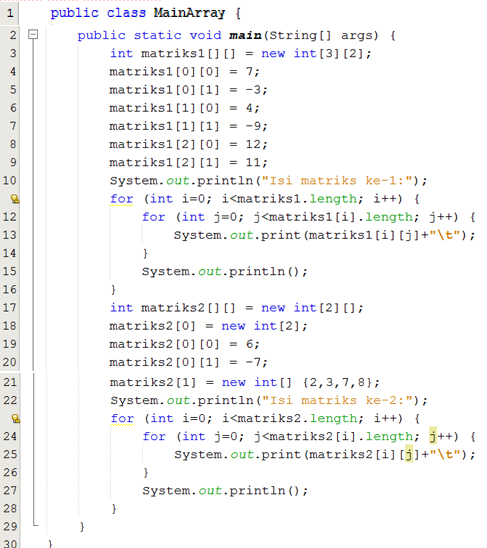
Jurnal Praktikum Pemrograman Berorientasi Obyek Pekan ke-3

Tujuan:

1. Mahasiswa dapat menjelaskan karakteristik array multidimensi di Java.
2. Mahasiswa dapat menerjemahkan isi dari class diagram.
3. Mahasiswa dapat menjelaskan perbedaan relasi agregasi dan komposisi.
4. Mahasiswa mampu mengimplementasikan relasi agregasi di Java.
5. Mahasiswa mampu mengimplementasikan relasi komposisi di Java

Soal berikut akan memperlihatkan bagaimana karakteristik array multidimensi di Java. Buatlah kode program berikut pada komputer Anda. Buatlah file baru dengan nama MainArray.java. Berikut kode programnya.



Jalankan program MainArray.java tersebut! Perhatikan output program yang muncul!

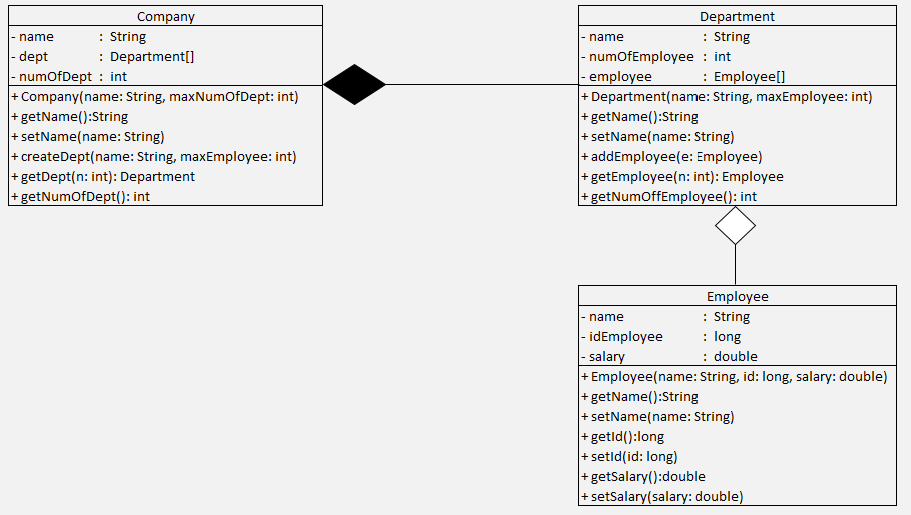
Jelaskan perbedaan isi matriks1 dan matriks2!

Jawab: Perbedaannya terletak pada polanya, jika matriks1 array yang digunakan bersifat konstan atau tetap. Sedangkan pada matriks2 array bersifat tidak tetap disetiap nilai array x nya, sehingga memiliki length atau panjang yang berubah-ubah.

Dari kode program di atas, berikan penjelasan bagaimanakah karakteristik array multidimensi dalam Java!

Jawab: Array pada java bersifat object refrence, yaitu berdasarkan pointer atau memori yang digunakan. Bisa dilihat disitu menggunkaan keyword new yang digunakan untuk membuat sebuah obyek dimana sebuah obyek diperlukan untuk menyewa atau menyimpan memori untuk nilai tersebut.

Perhatikan class diagram berikut ini.



Pada class diagram tersebut terdapat tiga kelas. Berikan penjelasan masing-masing class!

