**ASSIGNMENT NO.9.**

**Aim :-** 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.

**Program Code:-**

#include <iostream>

#include <fstream>

#include <cstring>

#include <iomanip>

#include<cstdlib>

#define max 50

using namespace std;

class Employee

{

char name[max];

int empid;

int sal;

char de[50];

friend class FileOperations;

public: Employee()

{

strcpy(name,"");

empid=sal==0;

strcpy(de,"");

}

Employee(char name[max],int empid,int sal,char de[max])

{

strcpy(this->de,de);

strcpy(this->name,name);

this->empid=empid;

this->sal=sal;

}

int getEmpId()

{

return empid;

}

void displayEmployeeData()

{

cout<<endl<<empid<<"\t\t\t"<<name<<"\t\t\t"<<sal<<"\t\t\t"<<de;

}

};

class FileOperations

{

fstream file;

public:FileOperations(char \*name)

{

//strcpy(this->name,name);

this->file.open(name,ios::in|ios::out|ios::ate|ios::binary);

}

void insertRecord(int empid,char name[max],int sal,char de[max])

{

Employee s=Employee(name,empid,sal,de);

file.seekp(0,ios::end);

file.write((char\*)&s,sizeof(Employee));

file.clear();

}

void displayAllRecords()

{

Employee s;

file.seekg(0,ios::beg);

while(file.read((char \*)&s,sizeof(Employee)))

{

s.displayEmployeeData();

}

file.clear();

}

void displayRecord(int empid)

{

Employee s;

file.seekg(0,ios::beg);

void \*p;

while(file.read((char \*)&s,sizeof(Employee)))

{

if(s.empid==empid)

{

s.displayEmployeeData();

break;

}

}

if(p==NULL)

throw "Element not present";

file.clear();

}

void deleteRecord(int empid)

{

ofstream newFile("new.txt",ios::binary);

file.seekg(0,ios::beg);

bool flag=false;

Employee s;

while(file.read((char \*)&s,sizeof(s)))

{

if(s.empid==empid)

{

flag=true;

continue;

}

newFile.write((char \*)&s,sizeof(s));

}

if(!flag)

{

cout<<"Element Not Present";

}

file.close();

newFile.close();

remove("Employee.txt");

rename("new.txt","Employee.txt");

file.open("Employee.txt",ios::in|ios::ate|ios::out|ios::binary);

}

~FileOperations()

{

file.close();

cout<<"Closing file..";

}

};

int main()

{

ofstream newFile("Employee.txt",ios::app|ios::binary);

newFile.close();

FileOperations file((char \*)"Employee.txt");

int empid,sal,choice=0;

char name[max],de[max];

while(choice!=5)

{

cout<<"\n\n1) 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 employee id and name : \n";

cin>>empid>>name;

cout<<"Enter sal \n";

cin>>sal;

cout<<"Enter designation : \n";

cin>>de;

file.insertRecord(empid,name,sal,de);

break;

case 2 :

cout<<"Employee ID"<<"\t\t"<<"Name"<<"\t\t"<<"Salary"<<"\t\t"<<"designation\n";

cout<<"----------------------------------------------------------------------";

file.displayAllRecords();

break;

case 3 :

cout<<"Enter employee id";

cin>>empid;

try

{

file.displayRecord(empid);

}

catch(const char \*str)

{

cout<<str;

}

break;

case 4:

cout<<"Enter employe id";

cin>>empid;

file.deleteRecord(empid);

break;

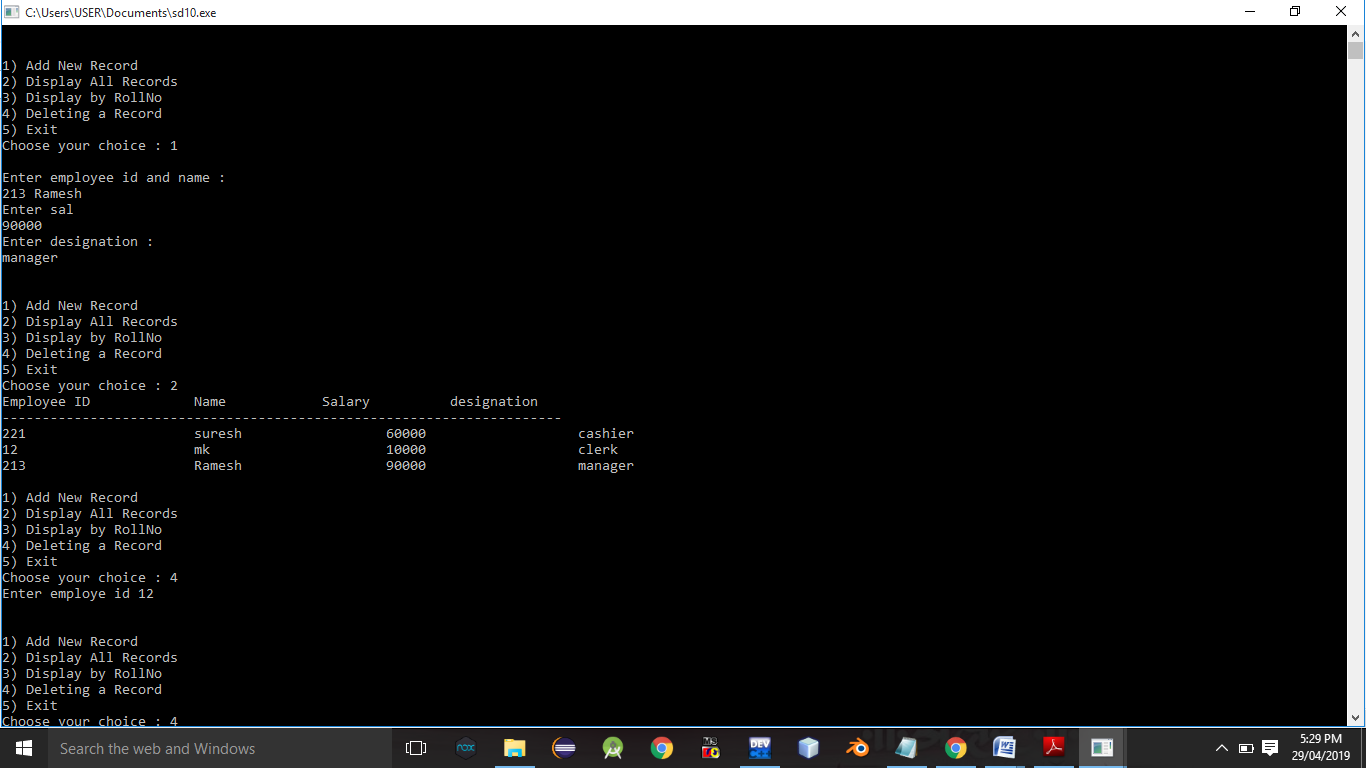
case 5 :break;

}

}

}

**Output Screenshots:-**

****

**Conclusion:-** Thus, this assignment is completed successfully.