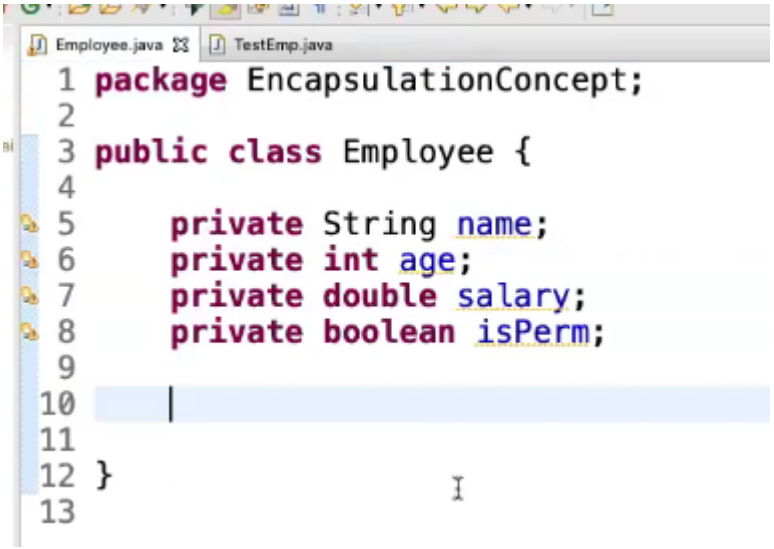
Encapsulation-

No unnecessary things shown to user.

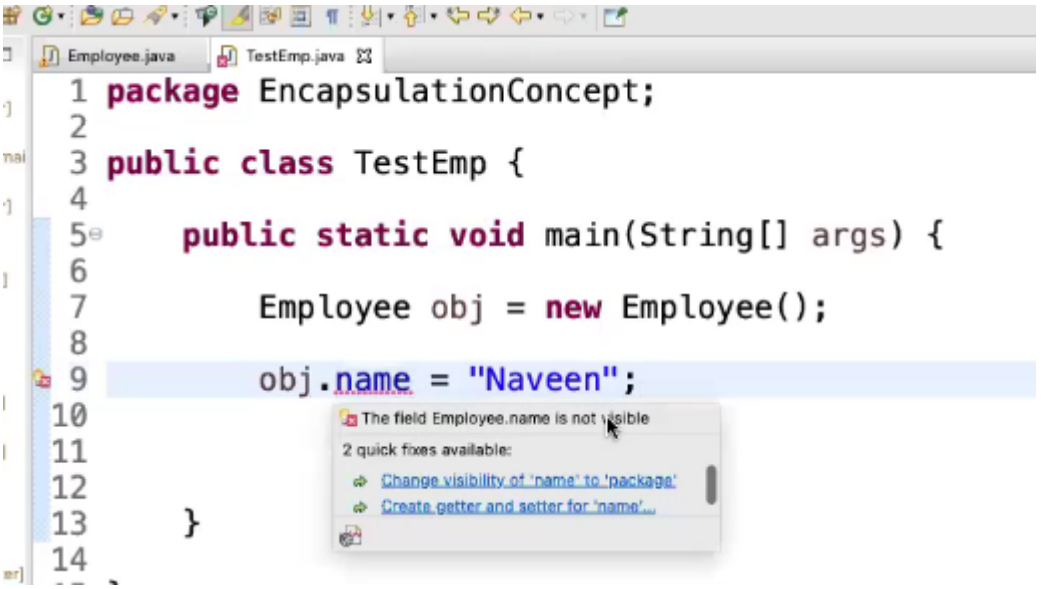
Data security.

We can keep all the variables as private.