1. Class Company memiliki atribut sebanyak 3 dan method sebanyak 6(satu contructor)

Daftar atribut (sesuaikan jumlah baris dengan jumlah atribut):

|  |  |  |
| --- | --- | --- |
| Nama atribut | Tipe data | Access modifier |
| Name  Dept  numofdept | String  Department  int | Private  Private  private |

Daftar method (sesuaikan jumlah baris dengan jumlah method):

|  |  |  |  |
| --- | --- | --- | --- |
| Nama method | Tipe balikan | Parameter | Access modifier |
| Setname | Void | Name | Public |
| Createdept | Void | Name, maxemlpoyee | Public |
| Getdept | Department | N | Public |
| Getnum | Int | Null | public |
| Company | Void | Name, maxnumofdept | Public |
| Getname | Void | Null | public |

1. Class Department memiliki atribut sebanyak 3 dan method sebanyak 6

Daftar atribut (sesuaikan jumlah baris dengan jumlah atribut):

|  |  |  |
| --- | --- | --- |
| Nama atribut | Tipe data | Access modifier |
| Name | String | Private |
| Numofemployee | Int | Private |
| Employee | Employee | private |

Daftar method (sesuaikan jumlah baris dengan jumlah method):

|  |  |  |  |
| --- | --- | --- | --- |
| Nama method | Tipe balikan | Parameter | Access modifier |
| deparment | Void | Name, maxempyee | Public |
| Getname | String | Null | Public |
| Setname | Void | Name | Public |
| Addemployee | Void | E | Public |
| Getemployee | Employee | N | Public |
| Getnumofemployee | Int | Null | public |

1. Class Employee memiliki atribut sebanyak 3 dan method sebanyak 7

Daftar atribut (sesuaikan jumlah baris dengan jumlah atribut):

|  |  |  |
| --- | --- | --- |
| Nama atribut | Tipe data | Access modifier |
| Name | String | Private |
| Idemployee | Long | Private |
| Salary | Double | private |

Daftar method (sesuaikan jumlah baris dengan jumlah method):

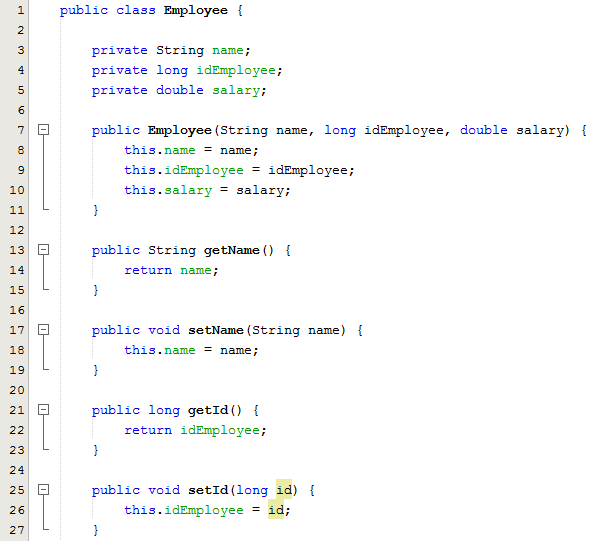
|  |  |  |  |
| --- | --- | --- | --- |
| Nama method | Tipe balikan | Parameter | Access modifier |
| Employee | Void | Name,id, salary | Public |
| Getname | String | Null | Public |
| Setname | Void | Name | Public |
| Getid | Long | Null | Public |
| Setid | Void | Id | Public |
| Getsalary | Double | Null | Public |
| Setsalary | Void | Salary | public |

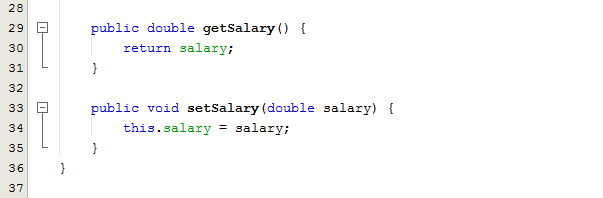
Relasi yang terbentuk antara class Company dan class Department adalah relasi komposisi

Relasi yang terbentuk antara class Department dan class Employee adalah relasi agregasi

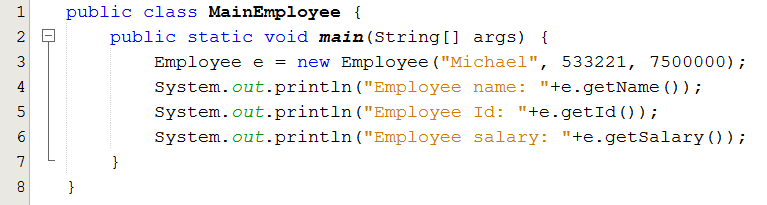
**Implementasi class Employee**

Buatlah sebuah file class baru bernama Employee.java. Berikut kode program untuk class Employee.





