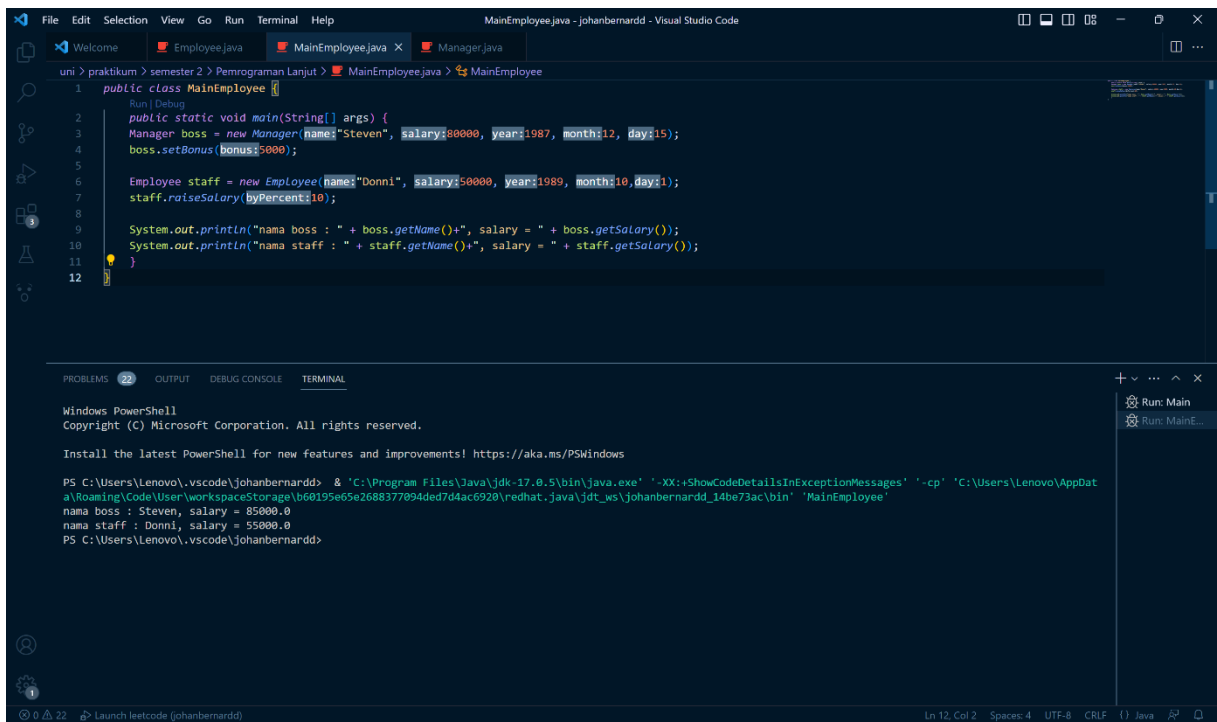


NAMA : Johanes Paulus Bernard Purek
NIM : 225150407111090
KELAS : B
BAB : 5 - Inheritance
ASISTEN : Fahru Setiawan Iskandar dan Adin Rama Ariyanto Putra

1. Data dan Analisis hasil percobaan

1. Jalankan code program diatas dan benahi jika menemukan kesalahan!



```
1 public class MainEmployee {
2     public static void main(String[] args) {
3         Manager boss = new Manager(name:"Steven", salary:80000, year:1987, month:12, day:15);
4         boss.setBonus(bonus:5000);
5
6         Employee staff = new Employee(name:"Donni", salary:50000, year:1989, month:10, day:1);
7         staff.raiseSalary(byPercent:10);
8
9         System.out.println("nama boss : " + boss.getName() + ", salary = " + boss.getSalary());
10        System.out.println("nama staff : " + staff.getName() + ", salary = " + staff.getSalary());
11    }
12 }
```

Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! <https://aka.ms/PSWindows>

PS C:\Users\Lenovo\.vscode\johanbernardd> & 'C:\Program Files\Java\jdk-17.0.5\bin\java.exe' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\Lenovo\AppData\Roaming\Code\User\workspaceStorage\b60195e05e2688377094ded7d4ac6920\redhat.java\jdt_ws\johanbernardd_14be73ac\bin\' 'MainEmployee'

nama boss : Steven, salary = 85000.0
nama staff : Donni, salary = 55000.0

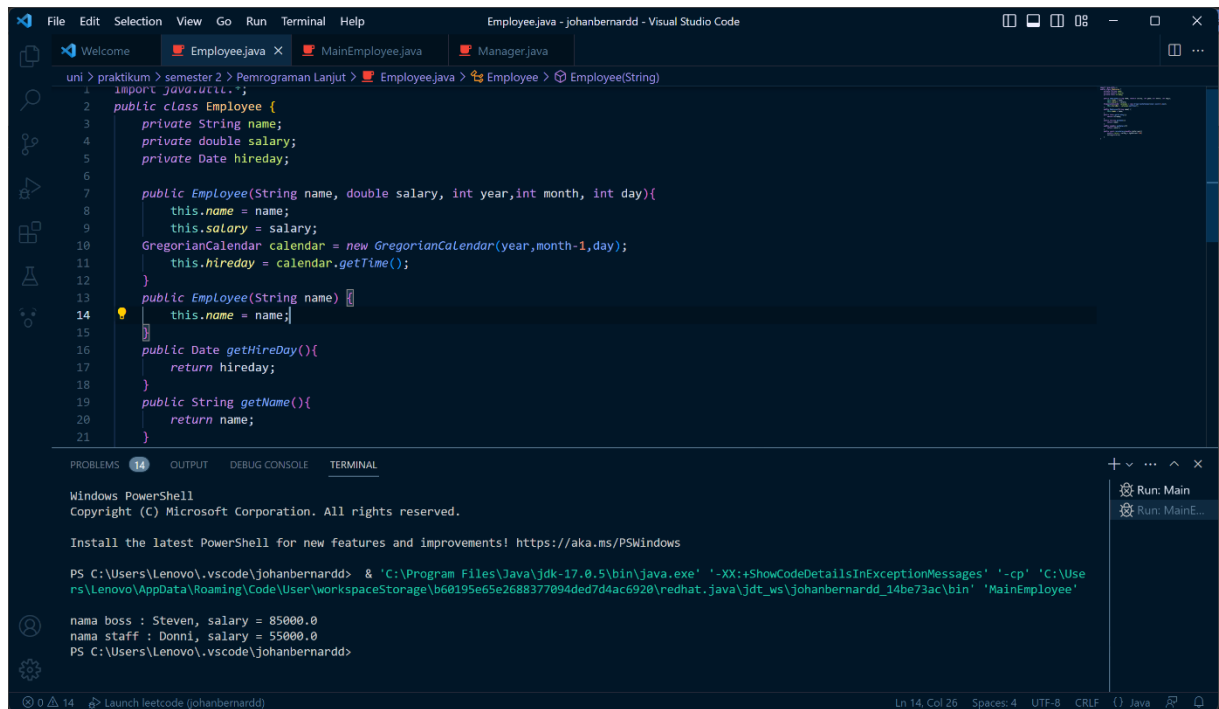
PS C:\Users\Lenovo\.vscode\johanbernardd>

