**Lab 9**

**Exercise 3:**

Design a system to manage employees within a company which has a name and sales tax number. Every **Employee** has a **name, ID**, **designation,** and **salary**. Managers have a department, engineer has a specialization and salesperson has a sales target.

Please decide the classes needed and their relationships. Write the prototype of these classes in a header file called company.h.

Implement these classes in a file called company.cpp

In a driver.cpp file write a main function that creates an object of company. Then creates a dynamic array of employees within the company. A sample is given below

int main()

{

// creates a company with name, sales tax number and number of employess

Company \*syst= new Company(“System Private Ltd”, “111-121-131”, 10);

for(int i=0; i< syst->getNumberEmployees(); i++)

{

//ask user the type of employee and add objects of manager, engineer or salesperson

}

Syst->DisplayEmployees();

return 0;

}

**Header File**

#ifndef COMPANY\_H

#define COMPANY\_H

#include <string>

using namespace std;

class Employee {

protected:

    string name;

    int ID;

    string designation;

    double salary;

public:

    Employee(string empName, int empID, string empDesignation, double empSalary);

    ~Employee();

   virtual void display();

};

class Manager : public Employee {

private:

    string department;

public:

    Manager(string empName, int empID, string empDesignation, double empSalary, string empDepartment);

    void display();

};

class Engineer : public Employee {

private:

    string specialization;

public:

    Engineer(string empName, int empID, string empDesignation, double empSalary, string empSpecialization);

    void display();

};

class Salesperson : public Employee {

private:

    double salesTarget;

public:

    Salesperson(string empName, int empID, string empDesignation, double empSalary, double empSalesTarget);

    void display();

};

class Company {

private:

    string name;

    string salesTaxNumber;

    int numberOfEmployees;

    Employee\*\* employees;

public:

    Company(string companyName, string companySalesTaxNumber, int numEmployees);

    ~Company();

    void addEmployee(Employee \*emp,int index);

    int getNumberEmployees();

    void displayEmployees();

};

#endif

// CPP

#include "company.h"

#include <iostream>

using namespace std;

// Employee class

Employee::Employee(string empName, int empID, string empDesignation, double empSalary)

    : name(empName), ID(empID), designation(empDesignation), salary(empSalary) {}

Employee::~Employee() {}

void Employee::display() {

    cout << "Name: " << name << ", ID: " << ID << ", Designation: " << designation << ", Salary: $" << salary << endl;

}

// Manager class

Manager::Manager(string empName, int empID, string empDesignation, double empSalary, string empDepartment)

    : Employee(empName, empID, empDesignation, empSalary), department(empDepartment) {}

void Manager::display() {

    cout << "Manager Information:" << endl;

    Employee::display();

    cout << "Department: " << department << endl;

}

// Engineer class

Engineer::Engineer(string empName, int empID, string empDesignation, double empSalary, string empSpecialization)

    : Employee(empName, empID, empDesignation, empSalary), specialization(empSpecialization) {}

void Engineer::display() {

    cout << "Engineer Information:" << endl;

    Employee::display();

    cout << "Specialization: " << specialization << endl;

}

// Salesperson class

Salesperson::Salesperson(string empName, int empID, string empDesignation, double empSalary, double empSalesTarget)

    : Employee(empName, empID, empDesignation, empSalary), salesTarget(empSalesTarget) {}

void Salesperson::display() {

    cout << "Salesperson Information:" << endl;

    Employee::display();

    cout << "Sales Target: $" << salesTarget << endl;

}

// Company class

Company::Company(string companyName, string companySalesTaxNumber, int numEmployees)

    : name(companyName), salesTaxNumber(companySalesTaxNumber){

    numberOfEmployees = numEmployees;

  employees = new Employee\*[numEmployees];

}

Company::~Company() {

    for (int i = 0; i < numberOfEmployees; i++) {

        delete employees[i];

    }

    delete[] employees;

}

void Company::addEmployee(Employee \* emp,int index) {

    employees[index] = emp;

}

int Company::getNumberEmployees() {

    return numberOfEmployees;

}

void Company::displayEmployees() {

    for (int i = 0; i < numberOfEmployees; i++) {

        cout << "Employee " << (i + 1) << ":" << endl;

        employees[i]->display();

        cout << endl;

    }

}

**Driver**

// driver.cpp

#include "company.h"

#include <iostream>

int main() {

    // Create a company with name, sales tax number, and number of employees

    Company\* syst = new Company("System Private Ltd", "111-121-131", 3);

    for (int i = 0; i < syst->getNumberEmployees(); i++) {

        // Ask user the type of employee and add objects of manager, engineer or salesperson

       string empType;

       cout << "Enter employee type (Manager, Engineer, or Salesperson): ";

       cin >> empType;

        string name;

        int ID;

        string designation;

        double salary;

        cout << "Enter " << empType << " details:" << std::endl;

        cout << "Name: ";

        cin >> name;

        cout << "ID: ";

        cin >> ID;

        cout << "Designation: ";

        cin >> designation;

        cout << "Salary: ";

        cin >> salary;

        if (empType == "Manager") {

            string department;

            cout << "Department: ";

            cin >> department;

            syst->addEmployee (new Manager(name, ID, designation, salary, department),i);

            std::cout << "Manager added successfully!" << endl;

        }

        else if (empType == "Engineer") {

            string specialization;

            cout << "Specialization: ";

            cin >> specialization;

            syst->addEmployee( new Engineer(name, ID, designation, salary, specialization),i);

            cout << "Engineer added successfully!" << endl;

        }

        else if (empType == "Salesperson") {

            double salesTarget;

            cout << "Sales Target: ";

            cin >> salesTarget;

            syst->addEmployee(new Salesperson(name, ID, designation, salary, salesTarget),i);

            cout << "Salesperson added successfully!" << endl;

        }

    }

    cout << "\nEmployees Information:\n";

    syst->displayEmployees();

    delete syst;

    return 0;

}