Buat file baru dengan nama MainEmployee.java. Berikut kode programnya.



Jalankan file MainEmployee.java. Apa output yang muncul?

Jawab:

run:

**Employee name: Michael**

**Employee name: 533221**

**Employee name: 7500000.0**

BUILD SUCCESSFUL (total time: 0 seconds)

Baris ke-3 pada program di atas melakukan pemanggilan constructor Employee. Jelaskan apa yang terjadi pada saat constructor Employee pada baris ke-3 dipanggil!

Jawab: saat contructor dipanggil melakukan set attribute untuk class employee yaitu 3 atribut tersebut yaitu name, idemployee, dan salary;

Lakukan overloading terhadap constructor Employee. Tambahkan kode berikut ke dalam class Employee.

public Employee(String name, long id) {

this.name = name;

idEmployee = id;

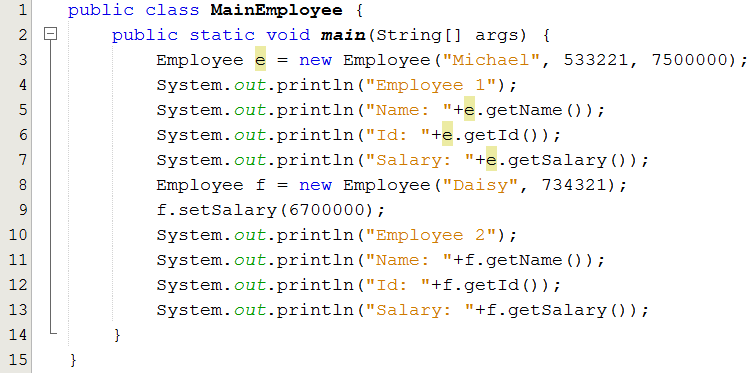
}

public Employee(String name) {

this.name = name;

}

Kemudian modifikasi class MainEmployee.java sebagai berikut.



Kembali jalankan program MainEmployee. Tuliskan output yang muncul!

Jawab:

run:

**Employee 1**

**Employee name: Michael**

**Employee name: 533221**

**Employee name: 7500000.0**

**Employee 2**

**name: Daisy**

**Id: 734321**

**Salary: 6.7E7**

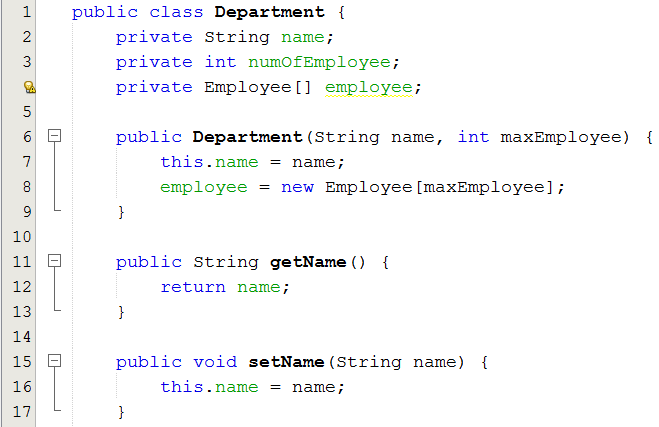
BUILD SUCCESSFUL (total time: 0 seconds)

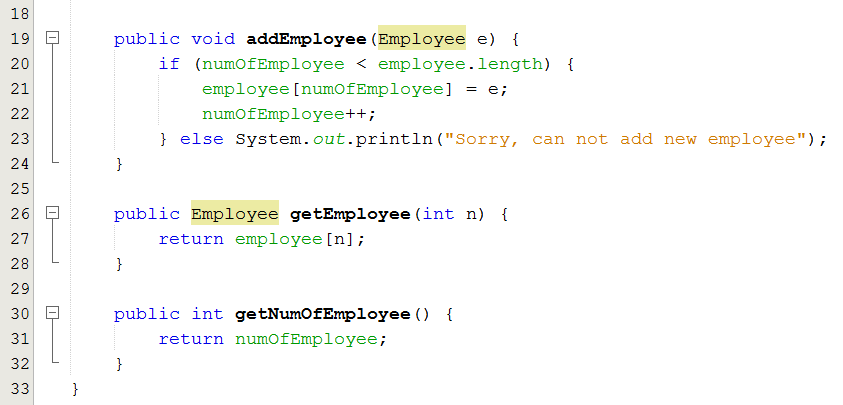
Dari percobaan di atas, apa yang dapat Anda simpulkan tentang overloading? Jelaskan apa yang dimaksud dengan overloading dan jelaskan kegunaannya!

