Sabancı University

Faculty of Engineering and Natural Sciences

CS204 Advanced Programming

Fall 2016-2017

Homework7 - Inheritance and Polymorphism

Due: 8/12/2016, 23:00

PLEASE NOTE:
Your program should be a robust one such that you have to consider all relevant programmer mistakes and extreme cases; you are expected to take actions accordingly!
You HAVE TO write down the code on your own.
You CANNOT HELP any friend while coding.
Plagiarism will not be tolerated!

1. Introduction

The aim of this assignment is to make you familiar with **inheritance** and **polymorphism**. You will implement classes that will be inherited another class (as is illustrated in Figure.1). The content and a brief description of each class will also be given to you in this document. Polymorphism will take part in the main() function. Although the main() function will be given to you (Fig.2), you still need to implement a function used in main() for polymorphism. Your implementation should be done in such a way that it will be consistent with the program flow (i.e. have exactly the same sample output) as it is described in section 3. Basically, the main idea of this homework is displaying the profiles and calculating the salaries of the people in the system. As stated above, this must be done by enabling inheritance and polymorphism behind the scenes, as described below.

2. Classes

Overall, you will have to implement **seven** classes. Below is a brief description of the member functions and variables each class should have. You should have in mind that some of those should not be declared or implemented at all; rather they can be simply inherited from their base class. It is up to you to decide which will be inherited, and which will be implemented.

1. **Person**: this is the base class for all other classes. It should have the following properties:
   - **Member variables:**
     i. string *name*: keeps the name of the shape;
     ii. int *ID:* id of person.
   - **Member functions: (the return types are also missing)**
     i. *calSalary()*:calculates and returns the salary of a person;
     ii. *displayProfile()*: print values of related member variables of each person;
     iii. *getName()*: returns the name of the person.

   You can put constructors and destructors as needed.
2. **Student:** *extends Person*. It has new member variable of type *bool* called *dormAccomodation* that is **true** if the student is staying in university campus. You can put constructors and destructors as needed, which (in order to avoid rewriting code) can call the constructor(s) of the base.
3. **Employee:** *extends Person*. It adds new member variable of type string called *department* which specifies the department an employee works in. You can implement constructors and destructors as needed, and call the base class constructor(s).

4. **Undergraduate:** *extends student*. It has a member variable *int year*(1 = freshman, 2 = sophomore, 3=junior, 4=senior). For an undergraduate student the salary is always zero and *displayProfile* function will print related member variables; *name*, *ID*, *dormAccomodation*, and *year*.
5. **Graduate**: *extends Student*. It has two new member variables *int scholarRank*(1=no scholar, 2= half scholar, 3=full scholar) and *string level which can be "master" or "phd".* The *salary* is defined as (*scholarRank-1)*x1250. If the level is "phd" then the salary will be increased by 600 and if the level is master then the salary will be increased by 500.
6. **Staff**: *extends Employee*. It has two new member variables called *int position (values can be 4=manager, 3=engineer, 2=specialist, 1=other)*and *int daysOfWork*. The salary of a staff is defined as 40 x *daysOfWork* x *position*.
7. **FacultyMember**: extends Employee. Furthermore, it has three new member variables called *int classestaught (number of classes the instructor taught)*,*int officeHours (number of hours an instructor stays in his office)* and *int rank (1=instructor,2=assistant, 3= associate, 4=professor)*. The salary for a faculty member is defined as *classestaught* x 2000+ officeHour x 100+ rank x 1000.