2. Bagaimana cara konstruktor pada subclass memanggil konstruktor di superclass nya?

Apakah hal itu perlu dilakukan? Sertakan alasan anda !

- Jika suatu konstruktor pada subclass memanggil konstruktor di superclass nya tentu dapat dilakukan. Caranya adalah dengan membuat konstruktor pada subclass sesuai dengan nama classnya kemudian diikuti dengan parameter, yang 100% isi parameter subclassnya sama dengan parameter di superclass. Kemudian, memanggil konstruktor di superclass dengan keyword **super** diikuti dengan variabel parameternya.

3. Tambahkan constructor pada class Employee dengan parameter String name! amati perubahan apa yang terjadi, jelaskan jawaban anda!



```

1  import java.util.*;
2  public class Employee {
3      private String name;
4      private double salary;
5      private Date hireday;
6
7      public Employee(String name, double salary, int year, int month, int day){
8          this.name = name;
9          this.salary = salary;
10         GregorianCalendar calendar = new GregorianCalendar(year, month-1, day);
11         this.hireday = calendar.getTime();
12     }
13     public Employee(String name) {
14         this.name = name;
15     }
16     public Date getHireDay(){
17         return hireday;
18     }
19     public String getName(){
20         return name;
21     }

```

Windows PowerShell
Copyright (c) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! <https://aka.ms/PSWindows>

```

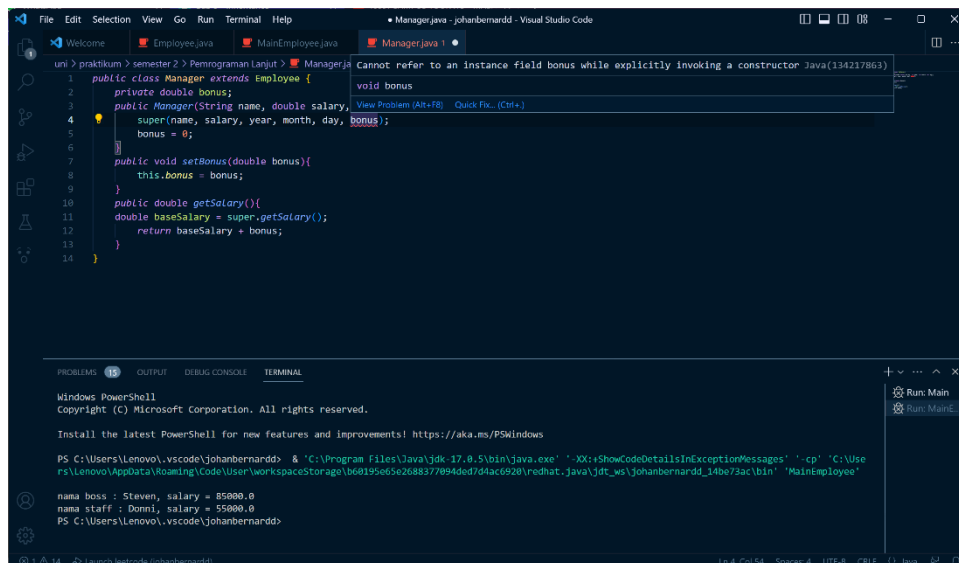
PS C:\Users\Lenovo\.vscode\johanbernardd> & 'C:\Program Files\Java\jdk-17.0.5\bin\java.exe' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\Lenovo\AppData\Roaming\Code\User\workspaceStorage\b68195e65e2688377094ded7d4ac6920\redhat.java\jdt_ws\johanbernardd_14be73ac\bin' 'MainEmployee'

nama boss : Steven, salary = 85000.0
nama staff : Donni, salary = 55000.0
PS C:\Users\Lenovo\.vscode\johanbernardd>

```

- Setelah dijalankan, tidak ada perubahan yang terjadi. Hal ini terjadi karena pada sebuah class, konstruktor yang ada boleh lebih dari 1 dimana hal ini kita katakan sebagai **overloading konstruktor**. Namun, konstruktor tersebut harus berbeda dengan konstruktor lainnya dimana perbedaannya dapat dilihat pada parameter setiap konstruktor dan tipe data dari parameter itu sendiri yang tidak boleh sama setiap konstruktornya.

4. Pada Class Manager baris ke 5, setelah variable day tambahkan variable bonus! Amati apa yang terjadi dan mengapa demikian?



```

1  public class Manager extends Employee {
2      private double bonus;
3      public Manager(String name, double salary,
4          super(name, salary, year, month, day, bonus);
5      bonus = 0;
6  }
7  public void setBonus(double bonus){
8      this.bonus = bonus;
9  }
10 public double getSalary(){
11     double baseSalary = super.getSalary();
12     return baseSalary + bonus;
13 }
14 }

```

Cannot refer to an instance field bonus while explicitly invoking a constructor Java(134217863)