Jawab: overloading adalah fitur dimana suatu function memiliki nama yang sama namun memiliki parameter yang perbeda. Kegunaannya digunakan untuk kondisi dimana terdapat attribute yang seharusnya tidak usah di set terlebih dahulu.

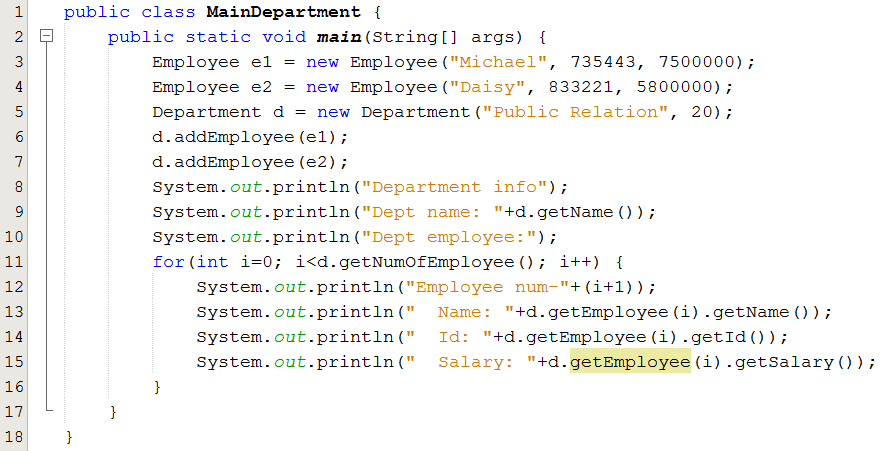
**Implementasi class Department**

Buat class baru dengan nama Department.java. Berikut isi class Department.





Kemudian buatlah sebuah file baru dengan nama MainDepartment.java. Berikut kode programnya.



Jalankan program MainDepartment.java. Apa output yang muncul?

Jawab: run:

**Departmen Info**

**Dept name: Public Relation**

**Dept employee:**

**Employee num-1**

**Name: Michael**

**Id: 735443**

**Salary: 7500000**

**Employee num-2**

**Name: Daisy**

**Id: 833221**

**Salary: 5800000**

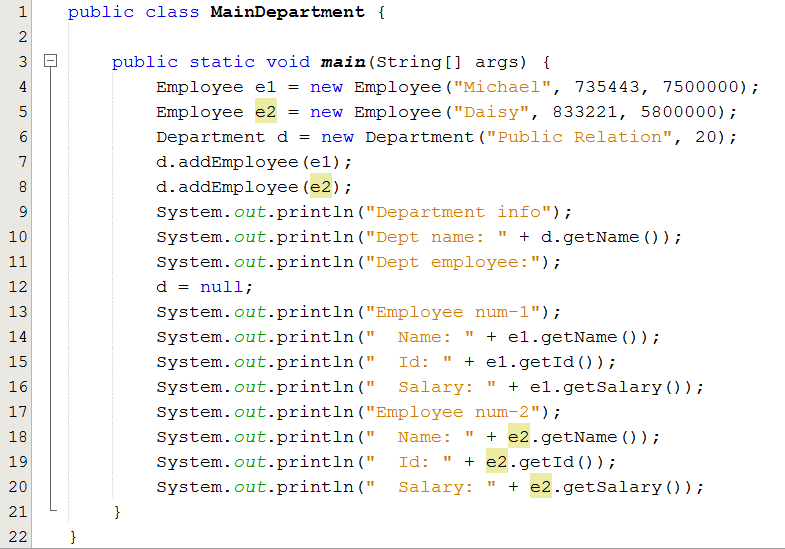
BUILD SUCCESSFUL (total time: 0 seconds)

..

Jelaskan apa yang terjadi pada baris ke-6 dan baris ke-7 pada kode MainDepartment.java di atas!

Jawab: menambah obyek employee ke dalam departmen

Lakukan modifikasi pada file MainDepartment.java menjadi sebagai berikut.



Jalankan file MainDepartment.java. Apakah hasilnya sama dengan sebelum dilakukan perubahan atau berbeda?

Jawab: berbeda

Apa yang terjadi pada baris ke-12 kode program MainDepartment.java yang dimodifikasi di atas?

