Object Oriented Programming Job Sheet 06

From:

AL AZHAR RIZQI RIFA'I FIRDAUS

Class:

2 I

Absence:

01

Student Number Identity:

2241720263

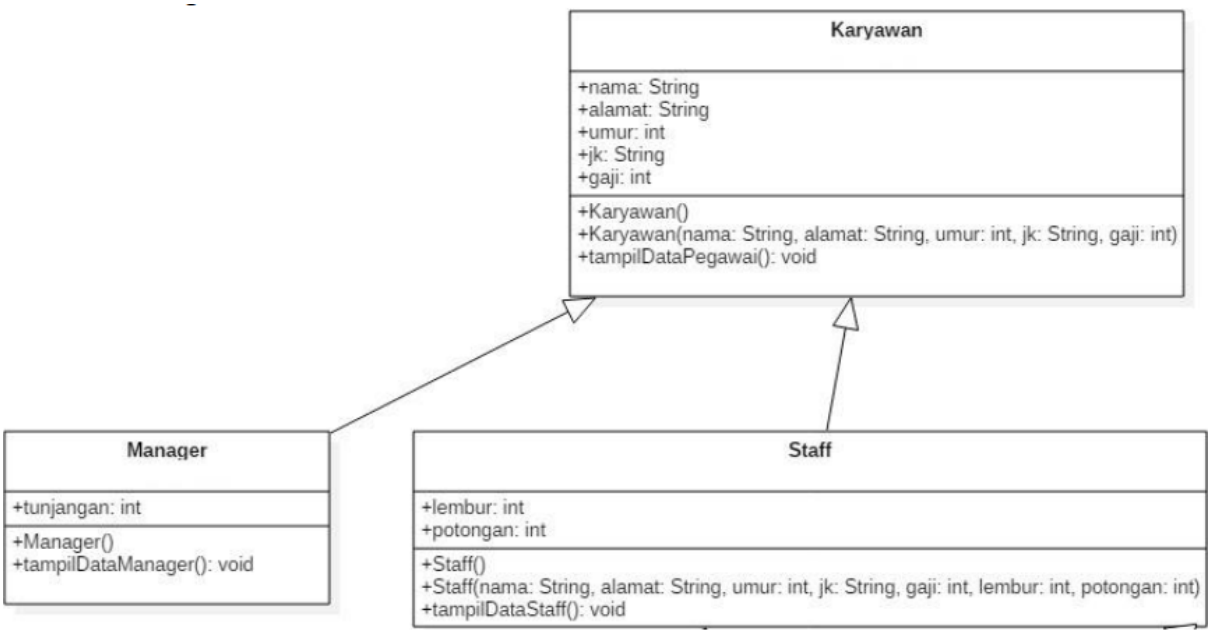
Department:

Information Technology

Study Program:

Informatics Engineering

Experiment 1



Code:

Employee.java | Manager.java | Main.java | Staff.java

src > main > java > com > azhar > exp1 > Employee.java > Employee > printDataEmployee()

```java
package com.azhar.exp1;

public class Employee {
    public String name, address, gender;
    public int age, salary;

    public Employee() {

    }

    public Employee(String name, String address, String gender, int age, int salary) {
        this.name = name;
        this.address = address;
        this.gender = gender;
        this.age = age;
        this.salary = salary;
    }

    Codeium: Refactor | Explain | Generate Javadoc
    public void printDataEmployee() {
        System.out.println("Name: " + name);
        System.out.println("Address: " + address);
        System.out.println("Gender: " + gender);
        System.out.println("Age: " + age);
        System.out.println("Salary: " + salary);
    }
}
```

