SKILL DEVELOPMENT LAB-II(2018-19)

Assignment 9 :

Aim :

Company maintains employee info. as employee ID , name , designation and salary . Allow user to add, delete innfo. of employee . Display info. of particular employee. If employee does not exist an appropriate message is displayed. If it is, then the system displays the emplyee details. Use index sequential file to maintain the data.

Objective :

We have to implement this using index sequential file organization.

Theory :

File is a collection of records related to each other. The file size is limited by the size of memory and storage medium.

File organization ensures that records are available for processing. It is used to determine an efficient file organization for each base relation.

Indexed sequential access file organization :

- Indexed sequential access file combines both sequential file and direct access file organization.
- In indexed sequential access file, records are stored randomly on a direct access device such as magnetic disk by a primary key.
- This file have multiple keys. These keys can be alphanumeric in which the records are ordered is called primary key.
- The data can be access either sequentially or randomly using the index. The index is stored in a file and read into memory when the file is opened.

Advantages of Indexed sequential access file organization :

- In indexed sequential access file, sequential file and random file access is possible.
- It accesses the records very fast if the index table is properly organized.
- The records can be inserted in the middle of the file.
- It provides quick access for sequential and direct processing.
- It reduces the degree of the sequential search.

Program :

```
#include<iostream>

#include<fstream>

#include<conio.h>

using namespace std;

int pass;

struct student

{

string name, dept,dsg;

long long int id, cell;

};



class llist

{

public: struct student s,s1;llist *address;

};

llist *start=NULL, *start1=NULL;

//Accept a element

void accept()

{

llist *nnode,*temp;

nnode=new(llist);

cout<<"\nEnter name:";

cin>>nnode->s.name;

cout<<"\nEnter division:";
```

```
cin>>nnode->s.dept;
cout<<"\nEnter roll:";
cin>>nnode->s.id;
cout<<"\nEnter cell number:";
cin>>nnode->s.cell;
cout<<"\nEnter address:";
cin>>nnode->s.dsg;
temp=start;
if(temp==NULL){start=nnode;nnode->address=NULL;}
else{
while(temp->address!=NULL)
{
temp=temp->address;
}
temp->address=nnode;
nnode->address=NULL;
}
}

//Display the elements
void display()
{
   llist *temp;
   temp=start;
   while(temp!=NULL)
```

```
    {
        cout<<"\nName is "<<temp->s.name;
        cout<<"\ndept is "<<temp->s.dept;
        cout<<"\nid is "<<temp->s.id;
        cout<<"\nCell number is "<<temp->s.cell;
        cout<<"\ndesignation is "<<temp->s.dsg;
        temp=temp->address;
    }
}
//Insert begin
void insert_begin()
{
llist *nnode,*temp;
nnode=new(llist);
cout<<"\nEnter name:";
cin>>nnode->s.name;
cout<<"\nEnter dept:";
cin>>nnode->s.dept;
cout<<"\nEnter id:";
cin>>nnode->s.id;
cout<<"\nEnter cell number:";
cin>>nnode->s.cell;
cout<<"\nEnter designation number:";
cin>>nnode->s.dsg;
temp=start;
```

```
nnode->address=temp;
start=nnode;
}
//Insert middle
void insert_mid()
{
llist *nnode,*temp,*prev;
nnode=new(llist);
int i,p;
cout<<"\nEnter name:";
cin>>nnode->s.name;
cout<<"\nEnter dept:";
cin>>nnode->s.dept;
cout<<"\nEnter id:";
cin>>nnode->s.id;
cout<<"\nEnter cell number:";
cin>>nnode->s.cell;
cout<<"\nEnter designation:";
cin>>nnode->s.dsg;
cout<<"\nEnter position:";
cin>>p;
temp=start;
for(i=0;i<p-1;i++)
{
prev=temp;
```

```
temp=temp->address;
}
prev->address=nnode;
nnode->address=temp;
}

//Delete begin
void del_begin()
{
llist *temp;
temp=start;
start=temp->address;
temp->address=NULL;
delete(temp);
}
//Delete at position
void del_mid()
{
llist *temp,*prev;
int i,p;
cout<<"\nEnter position:";
cin>>p;
temp=start;
for(i=0;i<p-1;i++)
{
```

```
prev=temp;

temp=temp->address;

}

prev->address=temp->address;

delete(temp);

}


//Search node

void search()

{

int i,j;

llist *temp;

cout<<"\nEnter id to be found:";

cin>>j;

temp=start;

i=1;

//cout<<j<<" "<<temp<<" "<<i;

while(temp->address!=NULL)

{

if(temp->s.id==j){break;}

else{i++;temp=temp->address;}

}

cout<<"Position is:"<<i;

}
```

```
void save()
{
   int m=0;
   llist *temp;
   temp=start;
   ofstream file1,file2,file3,file4,file5;
   file1.open("names.txt",ios_base::app);
   file2.open("dept.txt",ios_base::app);
   file3.open("id.txt",ios_base::app);
   file4.open("dsg.txt",ios_base::app);
   file5.open("cell.txt",ios_base::app);
   while(m<5)
   {
     switch(m)
   {
   case 0:while(temp!=NULL){file1<<temp->s.name<<endl;temp=temp->address;};break;
   case 1:while(temp!=NULL){file2<<temp->s.dept<<endl;temp=temp->address;};break;
   case 2:while(temp!=NULL){file3<<temp->s.id<<endl;temp=temp->address;};break;
   case 3:while(temp!=NULL){file4<<temp->s.dsg<<endl;temp=temp->address;};break;
   case 4:while(temp!=NULL){file5<<temp->s.cell<<endl;temp=temp->address;};break;
   }
   m++;
   temp=start;
   }
```

```
   file1.close();
   file2.close();
   file3.close();
   file4.close();
   file5.close();
}



void displayfile()
{
   llist *temp;
temp=start1;

while(temp->address!=NULL)
{
cout<<"--------------------------------------------------------------------------\n";
cout<<"\nName is:"<<temp->s1.name;
cout<<"\ndept is:"<<temp->s1.dept;
cout<<"\nid is:"<<temp->s1.id;
cout<<"\nCell number is:"<<temp->s1.cell;
cout<<"\ndesignation is:"<<temp->s1.dsg;
cout<<"\n--------------------------------------------------------------------------\n";
temp=temp->address;
}
}
```

```
void open()

{

   llist *temp,*nnode,*next;

   if(start1!=NULL){temp=start1->address;while(temp!=NULL){next=temp-
>address;delete(temp);temp=next;}start1=NULL;}

   ifstream file1,file2,file3,file4,file5;

   file1.open("names.txt");

   file2.open("dept.txt");

   file3.open("id.txt");

   file4.open("dsg.txt");

   file5.open("cell.txt");

   while(!file3.eof())

   {

      temp=start1;

      nnode=new(llist);

      if(temp==NULL){start1=nnode;nnode->address=NULL;}

      else{while(temp->address!=NULL){temp=temp->address;}temp->address=nnode;nnode-
>address=NULL;}

      getline(file1,nnode->s1.name);

      getline(file2,nnode->s1.dept);

      getline(file4,nnode->s1.dsg);

      file3>>nnode->s1.id;

      file5>>nnode->s1.cell;


   }
```

```
    file1.close();
    file2.close();
    file3.close();
    file4.close();
    file5.close();
    displayfile();
}

void del()
{
    ofstream file1,file2,file3,file4,file5;
    file1.open("names.txt",ios_base::trunc);
    file2.open("dept.txt",ios_base::trunc);
    file3.open("id.txt",ios_base::trunc);
    file4.open("dsg.txt",ios_base::trunc);
    file5.open("cell.txt",ios_base::trunc);
    file1.close();
    file2.close();
    file3.close();
    file4.close();
    file5.close();
}

void search1()
{
```

```
   int a,i=1,m,l=0;
   cout<<"\nEnter the id to be searched:";
   cin>>a;
   ifstream file;
   file.open("id.txt");
   while(!file.eof()){file>>m;if(m==a){cout<<"\nRecord found at:"<<i;l=1;break;} else{i++;}}
   if(l==0){cout<<"\nRecord not found!";}
   file.close();


}


//Main Function
int main()
{
   int j=0,k,flag=0,i;
while(flag==0)
{
   cout<<"\nWelcome"<<"\nWhat do you want to do?"<<"\n1.Enter new element to link\n"<<"2.Display the elements of array"<<"\n3.Insert at beginnng\n"
<<"4.Insert in middle\n"<<"5.Delete at begin\n"<<"6.Delete at position"<<"\n7.Search\n"<<"8.Save Data"<<"\n9.Open File & Display file\n"<<"10.Display Opened File"
<<"\n11.Clear the files"<<"\n12.Search For IDs"<<"\n13.Exit"<<"\nEnter your option:";
cin>>i;
switch(i)
{
case 1:accept();break;
```

```
case 2:display();break;

case 3:insert_begin();break;

case 4:insert_mid();break;

case 5:del_begin();break;

case 6:del_mid();break;

case 7:search();break;

case 8:save();break;

case 9:open();break;

case 10:displayfile();break;

case 11:del();break;

case 12:search1();break;

case 13:flag=1;

}

}

return 0;

}
```