Windows PowerShell
Copyright (c) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! <https://aka.ms/PSWindows>

```

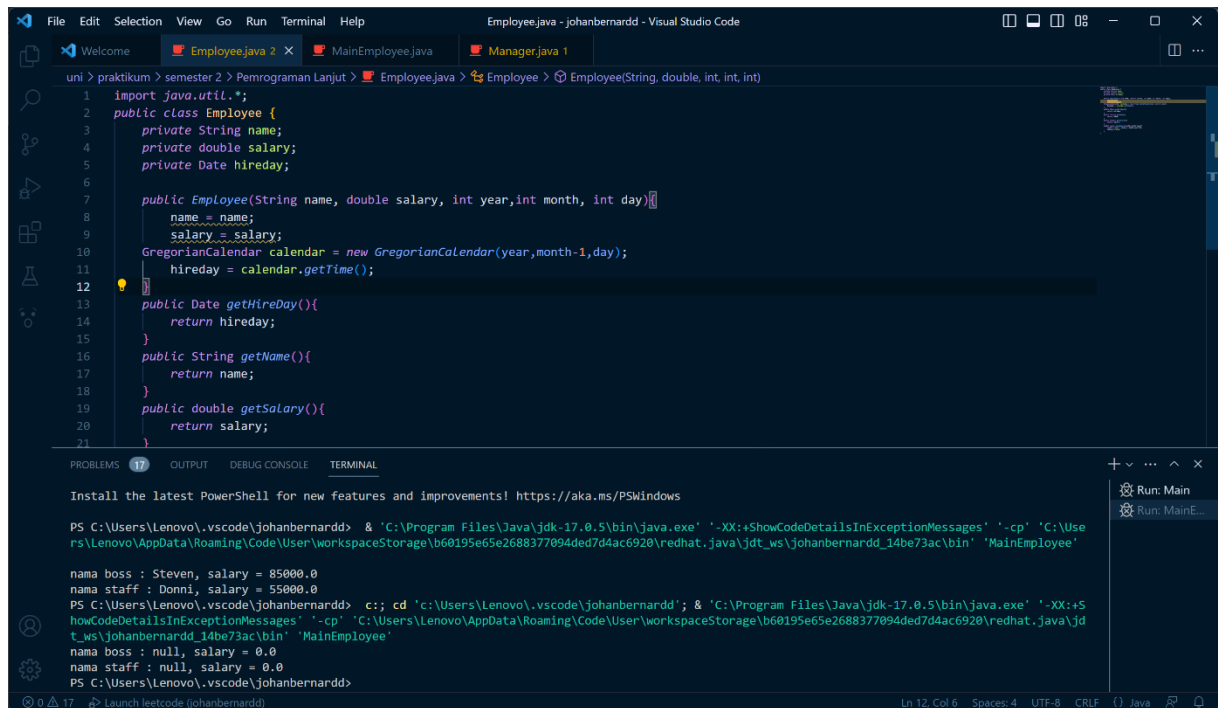
PS C:\Users\Lenovo\.vscode\johanbernardd> & 'C:\Program Files\Java\jdk-17.0.5\bin\java.exe' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\Lenovo\AppData\Roaming\Code\User\workspaceStorage\b68195e65e2688377094ded7d4ac6920\redhat.java\jdt_ws\johanbernardd_14be73ac\bin' 'MainEmployee'

nama boss : Steven, salary = 85000.0
nama staff : Donni, salary = 55000.0
PS C:\Users\Lenovo\.vscode\johanbernardd>

```

- Yang terjadi adalah java memberikan informasi bahwa kode tersebut mengalami masalah dikarenakan pada konstruktornya tidak terdapat parameter untuk tipe data dan variabel dari bonus itu sendiri. Terlebih konstruktor tersebut merupakan inheritance dari super-class Employee yang juga tidak memiliki konstruktor yang berparameterkan variabel bonus.

5. Untuk apa digunakan keyword this pada class manager dan employee? Hapus keyword this dan amati apa yang terjadi?



```

1  import java.util.*;
2  public class Employee {
3      private String name;
4      private double salary;
5      private Date hireday;
6
7      public Employee(String name, double salary, int year, int month, int day) {
8          name = name;
9          salary = salary;
10         GregorianCalendar calendar = new GregorianCalendar(year, month-1, day);
11         hireday = calendar.getTime();
12     }
13     public Date getHireDay() {
14         return hireday;
15     }
16     public String getName() {
17         return name;
18     }
19     public double getSalary() {
20         return salary;
21     }
22 }
  
```

```

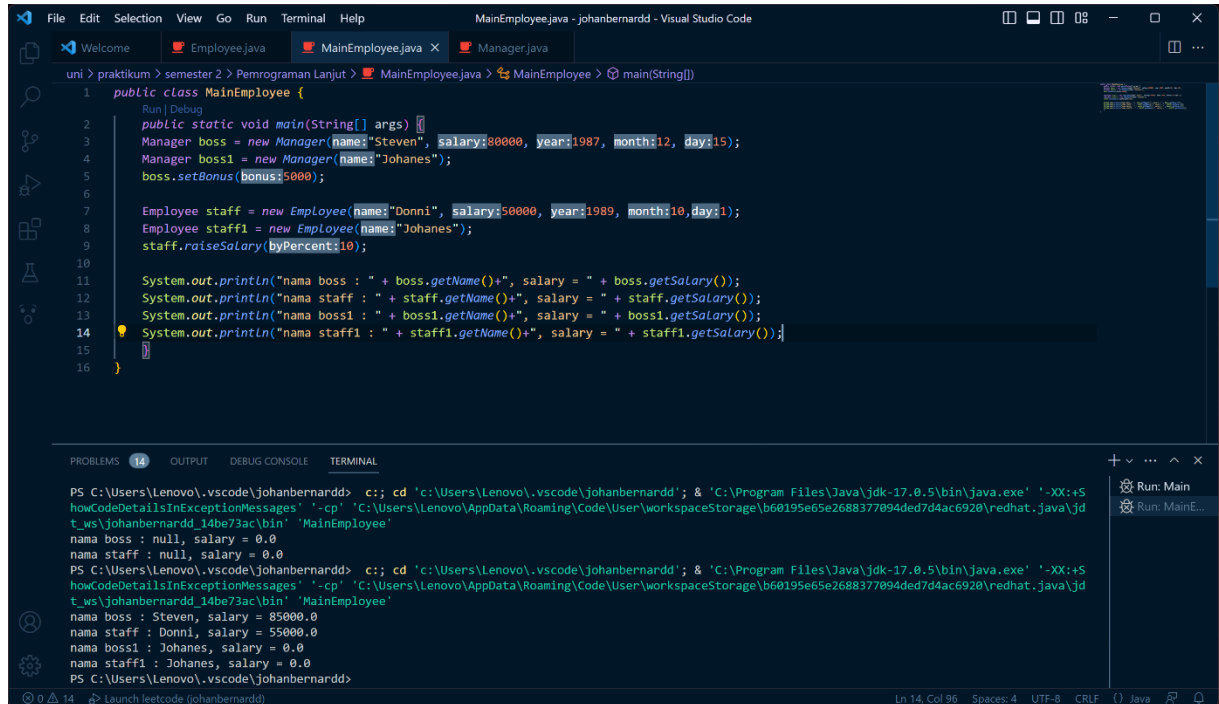
PS C:\Users\Lenovo\.vscode\johanbernardd> & 'C:\Program Files\Java\jdk-17.0.5\bin\java.exe' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\Lenovo\AppData\Roaming\Code\User\workspaceStorage\b60195e65e2688377094ded7d4ac6920\redhat.java\jdt_ws\johanbernardd_14be73ac\bin' 'MainEmployee'

nama boss : Steven, salary = 85000.0
nama staff : Donni, salary = 55000.0
PS C:\Users\Lenovo\.vscode\johanbernardd> c:; cd 'c:\Users\Lenovo\.vscode\johanbernardd'; & 'C:\Program Files\Java\jdk-17.0.5\bin\java.exe' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\Lenovo\AppData\Roaming\Code\User\workspaceStorage\b60195e65e2688377094ded7d4ac6920\redhat.java\jdt_ws\johanbernardd_14be73ac\bin' 'MainEmployee'
nama boss : null, salary = 0.0
nama staff : null, salary = 0.0
PS C:\Users\Lenovo\.vscode\johanbernardd>
  
