Quick revision in Binary Files in C++

Binary file: A file that contains data in the form of 0 and 1. It is neither readable nor printable using notepad. Character translation does not take place.

ifstream : A class whose objects are used to read from a file.

ofstream : A class whose objects are used to write in a file.

fstream : A class used for both reading and writing.

File Modes:

1. ios::in – for reading from a file
2. ios::out – for writing in a file (always starts fresh)
3. ios::app – for appending in a file
4. ios::ate – for appending by default and also writing any where in the file.
5. ios::in | ios::out – for reading and writing i.e for file modification
6. ios::binary – it has to be explicitly given for binary files

Explanation of few important lines:

1. f1.read((char*)&ob, sizeof(ob));

Suppose sizeof(ob) is 25 bytes. Read 25 bytes from the current position of get pointer in the file **f1** into the address of the object **ob** type casted to a string.

2. f1.write((char*)&ob,sizeof(ob));

Suppose sizeof(ob) is 25 bytes. Write 25 bytes from the address of the object ob type casted to a string from the current position of put pointer in the file f1.

3. f1.seekg(10);   or   *f1.seekg(10,ios::beg);*

sets the get pointer on the 10th byte from the beginning of the file, so that reading can take place from 10th byte onward.

4. f1.seekg(-15,ios::cur);

sets the get pointer on the 15th byte backward from the current position of the get pointer in the file, so that reading can take place from that position onward.

5. f1.tellg( );

returns the current position of the get pointer in the file f1.

```
eg. int pos;
    pos=f1.tellg( );
```

Write the function definitions for the following operations for the class defined below:

```
class student
{   int roll;
```

```
    char name[20];
    float total;
    char result[10];   // ‘Pass’ for pass and ‘Fail’ for fail
    public:
    void input( )
{   cout<<”\nEnter roll: “;        cin>>roll;
    cout<<”\nEnter name : “;    cin>>name;
    cout<<”\nEnter total marks: “;cin>>total;
    cout<<”\nEnter result  : “;   cin>>result; }
void output( )
{   cout<<”\n Roll: “<<roll;
    cout<<”\n Name : “<<name;
    cout<<”\n Total marks: “<<total;
    cout<<”\n Result : “<<result; }
void modify_marks( )
{   total + = 10;}
int ret_roll( )
{   return roll;              }
char *ret_result( )
{   return result;  }};
```

operations:

1. To write students in a binary file “stu.dat” .
2. To add students in a binary file “stu.dat”.
3. To display the students who has passed.
4. To give a grace of 10 marks to those students who has failed.
5. Copy those students in a file named “copy.dat” who has failed and roll number >20
6. To delete the student with a roll number passed as a parameter from the the file “stu.dat”
7. To search for a roll number in the file “stu.dat” and display its detail. If not found display “Unsuccessful”;

Ans1

```
void wtstudent( )
{   ofstream f(“stu.dat”, ios::binary);
  student s;
```

```
           char resp;
           do
           {     s.input( );
             f.write((char*)&s,sizeof(s));
             cout<<”\nWant to enter more (y/n): “;
             cin>>resp;
            }while(resp==’y’);
            f.close( );}

Ans2     void atstudent( )
         {   fstream f;
           f.open(“stu.dat”, ios::app | ios::binary);
           student s;
           char resp;
           do
           {    s.input( );
             f.write((char*)&s,sizeof(s));
             cout<<”\nWant to enter more (y/n): “;
             cin>>resp;
            }while(resp==’y’);
            f.close( ); }
Ans3     void disstudent( )
         {   ifstream f(“stu.dat”, ios::binary);
           student s;
           while(f.read((char*)&s,sizeof(s)))
           {     if(strcmp(s.ret_result(),”Pass”)==0)
                  s.output();   }
            f.close( ); }
Ans4.    void gracestudent( )
         {    fstream f;
           f.open(“stu.dat”, ios::in | ios::out | ios::binary);
           student s;
           while(f.read((char*)&s,sizeof(s)))
           {     if(strcmp(s.ret_result(),”Fail”)==0)
```

```
                {    s.modify_marks();
              f.seekg(-sizeof(s),ios::cur);   // or f.seekg(f.tellg()-sizeof(s),ios::beg)
                     f.write((char*)&s,sizeof(s));
                     break;      }   }
               f.close( );}
Ans5.        void copystudent( )
             {   ifstream f1("stu.dat", ios::binary);
               ofstream f2("copy.dat", ios::binary);
               student s;
               while(f1.read((char*)&s,sizeof(s)))
               {      if(strcmp(s.ret_result(),"Fail")==0 && s.ret_roll( ) > 20)
                      f2.write((char*)&s,sizeof(s));    }
               f1.close( );   f2.close( );}
Ans6.        void delstudent(int r )
             {    ifstream f1("stu.dat", ios::binary);
               ofstream f2("temp.dat", ios::binary);
               student s;
               while(f1.read((char*)&s,sizeof(s)))
               {      if(s.ret_roll()!=r)
                      f2.write((char*)&s,sizeof(s));     }
               f1.close( );    f2.close( );
               remove("stu.dat");     rename("temp.dat","stu.dat"); }
Ans7.        void serstudent( )
             {   ifstream f1("stu.dat", ios::binary);
               student s;
               int r, flag=0;
               cout<<"\nEnter roll number : ";
               cin>>r;
               while(f1.read((char*)&s,sizeof(s)))
               {      if(s.ret_roll()==r)
                 {          flag=1;          s.output();          break;       }    }
               f1.close( );
               if(flag==0)          cout<<"\n Unsuccessful";} }
```