CSE 335 Spring 2016: Project 3

Description: In project 3, you need to exercise visitor pattern. Your task is to design different units of a company. The different units of the company are defined as follows: An employee is a unit and has First Name, Last Name, Salary, Hiring Year, and ID. A Department is a unit and has a department name and multiple sub-departments. A Group is also a unit and has a group name and several employees as its group members. A Manager is an employee and has a rank. You will use visitor pattern to display the company's hierarchical structure and compute salary statistics. Using the visitor pattern implement two operations:

1. Print the company hierarchy. Your output should be properly indented as shown in the sample output.
2. Calculate the total salary given to all the employees in the company.

Coding Requirements:

1. Create a UML diagram. You must have all attributes and functions in your UML.
2. You cannot change the main.cpp file. Your program MUST compile and run with the main.cpp file.
3. Your output should match the sample output.
4. Implement your code to implement proper privacy and give clients only the interface they need. Clients should not access data members directly.
5. Reuse functions as much as possible. Ensure proper memory management.
6. Provide *Get* and *Set* functions for all data members.
7. Each class should be either defined and implemented in one header file or defined in a header file and implemented in a cpp file. No two classes should be defined or implemented in one file.
8. You must implement a default constructor, copy constructor, and assignment operator where necessary.

Submission Guidelines:

1. You need to form a team of two to work on the project. If you have not formed a team, please emailfarazah@cse.msu.edu ASAP so that we will team you up.
2. Each team should submit only once. Please do not make duplicate submission.
3. You must submit a zip file that contains two files and one directory:
   (a) Readme.txt including:
      i. netid of each group member,
      ii. full name of each group member,

iii. specific contributions of each individual group member.

(b) UML diagram in pdf format.

(c) The whole Netbeans project directory. You need to ensure that Netbeans can open your project, compile, and run. Your project will not be graded if it does not compile.

4. This project is due via handin (http://secure.cse.msu.edu/handin/) by 11:59 PM on 03/23/2016.

Output:

```
================================================== Test Visitor Pattern
Sales
Group1
          Tom Cruise; 40000; 2000; 5; 1
                    John Smith1; 10000; 2011; 1
                    John Smith2; 20000; 2012; 2
                    John Smith3; 30000; 2013; 3
Group2
          Alice Cooper; 45000; 2000; 9; 2
                    John Doe1; 15000; 2010; 6
                    John Doe2; 25000; 2011; 7
                    John Doe3; 35000; 2012; 8
                    Single Man; 50000; 2009; 10
Total Salary = 270000
```

Main.cpp

```
#include "Employee.h"
#include "Manager.h"
#include "PrintVisitor.h"
#include "Group.h"
#include "Department.h"
#include "SumsalaryVisitor.h"

int main(int argc, char *argv[]) {
    Employee JohnSmith1("John","Smith1",10000,2011, 1);
    Employee JohnSmith2("John","Smith2",20000,2012,2);
    Employee JohnSmith3("John","Smith3",30000, 2013,3);
    Manager TomCruiseManager("Tom","Cruise",40000, 2000, 5,1);

    Employee JohnDoe1("John", "Doe1", 15000, 2010,6);
    Employee JohnDoe2("John", "Doe2", 25000, 2011,7);
    Employee JohnDoe3("John", "Doe3", 35000, 2012,8);
    Manager AliceCooperManager("Alice","Cooper",45000, 2000, 9,2);



    cout<<"================================================== Test Visitor Pattern"<<endl;
    Group group1("Group1");
    group1.addGroupMember(&TomCruiseManager);
    group1.addGroupMember(&JohnSmith1);
    group1.addGroupMember(&JohnSmith2);
    group1.addGroupMember(&JohnSmith3);

    Group group2("Group2");
    group2.addGroupMember(&AliceCooperManager);
    group2.addGroupMember(&JohnDoe1);
    group2.addGroupMember(&JohnDoe2);
    group2.addGroupMember(&JohnDoe3);

    Employee SingleMan("Single","Man",50000,2009,10);

    Department salesDept("Sales");
    salesDept.addDepartmentMember(&group1);
    salesDept.addDepartmentMember(&group2);
    salesDept.addDepartmentMember(&SingleMan);

    PrintVisitor pv;
    salesDept.Accept(&pv);

    SumsalaryVisitor sv;
    salesDept.Accept(&sv);
    cout<<"Total Salary = "<<sv.getTotalSalary()<<endl;
    sv.restTotalSalary();
    }
```