```

- Keyword this pada class manager dan employee bertujuan untuk merujuk kepada sebuah variabel dari parameter yang telah dibuat beserta dengan tipe datanya, yang akan menampung **nilai dari suatu objek**, dimana nilai ini merupakan nilai yang telah dibuat menjadi objek pada parameter konstruktornya. Yang terjadi jika menghapus keyword this adalah program akan kebingungan karena keduanya memiliki nama variabel yang sama. Contohnya, jika this.name = name diganti menjadi name = name maka tentu akan menimbulkan suatu masalah pada program tersebut karena bingung untuk menentukan bagian mana yang berperan sebagai variabel penampung nilai objek yang telah didefinisikan sebagai parameter pada konstruktor, sehingga ketika program dijalankan menghasilkan output kosong(null / 0) pada name dan salary-nya.

6. Tambahkan constructor pada class Employee dengan parameter Bertipe data string

bernama name yang nantinya bila constructor ini akan dipanggil akan menginisialisasi variable name! Amati perubahannya pada class anak dan jelaskan! Benahi bila terjadi kesalahan!



```

1 public class MainEmployee {
2     public static void main(String[] args) {
3         Manager boss = new Manager(name:"Steven", salary:80000, year:1987, month:12, day:15);
4         Manager boss1 = new Manager(name:"Johanes");
5         boss.setBonus(bonus:5000);
6
7         Employee staff = new Employee(name:"Donni", salary:50000, year:1989, month:10, day:1);
8         Employee staff1 = new Employee(name:"Johanes");
9         staff.raiseSalary(byPercent:10);
10
11         System.out.println("nama boss : " + boss.getName() + ", salary = " + boss.getSalary());
12         System.out.println("nama staff : " + staff.getName() + ", salary = " + staff.getSalary());
13         System.out.println("nama boss1 : " + boss1.getName() + ", salary = " + boss1.getSalary());
14         System.out.println("nama staff1 : " + staff1.getName() + ", salary = " + staff1.getSalary());
15     }
16 }
  
```

```

PS C:\Users\Lenovo\.vscode\johanbernardd> c:\cd 'c:\Users\Lenovo\.vscode\johanbernardd'; & 'C:\Program Files\Java\jdk-17.0.5\bin\java.exe' '-XX:+S
howCodeDetailsInExceptionMessages' '-cp' 'C:\Users\Lenovo\AppData\Roaming\Code\User\workspaceStorage\b60195e65e2688377094ded7d4ac6920\redhat.java\jd
t_ws\johanbernardd_14be73ac\bin' 'MainEmployee'
nama boss : null, salary = 0.0
nama staff : null, salary = 0.0
PS C:\Users\Lenovo\.vscode\johanbernardd> c:\cd 'c:\Users\Lenovo\.vscode\johanbernardd'; & 'C:\Program Files\Java\jdk-17.0.5\bin\java.exe' '-XX:+S
howCodeDetailsInExceptionMessages' '-cp' 'C:\Users\Lenovo\AppData\Roaming\Code\User\workspaceStorage\b60195e65e2688377094ded7d4ac6920\redhat.java\jd
t_ws\johanbernardd_14be73ac\bin' 'MainEmployee'
nama boss : Steven, salary = 85000.0
nama staff : Donni, salary = 55000.0
nama boss1 : Johanes, salary = 0.0
nama staff1 : Johanes, salary = 0.0
PS C:\Users\Lenovo\.vscode\johanbernardd>
  
```

- Jika ingin membuat konstruktor baru dengan variabel name dan bertipe data String pada class Employee, maka tentu jika ingin menggunakan dan menjalankannya dibuatkan juga konstruktor pada sub-classnya yakni class Manager dengan menggunakan keyword **super**. Ketika ingin menjalankannya pada class MainEmployee, maka tentu harus dibuatkan juga objek baru dari konstruktor baru String name itu tadi. Lalu, program dijalankan dan tentu hanya akan menampilkan output name-nya karena pada konstruktor baru yang dibuat tidak menampung untuk parameter salary.

7. Pada bab sebelumnya anda telah belajar mengenai konsep encapsulation, jelaskan mengapa pada super class menggunakan modifier protected? Apa yang terjadi jika modifier anda ubah menjadi private atau public? Jelaskan !

- Modifier protected yang digunakan pada super class bertujuan untuk menyembunyikan informasi kode agar tidak diketahui oleh orang lain. Penyembunyian informasi ini berbeda penerapannya jika menggunakan modifier private dimana apabila super class ini tidak memiliki sub class maka modifier private lebih tepat untuk digunakan karena hanya berlaku untuk kelas tersebut saja.

