl;
  Fil.close();
}
```

```
void Display(char Fname[])
{
  fstream Fil; Stock S;
  int InoS,Found=0;
  Fil.open(Fname,ios::binary|ios::in);
  cout<<"Enter ItemNo to search:";
  cin>>InoS;
  while(Fil.read((char*)&S,sizeof(S)))
  {
   if(S.Rino()==InoS)
   {
    S.Output();

    Found++;
   }
  }
  if (Found==0)
   cout<<"Record Not Found"<<endl;
  Fil.close();
}
```

Counting no. of records in a binary file

```
int CountRecord(char Fname[])
{
  fstream Fil; Stock S;
  int Count=0;
  Fil.open(Fname,ios::binary|ios::in);
  while(Fil.read((char*)&S,sizeof(S)))
    Count++;
  Fil.close();
  return Count;
}
```

```
int CountRecord(char Fname[])
{
  fstream Fil; Stock S;
  int Count=0;
  Fil.open(Fname,ios::binary|ios::in);
  while(Fil.read((char*)&S,sizeof(S)))
    Count++;
  Fil.close();
  return Count;
}
```

Transferring the content of a binary file to another

```
void Transfer(char FN1[],char FN2[])
{
  fstream Fil1,Fil2;
  Stock S;
  Fil1.open(FN1,ios::binary|ios::in);
  Fil2.open(FN2,ios::binary|ios::out);
  while (Fil1.read((char*)&S,sizeof(S)))
    Fil2.write((char*)&S,sizeof(S));
  Fil1.close();
  Fil2.close();
}
```

```
void Tranfer(char FN1[],char FN2[])
{
  fstream Fil1,Fil2;
  Stock S;
  Fil1.open(FN1,ios::binary|ios::in);
  Fil2.open(FN2,ios::binary|ios::out);
  while (Fil1.read((char*)&S,sizeof(S)))
    Fil2.write((char*)&S,sizeof(S));
  Fil1.close();
  Fil2.close();
}
```

Insertion of a record in a Binary File (Sequential)

(Assuming file to be in ascending order of Item No)

```
void Insertion(char FN1[],char FN2[])
{
  fstream Fil1,Fil2;
  Stock S,Snew;
  Fil1.open(FN1,ios::binary|ios::in);
  Fil2.open(FN2,ios::binary|ios::out);
  cout<<"Enter New Record"<<endl;
  cout<<"Item No?";cin>>Snew.INo;
  cout<<"Item   ?";gets(Snew.Item);
  cout<<"Qty    ?";cin>>Snew.Qty;
  int Inse=0;
  while (Fil1.read((char*)&S,sizeof(S)))
  {
    if (Snew.INo<S.INo && !Inse)
    {
    Fil2.write((char*)&Snew,sizeof(Snew));
    Inse++;
    }
    Fil2.write((char*)&S,sizeof(S));
  }
  if (!Inse)
    Fil2.write((char*)&Snew,sizeof(Snew));
  Fil1.close();
  Fil2.close();
}
```

```
void Insertion(char FN1[],char FN2[])
{
  fstream Fil1,Fil2;
  Stock S,Snew;
  Fil1.open(FN1,ios::binary|ios::in);
  Fil2.open(FN2,ios::binary|ios::out);
  cout<<"Enter New Record"<<endl;
  Snew.Input();


  int Inse=0;
  while (Fil1.read((char*)&S,sizeof(S)))
  {
    if (Snew.Rino()<S.RINo() && !Inse)
    {
    Fil2.write((char*)&Snew,sizeof(Snew));
    Inse++;
    }
    Fil2.write((char*)&S,sizeof(S));
  }
  if (!Inse)
    Fil2.write((char*)&Snew,sizeof(Snew));
  Fil1.close();
  Fil2.close();
}
```

Deletion of a record from binary file

```
void Deletion(char FN1[],char FN2[])
{
  fstream Fil1,Fil2;
  Stock S;
  int InoD;//Itemno to be deleted
  Fil1.open(FN1,ios::binary|ios::in);
  Fil2.open(FN2,ios::binary|ios::out);
  cout<<"Enter ItemNo to be deleted:";
  cout<<"Item No?";cin>>InoD;
  int Dele=0;
  while (Fil1.read((char*)&S,sizeof(S)))
  {
    if (S.INo!=InoD)
      Fil2.write((char*)&S,sizeof(S));
    else
      Dele++;
  }
  if (!Dele)
    cout<<"Record match not found…"<<endl;
  else
    cout<<"Record deleted…"<<endl;
  Fil1.close();
  Fil2.close();
}
```

