ASSIGNMENT 9

AIM:-

Company maintain employee information as employee ID ,name,designation and salary.Allow user to add,delete information of employee.Display information of particular employee.If employee doesn't exists an appropriate message is displayed.Use index sequential file to maintain the data.

OBJECTIVE:-

To implement file handling and perform functions like insertion,deletion and display of data using index sequential file.

THEORY:-

Records in indexed sequential files are stored in the order that they are written to the disk. Records may be retrieved in sequential order or in random order using a numeric index to represent the record number in the file.The record size, specified when the file is created, may range from 1 to 8000 bytes.When an Internet Basic program opens an indexed sequential file, the Comet operating system assigns a unique record pointer to the file. Each user opening the file is assigned a unique pointer, allowing multiple users to access data from the same file at the same time. To avoid data integrity problems when more than one user is accessing a file, Comet provides a record locking mechanism. The EXTRACT statement is used to read and lock individual data records.When an indexed sequential file is opened, the record pointer is positioned at the first record. Subsequent I/O operations change the location of the pointer. Note: Some I/O operations do not move the pointer.

EXAMPLE:

For example, to read all the records from an indexed sequential file in order, you would open the file and read the records without specifying an index. This would move through the file in sequential order and end when the last record was read.To read a specific record from an indexed sequential file, you would include the KEY= parameter in the READ (or associated input) statement. The "key" in this case would be a specific record number (e.g., the number 35 would represent the 35th record in the file). The direct access to a record moves the record pointer, so that subsequent sequential access would take place from the new record pointer location, rather than the beginning of the file.

Application :

Indexed sequential files are commonly used for transaction files because they take less disk space than keyed files, and are faster to read from beginning to end than a keyed file.

ALGORITHM:

1.INSERT NODE IN FILE

```
void Create()
{
 char ch='y';
 ofstream seqfile;
 ofstream indexfile;
 int i=0;
 indexfile.open("IND.DAT",ios::out|ios::binary);
 seqfile.open("EMP.DAT",ios::out|ios::binary);
 do
 {
  cout<<"\n Enter Name: ";
  cin>>Records.name;
  cout<<"\n Enter Emp_ID: ";
  cin>>Records.emp_id;
  cout<<"\n Designation";
  cin>>Records.des;
  cout<<"\n Enter Salary: ";
  cin>>Records.salary;
  cout<<Records.name<<" "<<Records.emp_id<<" "<<Records.salary;
  seqfile.write((char*)&Records,sizeof(Records));
  Ind_Records.emp_id=Records.emp_id;
  Ind_Records.position=i;
  indexfile.write((char*)&Ind_Records,sizeof(Ind_Records));
  i++;
  cout<<"\nDo you want to add more records?";
  cin>>ch;
```

```
  }while(ch=='y');
  seqfile.close();
  indexfile.close();
}
```

2.DISPLAY FILE

```
void Employee::Display()
{
 ifstream seqfile;
 ifstream indexfile;

 seqfile.open("EMP.DAT",ios::in|ios::binary);
 indexfile.open("IND.DAT",ios::in|ios::binary);
 cout<<"\n The Contents of file are ..."<<endl;
 int i=0;
 while(indexfile.read((char
*)&Ind_Records,sizeof(Ind_Records)))
 {
   i=Ind_Records.position*sizeof(Rec);
   seqfile.seekg(i,ios::beg);
   seqfile.read((char *)&Records,sizeof(Records));
   if(Records.emp_id!=-1)
   {
   cout<<"\nName: "<<Records.name<<flush;
   cout<<"\nEmp_ID: "<<Records.emp_id;
   cout<<"\nDesignation :"<<Records.des;
   cout<<"\nSalary: "<<Records.salary;
   cout<<"\n";
    }

 }
 seqfile.close();
 indexfile.close();
```

```
}
```

3.SEARCH A RECORD FROM FILE

```
Void Search()
{
 fstream seqfile;
 fstream indexfile;
 int id,pos,offset;
 cout<<"\n Enter the Emp_ID for searching the record ";
 cin>>id;
 indexfile.open("IND.DAT",ios::in|ios::binary);
 pos=-1;
 while(indexfile.read((char
*)&Ind_Records,sizeof(Ind_Records)))
 {
  if(id==Ind_Records.emp_id)
  {
   pos=Ind_Records.position;
   break;
  }
 }
  if(pos==-1)
  {
  cout<<"\n Record is not present in the file";
  return;
  }
  offset=pos*sizeof(Records);
  seqfile.open("EMP.DAT",ios::in|ios::binary);
  seqfile.seekg(offset,ios::beg);
  seqfile.read((char *)&Records,sizeof(Records));
  if(Records.emp_id==-1)
```

```
  {
  cout<<"\n Record is not present in the file";
  return;
  }
  else
  {
  cout<<"\n The Record is present in the file and it is...";
  cout<<"\n Name: "<<Records.name;
  cout<<"\n Emp_ID: "<<Records.emp_id;
  cout<<"\n Designation: "<<Records.des;
  cout<<"\n Salary: "<<Records.salary;
  }
  seqfile.close();
  indexfile.close();
}
```