Namun, jika menggunakan modifier protected maka sub class yang dimiliki oleh super class dapat mengakses seluruh informasi pada super class tersebut. Modifier protected ini tepat digunakan jika ingin kelas lain yang berada dalam package yang sama dan sub class (child)-nya untuk dapat mengakses informasinya.

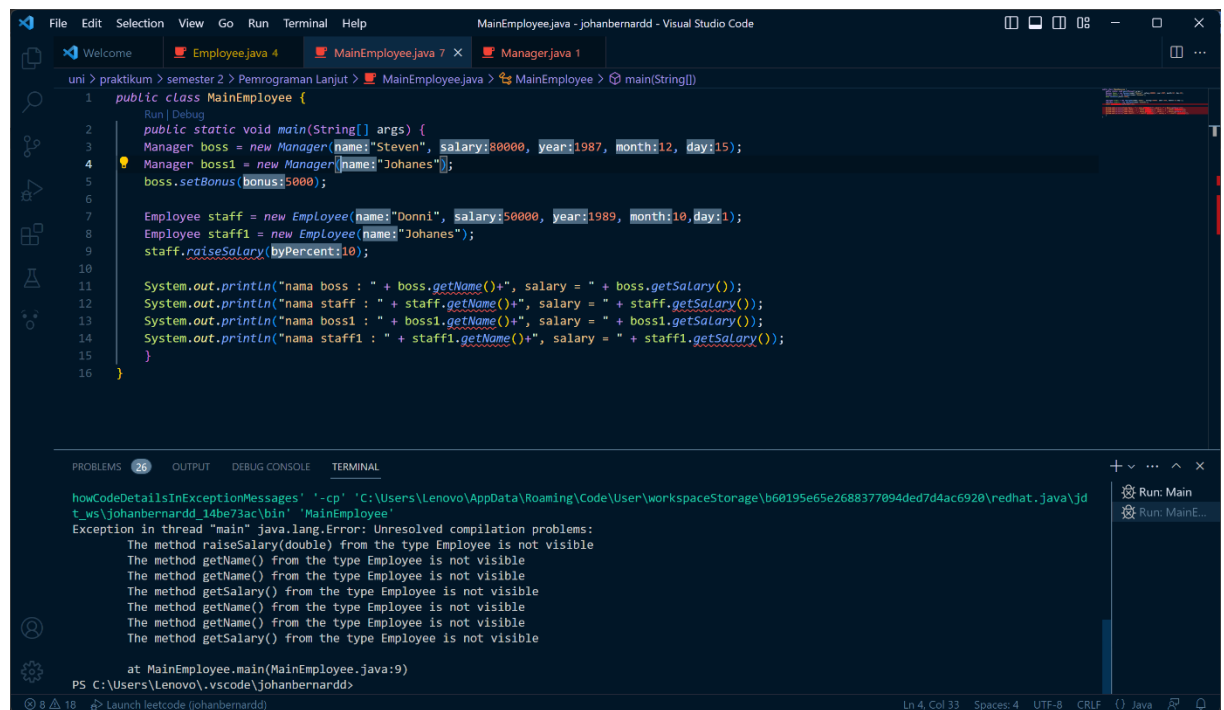
8. Ubahlah akses modifier method pada kelas employee menjadi :

a. Private

b. Protected

Amati perubahan apa yang terjadi? Jelaskan jawaban anda dengan detail!

a. Private



```
1 public class MainEmployee {
2     public static void main(String[] args) {
3         Manager boss = new Manager(name:"Steven", salary:80000, year:1987, month:12, day:15);
4         Manager boss1 = new Manager(name:"Johanes");
5         boss.setBonus(bonus:5000);
6
7         Employee staff = new Employee(name:"Donni", salary:50000, year:1989, month:10, day:1);
8         Employee staff1 = new Employee(name:"Johanes");
9         staff.raiseSalary(byPercent:10);
10
11         System.out.println("nama boss : " + boss.getName() + ", salary = " + boss.getSalary());
12         System.out.println("nama staff : " + staff.getName() + ", salary = " + staff.getSalary());
13         System.out.println("nama boss1 : " + boss1.getName() + ", salary = " + boss1.getSalary());
14         System.out.println("nama staff1 : " + staff1.getName() + ", salary = " + staff1.getSalary());
15     }
16 }
```

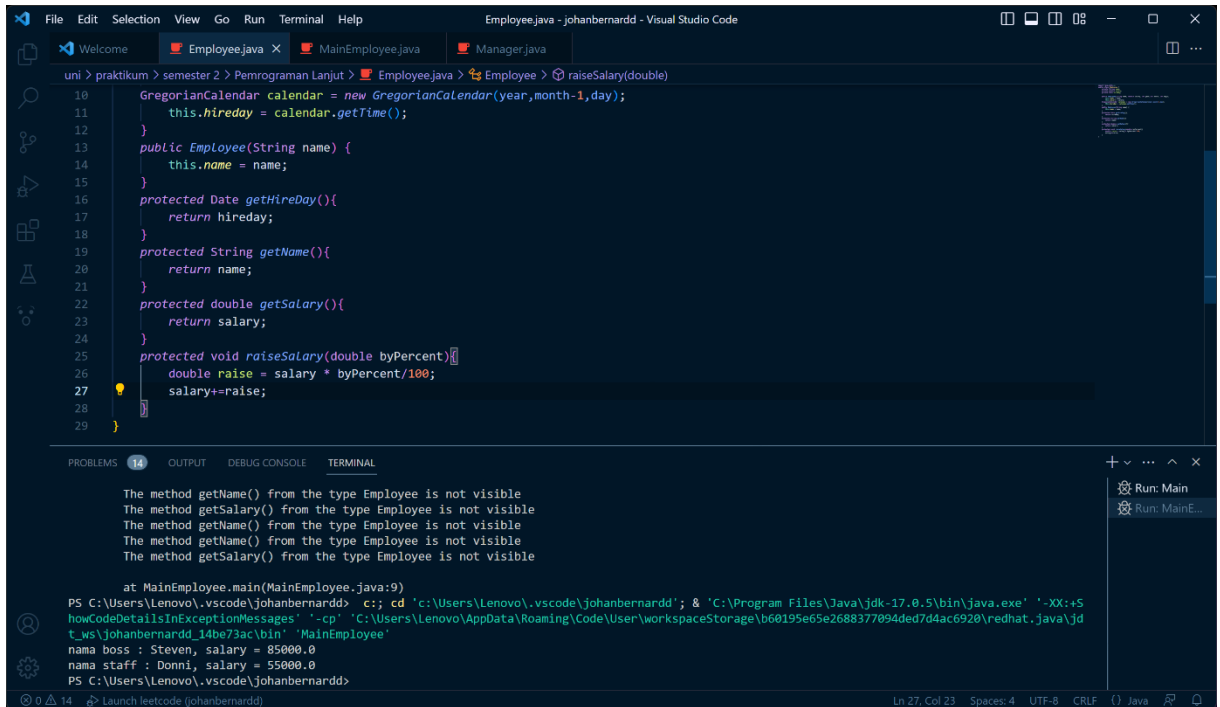
Exception in thread "main" java.lang.Error: Unresolved compilation problems:

