Institute of Computer Technology

B. Tech. Computer Science and Engineering

Sub: ESFP – II

Course Code: 2CSE203

Practical – 9

Name: Jaymin Gondaliya

Enrollment No: 23162171007

Sem - 2

Branch: CS

Class: B

Batch: 25

Objective:

To implement friend class and friend function concept.

Problem Definition-1: Complete the code for the object assigned to you to satisfy following specifications.

Code:

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

const int MAX_EMPLOYEES = 100;

class EmployeeManager;
class EmployeeDatabase;

class Employee
{
    friend class EmployeeManager;
    friend class EmployeeDatabase;
    friend void displayEmployee(const Employee &emp);
```

```
private:
    string name;
    int id;
    string department;
    double salary;
    bool isPresent;

public:
    Employee() : isPresent(false) {}
    ~Employee()
    {
        cout << "Destroying employee: " << name << endl;
    }

    void markAttendance(bool present)
    {
        isPresent = present;
        if (!isPresent)
        {
            salary -= 50;
        }
    }

    void resetSalary()
    {
        salary = 0;
    }

    void increaseSalary(double amount)
    {
        salary += amount;
    }
};

class EmployeeManager
{
private:
    Employee employees[MAX_EMPLOYEES];
    int numEmployees;

public:
    EmployeeManager() : numEmployees(0) {}
    ~EmployeeManager()
    {
        cout << "Destroying EmployeeManager..." << endl;
    }

    int getNumEmployees() const
```

```
    {
        return numEmployees;
    }

    const Employee &getEmployee(int index) const
    {
        return employees[index];
    }
    void addEmployee(const string &name, int id, const string &department,
double salary)
    {
        if (numEmployees < MAX_EMPLOYEES)
        {
            employees[numEmployees].name = name;
            employees[numEmployees].id = id;
            employees[numEmployees].department = department;
            employees[numEmployees].salary = salary;
            numEmployees++;
        }
        else
        {
            cout << "Maximum number of employees reached." << endl;
        }
    }

    void markAttendance(int id, bool present)
    {
        for (int i = 0; i < numEmployees; ++i)
        {
            if (employees[i].id == id)
            {
                employees[i].markAttendance(present);
                cout << "Attendance marked for employee with ID " << id <<
endl;
                return;
            }
        }
        cout << "Employee not found with ID " << id << endl;
    }

    void resetSalary(int id)
    {
        for (int i = 0; i < numEmployees; ++i)
        {
            if (employees[i].id == id)
            {
                employees[i].resetSalary();
                cout << "Salary reset for employee with ID " << id << endl;
```

```
                return;
            }
        }
        cout << "Employee not found with ID " << id << endl;
    }

    void increaseSalary(int id, double amount)
    {
        for (int i = 0; i < numEmployees; ++i)
        {
            if (employees[i].id == id)
            {
                employees[i].increaseSalary(amount);
                cout << "Salary increased for employee with ID " << id <<
endl;
                return;
            }
        }
        cout << "Employee not found with ID " << id << endl;
    }

    friend void displayEmployee(const Employee &emp);
};

class EmployeeDatabase
{
    friend class EmployeeManager;

public:
    void displayByCategory(const EmployeeManager &empManager)
    {
        int choice;
        cout << "Display employees by category:" << endl;
        cout << "1. CS" << endl;
        cout << "2. BDA" << endl;
        cout << "3. CBA" << endl;
        cout << "Enter choice: ";
        cin >> choice;

        cout << "Employees:" << endl;
        for (int i = 0; i < empManager.getNumEmployees(); ++i)
        {
            const Employee &employee = empManager.getEmployee(i);
            if (choice == 1 && employee.department == "CS")
            {
                displayEmployee(employee);
            }
            else if (choice == 2 && employee.department == "BDA")
```