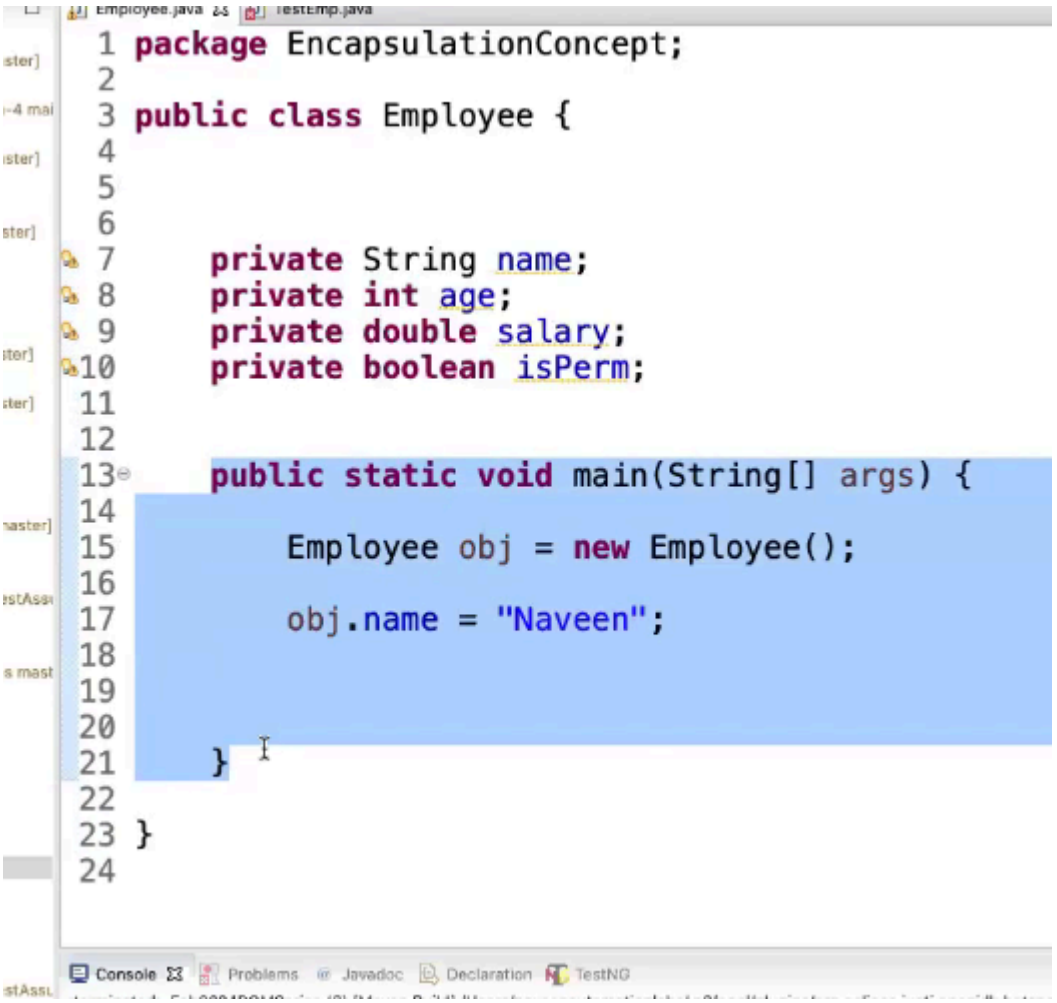
Methods will be public.