Employee.java | Manager.java | Main.java | Staff.java

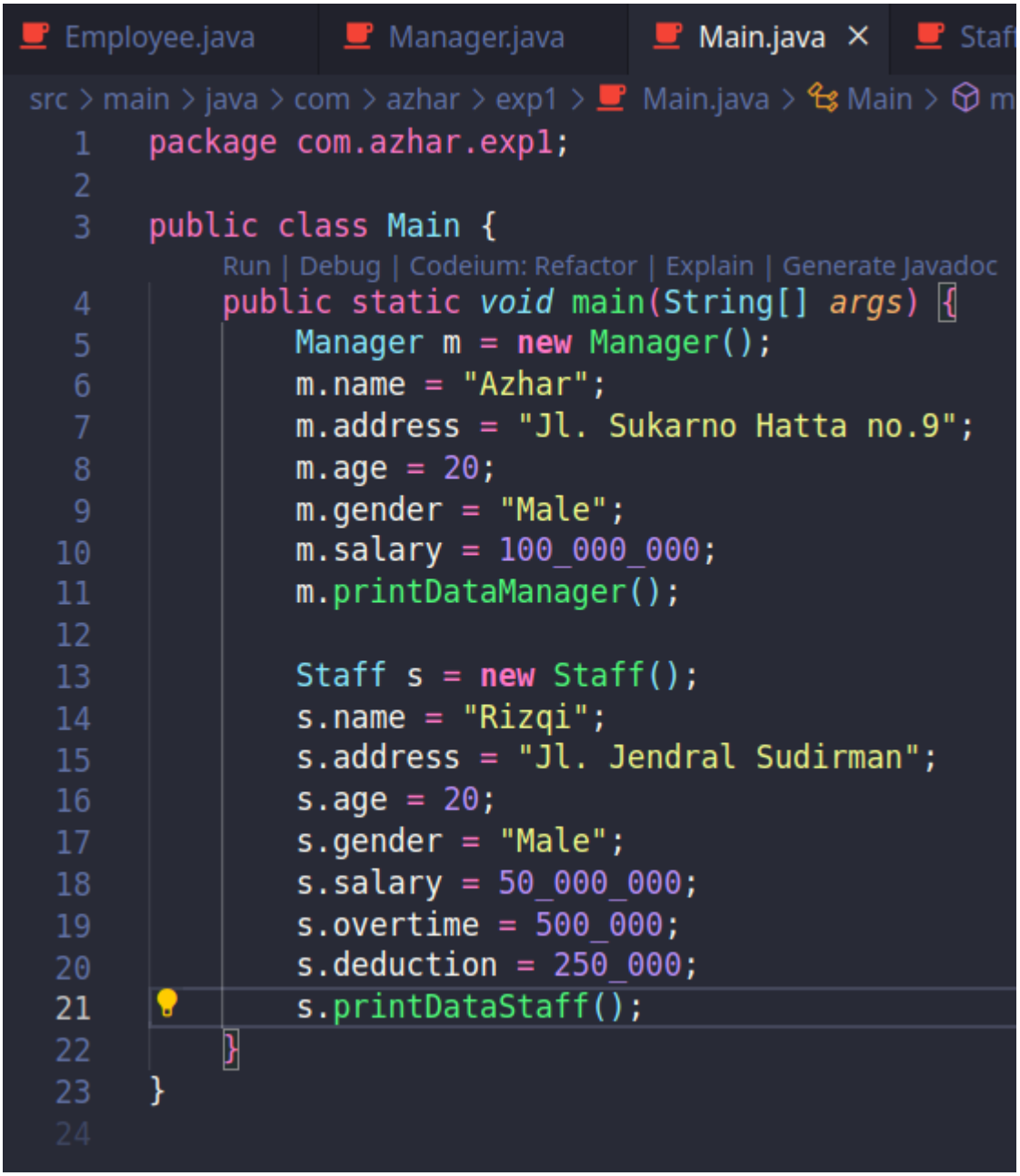
src > main > java > com > azhar > exp1 > Manager.java > Manager > printDataManager()

```java
package com.azhar.exp1;

public class Manager extends Employee {
    public int allowance;

    public Manager() {

    }

    Codeium: Refactor | Explain | Generate Javadoc
    public void printDataManager() {
        super.printDataEmployee();
        System.out.println("Allowance: " + allowance);
        System.out.println("Total Salary: " + (super.salary + allowance));

    }
}
```