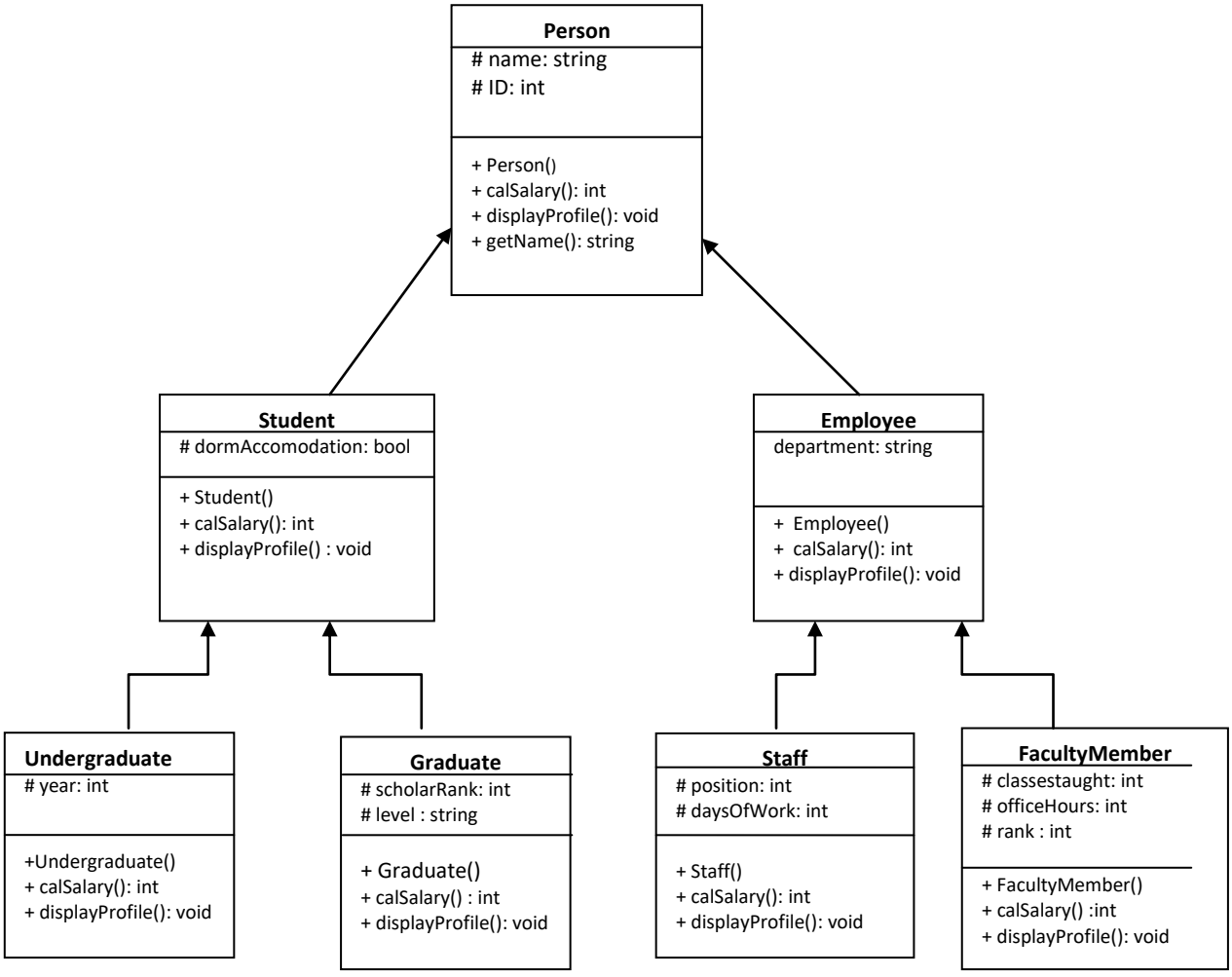

Figure 1. UML class diagram for the described classes.

Figure 1 shows the UML class diagrams for the described classes. It is up to you to decide which (or whether) certain members will be **virtual** or not. You should do as much of the job as you can in the base classes, thus avoiding the repetitions in the inherited classes. Some of the classes need to be ***abstract* classes**. Furthermore, you should decide about access-specifier (*public*, *protected* or *private*) for both class variables and methods. The inheritance access-specifier (*public*, *protected* or *private*) is also left to you. Finally, you should decide which functions will be overridden, and which will remain the same as the base class during the inheritance process.

3. Program flow

Main function is given in Figure.2. You should implement the getPerson() function used inside main() in such a way that the output will be consistent with the sample run given in Figure 3. You may assume that the inputs are given correctly, e.g. if an integer number is expected, the input will be integer. In addition we assume that the user enters the values for member variables correctly. In order to avoid using *getline*() function, the name of the person should not contain empty spaces (e.g. empty space, tab, …) . Other unspecified issues, should be handled as you think is a proper way of dealing with them.

```
int main()
{
	cout<<"WELCOME TO THE PERSON PROGRAM"<<endl;
	cout<<"FOR EXITIING PRESS Y/y, OTHERWISE PRESS ANY KEY"<<endl;
	person *person_1; /* define person_1 of the class person.*/
	char c;
	while (tolower(c = getchar())!='y')
	{
		cout<<"Getting person "<<endl;
		person_1 = getPerson();

	cout<<"**********************************************************************"<<e
ndl<<endl;
	    cout<<"The profile of a person:"<<endl;
		person_1->displayProfile();
		cout<<"Salary: "<<person_1->calSalary()<<endl;
	cout<<"**********************************************************************"<<e
ndl;
	cout<<"FOR EXITIING PRESS Y/y, OTHERWISE, FOR ANOTHER COMPARISON PRESS ANY
KEY"<<endl<<endl;
      cin.ignore();//flushing the buffer for remaining character(s), in order
getchar() to work
	}
	cout<<"PROGRAM EXITING. THANKS FOR USING IT."<<endl;
    system("pause");
    return 0;
}
```

Figure 2. Main function

Sample runs:

```
WELCOME TO THE PERSON PROGRAM
FOR EXITIING PRESS Y/y, OTHERWISE, FOR ANOTHER COMPARISON PRESS ANY KEY


Getting person
```