Cant access private from other class-

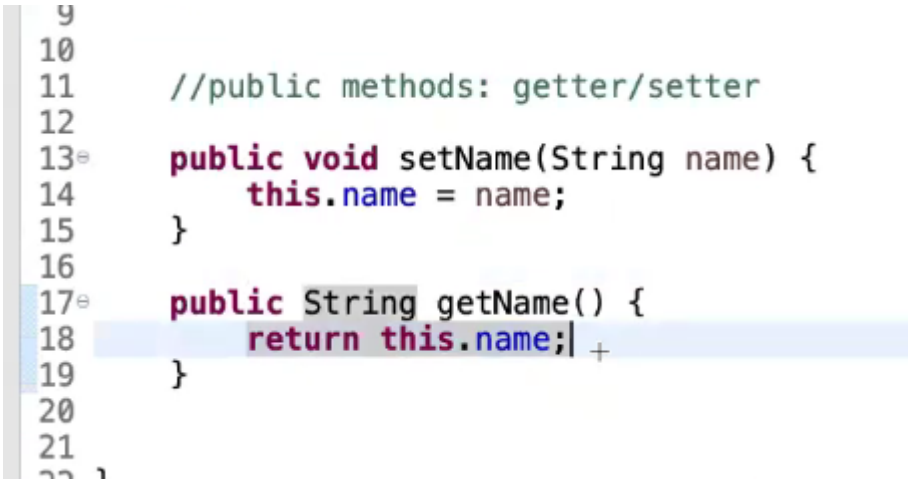


Within class everything allowed-

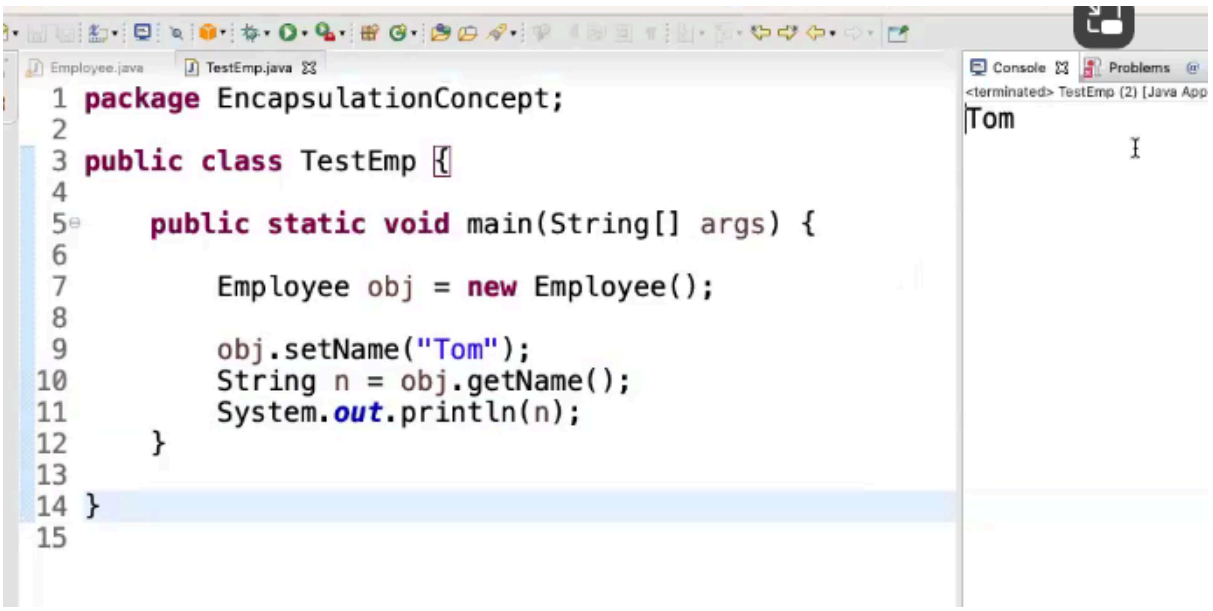


```java
package EncapsulationConcept;

public class Employee {



    private String name;
    private int age;
    private double salary;
    private boolean isPerm;


    public static void main(String[] args) {

        Employee obj = new Employee();

        obj.name = "Naveen";



    }

}
```

Getter and setter-



```java
    //public methods: getter/setter

    public void setName(String name) {
        this.name = name;
    }

    public String getName() {
        return this.name;
    }
```



Getters and setters for all-

```
9
10    // public methods: getter/setter
11
12    public String getName() {
13        return name;
14    }
15
16    public void setName(String name) {
17        this.name = name;
18    }
19
20    public int getAge() {
21        return age;
22    }
23
24    public void setAge(int age) {
25        this.age = age;
26    }
27
28    public double getSalary() {
29        return salary;
30    }
31
32    public void setSalary(double salary) {
33        this.salary = salary;
34    }
35
36    public boolean isPerm() {
37        return isPerm;
38    }
39
40    public void setPerm(boolean isPerm) {
41        this.isPerm = isPerm;
42    }
43
```

this is test emp class-

```
12
13
14          obj.setAge(20);
15          System.out.println(obj.getAge());
16
```

20

Constructor created-

```
 8      private double salary;
 9      private boolean isPerm;
10
11      //public const..
12      public Employee(String name, int age, double salary, boolean isPerm) {
13          this.name = name;
14          this.age = age;
15          this.salary = salary;
16          this.isPerm = isPerm;
17      }
18
19      // public methods: getter/setter
```

```
10
11      //public const.. is also like setter
12      public Employee(String name, int age, double salary, boolean isPerm) {
13          this.name = name;
14          this.age = age;
15          this.salary = salary;
16          this.isPerm = isPerm;
17      }
```

test emp class-

```
16
17          Employee obj = new Employee("Pooja", 20, 12.33, true);
18
19          System.out.println(obj.getName() + " " + obj.getAge() + " " + obj.getSalary() + " " + obj.isPerm());
20
21
```

```
Console  Problems  Java
<terminated> TestEmp (2) [Java Applica
Pooja 20 12.33 true
```

Setter can be done by constructor so not mandatory to write setters individually.

Setter used for updating values-

test emp class-

```
21
22          obj.setAge(21);
23          obj.setSalary(25);
24
25          System.out.println(obj.getName() + " " + obj.getAge() + " " + obj.getSalary() + " " + obj.isPerm());
26
27      }
28
29  }
30
```

```
Pooja 21 25.0 true
```