Employee.java | Manager.java | Main.java | Staff.java

src > main > java > com > azhar > exp1 > Staff.java > Staff > printDataStaff()

```java
package com.azhar.exp1;

public class Staff extends Employee {
    public int overtime, deduction;

    public Staff() {

    }

    public Staff(String name, String address, String gender, int age, int salary, int overtime, int deduction) {
        super(name, address, gender, age, salary);
        this.overtime = overtime;
        this.deduction = deduction;
    }

    Codeium: Refactor | Explain | Generate Javadoc
    public void printDataStaff() {
        super.printDataEmployee();
        System.out.println("Overtime: " + overtime);
        System.out.println("Deduction: " + deduction);
        System.out.println("Total Salary: " + (salary + overtime - deduction));
    }
}
```


Employee.java
Manager.java
Main.java
Staf
src > main > java > com > azhar > exp1 > Main.java > Main > m
1 package com.azhar.exp1;
2
3 public class Main {
Run | Debug | Codeium: Refactor | Explain | Generate Javadoc
4 public static void main(String[] args) {
5 Manager m = new Manager();
6 m.name = "Azhar";
7 m.address = "Jl. Sukarno Hatta no.9";
8 m.age = 20;
9 m.gender = "Male";
10 m.salary = 100_000_000;
11 m.printDataManager();
12
13 Staff s = new Staff();
14 s.name = "Rizqi";
15 s.address = "Jl. Jendral Sudirman";
16 s.age = 20;
17 s.gender = "Male";
18 s.salary = 50_000_000;
19 s.overtime = 500_000;
20 s.deduction = 250_000;
21 s.printDataStaff();
22 }
23 }
24

Result:

```
→ zharsuke@box  ~/Documents/College/Semester_3/oop/meet-7/coding git:(master) ✗
ExceptionMessages -cp /home/zharsuke/Documents/College/Semester_3/oop/meet-7/cod
Picked up _JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=on -Dswing.aatext=true
Name: Azhar
Address: Jl. Sukarno Hatta no.9
Gender: Male
Age: 20
Salary: 100000000
Allowance: 0
Total Salary: 100000000
Name: Rizqi
Address: Jl. Jendral Sudirman
Gender: Male
Age: 20
Salary: 50000000
Overtime: 500000
Deduction: 250000
Total Salary: 50250000
→ zharsuke@box  ~/Documents/College/Semester_3/oop/meet-7/coding git:(master) ✗
```

Question

1. Name which class is a superclass and subclass from experiment 1 above!

   - From code above, Employee is a superclass, then Manager and Staff is a subclass because both extends Employee.

2. What keywords are used to derive one class to another?

   - extends.

3. Look at the program code in the Manager class, what attributes does the class have? Mention which attributes are inherited from the Employee class!

   - From code above, Manager class has one attribute that is allowance, then has one attribute that inherited from Employee class which is salary. It because use super to call it.