```
void Deletion(char FN1[],char FN2[])
{
  fstream Fil1,Fil2;
  Stock S;
  int InoD; //Itemno to be deleted
  Fil1.open(FN1,ios::binary|ios::in);
  Fil2.open(FN2,ios::binary|ios::out);
  cout<<"Enter ItemNo to be deleted:";
  cout<<"Item No?";cin>>InoD;
  int Dele=0;
  while (Fil1.read((char*)&S,sizeof(S)))
  {
    if (S.Rino()!=InoD)
      Fil2.write((char*)&S,sizeof(S));
    else
      Dele++;
  }
  if (!Dele)
    cout<<"Record match not found…"<<endl;
  else
    cout<<"Record deleted…"<<endl;
  Fil1.close();
  Fil2.close();
}
```

Adding a record at the end of an existing binary file

```
void Create(char Fname[])
{
  fstream Fil; Stock S;
  Fil.open(Fname,ios::binary|ios::app);
  cout<<"Item No?";cin>>S.INo;
  cout<<"Item   ?";gets(S.Item);
  cout<<"Qty    ?";cin>>S.Qty;
  Fil.write((char*)&S,sizeof(S));
  Fil.close();
}
```

```
void Create(char Fname[])
{
  fstream Fil; Stock S;
  Fil.open(Fname,ios::binary|ios::app);
  S.Input();


  Fil.write((char*)&S,sizeof(S));
  Fil.close();
}
```

```
#include <fstream.h>
#include <conio.h>
#include <stdio.h>
#include <string.h>
#include <ctype.h>
class EMP
{
  int Eno;
  char Name[20], Address[40];
public:
  int GetEno(){return Eno;}
  int Check(char NAMES[]){return strcmp(Name,NAMES);}
  char *GetEname(){return Name;}
  void In();
  void EditAddress();
  void Out();
};
void EMP::In()
{
  cout<<"Eno    :";cin>>Eno;
  cout<<"Name   :";gets(Name);
  cout<<"Address       :";gets(Address);
}
void EMP::EditAddress()
{
  cout<<"Address :";gets(Address);
}
void EMP::Out()
{
  cout<<"Eno     :"<<Eno<<endl;
  cout<<"Name    :"<<Name<<" Address :"<<Address<<endl;
}
void Create();              //To create a binary file with records of EMP class
void Addatend();            //To add new records in a binary file with records of EMP class
void Display();             //To display the contents of binary file
void Search();              //To search
void SearchName();
int Count(char FN[]);
```

To sort a binary file in ascending order of Eno (Numeric Field)

```
void SortEno()
{
  fstream F;
  F.open("EMP.DAT",ios::binary|ios::in|ios::out);
  //Move to the last record
  F.seekg(0,ios::end);           //To move the record pointer to end of file
  int NOR=F.tellg()/sizeof(EMP); //To find number of records in the file
  EMP EJ,EJP1;
  for (int i=0;i<NOR-1;i++)
    for (int j=0;j<NOR-i-1;j++)
    {
      F.seekg(j*sizeof(EMP));    //To move the file pointer to jth position
      F.read((char*)&EJ,sizeof(EMP));   //reads jth record
      F.read((char*)&EJP1,sizeof(EMP)); //reads (j+1)th record
      if (EJ.GetEno()>EJP1.GetEno())
      {
        F.seekp(j*sizeof(EMP));
        F.write((char*)&EJP1,sizeof(EMP));
        F.write((char*)&EJ,sizeof(EMP));
      }
    }
  F.close();
}
```

To sort a binary file in ascending order of Ename (String)

```
void SortEname()
{
  fstream F;
  F.open("EMP.DAT",ios::binary|ios::in|ios::out);
  F.seekg(0,ios::end);          //To move the record pointer to end of file
  int NOR=F.tellg()/sizeof(EMP);//To find number of records in the file
  EMP EJ,EJP1;
  for (int i=0;i<NOR-1;i++)
    for (int j=0;j<NOR-i-1;j++)
    {
      F.seekg(j*sizeof(EMP));   //To move the file pointer to jth position
      F.read((char*)&EJ,sizeof(EMP));  //reads (j)th record
      F.read((char*)&EJP1,sizeof(EMP));//reads (j+1)th record
      if (strcmpi(EJ.GetEname(),EJP1.GetEname())>0)
      {
        F.seekp(j*sizeof(EMP));
        F.write((char*)&EJP1,sizeof(EMP));
        F.write((char*)&EJ,sizeof(EMP));
      }
    }
  F.close();
}
```

To edit <Address> field of a particular Record in a binary file

