ASSIGNMENT 8

1. **Aim** – Department maintains student information as roll number ,name,division and address. Allow user to add,delete info. Of employee.Display info. Of particular employee.IF employee does not exist an appropriate message is displayed.Use sequential data file.

2. **Objective** – To implement the student database and functions to add, delete student.

3. Theory –

A database consist of a huge amount of data. The data is grouped within a table in RDBMS, and each table have related records. A user can see that the data is stored in form of tables, but in acutal this huge amount of data is stored in physical memory in form of files.

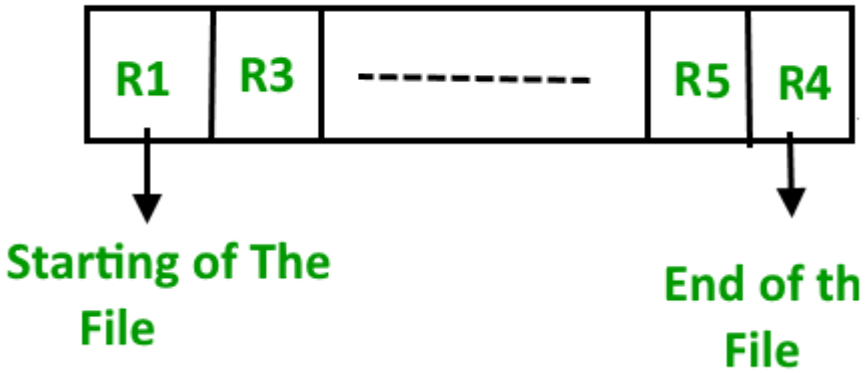
File – A file is named collection of related information that is recorded on secondary storage such as magnetic disks, magnetic tables and optical disks.

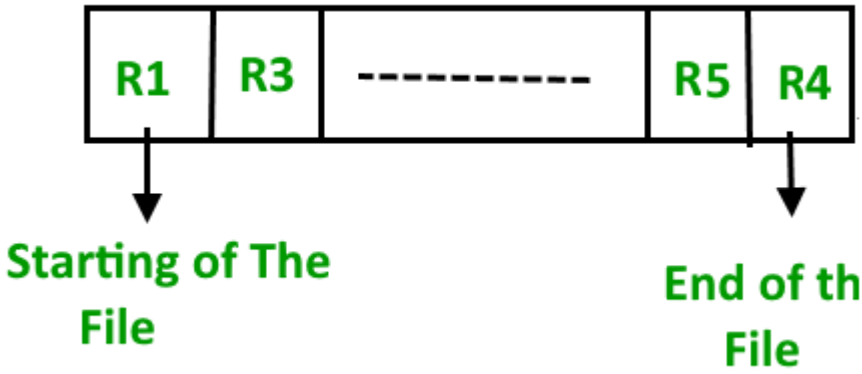
What is File Organization?
File Organization refers to the logical relationships among various records that constitute the file, particularly with respect to the means of identification and access to any specific record. In simple terms, Storing the files in certain order is called file Organization. **File Structure** refers to the format of the label and data blocks and of any logical control record.

Sequential File Organization –

The easiest method for file Organization is Sequential method. In this method the the file are stored one after another in a sequential manner. There are two ways to implement this method:

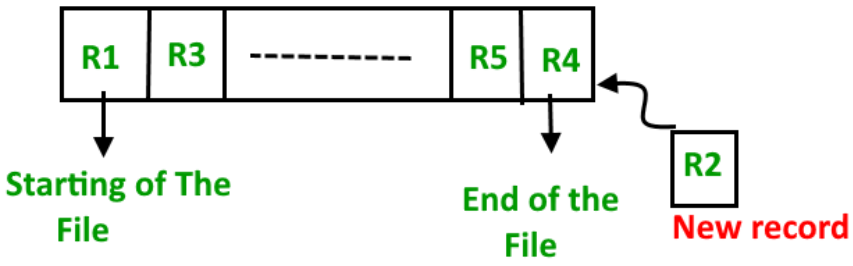
1. **Pile File Method –** This method is quite simple, in which we store the records in a sequence i.e one after other in the order in which they are inserted into the tables.