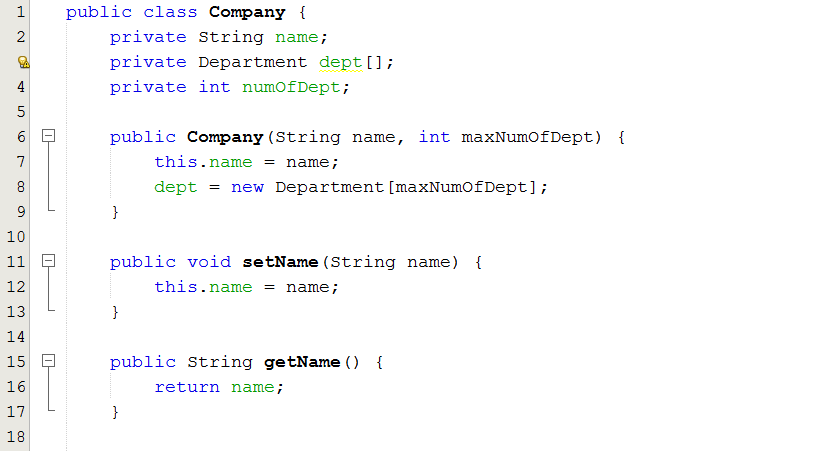
Jawab: d diset null

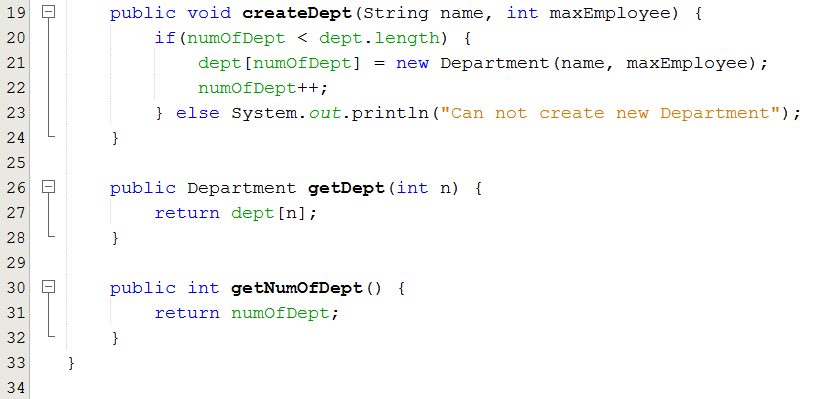
Apa perbedaan antara kode program MainDepartment.java sebelum dilakukan modifikasi dan setelah dilakukan modifikasi?

Jawab: setelah dimodifikasi employee menjadi bernilai kosong

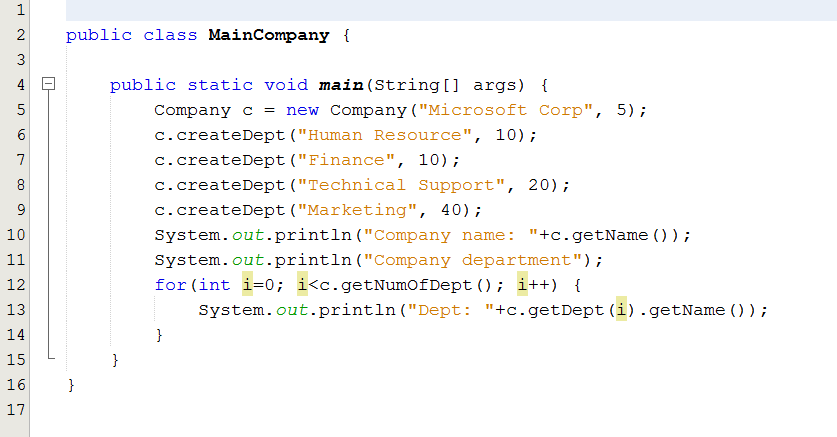
**Implementasi class Company**

Buat file baru dengan nama Company.java. Berikut kode programnya.