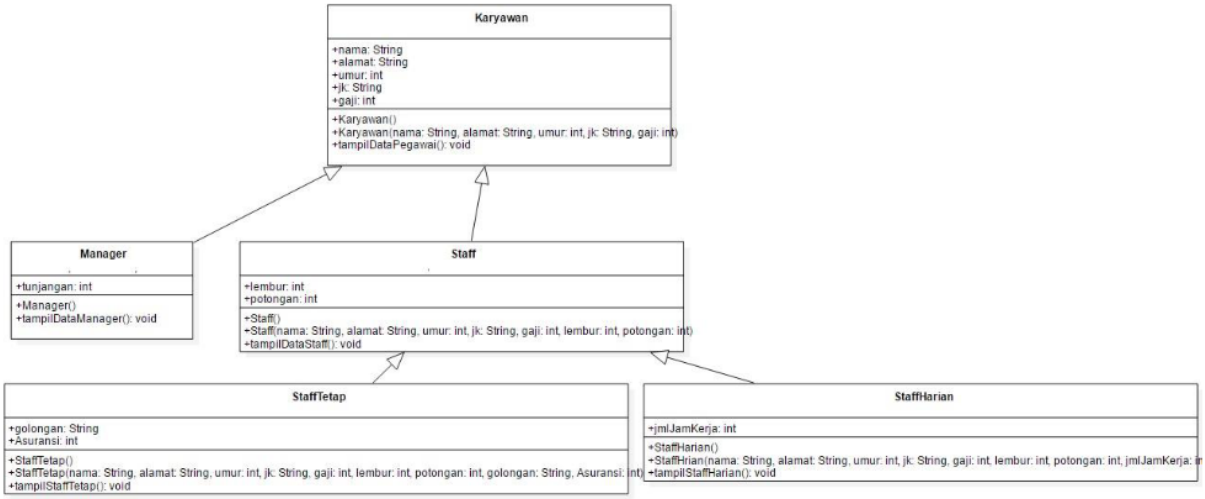
4. Explain the super keyword in the program snippet below which is contained in the Manager class!

   - super is keyword that used to call attribute from superclass / parent class.

5. The program in experiment 1 above belongs to what type of inheritance? Explain why!

   - From code above, it is hierarchical inheritance because super class which is Employee has sub class more than one that is Manager and Staff and both in same level.

Experiment 2


Karyawan
+nama: String
+alamat: String
+umur: int
+jk: String
+gaji: int
+Karyawan()
+Karyawan(nama: String, alamat: String, umur: int, jk: String, gaji: int)
+tampilDataPegawai(): void
Manager
+tunjangan: int
+Manager()
+tampilDataManager(): void
Staff
+lembur: int
+potongan: int
+Staff()
+Staff(nama: String, alamat: String, umur: int, jk: String, gaji: int, lembur: int, potongan: int)
+tampilDataStaff(): void
StaffTetap
+golongan: String
+Asuransi: int
+StaffTetap()
+StaffTetap(nama: String, alamat: String, umur: int, jk: String, gaji: int, lembur: int, potongan: int, golongan: String, Asuransi: int)
+tampilStaffTetap(): void
StaffHarian
+jmlJamKerja: int
+StaffHarian()
+StaffHrian(nama: String, alamat: String, umur: int, jk: String, gaji: int, lembur: int, potongan: int, jmlJamKerja: in
+tampilStaffHarian(): void

Code:

Manager.java | FixedStaff.java | DailyStaff.java | Main.java | Staff.java

src > main > java > com > azhar > exp1 > FixedStaff.java > FixedStaff > printDataFixedStaff()

```java
package com.azhar.exp1;

public class FixedStaff extends Staff {
    public String group;
    public int assurance;

    public FixedStaff() {

    }

    public FixedStaff(String name, String address, String gender, int age, int salary, int overtime, int deduction , String group, int assurance) {
        super(name, address, gender, age, salary, overtime, deduction);
        this.group = group;
        this.assurance = assurance;
    }

    Codeium: Refactor | Explain | Generate Javadoc
    public void printDataFixedStaff() {
        System.out.println("==========Data Fixed Staff==========");
        super.printDataStaff();
        System.out.println("Group: " + group);
        System.out.println("Assurance: " + assurance);
        System.out.println("Net Salary: " + (salary + overtime - deduction - assurance));
    }
}
```

Manager.java | FixedStaff.java | DailyStaff.java | Main.java | Staff.java

src > main > java > com > azhar > exp1 > DailyStaff.java > DailyStaff > printDataDailyStaff()

```java
package com.azhar.exp1;

public class DailyStaff extends Staff {
    public int amountWorkingHours;

    public DailyStaff() {

    }

    public DailyStaff(String name, String address, String gender, int age, int salary, int overtime, int deduction, int amountWorkingHours) {
        super(name, address, gender, age, salary, overtime, deduction);
        this.amountWorkingHours = amountWorkingHours;
    }

    Codeium: Refactor | Explain | Generate Javadoc
    public void printDataDailyStaff() {
        System.out.println("==========Data Daily Staff==========");
        super.printDataStaff();
        System.out.println("Amount Working Hours: " + amountWorkingHours);
        System.out.println("Net Salary: " + (salary * amountWorkingHours + overtime - deduction));
    }
}
```

