

LP2A Written Exam
Thursday May 15th, 2025
Duration 1h30
No documents

Exercise #0 Back to the lectures (4 points)

Question#1: How does inheritance promote software reusability?

Question#2: What is the purpose of interfaces? How to write a class which implements an interface in java? How many interfaces a class can implement?

Exercise #1 Can AI help me? (12 points)

This the subject which has been proposed to AI.

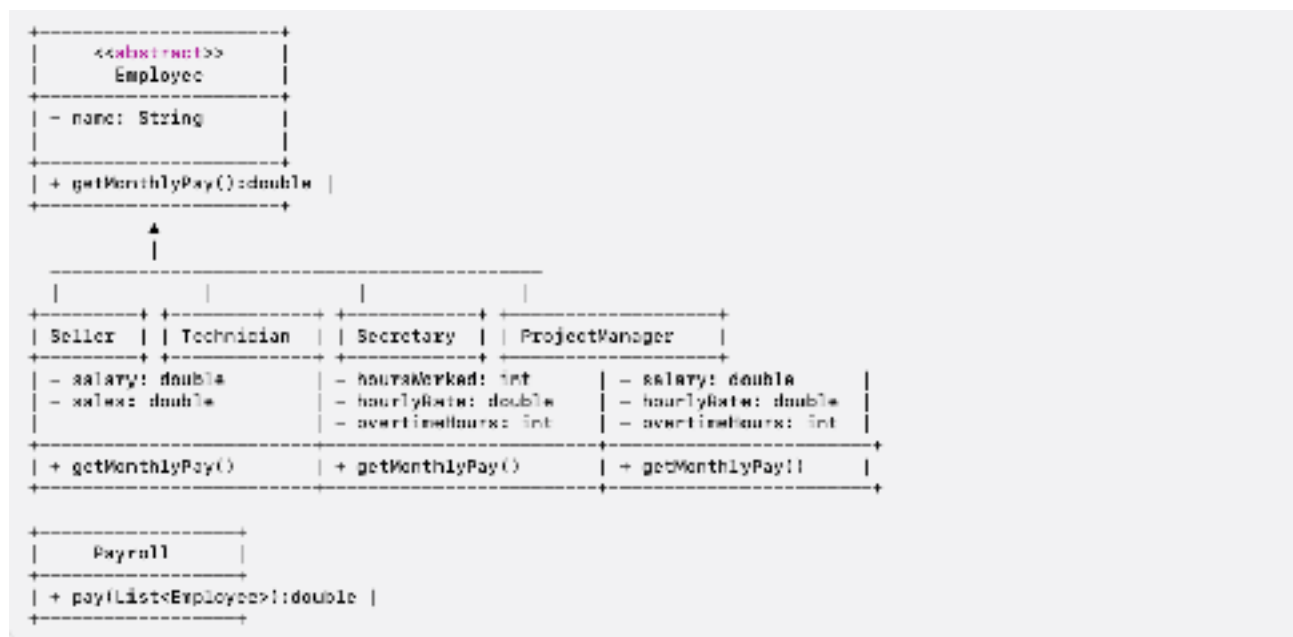
You will program the calculation of the monthly wages of employees of a company.
This company has several types of employees:

- Seller: which have a fixed salary and a bonus of 1% of their sales for the month,
- Technician: whose salary is calculated according to the number of hours worked in the month with a fixed hourly rate and additional hours, paid 30% more than the normal hours,
- Secretary: paid in the same way as technicians, but with a 20% rate for overtime,
- Project manager: paid a fixed salary regardless of the number of hours worked.

Propose a UML architecture for these classes and for the class « Payroll » which has a method « pay » whose parameter is the list of employees and returns the total payroll for the company. Implements it in Java.