- The method raiseSalary(double) from the type Employee is not visible
- The method getName() from the type Employee is not visible
- The method getName() from the type Employee is not visible
- The method getSalary() from the type Employee is not visible
- The method getName() from the type Employee is not visible
- The method getName() from the type Employee is not visible
- The method getSalary() from the type Employee is not visible

at MainEmployee.main(MainEmployee.java:9)
PS C:\Users\Lenovo\.vscode\johanbernardd>

- Yang terjadi adalah kode program pada class utama MainEmployee mengalami masalah karena tidak dapat dijalankan dan menghasilkan output yang ditampilkan seperti diatas. Ini disebabkan karena method-method yang diganti access modifiernya menjadi private pada class Employee dimana hal ini berakibat pada pengaksesan dari method-method tersebut pada class lain karena prinsip dasar dari access modifier private adalah isi dari class tersebut hanya dapat digunakan pada class tersebut dan tidak dapat digunakan ataupun diakses oleh class yang lain.

b. Protected



```

10  GregorianCalendar calendar = new GregorianCalendar(year, month-1, day);
11  this.hireday = calendar.getTime();
12  }
13  public Employee(String name) {
14  this.name = name;
15  }
16  protected Date getHireDay(){
17  return hireday;
18  }
19  protected String getName(){
20  return name;
21  }
22  protected double getSalary(){
23  return salary;
24  }
25  protected void raiseSalary(double byPercent){
26  double raise = salary * byPercent/100;
27  salary+=raise;
28  }
29  }

```

```

at MainEmployee.main(MainEmployee.java:9)
PS C:\Users\Lenovo\.vscode\johanbernardd> c:\cd 'c:\Users\Lenovo\.vscode\johanbernardd'; & 'C:\Program Files\Java\jdk-17.0.5\bin\java.exe' '-XX:+$
howCodeDetailsInExceptionMessages' '-cp' 'C:\Users\Lenovo\AppData\Roaming\Code\User\workspaceStorage\b60195e65e2688377094ded7d4ac6920\redhat.java\jd
t_ws\johanbernardd_14be73ac\bin' 'MainEmployee'
nama boss : Steven, salary = 85000.0
nama staff : Donni, salary = 55000.0
PS C:\Users\Lenovo\.vscode\johanbernardd>

```

- Yang terjadi adalah program dapat dijalankan dengan baik. Berbeda dengan ketika menggunakan access modifier private sebelumnya, method-method yang ada pada class Employee diganti dengan modifier protected yang tentu prinsip dasar dari penggunaan protected sendiri dapat digunakan pada sub-classnya (turunan class) dan juga dapat digunakan jika berada dalam package yang sama.

2. Tugas Praktikum

2.1 Source code

```

3  package TugasInheritance;
4
5  public class Manusia {
6      private String nama;
7      private boolean jenisKelamin;
8      private String nik;
9      private boolean menikah;
10
11      public Manusia(String nama, boolean jenisKelamin, String
    nik, boolean menikah) {
12          this.nama = nama;
13          this.jenisKelamin = jenisKelamin;
14          this.nik = nik;
15          this.menikah = menikah;

```

```
16     }
17     public String getNama() {
18         return nama;
19     }
20     public void setNama(String nama) {
21         this.nama = nama;
22     }
23     public boolean getJenisKelamin() {
24         return jenisKelamin;
25     }
26     public void setJenisKelamin(boolean jenisKelamin) {
27         this.jenisKelamin = jenisKelamin;
28     }
29     public String getNik() {
30         return nik;
31     }
32     public void setNik(String nik) {
33         this.nik = nik;
34     }
35     public boolean getMenikah() {
36         return menikah;
37     }
38     public void setMenikah(boolean menikah) {
39         this.menikah = menikah;
40     }
41     public double getTunjangan() {
42         if(this.menikah) {
43             if(this.jenisKelamin) {
44                 return 25;
45             } else {
46                 return 20;
47             }
48         } else {
49             return 15;
50         }
51     }
52     public double getPendapatan() {
53         return getTunjangan();
54     }
55     public String toString() {
56         String jenisKelamin;
57         if(this.jenisKelamin) {
58             jenisKelamin = "Laki-laki";
59         } else {
60             jenisKelamin = "Perempuan";
61         }