4.DELETION OF RECORD

```
void Employee::deletion()
{
    int id,pos;
    cout<<"For deletion"<<endl;
    cout<<"\n Enter the employee id for searching"<<endl;
    cin>>id;
    fstream seqfile;
    fstream indexfile;
  seqfile.open("EMP.DAT",ios::in|ios::binary|ios::out);
 indexfile.open("IND.DAT",ios::in|ios::binary|ios::out);
 seqfile.seekg(0,ios::beg);
 indexfile.seekg(0,ios::beg);
 pos=-1;
 while(indexfile.read((char
*)&Ind_Records,sizeof(Ind_Records)))
 {
```

```
  if(id==Ind_Records.emp_id)
  {
   pos=Ind_Records.position;
   Ind_Records.emp_id=-1;
   break;
  }
 }
  if(pos==-1)
  {
  cout<<"\n Record is not present in the file";
  return;
  }
  int offset=pos*sizeof(Rec);
  seqfile.seekp(offset);
  strcpy(Records.name,"");
  Records.emp_id=-1;
  Records.salary=-1;
  strcpy(Records.des,"");
  seqfile.write((char *)&Records,sizeof(Records))<<flush;
  offset=pos*sizeof(Ind_Rec);
  indexfile.seekp(offset);
  Ind_Records.emp_id=-1;
  Ind_Records.position=pos;
  indexfile.write((char *)&Ind_Records,sizeof(Ind_Records));
  seqfile.seekg(0);
  indexfile.close();
  seqfile.close();
}
```