Output :

```
Enter name:sanket

Enter division:c

Enter roll:223033

Enter cell number:123456789

Enter address:pune

Welcome
What do you want to do?
1.Enter new element to link
2.Display the elements of array
3.Insert at beginnng
4.Insert in middle
5.Delete at begin
6.Delete at position
7.Search
8.Save Data
9.Open File & Display file
10.Display Opened File
11.Clear the files
12.Search For IDs
13.Exit
Enter your option:8

Welcome
What do you want to do?
1.Enter new element to link
2.Display the elements of array
3.Insert at beginnng
4.Insert in middle
5.Delete at begin
6.Delete at position
7.Search
8.Save Data
9.Open File & Display file
10.Display Opened File
11.Clear the files
12.Search For IDs
13.Exit
Enter your option:
```

```
Welcome
What do you want to do?
1.Enter new element to link
2.Display the elements of array
3.Insert at beginnng
4.Insert in middle
5.Delete at begin
6.Delete at position
7.Search
8.Save Data
9.Open File & Display file
10.Display Opened File
11.Clear the files
12.Search For IDs
13.Exit
Enter your option:9
--------------------------------------------------------------------------------

Name is:sanket
dept is:c
id is:223033
Cell number is:123456789
designation is:pune
--------------------------------------------------------------------------------

Welcome
What do you want to do?
1.Enter new element to link
2.Display the elements of array
3.Insert at beginnng
4.Insert in middle
5.Delete at begin
6.Delete at position
7.Search
8.Save Data
9.Open File & Display file
10.Display Opened File
11.Clear the files
12.Search For IDs
13.Exit
Enter your option:
```

Conclusion :

Thus we implemented this example using index sequential file organization.