```

```
62         return "Nama : " + this.nama + "\nNIK : " + this.nik
+ "\nJenis Kelamin : " + jenisKelamin + "\nJumlah Pendapatan : "
+ getPendapatan() + "\n";
63     }
64 }
```

```
package TugasInheritance;

public class MahasiswaFILKOM extends Manusia {
    private String nim;
    private double ipk;
    public MahasiswaFILKOM(String nama, boolean jenisKelamin,
String nik, boolean menikah, String nim, double ipk) {
        super(nama, jenisKelamin, nik, menikah);
        this.nim = nim;
        this.ipk = ipk;
    }
    public String getNim() {
        return nim;
    }
    public void setNim(String nim) {
        this.nim = nim;
    }
    public double getIpk() {
        return ipk;
    }
    public void setIpk(double ipk) {
        this.ipk = ipk;
    }
    public double getBeasiswa() {
        double beasiswa = 0;
        if (ipk > 3.0 && ipk <= 3.5) {
            beasiswa = 50;
        } else if (ipk > 3.5 && ipk <= 4.0) {
            beasiswa = 75;
        } else {
            beasiswa = 0;
        }
        return beasiswa;
    }
    public String getStatus() {
        String status = "";
        String angkatan = "";
        String prodi = "";
        String digitAngkatan = nim.substring(0, 2);
        String digitProdi = nim.substring(6, 7);
    }
}
```



```
switch (digitProdi) {
    case "2":
        prodi = "Teknik Informatika";
        break;
    case "3":
        prodi = "Teknik Komputer";
        break;
    case "4":
        prodi = "Sistem Informasi";
        break;
    case "6":
        prodi = "Pendidikan Teknologi Informasi";
        break;
    case "7":
        prodi = "Teknologi Informasi";
        break;
    default: prodi = "Prodi tidak diketahui";
}

angkatan = "20" + digitAngkatan;
status = prodi + " " + angkatan;

return status;
}

@Override
public String toString() {
    return super.toString() + "NIM : " + nim + "\nIPK : " +
ipk + "\nStatus : " + getStatus() + "\n";
}
}

package TugasInheritance;
import java.time.LocalDate;

public class Pekerja extends Manusia {
    private double gaji;
    private LocalDate tahunMasuk;
    private int jumlahAnak;
    public Pekerja(String nama, boolean jenisKelamin, String nik,
boolean menikah, double gaji, LocalDate tahunMasuk, int jumlahAnak)
{
    super(nama, jenisKelamin, nik, menikah);
    this.gaji = gaji;
    this.tahunMasuk = tahunMasuk;
    this.jumlahAnak = jumlahAnak;
}
```

```
}  
public double getGaji() {  
    return gaji;  
}  
public void setGaji(double gaji) {  
    this.gaji = gaji;  
}  
public LocalDate getTahunMasuk() {  
    return tahunMasuk;  
}  
public void setTahunMasuk(LocalDate tahunMasuk) {  
    this.tahunMasuk = tahunMasuk;  
}  
public int getJumlahAnak() {  
    return jumlahAnak;  
}  
public void setJumlahAnak(int jumlahAnak) {  
    this.jumlahAnak = jumlahAnak;  
}  
public double getBonus() {  
    LocalDate now = LocalDate.now();  
    int lamaBekerja = now.getYear() - tahunMasuk.getYear();  
    if (lamaBekerja <= 5) {  
        return 0.05 * gaji;  
    } else if (lamaBekerja > 5 && lamaBekerja <= 10) {  
        return 0.1 * gaji;  
    } else {  
        return 0.15 * gaji;  
    }  
}  
public double getTunjanganAnak() {  
    return jumlahAnak * 20;  
}  
public double getGajiTotal() {  
    return gaji + getBonus() + getTunjanganAnak();  
}  
@Override  
public String toString() {  
    return super.toString() + "Tahun Masuk : " + tahunMasuk +  
"\nJumlah Anak : " +  
    jumlahAnak + "\nGaji : " + getGajiTotal() + "\n";  
}  
}  
  
package TugasInheritance;  
import java.time.LocalDate;
```

```
public class Manager1 extends Pekerja {
    private String departemen;
    public Manager1(String nama, boolean jenisKelamin, String nik,
boolean menikah, double gaji, LocalDate tahunMasuk, int jumlahAnak,
String departemen) {
        super(nama, jenisKelamin, nik, menikah, gaji, tahunMasuk,
jumlahAnak);
        this.departemen = departemen;
    }
    public String getDepartemen() {
        return departemen;
    }
    public void setDepartemen(String departemen) {
        this.departemen = departemen;
    }
    public double getTunjanganTotal() {
        return 0.1 * getGajiTotal();
    }
    @Override
    public String toString() {
        return super.toString() + "Departemen : " + getDepartemen()
+ "\n";
    }
}
```

```
package TugasInheritance;
import java.time.LocalDate;

public class TestCaseTugasInheritance {
    public static void main(String[] args) {
        //Manusia a)Laki-Laki telah menikah
        Manusia manusia1 = new Manusia("Cristiano", true,
"500100010119700001", true);
        System.out.println(manusia1.toString());
        //Manusia b)Perempuan telah menikah
        Manusia manusia2 = new Manusia("Jessica", false,
"50010102021970", true);
        System.out.println(manusia2.toString());
        //Manusia c)Belum menikah
        Manusia manusia3 = new Manusia("Georgina", false,
"50110003031970", false);
        System.out.println(manusia3.toString());

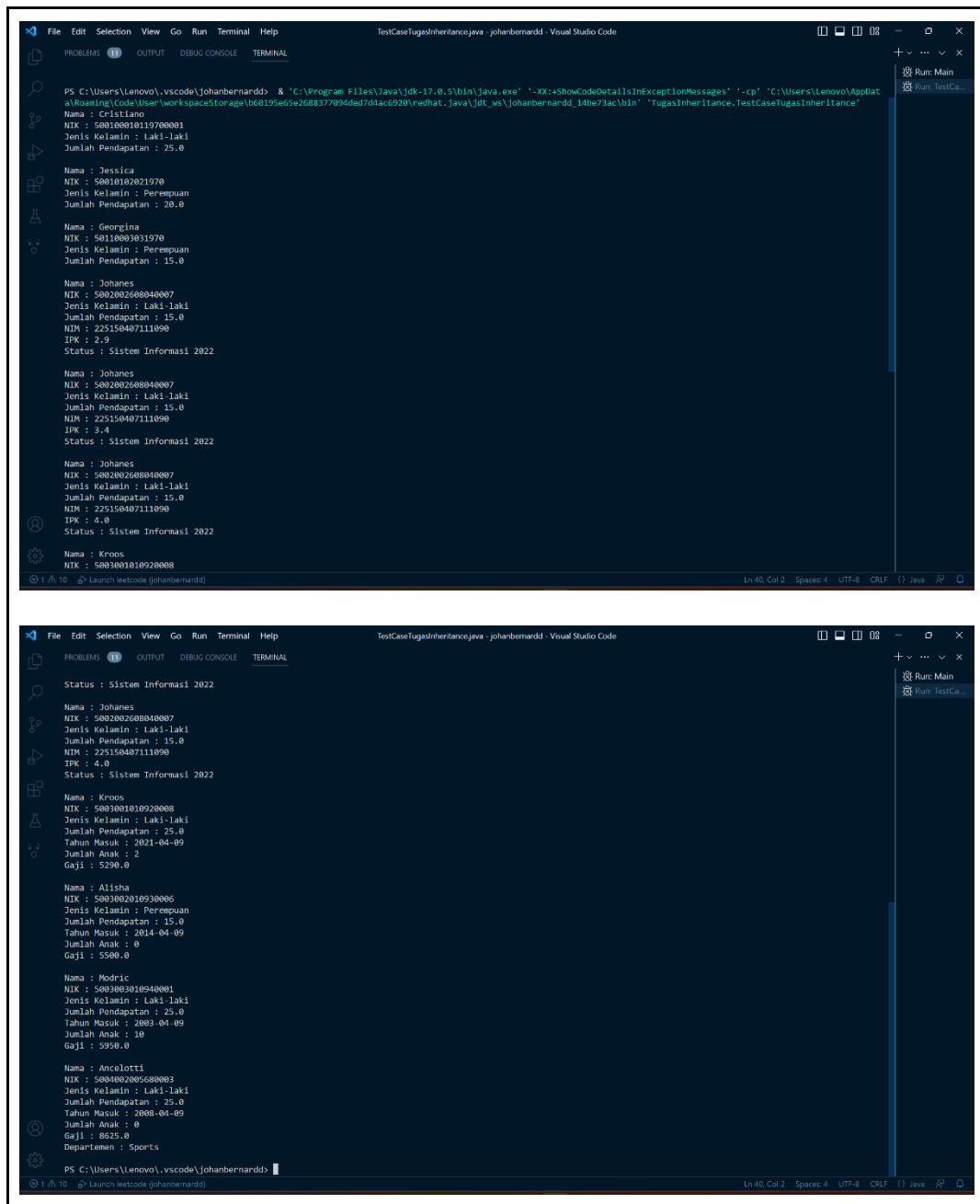
        //MahasiswaFILKOM a)Ipk < 3
    }
}
```

```
MahasiswaFILKOM mahasiswa1 = new MahasiswaFILKOM("Johanes",
true, "5002002608040007", false, "225150407111090", 2.9);
System.out.println(mahasiswa1.toString());
//MahasiswaFILKOM b)Ipk 3 - 3.5
MahasiswaFILKOM mahasiswa2 = new MahasiswaFILKOM("Johanes",
true, "5002002608040007", false, "225150407111090", 3.4);
System.out.println(mahasiswa2.toString());
//MahasiswaFILKOM c)Ipk 3.5 - 4
MahasiswaFILKOM mahasiswa3 = new MahasiswaFILKOM("Johanes",
true, "5002002608040007", false, "225150407111090", 4.0);
System.out.println(mahasiswa3.toString());

//Pekerja a)Lama bekerja 2 tahun, anak 2
Pekerja pekerja1 = new Pekerja("Kroos", true,
"5003001010920008", true, 5000, LocalDate.of(2021, 4, 9), 2);
System.out.println(pekerja1.toString());
//Pekerja b)Lama bekerja 9 tahun
Pekerja pekerja2 = new Pekerja("Alisha", false,
"5003002010930006", false, 5000, LocalDate.of(2014, 4, 9), 0);
System.out.println(pekerja2.toString());
//Pekerja c)Lama bekerja 20 tahun, anak 10
Pekerja pekerja3 = new Pekerja("Modric", true,
"5003003010940001", true, 5000, LocalDate.of(2003, 4, 9), 10);
System.out.println(pekerja3.toString());

//Manager Lama bekerja 15 tahun dengan gaji $7500
Manager1 manager1 = new Manager1("Ancelotti", true,
"5004002005680003", true, 7500, LocalDate.of(2008, 4, 9), 0,
"Sports");
System.out.println(manager1.toString());
}
}
```