CODE:-

```
#include<iostream>
#include<fstream>
```

```
#include<string.h>
using namespace std;

typedef struct EMP_REC
 {
 char name[10];
 int emp_id;
 int salary;
 char des[10];
 }Rec;

typedef struct INDEX_REC
  {
  int emp_id;
  int position;
  }Ind_Rec;

class Employee
{

 Rec Records;
 Ind_Rec Ind_Records;
 public:
  void Create();
  void Display();
  void Search();
  void deletion();
};

void Employee::Create()
{
 char ch='y';
```

```
 ofstream seqfile;
 ofstream indexfile;
 int i=0;
 indexfile.open("IND.DAT",ios::out|ios::binary);
 seqfile.open("EMP.DAT",ios::out|ios::binary);
 do
 {
  cout<<"\n Enter Name: ";
  cin>>Records.name;
  cout<<"\n Enter Emp_ID: ";
  cin>>Records.emp_id;
  cout<<"\n Designation";
  cin>>Records.des;
  cout<<"\n Enter Salary: ";
  cin>>Records.salary;
  cout<<Records.name<<" "<<Records.emp_id<<" "<<Records.salary;

  seqfile.write((char*)&Records,sizeof(Records));

  Ind_Records.emp_id=Records.emp_id;
  Ind_Records.position=i;
  indexfile.write((char*)&Ind_Records,sizeof(Ind_Records));
  i++;
  cout<<"\nDo you want to add more records?";
  cin>>ch;
  }while(ch=='y');
  seqfile.close();
  indexfile.close();
}

void Employee::Display()
{
```

```
 ifstream seqfile;
 ifstream indexfile;

 seqfile.open("EMP.DAT",ios::in|ios::binary);
 indexfile.open("IND.DAT",ios::in|ios::binary);
 cout<<"\n The Contents of file are ..."<<endl;
 int i=0;
 while(indexfile.read((char
*)&Ind_Records,sizeof(Ind_Records)))
 {
   i=Ind_Records.position*sizeof(Rec);
   seqfile.seekg(i,ios::beg);
   seqfile.read((char *)&Records,sizeof(Records));
   if(Records.emp_id!=-1)
   {
   cout<<"\nName: "<<Records.name<<flush;
   cout<<"\nEmp_ID: "<<Records.emp_id;
   cout<<"\nDesignation :"<<Records.des;
   cout<<"\nSalary: "<<Records.salary;
   cout<<"\n";
    }

 }
 seqfile.close();
 indexfile.close();
}
void Employee::Search()
{
 fstream seqfile;
 fstream indexfile;
 int id,pos,offset;
 cout<<"\n Enter the Emp_ID for searching the record ";
```

```
 cin>>id;
 indexfile.open("IND.DAT",ios::in|ios::binary);
 pos=-1;
 while(indexfile.read((char
*)&Ind_Records,sizeof(Ind_Records)))
 {
  if(id==Ind_Records.emp_id)
  {
   pos=Ind_Records.position;
   break;
  }
 }
  if(pos==-1)
  {
  cout<<"\n Record is not present in the file";
  return;
  }
  offset=pos*sizeof(Records);
  seqfile.open("EMP.DAT",ios::in|ios::binary);
  seqfile.seekg(offset,ios::beg);
  seqfile.read((char *)&Records,sizeof(Records));
  if(Records.emp_id==-1)
  {
  cout<<"\n Record is not present in the file";
  return;
  }
  else
  {
  cout<<"\n The Record is present in the file and it is...";
  cout<<"\n Name: "<<Records.name;
  cout<<"\n Emp_ID: "<<Records.emp_id;
  cout<<"\n Designation: "<<Records.des;
```

```
  cout<<"\n Salary: "<<Records.salary;
  }
  seqfile.close();
  indexfile.close();
}
void Employee::deletion()
{
    int id,pos;
    cout<<"For deletion"<<endl;
    cout<<"\n Enter the employee id for searching"<<endl;
    cin>>id;
    fstream seqfile;
    fstream indexfile;
  seqfile.open("EMP.DAT",ios::in|ios::binary|ios::out);
 indexfile.open("IND.DAT",ios::in|ios::binary|ios::out);
 seqfile.seekg(0,ios::beg);
 indexfile.seekg(0,ios::beg);
 pos=-1;
 while(indexfile.read((char
*)&Ind_Records,sizeof(Ind_Records)))
 {
  if(id==Ind_Records.emp_id)
  {
   pos=Ind_Records.position;
   Ind_Records.emp_id=-1;
   break;
  }
 }
  if(pos==-1)
  {
  cout<<"\n Record is not present in the file";
  return;
```

```
  }
  int offset=pos*sizeof(Rec);
  seqfile.seekp(offset);
  strcpy(Records.name,"");
  Records.emp_id=-1;
  Records.salary=-1;
  strcpy(Records.des,"");
  seqfile.write((char *)&Records,sizeof(Records))<<flush;
  offset=pos*sizeof(Ind_Rec);
  indexfile.seekp(offset);
  Ind_Records.emp_id=-1;
  Ind_Records.position=pos;
  indexfile.write((char *)&Ind_Records,sizeof(Ind_Records));
  seqfile.seekg(0);
  indexfile.close();
  seqfile.close();
}
int main()
{
 Employee e;
 char ans='y';
 int choice,key;
 do
 {
    cout<<"1.Create"<<endl;
    cout<<"2.Display"<<endl;
    cout<<"3.Search"<<endl;
    cout<<"4.Delete"<<endl;
    cout<<"Enter your choice"<<endl;
    cin>>choice;
        switch(choice)
        {
```

```
        case 1:
           e.Create();
           break;
        case 2:
          e.Display();
          break;
        case 3:
            e.Search();
            break;
        case 4:
           e.deletion();
           break;
        }
        cout<<"Do you want to continue"<<endl;
        cin>>ans;
 }while (ans=='y');
return 0;
}
```

OUTPUT:-

```
jugal@ubuntu:~/17u183/sem2/SD$ g++ EmployeeIndexSequential.cpp
jugal@ubuntu:~/17u183/sem2/SD$ ./a.out
1.Create
2.Display
3.Search
4.Delete
Enter your choice
1

 Enter Name: jugal

 Enter Emp_ID: 42

 Designationffsdf

 Enter Salary: 2222
jugal 42 2222
Do you want to add more records?y

 Enter Name: chakka

 Enter Emp_ID: 34

 Designationfdfsdf

 Enter Salary: 11
chakka 34 11
Do you want to add more records?n
Do you want to continue
y
1.Create
2.Display
3.Search
4.Delete
Enter your choice
2

 The Contents of file are ...

Name: jugal
Emp_ID: 42
Designation :ffsdf
Salary: 2222

Name: chakka
Emp_ID: 34
Designation :fdfsdf
Salary: 11
Do you want to continue
y
1.Create
2.Display
3.Search
4.Delete
Enter your choice
3

 Enter the Emp_ID for searching the record 34

 The Record is present in the file and it is...
 Name: chakka
 Emp_ID: 34
 Designation: fdfsdf
 Salary: 11Do you want to continue
```

CONCLUSION:-

We have successfully implemented file handling and performed functions like insertion,deletion and display of employee data using index sequential file.