



Object-Oriented Modeling and Design 4th Assignment Design with GoF 2

Problem:

We will design a part of the **company's payroll system**.

Requirements:

- The company has different types of employees, i.e., *workers* and *researchers*.
- The salaries of employees are calculated differently depending on their types.
For example, the monthly salary of a worker is calculated by multiplying the hourly wage of that worker by the total number of hours worked in the month.
The monthly salary of a researcher is calculated based on their active projects. For example, it is 30000TL for one project, 50000 for two projects, and 65000TL for three or more projects.
- A company can have departments that have employees and other sub-departments.
- Depending on certain conditions, a class **Accounting** sometimes gets the *total* monthly salaries of all company employees and sometimes the *maximum* or *minimum* salaries.
- A new employee type may be added to the system in the future with a different method for calculating the monthly salary. For example, the *managers* can have a fixed base salary plus a promotion based on the profit.
- In the future, the class **Accounting** may need the *average* of all salaries in the company.

To Do:

- a) **Design the system** using GoF software design patterns to achieve the required flexibility and draw the **UML class diagram**. Show all necessary members of the classes and the parameters of the methods. Mention the GoF design patterns used in your solution.
For this part, create a file `class_diagram.pdf`.
- b) **Write the program** `oomd2324h4.cpp` based on your design using the C++ programming language.
 - Your program may contain two employee types, i.e., **Worker** and **Researcher**.
 - You can initialize objects using the necessary data for salary calculation.Examples:

```
// Worker object: hourly wage = 200TL, total of worked hours = 160  
Worker worker1{200, 160};  
  
// Researcher object: the number of active projects = 2  
Researcher researcher1{2};
```

 - The `getSalary` methods of the classes can calculate and return the salaries of the related employees.
 - You do not need to develop a factory class for this assignment. You may create required objects in your main function as hard-coded data.

- For testing, create the following structure:
 Company:
 worker1 hourly wage = 500TL, total of worked hours = 100
 Department1:
 worker2, hourly wage = 200TL, total of worked hours = 160
 worker3, hourly wage = 300TL, total of worked hours = 160
 Department1.1: (Sub-department in the Department1)
 worker4, hourly wage = 100TL, worked hours = 120
 researcher1, the number of active projects = 2
 researcher2, the number of active projects = 3
- Perform the following operations in your main function:
 - Create the given structure of the company.
 - An `Accounting` object gets the total of salaries and prints it.
 - The same `Accounting` object gets the average of the salaries of the employees in the company and prints it.

SUBMISSION:

- Upload the files `class_diagram.pdf` and `oofd2324h4.cpp` to Ninova by **Saturday, 23:00 May 18, 2024**.
- **Late submitted assignments are not accepted.** Do not send your solutions by e-mail. We will only accept files uploaded to the official Ninova e-learning system before the deadline. Do not risk leaving your submission to the last few minutes.
- **Cheating** will not be tolerated. Any cheating is subject to the University disciplinary proceedings.
 It is allowed to discuss how to solve a problem with your classmates; however, **this assignment is not group homework. The actual solution should be an independent effort.**