```
            {
                displayEmployee(employee);
            }
            else if (choice == 3 && employee.department == "CBA")
            {
                displayEmployee(employee);
            }
            else
            {
                cout << "No employees found in this category." << endl;
            }
        }
    }
};

void displayEmployee(const Employee &emp)
{
    cout << "Name: " << emp.name << endl;
    cout << "ID: " << emp.id << endl;
    cout << "Department: " << emp.department << endl;
    cout << "Salary: " << emp.salary << endl;
    cout << "Is Present: " << (emp.isPresent ? "Yes" : "No") << endl;
}

int main()
{
    if (!checkLogin())
    {
        return 0;
    }

    EmployeeManager empManager;
    EmployeeDatabase empDatabase;

    int choice;

    do
    {
        cout << "\nEmployee Management System\n";
        cout << "1. Enter Employee Details\n";
        cout << "2. Display All Employees\n";
        cout << "3. Mark Attendance\n";
        cout << "4. Reset Salary\n";
        cout << "5. Increase Salary\n";
        cout << "6. Display Employees by Category\n";
        cout << "7. Exit\n";
        cout << "Enter your choice: ";
        cin >> choice;
```

```
        switch (choice)
        {
        case 1:
        {
            string name, department;
            int id;
            double salary;

            cout << "Enter employee name: ";
            cin >> name;
            cout << "Enter employee ID: ";
            cin >> id;
            cout << "Enter employee department: ";
            cin >> department;
            cout << "Enter employee salary: ";
            cin >> salary;

            empManager.addEmployee(name, id, department, salary);
            break;
        }
        case 2:
        {
            cout << "\nEmployee Details:" << endl;
            for (int i = 0; i < empManager.getNumEmployees(); ++i)
            {
                displayEmployee(empManager.getEmployee(i));
                cout << endl;
            }
            break;
        }
        case 3:
        {
            int employeeId;
            bool present;
            cout << "\nEnter employee ID to mark attendance: ";
            cin >> employeeId;
            cout << "Enter 1 for present, 0 for absent: ";
            cin >> present;
            empManager.markAttendance(employeeId, present);
            break;
        }
        case 4:
        {
            int employeeId;
            cout << "\nEnter employee ID to reset salary: ";
            cin >> employeeId;
            empManager.resetSalary(employeeId);
```

```
            break;
        }
        case 5:
        {
            int employeeId;
            double amount;
            cout << "\nEnter employee ID to increase salary: ";
            cin >> employeeId;
            cout << "Enter amount to increase: ";
            cin >> amount;
            empManager.increaseSalary(employeeId, amount);
            break;
        }
        case 6:
        {
            empDatabase.displayByCategory(empManager);
            break;
        }
        case 7:
        {
            cout << "Exiting program..." << endl;
            break;
        }
        default:
            cout << "Invalid choice. Please try again." << endl;
        }

    } while (choice != 7);

    return 0;
}
```

Output:

```
Enter your choice: 1
Enter employee name: test
Enter employee ID: 4321
Enter employee department: BDA
Enter employee salary: 3000

Employee Management System
1. Enter Employee Details
2. Display All Employees
3. Mark Attendance
4. Reset Salary
5. Increase Salary
6. Display Employees by Category
7. Exit
Enter your choice: 2

Employee Details:
Name: jaimin
ID: 1234
Department: CS
Salary: 1000
Is Present: No

Name: test
ID: 4321
Department: BDA
Salary: 3000
Is Present: No
```

```
Enter your choice: 2

Employee Details:
Name: jaimin
ID: 1234
Department: CS
Salary: 1000
Is Present: Yes

Name: test
ID: 4321
Department: BDA
Salary: 3000
Is Present: No
```

```
7. Exit
Enter your choice: 5

Enter employee ID to increase salary: 1234
Enter amount to increase: 2000
Salary increased for employee with ID 1234

Employee Management System
1. Enter Employee Details
2. Display All Employees
3. Mark Attendance
4. Reset Salary
5. Increase Salary
6. Display Employees by Category
7. Exit
Enter your choice: 2

Employee Details:
Name: jaimin
ID: 1234
Department: CS
Salary: 3000
Is Present: Yes
```

```
5. Increase Salary
6. Display Employees by Category
7. Exit
Enter your choice: 6
Display employees by category:
1. CS
2. BDA
3. CBA
Enter choice: 1
Employees:
Name: jaimin
ID: 1234
Department: CS
Salary: 3000
Is Present: Yes
No employees found in this category.

Employee Management System
1. Enter Employee Details
2. Display All Employees
3. Mark Attendance
4. Reset Salary
5. Increase Salary
6. Display Employees by Category
7. Exit
Enter your choice:
```