```
void Modify()
{
  fstream F;
  F.open("EMP.DAT",ios::binary|ios::in|ios::out);
  EMP E;  int Seno, Found=0;
  cout<<"Enter the emp no:";cin>>Seno;
//Search for a record matching with Eno whose Address is to be changed
  while (!Found && F.read((char*)&E,sizeof(E)))
    if (E.GetEno()==Seno) Found++;
  if (Found)
  {
    E.EditAddress();
    int N=F.tellg()-sizeof(E);//Finds the searched record's position
    F.seekp(N);              //Moves file pointer for re-writing edited record
    F.write((char*)&E,sizeof(EMP));
  }
  else
    cout<<"No record matching for editing ... "<<endl;
  F.close();
}
void main()
{
  char Ch;
  do
  { cout<<"C:Create/D:display/A:Append/S;Search/N:Name/E:Enowise/M:Namewise/F:Modify/Q:Quit ";
    cin>>Ch;
    switch(Ch)
    {case 'C':Create();break;
     case 'A':Addatend();break;
     case 'D':Display();break;
     case 'S':Search();break;
     case 'N':SearchName();break;
     case 'E':SortEno();break;
     case 'M':SortEname();break;
     case 'F':Modify();break;
    }
  }
  while (Ch!='Q');
}
```

Functions/Keywords associated with File Handling in C++

`sizeof`

A **keyword** used in C++, which returns the no. of bytes required by a particular variable/object or a data type.

Example

```
int A; float B;char T,P[25];
cout<<sizeof(A);//will display 2
cout<<sizeof(B);//will display 4
cout<<sizeof(T);//will display 1
cout<<sizeof(P);//will display 25
```

`open`

A member function of fstream, ifstream and ofstream used to open a file object in a particular stream.

Header File <fstream.h>

Syntax

```
<fstream object>.open(<char*- Filename>,<int- Filemode>);
```

Example

```
fstream Fil;
Fil.open("STDNT.DAT",ios::binary|ios::in);
```

`close`

A member function of fstream, ifstream and ofstream used to close a file object.

Header File <fstream.h>

Syntax:

```
<fstream object>.close();
```

Example

```
fstream Fil;
Fil.open("STDNT.DAT",ios::binary|ios::in);
:
Fil.close();
```

`read`

A member function of istream/ifstream/fstream, used to extract a no. of characters.

Header Files <iostream.h> and <fstream.h>

Syntax

```
<fstream object>.read(<char*>,<int>);
```

Example

```
fstream Fil;Student S;
Fil.open("STDNT.DAT",ios::binary|ios::in);
Fil.read((char*)&S,sizeof(S));
S.Output();
```

`write`

A member function of ostream/ofstream/fstream, used to insert a no. of characters.

Header Files <iostream.h> and <fstream.h>

Syntax

```
<fstream object>.write(<char*>,<int>);
```

Example

```
fstream Fil;Student S;
Fil.open("STDNT.DAT",ios::binary|ios::out);
S.Input();
Fil.write((char*)&S,sizeof(S));
Fil.close();
```

`eof`

A member function of ios, returns a non-zero value at the end of file.

Header Files <iostream.h> and <fstream.h>

Syntax

```
eof(<fstream object>)
```

Example

```
fstream Fil;Student S;
Fil.open("STDNT.DAT",ios::binary|ios::in);
Fil.read((char*)&S,sizeof(S));
if (!Fil.eof()) S.Output();
Fil.close();
```

`get`

A member function of istream/ifstream/fstream, used to extract next character/eof.

Header Files <iostream.h> and <fstream.h>

Syntax

```
<ifstream/istream object>.get();
```

Example

```
ifstream Fil("LETTER.MYN");char ch;
ch=Fil.get();
while (!Fil.eof())
{
  cout<<ch;
  ch=Fil.get();
}
Fil.close();
```

put

A member function of ostream/ofstream, used to insert a character.

Header Files <iostream.h> and <fstream.h>

Syntax

```
<ofstream/ostream object>.put(<char>);
```

Example

```
ofstream Fil("LETTER.MYN");char ch;
do
{
  ch=getche();
  if (ch!=27) Fil.put(ch);
}
while (ch!=27);
Fil.close();
```

getline

A member function of istream, used to extract characters upto the size given/upto the delimeter whichever encountered first.

Header Files <iostream.h> and <fstream.h>

Syntax

```
<istream/ifstream object>.getline(<char*>,<int>);
<istream/ifstream object>.getline(<char*>,<int>,<char>);
```

Example

