\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Data Structure Lab

CEN-391

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Program 4

Code :-

#include <iostream>

#include <string.h>

using namespace std;

int Max\_Size = 0;

struct Employee

{

    int Eid;

    char Name[30];

    float Salary;

};

void Add\_Employee(Employee \*Emp\_Data, int &size)

{

    cout << endl

         << "Add Employee..." << endl;

    if (size == Max\_Size)

    {

        cout << "Overflow" << endl;

        return;

    }

    int Eid;

    bool check = false;

    do

    {

        cout << "Enter The Employee Eid : ";

        cin >> Eid;

        for (int i = 0; i < size; i++)

        {

            if (Eid == (Emp\_Data + i)->Eid)

            {

                cout << endl

                     << "Eid Already Exist!" << endl;

                cout << "Try Again!" << endl

                     << endl;

                check = true;

            }

        }

    } while (check);

    (Emp\_Data + size)->Eid = Eid;

    fflush(stdin);

    cout << "Enter The Employee Name : ";

    gets((Emp\_Data + size)->Name);

    cout << "Enter The Employee Salary : ";

    cin >> (Emp\_Data + size)->Salary;

    size++;

}

void Display\_Employee(Employee \*Emp\_Data, int &size)

{

    if (size == 0)

    {

        cout << endl

             << "Empty!" << endl;

        return;

    }

    cout << endl

         << "Display All Employee..." << endl;

    cout << "|\tEid \t|"

         << "\t    Name     \t|"

         << "\t Salary \t|" << endl;

    for (int i = 0; i < size; i++)

    {

        cout << "\t" << (Emp\_Data + i)->Eid << "\t";

        cout << "\t" << (Emp\_Data + i)->Name << "\t";

        cout << "\t" << (Emp\_Data + i)->Salary << "\t" << endl;

    }

}

void Search\_Employee\_Eid(Employee \*Emp\_Data, int &size)

{

    cout << endl

         << "Search Employee By Eid..." << endl;

    if (size == 0)

    {

        cout << "Empty!" << endl;

        return;

    }

    int Eid;

    cout << "Enter The Employee Eid : ";

    cin >> Eid;

    int i;

    cout << endl;

    for (i = 0; i < size; i++)

    {

        if ((Emp\_Data + i)->Eid == Eid)

        {

            cout << "Employee Found!\n\nDetails..." << endl;

            cout << "Eid : " << (Emp\_Data + i)->Eid << "\t  ";

            cout << "Name : " << (Emp\_Data + i)->Name << "\t  ";

            cout << "Salary : " << (Emp\_Data + i)->Salary << endl;

            break;

        }

    }

    if (i == size)

    {

        cout << "Employee Not Found!" << endl;

    }

}

void Search\_Employee\_Name(Employee \*Emp\_Data, int &size)

{

    cout << endl

         << "Search Employee By Name..." << endl;

    if (size == 0)

    {

        cout << "Empty!" << endl;

        return;

    }

    char Name[30];

    cout << "Enter The Name Of Your Employee : ";

    fflush(stdin);

    gets(Name);

    int i;

    cout << endl;

    for (i = 0; i < size; i++)

    {

        int j;

        if (!strcmp(Name, (Emp\_Data + i)->Name))

        {

            cout << "Employee Found!\n\nDetails..." << endl;

            cout << "Eid : " << (Emp\_Data + i)->Eid << "\t  ";

            cout << "Name : " << (Emp\_Data + i)->Name << "\t  ";

            cout << "Salary : " << (Emp\_Data + i)->Salary << endl;

            break;

        }

    }

    if (i == size)

    {

        cout << "Employee Not Found!" << endl;

    }

}

void Highest\_Salary(Employee \*Emp\_Data, int &size)

{

    cout << endl

         << "Highest Salary Of Employee" << endl;

    if (size == 0)

    {

        cout << "Empty!" << endl;

        return;

    }

    float Max\_Salary = 0;

    for (int i = 0; i < size; i++)

    {

        if (Max\_Salary < (Emp\_Data + i)->Salary)

        {

            Max\_Salary = (Emp\_Data + i)->Salary;

        }

    }

    for (int i = 0; i < size; i++)

    {

        if (Max\_Salary == (Emp\_Data + i)->Salary)

        {

            cout << "Eid : " << (Emp\_Data + i)->Eid << "\t  ";

            cout << "Name : " << (Emp\_Data + i)->Name << "\t ";

            cout << "Salary : " << (Emp\_Data + i)->Salary << endl;

        }

    }

}

void Total\_Employee(int &size)

{

    cout << endl

         << "No Of Employee..." << endl;

    cout << endl

         << "Total No Of Employee : ";

    cout << size << endl;

}

void Menu()

{

    cout << endl

         << endl

         << "\_\_\_Operations\_\_\_" << endl;

    cout << "1.Add Employee" << endl;

    cout << "2.Display Employee" << endl;

    cout << "3.Search Employee Byy Eid" << endl;

    cout << "4.Search Employee By Name" << endl;

    cout << "5.Employee having Higest Salary" << endl;

    cout << "6.Total No Of Employee" << endl;

    cout << "7.Exit" << endl;

    cout << "Enter Your Choice : ";

}

bool Options(Employee \*Emp\_Data, int &size)

{

    int opt;

    cin >> opt;

    switch (opt)

    {

    case 1:

        Add\_Employee(Emp\_Data, size);

        break;

    case 2:

        Display\_Employee(Emp\_Data, size);

        break;

    case 3:

        Search\_Employee\_Eid(Emp\_Data, size);

        break;

    case 4:

        Search\_Employee\_Name(Emp\_Data, size);

        break;

    case 5:

        Highest\_Salary(Emp\_Data, size);

        break;

    case 6:

        Total\_Employee(size);

        break;

    case 7:

        return 0;

    default:

        cout << "Invalid Input!\nTry Again!" << endl;

    }

    return 1;

}

int main()

{

    system("cls");

    cout << "\_\_Vicky Gupta 20BCS070\_\_" << endl;

    cout << "Enter The No Of Employee : ";

    cin >> Max\_Size;

    struct Employee \*Emp\_Data = (Employee \*)malloc(Max\_Size \* sizeof(Employee));

    int size = 0;

    while (true)

    {

        Menu();

        if (!Options(Emp\_Data, size))

            break;

    }

    cout << endl

         << "Exiting..." << endl;

    return 0;

}

Output :-

Text

Description automatically generated

Text

Description automatically generated

Text

Description automatically generated

Text

Description automatically generated