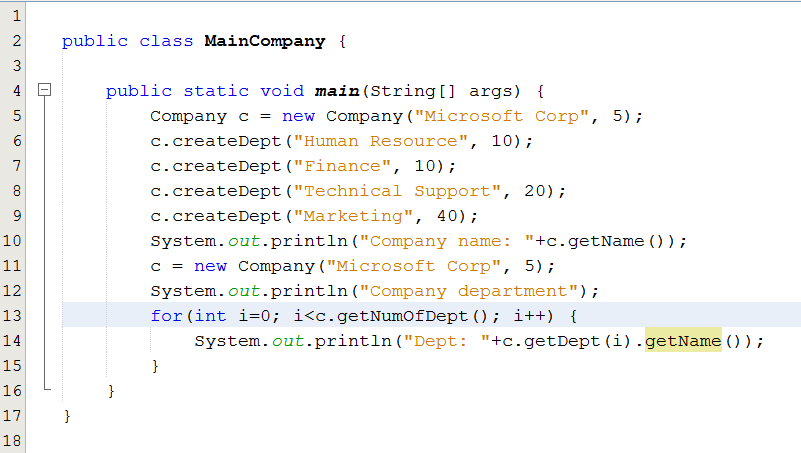
Buatlah sebuah file MainCompany.java. Berikut kode programnya.



Jalankan MainCompany.java. Apa output yang muncul?

Jawab: ……………………………………………………………………………………………..

Lakukan modifikasi pada MainCompany.java sebagai berikut.



Jalankan kembali MainCompany.java. Apa output yang muncul?

Jawab: ……………………………………………………………………………………………..

Apa yang terjadi pada kode baris ke-11 yang dimodifikasi di atas? Apa efeknya terhadap output program?

Jawab: ……………………………………………………………………………………………..

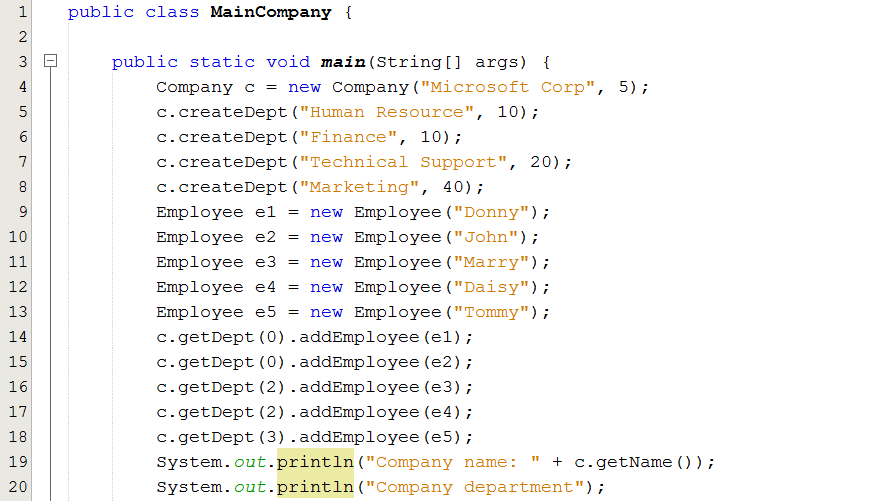
Apa perbedaan output yang muncul sebelum dimodifikasi dan setelah dimodifikasi? Apa yang menyebabkan perbedaan tersebut?

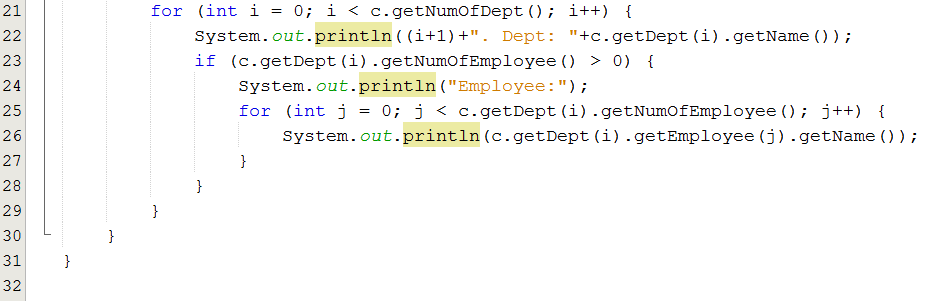
Jawab: ……………………………………………………………………………………………..

Pada file MainDepartment.java terlihat relasi antara class Department dan class Employee. Sedangkan pada file MainCompany.java terlihat relasi antara class Company dan class Department. Menurut Anda, apa perbedaan karakteristik antara relasi class Department dan class Employee pada file MainDepartment dan relasi class Company dan class Department pada file MainCompany?

Jawab: ……………………………………………………………………………………………..

Modifikasi file MainCompany.java menjadi sebagai berikut.





Jalankan file MainCompany.java. Apa output yang muncul?

Jawab: ……………………………………………………………………………………………..

Jelaskan apa yang terjadi pada baris ke-26 file MainCompany.java yang telah dimodifikasi di atas?

Jawab: ……………………………………………………………………………………………..