```
ifstream Fil;char Str[80];
Fil.getline(Str,80);
while (!Fil.eof())
{
  cout<<Str<<endl;
  Fil.getline(Str,80);
}
Fil.close();
```

tellg

A member function of ifstream, used to return the current byte position in a binary file.

Header Files <fstream.h>

Syntax

```
int <ifstream object>.tellg();
```

Example

```
fstream Fil;Fil.open(“EMP.DAT”,ios::binary|ios::in);
EMP E;
Fil.read((char*)&E,sizeof(E));//Reading first record
Fil.read((char*)&E,sizeof(E));//Reading second record
int N=Fil.tellg()/sizeof(EMP);
cout<<”Record Number:”<<N<<endl;
Fil.close();
```

tellp

A member function of ofstream, used to return the current byte position in a binary file.

Header Files <fstream.h>

Syntax

```
int <ifstream object>.tellp();
```

Example

```
fstream Fil;
Fil.open(“EMP.DAT”,ios::binary|ios::in|ios::out);
EMP E,F;
Fil.read((char*)&E,sizeof(E));//Reading first record
```

```
F.Input();
Fil.write((char*)&F,sizeof(F));//Writing second record
int N=Fil.tellp()/sizeof(EMP);
cout<<"Record Number:"<<N<<endl;
Fil.close();
```

seekg A member function of ifstream, used to move the file pointer to a specified byte position in a binary file.

Header Files <fstream.h>

Syntax

```
<ifstream object>.seekg(<Byte Position>);
```

Example

```
fstream Fil;Fil.open("EMP.DAT",ios::binary|ios::in);
EMP E;
Fil.seekg(3*sizeof(EMP));//Moves the file pointer to 4th record
Fil.read((char*)&E,sizeof(E));//Reading 4th record
E.Display();
Fil.close();
```

seekp A member function of ofstream, used to move the file pointer to a specified byte position in a binary file.

Header Files <fstream.h>

Syntax

```
<ofstream object>.seekp(<Byte Position>);
```

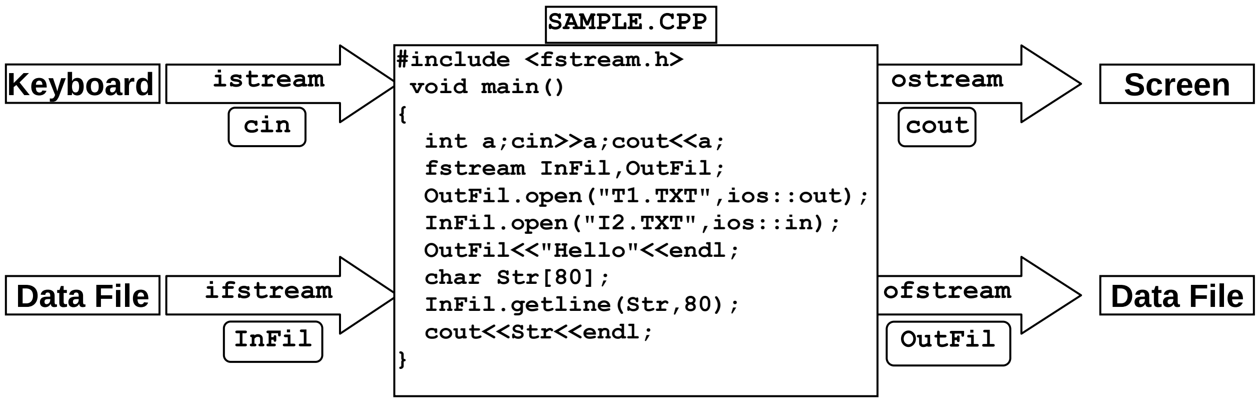
Example

```
fstream Fil;
Fil.open("EMP.DAT",ios::binary|ios::in|ios::out);
EMP E;
E.Input();
Fil.seekp(3*sizeof(EMP));//Moves the file pointer to 4th record
Fil.write((char*)&E,sizeof(E));//Writing 4th record
Fil.close();
```

Streams

A stream is used to manage flow of bytes. An input stream (**istream**) manages bytes flowing into the program, and an output stream (**ostream**) manages bytes flowing out of the program. Instance (object) of the class istream (**cin**) represents the standard input device (i.e. the keyboard). Instance (object) of the class ostream (**cout**) manages the output of data from the program to the standard output device (i.e. screen/monitor).

In C++, a data file is simply an external stream (a sequence of bytes stored on a disk). If a file is opened for output (for writing on to the file), then it is an output file stream (**ofstream**). If the file is opened for input (for reading from file), then it is an input file stream (**ifstream**).