2.2 Screenshot hasil



```

PS C:\Users\Lenovo\vscode\johanbernardd> & 'C:\Program Files\Java\jdk-17.0.5\bin\java.exe' "-Xc+ShowCodeDetailsInExceptionMessages" "-cp" 'C:\Users\Lenovo\AppData\Local\Temp\Code\workspaceStorage\b60195e5e2688377894ded7d4ac920\redhat.java\jdt_ws\johanbernardd_14be73ac\bin' 'TugasInheritance.TestCaseTugasInheritance'

Nama : Cristiano
NIK : 50010001010700001
Jenis Kelamin : Laki-laki
Jumlah Pendapatan : 25.0

Nama : Jessica
NIK : 50010102021970
Jenis Kelamin : Perempuan
Jumlah Pendapatan : 20.0

Nama : Georgina
NIK : 50110003031970
Jenis Kelamin : Perempuan
Jumlah Pendapatan : 15.0

Nama : Johannes
NIK : 5002002600040007
Jenis Kelamin : Laki-laki
Jumlah Pendapatan : 15.0
NIM : 225150407111090
IPK : 2.9
Status : Sistem Informasi 2022

Nama : Johannes
NIK : 5002002600040007
Jenis Kelamin : Laki-laki
Jumlah Pendapatan : 15.0
NIM : 225150407111090
IPK : 3.4
Status : Sistem Informasi 2022

Nama : Johannes
NIK : 5002002600040007
Jenis Kelamin : Laki-laki
Jumlah Pendapatan : 15.0
NIM : 225150407111090
IPK : 4.0
Status : Sistem Informasi 2022

Nama : Kroos
NIK : 5003001010920008
Jenis Kelamin : Laki-laki
Jumlah Pendapatan : 25.0
Tahun Masuk : 2021-04-09
Jumlah Anak : 2
Gaji : 5290.0

Nama : Alisha
NIK : 5003002010930006
Jenis Kelamin : Perempuan
Jumlah Pendapatan : 15.0
Tahun Masuk : 2014-04-09
Jumlah Anak : 0
Gaji : 5500.0

Nama : Modric
NIK : 5003003010940001
Jenis Kelamin : Laki-laki
Jumlah Pendapatan : 25.0
Tahun Masuk : 2003-04-09
Jumlah Anak : 10
Gaji : 5950.0

Nama : Ancelotti
NIK : 5004002005600003
Jenis Kelamin : Laki-laki
Jumlah Pendapatan : 25.0
Tahun Masuk : 2008-04-09
Jumlah Anak : 0
Gaji : 8625.0
Departemen : Sports

PS C:\Users\Lenovo\vscode\johanbernardd>

```