Manager.java | FixedStaff.java | DailyStaff.java | Main.java | Staff.java

src > main > java > com > azhar > exp1 > Main.java > Main > main(String[])

```java
package com.azhar.exp1;

public class Main {
    Run | Debug | Codeium: Refactor | Explain | Generate Javadoc
    public static void main(String[] args) {
        // Manager m = new Manager();
        // m.name = "Azhar";
        // m.address = "Jl. Sukarno Hatta no.9";
        // m.age = 20;
        // m.gender = "Male";
        // m.salary = 100_000_000;
        // m.printDataManager();

        // Staff s = new Staff();
        // s.name = "Rizqi";
        // s.address = "Jl. Jendral Sudirman";
        // s.age = 20;
        // s.gender = "Male";
        // s.salary = 50_000_000;
        // s.overtime = 500_000;
        // s.deduction = 250_000;
        // s.printDataStaff();

        FixedStaff fs = new FixedStaff(name:"Rizqi", address:"Jl. Jendral Sudirman", gender:"Male", age:20, salary:50_000_000, overtime:500_000, deduction:250_000, group:"IT", assurance:100_000);
        fs.printDataFixedStaff();

        DailyStaff ds = new DailyStaff(name:"Rizqi", address:"Jl. Jendral Sudirman", gender:"Male", age:20, salary:50_000_000, overtime:500_000, deduction:250_000, amountWorkingHours:8);
        ds.printDataDailyStaff();
    }
}
```

Result:

```
→ zharsuke@box  ~/Documents/College/Semester_3/oop/meet-7/coding git:(master) ✗
ExceptionMessages -cp /home/zharsuke/Documents/College/Semester_3/oop/meet-7/cod
Picked up _JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=on -Dswing.aatext=true
==========Data Fixed Staff==========
Name: Rizqi
Address: Jl. Jendral Sudirman
Gender: Male
Age: 20
Salary: 50000000
Overtime: 500000
Deduction: 250000
Total Salary: 50250000
Group: IT
Assurance: 100000
Net Salary: 50150000
==========Data Daily Staff==========
Name: Rizqi
Address: Jl. Jendral Sudirman
Gender: Male
Age: 20
Salary: 50000000
Overtime: 500000
Deduction: 250000
Total Salary: 50250000
Amount Working Hours: 8
Net Salary: 400250000
→ zharsuke@box  ~/Documents/College/Semester_3/oop/meet-7/coding git:(master) ✗
```

Question

6. Based on the class above, which of the following is single inheritance and which one is multilevel inheritance?

- Class that single inheritance is:
  - Manager => Employee
- Class that multilevel inheritance is:
  - DailyStaff => Staff => Employee
  - FixedStaff => Staff => Employee

7. Look at the program code of the StaffFixed and StaffDaily classes, what attributes are owned by the class? Mention which attributes are inherited from the class Staff class!

- FixedStaff:
  - Attribute = group, assurance
  - Attribute that inherited = name, address, gender, age, salary, overtime, deduction.
- DailyStaff:
  - Attribute = amountWorkingHours
  - Attribute that inherited = name, address, gender, age, salary, overtime, deduction.

8. What is the function of the following program snippet in the Dailystaff class

```
super(nama,alamat,jk,umur,gaji,lembur,potongan);
```

- It is to call parent class constructor that has parameters that is from Staff class.

9. What is the function of the following program snippet in the Dailystaff class?

```
super.tampilDataStaff();
```

- It is to call parent class method that is from Staff class.