Choose an option (1,2,3, or 4):
1. undergraduate student
2. graduate student
3. staff
4. faculty member
1
You chose undergraduate. Give person's name, id, accommodation status (1 for yes and 0 for no)
, and year:
Burcu
1233
0
1
****

The profile of a person:
Name: Burcu
ID: 1233
Dormitory accommodation: No
Year : 1
Salary: 0
****
FOR EXITIING PRESS Y/y, OTHERWISE, FOR ANOTHER COMPARISON PRESS ANY KEY

Getting person

Choose an option (1,2,3, or 4):
1. undergraduate student
2. graduate student
3. staff
4. faculty member
2
You chose graduate. Give person's name, id, accommodation status (1 for yes and 0 for no),scho
lar Rank(btw 1 and 3) and level(phd or master):
Kemal
4564
1
2
phd
****

The profile of a person:
Name: Kemal
ID: 4564
Dormitory accommodation: Yes
Scholarship rank: 2
graduate level: phd
Salary: 1850
****
FOR EXITIING PRESS Y/y, OTHERWISE, FOR ANOTHER COMPARISON PRESS ANY KEY

Getting person

Choose an option (1,2,3, or 4):
1. undergraduate student
2. graduate student
3. staff
4. faculty member
3
You chose staff. Give person's name, id, department, position(btw 1 and 4) and Days of work:
Buket
8592
SR
2
28
****

The profile of a person:
Name: Buket
ID: 8592
Department: SR
Position: 2
Days of work: 28
Salary: 2240
****
FOR EXITIING PRESS Y/y, OTHERWISE, FOR ANOTHER COMPARISON PRESS ANY KEY

Getting person

Choose an option (1,2,3, or 4):
1. undergraduate student
2. graduate student
3. staff
4. faculty member
4
You chose faculty member. Give person's name, id, department, number of classes taught,office hours, and rank(btw 1 and 4):
Cengiz
85961
CS
3
10
2
****

The profile of a person:
Name: Cengiz
ID: 85961
Department: CS
Office hours: 10

```
# of classes taught: 3
Rank: 2
Salary: 9000
*****************************************************************
FOR EXITIING PRESS Y/y, OTHERWISE, FOR ANOTHER COMPARISON PRESS ANY KEY
y
PROGRAM EXITING. THANKS FOR USING IT.
Press any key to continue . . .
```

Figure 3. Sample runs

Some Important Rules: In order to get a full credit, your programs must be efficient and well presented, presence of any redundant computation or bad indentation, or missing, irrelevant comments are going to decrease your grades. You also have to use understandable identifier names, informative introduction and prompts. Modularity is also important; you have to use functions wherever needed and appropriate.

When we grade your homeworks we pay attention to these issues. Moreover, in order to observe the real performance of your codes, we may run your programs in *Release* mode and **we may test your programs with very large test cases**.

What and where to submit (PLEASE READ, IMPORTANT): You should prepare (or at least test) your program using MS Visual Studio 2012 C++. We will use the standard C++ compiler and libraries of the abovementioned platform while testing your homework. It'd be a good idea to write your name and last name in the program (as a comment line of course).

Submissions guidelines are below. Some parts of the grading process are automatic. Students are expected to strictly follow these guidelines in order to have a smooth grading process. If you do not follow these guidelines, depending on the severity of the problem created during the grading process, 5 or more penalty points are to be deducted from the grade.

Name your cpp file that contains your program as follows:

"SUCourseUserName_YourLastname_YourName_HWnumber.cpp"

Your SUCourse user name is actually your SUNet username that is used for checking sabanciuniv e-mails. Do NOT use any spaces, non-ASCII and Turkish characters in the file name. For example, if your SUCourse user name is cago, name is Çağlayan, and last name is Özbugsızkodyazaroğlu, then the file name must be:

Cago_Ozbugsizkodyazaroglu_Caglayan_hw2.cpp

Do not add any other character or phrase to the file name. Make sure that this file is the latest version of your homework program. Compress this cpp file using WINZIP or WINRAR programs. Please use "zip" compression. "rar" or another compression mechanism is NOT allowed. Our homework processing system works only with zip files. Therefore, make sure that the resulting compressed file has a zip extension. Check that your compressed file opens up correctly and it contains your cpp file.

You will receive no credits if your compressed zip file does not expand or it does not contain the correct file. The naming convention of the zip file is the same as the cpp file (except the extension of the file of course). The name of the zip file should be as follows:

SUCourseUserName_YourLastname_YourName_HWnumber.zip

For example zubzipler_Zipleroglu_Zubeyir_hw1.zip is a valid name, but

hw1_hoz_HasanOz.zip, HasanOzHoz.zip

are **NOT** valid names.

Submit via SUCourse ONLY! You will receive no credits if you submit by other means (e-mail, paper, etc.).

Successful submission is one of the requirements of the homework. If, for some reason, you cannot successfully submit your homework and we cannot grade it, your grade will be 0.

Good Luck!

CS204 Team -Leyli Javid Khayati