This is the result of the AI proposal:

UML diagram



Code

```
abstract class Employee {
    protected String name;

    public Employee(String name) {
        this.name = name;
    }

    public abstract double getMonthlyPay();
}
```

```
// Seller
class Seller extends Employee {
    private double fixedSalary;
    private double sales;

    public Seller(String name, double fixedSalary, double
sales) {
        super(name);
        this.fixedSalary = fixedSalary;
        this.sales = sales;
    }

    @Override
    public double getMonthlyPay() {
        return fixedSalary + 0.01 * sales;
    }
}
```

```
// Technician
class Technician extends Employee {
    protected int hoursWorked;
    protected double hourlyRate;
    protected int overtimeHours;

    public Technician(String name, int hoursWorked,
double hourlyRate, int overtimeHours) {
        super(name);
        this.hoursWorked = hoursWorked;
        this.hourlyRate = hourlyRate;
        this.overtimeHours = overtimeHours;
    }

    @Override
    public double getMonthlyPay() {
        double regularPay = (hoursWorked - overtimeHours)
* hourlyRate;
        double overtimePay = overtimeHours * hourlyRate *
1.3;
        return regularPay + overtimePay;
    }
}
```

```
// Secretary
class Secretary extends Technician {

    public Secretary(String name, int hoursWorked, double
hourlyRate, int overtimeHours) {
        super(name, hoursWorked, hourlyRate,
overtimeHours);
    }

    @Override
    public double getMonthlyPay() {
        double regularPay = (hoursWorked - overtimeHours)
* hourlyRate;
        double overtimePay = overtimeHours * hourlyRate *
1.2;
        return regularPay + overtimePay;
    }
}
```

```
// Project Manager
class ProjectManager extends Employee {
    private double salary;

    public ProjectManager(String name, double salary) {
        super(name);
        this.salary = salary;
    }

    @Override
    public double getMonthlyPay() {
        return salary;
    }
}
```

```
// Payroll Class
class Payroll {
    public double pay(List<Employee> employees) {
        double total = 0;
        for (Employee e : employees) {
            total += e.getMonthlyPay();
        }
        return total;
    }
}
```

```
// Example main class
public class Main {
    public static void main(String[] args) {
        List<Employee> employees = new ArrayList<>();
        employees.add(new Seller("Alice", 2000, 30000));
        employees.add(new Technician("Bob", 160, 20,
10));
        employees.add(new Secretary("Carol", 160, 18,
8));
        employees.add(new ProjectManager("Dave", 3500));

        Payroll payroll = new Payroll();
        double total = payroll.pay(employees);
        System.out.println("Total payroll: " + total);
    }
}
```

Question#1: Criticise the result obtained by the AI chat on UML point of view.

Question#2: Propose your UML diagram correcting the one proposed by AI

Question#3: Criticise the provided code and indicate the modifications

Like all businesses, it has a director. The director also receives a salary, calculated from a fixed salary and a bonus of 0.4% of the results of the company.

Question #4: Propose a class Director and add it into the list of « Payroll » class to calculate payroll.

Exercise #2 Abstraction is Abstract (4 points)

Gives the result written by this program on screen after having corrected possible mistakes.

```
//Main class definition

class AbstractStatic {
public static void main(String[]
args) {
    N n1 = new N() ;
    N n2 = new N(4) ;
    N n2 = new N(3) ;
    C c1= new C() ;
    n1.m1(2) ;
    n2.m1(true) ;

    System.out.println(c1.n) ;
    System.out.println(n1.n) ;
    System.out.println(n1.i) ;
    System.out.println(n2.n) ;
    System.out.println(n2.i) ;
}
}

// Class C definition
abstract class C {
    private int n = 0 ;
    C() {
        n++ ; }
}

//Class N definition

class N extends C{
    private int i = 1 ;
    private boolean b= true ;

    N(int j) {
        this.i=i+2 ;
        this.n=i+5 ;
    }

    N(int i) {
        this.i=i+3 ;
        this.n= i+6 ;
    }

    N(boolean b) {
        this.b= b ;
        i++ ;
    }

    void m1(int j) {
        i++ ;
    }

    void m1(boolean bool) {
        if(bool)
        {
            i++ ;
        }
    }
}
```