10. Note the program code below which is contained in the StaffTetap class.

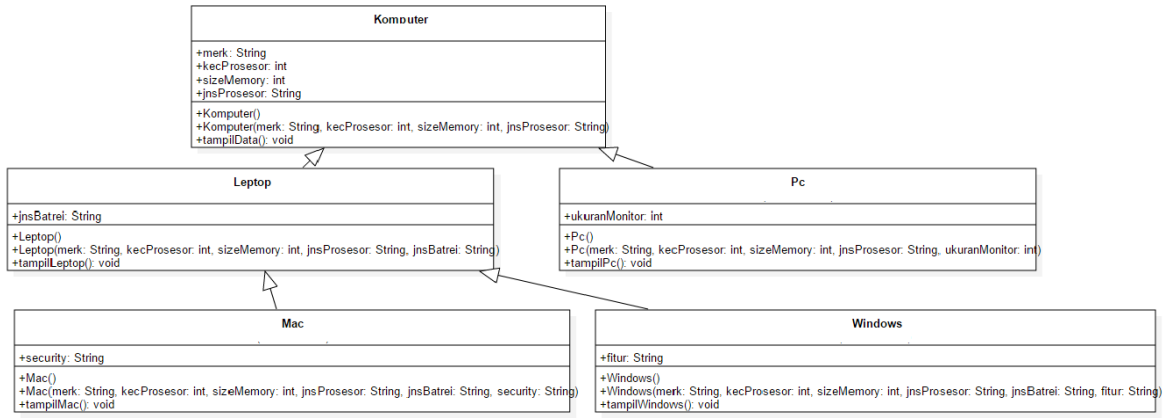
```
System.out.println("Gaji Bersih      ="+(gaji+lembur-potongan-asuransi));
```

It can be seen in the program snippet above that the salary, overtime and deduction attributes can be accessed directly. Why does this happen and how does the StaffTetap class have the attributes salary, overtime, and deductions when there are no salary, overtime, and deductions attributes declared in the class? and deductions?

- It happen because attribute above already called at FixedStaff constructor that has super constructor that has parameter that inherited from Staff class constructor.

Assignment

1. Create a program with the concept of inheritance as in the following class diagram. Then make an object instantiation to display data in the Mac, Windows and Pc!



Code:

Computer.java | Laptop.java | Pc.java | Mac.java | Windows.java | Main.java

src > main > java > com > azhar > asg > Computer.java > ...

```java
package com.azhar.asg;

public class Computer {
    public String brand, processorType;
    public int memorySize, processorSpeed;

    public Computer() {

    }

    public Computer(String brand, String processorType, int memorySize, int processorSpeed) {
        this.brand = brand;
        this.processorType = processorType;
        this.memorySize = memorySize;
        this.processorSpeed = processorSpeed;
    }

    Codeium: Refactor | Explain | Generate Javadoc
    public void printDataComputer() {
        System.out.println("Brand = " + brand);
        System.out.println("Processor Type = " + processorType);
        System.out.println("Memory Size = " + memorySize);
        System.out.println("Processor Speed = " + processorSpeed);
    }
}
```

Computer.java | Laptop.java | Pc.java | Mac.java | Windows.java | Main.java

src > main > java > com > azhar > asg > Laptop.java > Laptop > printDataLaptop()

```java
package com.azhar.asg;

public class Laptop extends Computer {
    public String batteryType;

    public Laptop() {

    }

    public Laptop(String brand, String processorType, int memorySize, int processorSpeed, String batteryType) {
        super(brand, processorType, memorySize, processorSpeed);
        this.batteryType = batteryType;
    }

    Codeium: Refactor | Explain | Generate Javadoc
    public void printDataLaptop() {
        System.out.println("==========Data Laptop==========");
        super.printDataComputer();
        System.out.println("Battery Type = " + batteryType);
    }
}
```

