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

```
#include <iostream>
#include <string.h>
using namespace std;
class Emp
{
   int eid;
   char name[20];
   float basic;

public:
   Emp() //D
   {
      eid = 00;
      strcpy(name, "Not Given");
      basic = 00;
   }

   Emp(int d, const char *nm, float bs)//PARA
   {
      eid = d;
      strcpy(name, nm);
      basic = bs;
   }
   void display()
   {
      cout << "\n***********************************";
      cout << "\nEmployee Id : " << eid;

      cout << "\nName : " << name;

      cout << "\nSalary : " << basic;

      cout << "\n***********************************";

   }



   int getid()

   {

      return eid;

   }

   const char *getname()
```

```
    {
        return name;
    }
    float getbasic()
    {
        return basic;
    }
    void setid(int d)
    {
        this->eid = d;
    }
    void setename(const char *nm)
    {
        strcpy(this->name, nm);
    }
    void setbasic(float bs)
    {
        this->basic = bs;
    }
}; // emp class end

class Node
{
    Emp data;
    Node *next;

public:
    Node(Emp d)
    {
        data = d;
```

```
    next = NULL;
  }
  Emp getdata()
  {
    return data;
  }
  Node *getnext()
  {
    return next;
  }

  void setdata(Emp d)
  {
    this->data = d;
  }
  void setnext(Node *n)
  {
    this->next = n;
  }
}; // node class end

class Company
{
  Node *start;

public:
  Company()
  {
    start = NULL;
  }
```

```
void addemp(Emp e)
{
   Node *temp = new Node(e);
   temp->setnext(start);
   start = temp;
}

void removeempById(int id)
{
   if (start == NULL)
   {
      cout << "\nNo employee data is here ";
      return;
   }
   Node *p = start; // when node to be deleted first node
   if (id == p->getdata().getid())
   {
      start = start->getnext();
      p->getdata().display();
      cout << "\nthis employee is deleted ...";
      delete p;
      return;
   }
   if (p->getnext() == NULL)
   {
      cout << "\nnot Found ";
      return;
   }
   while (p->getnext()!= NULL)
   {
```

```
        Emp e1 = p->getnext()->getdata();
        if (id == e1.getid())
        {
            Node *q = p->getnext();
            p->setnext(q->getnext());
            q->getdata().display();
            cout << "\nNow Deleted this employee";
            delete q;
            return;
        }
        p = p->getnext();
    }
    cout << "\nnot Found ";
}

void editiEmp(int id)
{
    if (start == NULL)
    {
        cout << "\nNo Employee is here";
        return;
    }
    Node *p = start;
    while (p != NULL)
    {
        Emp e = p->getdata();
        if (p->getdata().getid() == id)
        {
            Emp e = p->getdata();
            char ans;
```

```
            char name[20];
            float sal;
            cout << "\nDo you wan change name : (Y/N)";
            cin >> ans;
            if (ans == 'Y' || ans == 'y')
            {
               cout << "\n Enter New Name : ";
               cin >> name;
               e.setename(name);
            }
            cout << "\nDo you wan change Salary : (Y/N)";
            cin >> ans;
            if (ans == 'Y' || ans == 'y')
            {
               cout << "\n Enter new salary : ";
               cin >> sal;
               e.setbasic(sal);
            }
            p->setdata(e);
            return;
         }
         p = p->getnext();
      }

      cout << "\nRecord not Found ";
}

void searchEmpById(int id)
{
   if (start == NULL)
```

```
    {
        cout << "\nNot Found";
        return;
    }
    Node *p = start;
    while (p != NULL)
    {
        Emp e1 = p->getdata();
        if (e1.getid() == id)
        {
            cout << "\nEmployee Found ...";
            e1.display();
            return;
        }
        p = p->getnext();
    }
    cout << "\nEmployee is not Found";
}

void DisplayallEmp()
{
    if (start == NULL)
    {
        cout << "\n\n No employee data here ";
        cout << "\n\n";
        return;
    }
    Node *p = start;
    while (p != NULL)
    {
```

```
            /* code */ p->getdata().display();
            p = p->getnext();
        }
    }

}; // Company class End

int main()
{
    int ch = 0;
    Company lt;
    while (ch != 6)
    {
        cout << "\n\n*************************************";
        cout << "\n\t1.Add Employee.";
        cout << "\n\t2.Display Employee.";
        cout << "\n\t3.Search By id.";
        cout << "\n\t4.Delete by Id.";
        cout << "\n\t5.Edit Employee data.";
        cout << "\n\t6.Exit";
        cout << "\n*************************************\n\n";
        cout << "\n\tEnter the choice: ";
        cin >> ch;
        switch (ch)
        {
        case 1:
        {
            int id;
            char name[20];
            float bs;
```

```
    cout << "\n Enter the ID : ";
    cin >> id;
    cout << "\n Enter the name :";
    cin >> name;
    cout << "\n Enter the Salary: ";
    cin >> bs;
    Emp e1(id, name, bs);
    lt.addemp(e1);
}
break;
case 2:
    lt.DisplayallEmp();
    /* code */
    break;
case 3:
{
    int id;
    cout << "\nEnter the id to search : ";
    cin >> id;
    lt.searchEmpById(id);
}
/* code */
break;

case 4:
{
    int id;
    cout << "\nEnter the id to Delete  : ";
    cin >> id;
    lt.removeempById(id);
```

```
        }
        /* code */
        break;

        case 5:
        {
           int id;
           cout << "\nEnter the id for edit user  : ";
           cin >> id;
           lt.editiEmp(id);
        }

        /* code */
        break;
        case 6:
        {
           cout << "\nEnd the Programm!!!";
        }
        /* code */
        break;

        default:
           cout << "\nInvalid choice:";
           break;
        }
    }
}
```