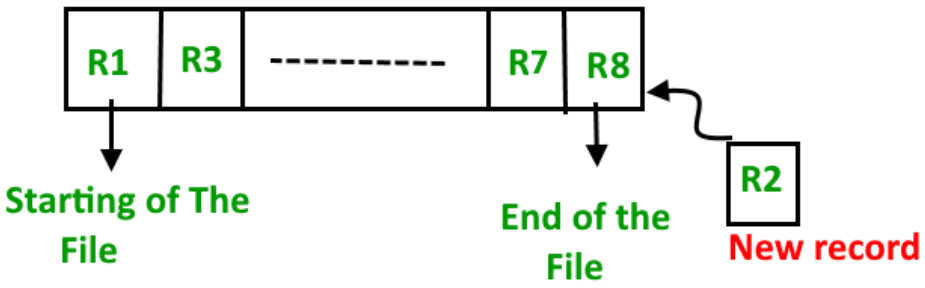


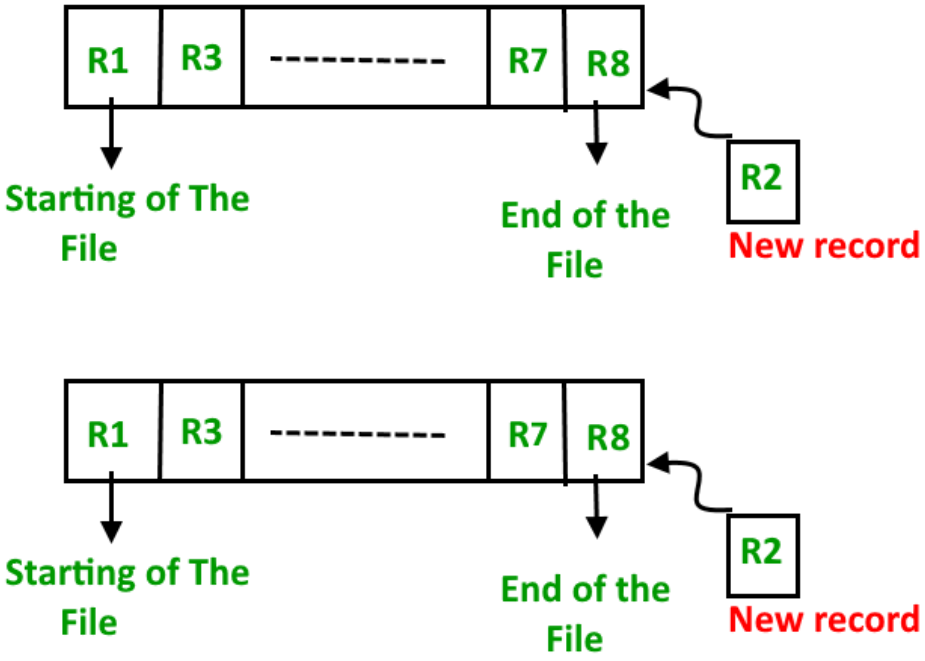
Insertion of new record –

Let the R1, R3 and so on upto R5 and R4 be four records in the sequence. Here, records are nothing but a row in any table. Suppose a new record R2 has to be inserted in the sequence, then it is simply placed at the end of the file.

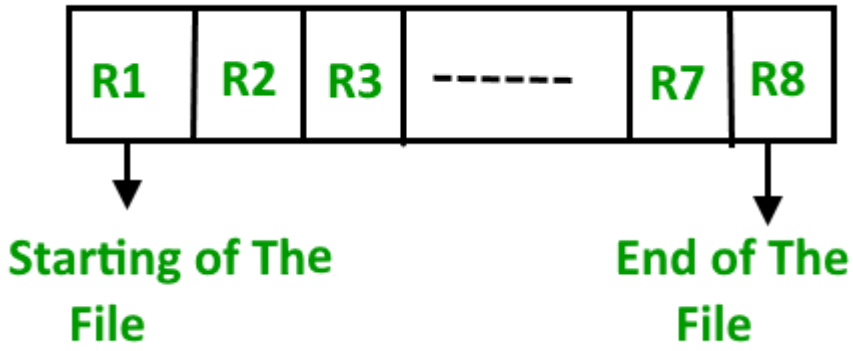


2. **Sorted File Method –**In this method, As the name itself suggest whenever a new record has to be inserted, it is always inserted in a sorted (ascending or descending) manner. Sorting of records may be based on any primary key or any other key.





Insertion of new record –
Let us assume that there is a preexisting sorted sequence of four records R1, R3, and so on upto R7 and R8. Suppose a new record R2 has to be inserted in the sequence, then it will be inserted at the end of the file and then it will sort the sequence .