Computer.java | Laptop.java | Pc.java | Mac.java | Windows.java | Main.java

src > main > java > com > azhar > asg > Pc.java > Pc > printDataPc()

```java
package com.azhar.asg;

public class Pc extends Computer {
    public int monitorSize;

    public Pc() {

    }

    public Pc(String brand, String processorType, int memorySize, int processorSpeed, int monitorSize) {
        super(brand, processorType, memorySize, processorSpeed);
        this.monitorSize = monitorSize;
    }

    Codeium: Refactor | Explain | Generate Javadoc
    public void printDataPc() {
        System.out.println("==========Data Pc==========");
        super.printDataComputer();
        System.out.println("Monitor Size = " + monitorSize);
    }
}
```

Computer.java | Laptop.java | Pc.java | Mac.java | Windows.java | Main.java

src > main > java > com > azhar > asg > Mac.java > Mac > printDataMac()

```java
package com.azhar.asg;

public class Mac extends Laptop {
    public String security;

    public Mac() {

    }

    public Mac(String brand, String processorType, int memorySize, int processorSpeed, String batteryType , String security) {
        super(brand, processorType, memorySize, processorSpeed, batteryType);
        this.security = security;
    }

    Codeium: Refactor | Explain | Generate Javadoc
    public void printDataMac() {
        System.out.println("==========Data Mac==========");
        super.printDataLaptop();
        System.out.println("Security = " + security);
    }
}
```

Computer.java | Laptop.java | Pc.java | Mac.java | Windows.java | Main.java

src > main > java > com > azhar > asg > Windows.java > Windows > printDataWindows()

```java
package com.azhar.asg;

public class Windows extends Laptop {
    public String feature;

    public Windows() {

    }

    public Windows(String brand, String processorType, int memorySize, int processorSpeed, String batteryType , String feature) {
        super(brand, processorType, memorySize, processorSpeed, batteryType);
        this.feature = feature;
    }

    Codeium: Refactor | Explain | Generate Javadoc
    public void printDataWindows() {
        System.out.println("==========Data Windows==========");
        super.printDataLaptop();
        System.out.println("Feature = " + feature);
    }
}
```

Computer.java | Laptop.java | Pc.java | Mac.java | Windows.java | Main.java

src > main > java > com > azhar > asg > Main.java > Main > main(String[])

```java
package com.azhar.asg;

public class Main {
    Run | Debug | Codeium: Refactor | Explain | Generate Javadoc
    public static void main(String[] args) {
        Pc pc = new Pc(brand:"Lenovo", processorType:"Intel", memorySize:8, processorSpeed:150, monitorSize:15);
        pc.printDataPc();

        Mac mac = new Mac(brand:"Apple", processorType:"Intel", memorySize:8, processorSpeed:150, batteryType:"Li-ion", security:"Secured");
        mac.printDataMac();

        Windows windows = new Windows(brand:"Dell", processorType:"Intel", memorySize:8, processorSpeed:150, batteryType:"Li-ion",
        feature:"There is dedicated vga card");
        windows.printDataWindows();
    }
}
```

Result:

```
➜ zharsuke@box  ~/Documents/College/Semester_3/oop/meet-7/coding git:(master) ✗
ExceptionMessages -cp /home/zharsuke/Documents/College/Semester_3/oop/meet-7/cod
Picked up _JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=on -Dswing.aatext=true
==========Data Pc==========
Brand = Lenovo
Processor Type = Intel
Memory Size = 8
Processor Speed = 150
Monitor Size = 15
==========Data Mac==========
==========Data Laptop==========
Brand = Apple
Processor Type = Intel
Memory Size = 8
Processor Speed = 150
Battery Type = Li-ion
Security = Secured
==========Data Windows==========
==========Data Laptop==========
Brand = Dell
Processor Type = Intel
Memory Size = 8
Processor Speed = 150
Battery Type = Li-ion
Feature = There is dedicated vga card
➜ zharsuke@box  ~/Documents/College/Semester_3/oop/meet-7/coding git:(master) ✗
```