```
#include <fstream.h>
 void main()
{
  int a;cin>>a;cout<<a;
  fstream InFil,OutFil;
  OutFil.open("T1.TXT",ios::out);
  InFil.open("I2.TXT",ios::in);
  OutFil<<"Hello"<<endl;
  char Str[80];
  InFil.getline(Str,80);
  cout<<Str<<endl;
}
```

Sample Questions (as per CBSE Board Examination Papers)

1. “SHOW.DAT” is a binary file containing objects of a class PLAY. The class PLAY is defined as follows:

```
class PLAY
{
      char PlayTitle[20];
      int Duration;//Assuming duration in hours
public:
      void NewPlay(){gets(PlayTitle);cin>>Dur;}
      void PlayList(){cout<<PlayTitle<<’:’<<Duration<<endl;}
      int DurationLook(int Dur) {return (Dur==Duration)?1:0;}
};
```

   Write a function in C++ to read the content from “SHOW.DAT” and display those Titles of those whose duration is 3 hours.

2. Assuming a binary file STOCK.DAT containing objects of class STOCK, write functions in C++ to perform the following:
   (i) To get the data values for objects of STOCK from the user and write them into the binary file.
   (ii) To read the objects of STOCK from the binary file and display them on screen.

```
class STOCK
{
      int ITNO;
      char ITEM[10];
public:
      void GETIT(){cin>>ITNO;gets(ITEM);}
      void SHOWIT(){cout<<ITNO<<”   ”<<ITEM<<endl;}
};
```

3. Assuming the class DRINKS defined below; write functions in C++ to perform the following:
   (i) To get the data values for objects of DRINKS from the user and write them into a binary file “DRINKS.DAT”.
   (ii) To read the objects of DRINKS from “DRINKS.DAT” and display the detail the detail of drink, where the name is “INDY COLA”.

```
class DRINKS
{
      int DCODE;
      char  DNAME[13];
      int DSIZE;             // size in litres
      float DPRICE;
public:
      void Getdrinks() {cin>>DCODE>>DNAME>>DSIZE>>DPRICE;}
      void Showdrinks(){cout<<DCODE<<DNAME<<DSIZE<<DPRICE<<endl;}
      char *Getname()  {return DNAME;}
};
```

4. Consider the following class definition

```
class EMPLOYEE
{
   int Code;
   char Name[20];
   float Salary;
public:
   void Input() {cin>>code>>name>>salary;}
   void Show() {cout<<code<<name<<salary<<endl;}
   float Retsal() {return salary;}
};
```

Write functions in C++

i) To get the data values for objects of **EMPLOYEE** from the user and write the objects of **EMPLOYEE** to a binary file "EMP.DAT".

ii) To read the objects of **EMPLOYEE** from binary file "EMP.DAT" and display all the objects on the screen where salary is between Rs. 10000 and Rs. 20000.

5. Consider the following class definition

```
class Bank
{
   int Accno;
   char Name[20];
   float Balance;
public:
   void Input() {cin>>accno>>name>>balance;}
   void Display() {cout<<accno<<name<<balance<<endl;}
   float Getbalance() {return balance;}
};
```

Write functions in C++

i) To accept the objects of class **Bank** from the user and write to a binary file "BANK.DAT".

ii) To read the objects of **Bank** from binary file and display all the objects on the screen where balance is more than Rs. 25000.

6. A binary file "BOOK.DAT" is containing objects of class LIBRARY, which is defined as follows:

```
class LIBRARY
{
      char Title[20], Subject[20];
public:
      void Display(){cout<<Title<<"→"<<Subject<<endl;
      int CheckTitle(char T[]) {strcmp(T,Title);}
};
```

Write a function in C++ to search for the Title of a book desired by the user from the binary file "BOOK.DAT" and display Title and Subject, if the Title of book is present in the file.

7. A binary file "DRUG.DAT" is containing objects of class MEDICAL which is defined as follows:

```
class MEDICAL
{
      char Drug[20], Disease[20];
public:
      void Display(){cout<<Drug<<"→"<<Disease<<endl;
      int CheckDrug(char Dname[]) {strcmp(Drug,DrugName);}
};
```

Write a function in C++ to search for a name of the medicine (Drug) from the binary file "DRUG.DAT" and display Drug and Disease if the medicine is present in the file.

8. Write a program in C++ to perform the following operations related to text file:

- To create a text file "STORY.TXT" from the content typed by the user.
- To display the content of "STORY.TXT" on screen
- To display and count the number of alphabets present in the file "STORY.TXT"