4. **Algorithm** –

5. **Program code –**

#include <iostream>

```
#include<fstream>
#include<cstring>
#include<iomanip>
using namespace std;
const int MAX=20;
class Student
{
 int rollno;
 char name[20],city[20];
 char div;
 int year;
public:
 Student()
{
  strcpy(name,"");
  strcpy(city,"");
  rollno=year=div=0;
}
 Student(int rollno,char name[MAX],int year,char div,char city[MAX])
 {
  strcpy(this->name,name);
  strcpy(this->city,city);
  this->rollno=rollno;
  this->year=year;
  this->div=div;
 }
```

```
 int getRollNo()
 {
  return rollno;
 }
 void displayRecord()
 {



cout<<endl<<setw(5)<<rollno<<setw(20)<<name<<setw(5)<<year<<setw(5)<<
div<<setw(10)<<city;
 }
};
//==========File Operations ===========
class FileOperations
{
 fstream file;
public:
 FileOperations(char* filename)
{
file.open(filename,ios::in|ios::out|ios::ate|ios::binary);
}
 void insertRecord(int rollno, char name[MAX],int year, char div,char
city[MAX])
 {
  Student s1(rollno,name,year,div,city);
  file.seekp(0,ios::end);
  file.write((char *)&s1,sizeof(Student));
```

```
 file.clear();
}
void displayAll()
{
 Student s1;
 file.seekg(0,ios::beg);
 while(file.read((char *)&s1, sizeof(Student)))
 {
  s1.displayRecord();
 }
 file.clear();
}
void displayRecord(int rollNo)
{
 Student s1;
 file.seekg(0,ios::beg);
 bool flag=false;
 while(file.read((char*)&s1,sizeof(Student)))
 {
  if(s1.getRollNo()==rollNo)
  {
   s1.displayRecord();
   flag=true;
   break;
  }
 }
```

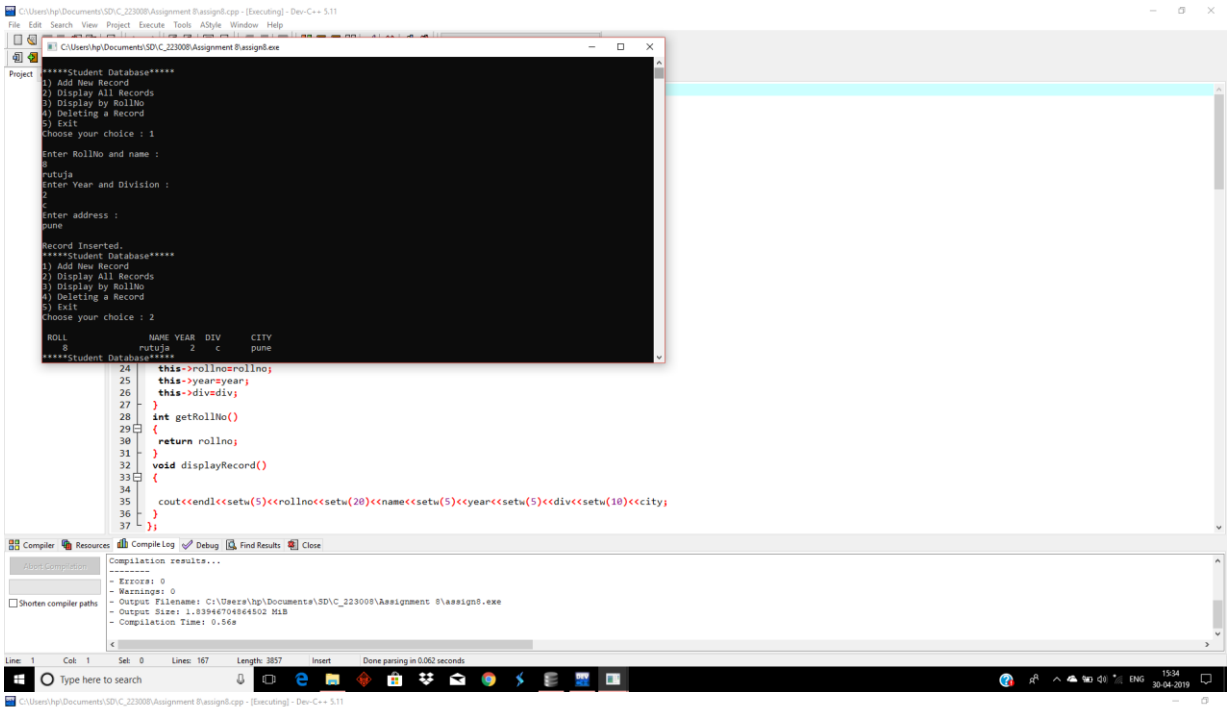
```
 if(flag==false)
 {
  cout<<"\nRecord of "<<rollNo<<"is not present.";
 }
 file.clear();
}
void deleteRecord(int rollno)
{
 ofstream outFile("new.dat",ios::binary);
 file.seekg(0,ios::beg);
 bool flag=false;
 Student s1;

 while(file.read((char *)&s1, sizeof(Student)))
 {
  if(s1.getRollNo()==rollno)
  {
   flag=true;
   continue;
  }
  outFile.write((char *)&s1, sizeof(Student));
 }
 if(!flag)
 {
  cout<<"\nRecord of "<<rollno<<" is not present.";
 }
```

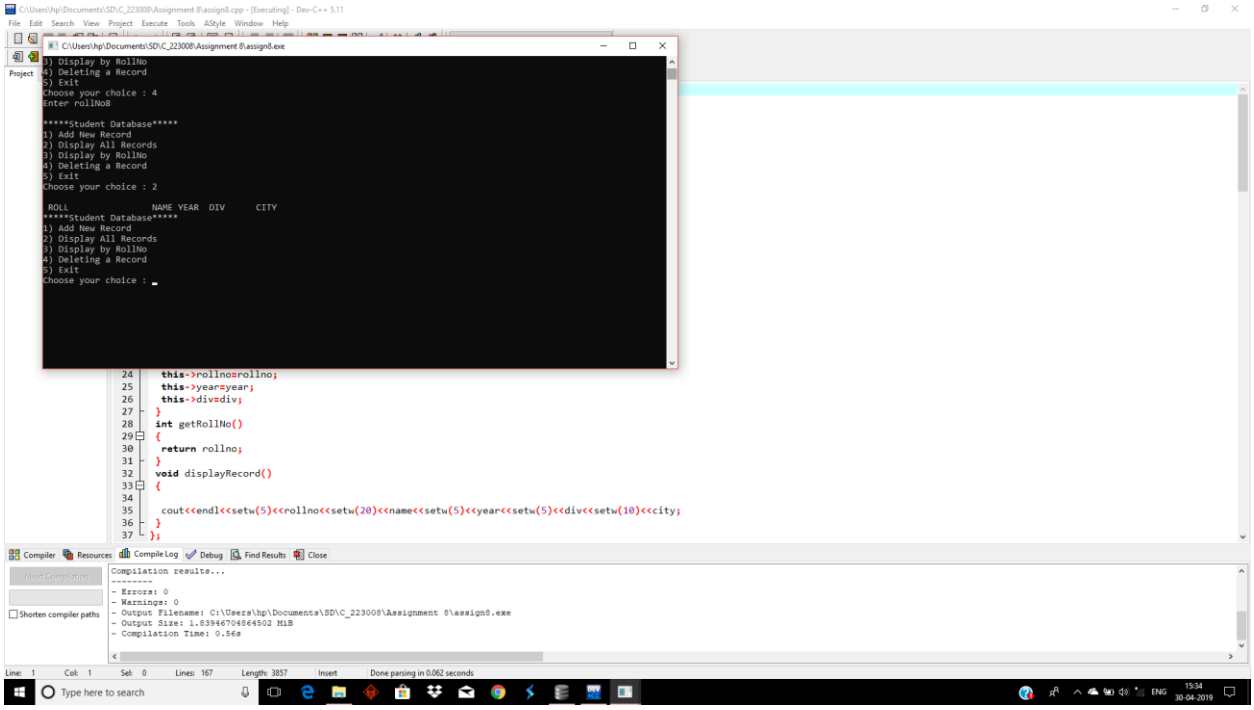
```
  file.close();
  outFile.close();
  remove("student.dat");
  rename("new.dat","student.dat");
  file.open("student.dat",ios::in|ios::out|ios::ate|ios::binary);
 }
 ~FileOperations()
 {
  file.close();
  cout<<"\nFile Closed.";
 }
};
int main() {
 ofstream newFile("student.dat",ios::app|ios::binary);
  newFile.close();
  FileOperations file((char*)"student.dat");
     int rollNo,year,choice=0;
     char div;
     char name[MAX],address[MAX];
     while(choice!=5)
     {

         cout<<"\n*****Student Database*****\n";
         cout<<"1) Add New Record\n";
         cout<<"2) Display All Records\n";
         cout<<"3) Display by RollNo\n";
```

```
cout<<"4) Deleting a Record\n";
cout<<"5) Exit\n";
cout<<"Choose your choice : ";
cin>>choice;
switch(choice)
{
   case 1 : //New Record
    cout<<endl<<"Enter RollNo and name : \n";
    cin>>rollNo>>name;
    cout<<"Enter Year and Division : \n";
    cin>>year>>div;
    cout<<"Enter address : \n";
    cin>>address;
    file.insertRecord(rollNo,name,year,div,address);
    cout<<"\nRecord Inserted.";
    break;
  case 2 :

cout<<endl<<setw(5)<<"ROLL"<<setw(20)<<"NAME"<<setw(5)<<"YEAR"<<set
w(5)<<"DIV"<<setw(10)<<"CITY";
     file.displayAll();
     break;
  case 3 :
    cout<<"Enter Roll Number";
    cin>>rollNo;
     file.displayRecord(rollNo);
```

```
            break;
          case 4:
            cout<<"Enter rollNo";
            cin>>rollNo;
            file.deleteRecord(rollNo);
            break;
         case 5 :break;
       }

   }

 return 0;
}
```

6. **Output screen shots/Copy Past From Terminal –**





7. **Conclusion** –

Every **file** record contains a data field (attribute) to uniquely identify that record.
In **sequential file organization**, records are placed in the **file** in some **sequential** order based on the unique key field or search key.