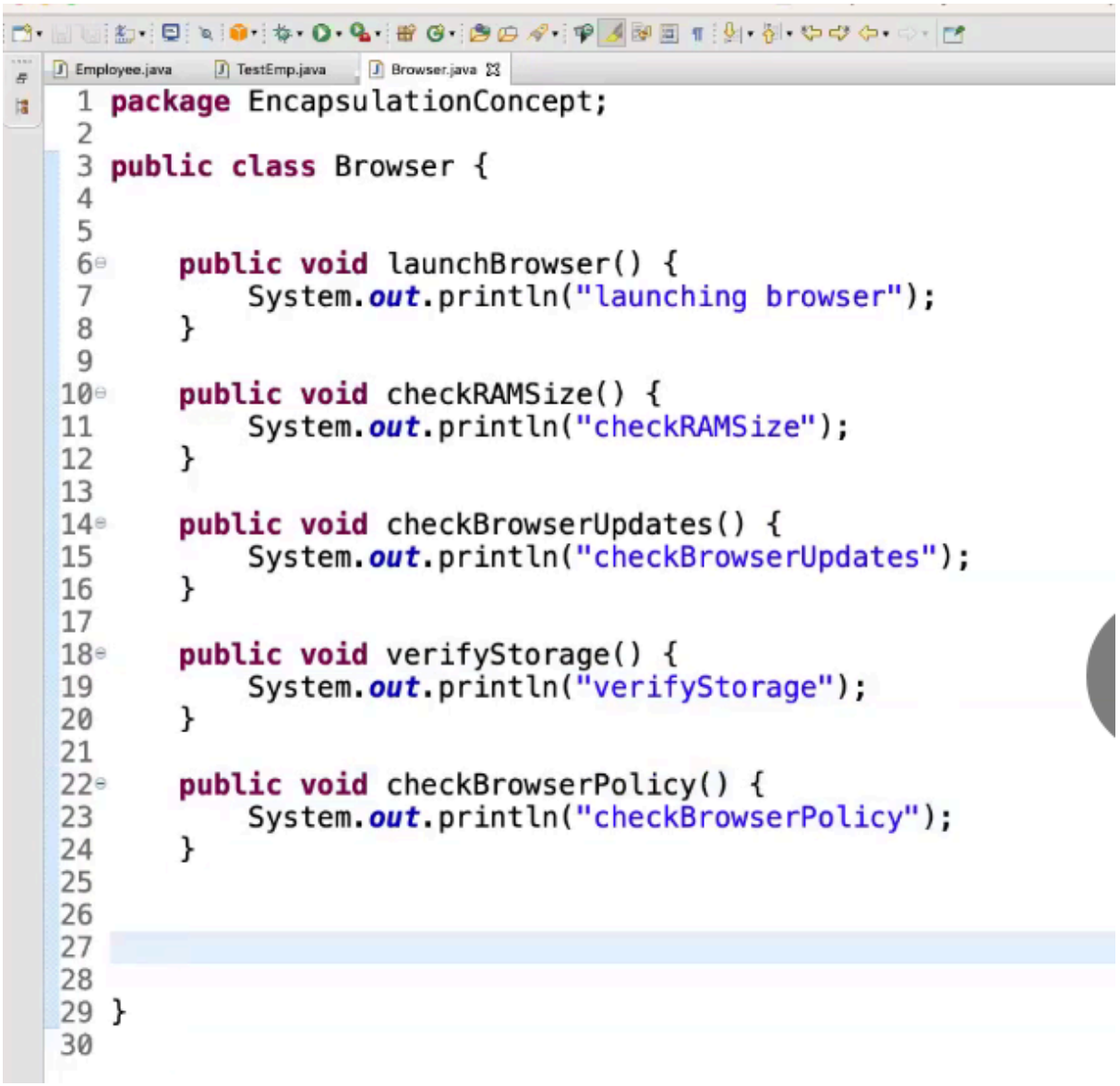
Another object-

```
31
32          //
33          Employee obj1 = new Employee("Veena", 30, 22.33, false);
34          System.out.println(obj1.getName() + " " + obj1.getAge() + " " + obj1.getSalary() + " " + obj1.isPerm());
35
```

```
35
36          obj1.setPerm(true);
37
38          System.out.println(obj1.getName() + " " + obj1.getAge() + " " + obj1.getSalary() + " " + obj1.isPerm());
```

```
Veena 30 22.33 false
Veena 30 22.33 true
```

Another class-



Bad design. Why to keep all methods as public which is not useful.

Encapsulate-

Employee.java *TestEmp.java Browser.java

```
1  package EncapsulationConcept;
2
3  public class Browser {
4
5
6      public void launchBrowser() {
7          System.out.println("launching browser");
8          checkRAMSize();
9          checkBrowserUpdates();
10         checkBrowserPolicy();
11         System.out.println("chrome is launched");
12     }
13
14     private void checkRAMSize() {
15         System.out.println("checkRAMSize");
16     }
17
18     private void checkBrowserUpdates() {
19         System.out.println("checkBrowserUpdates");
20     }
21
22     private void verifyStorage() {
23         System.out.println("verifyStorage");
24     }
25
26     private void checkBrowserPolicy() {
27         System.out.println("checkBrowserPolicy");
28     }
29
30
31
32
33 }
34
```

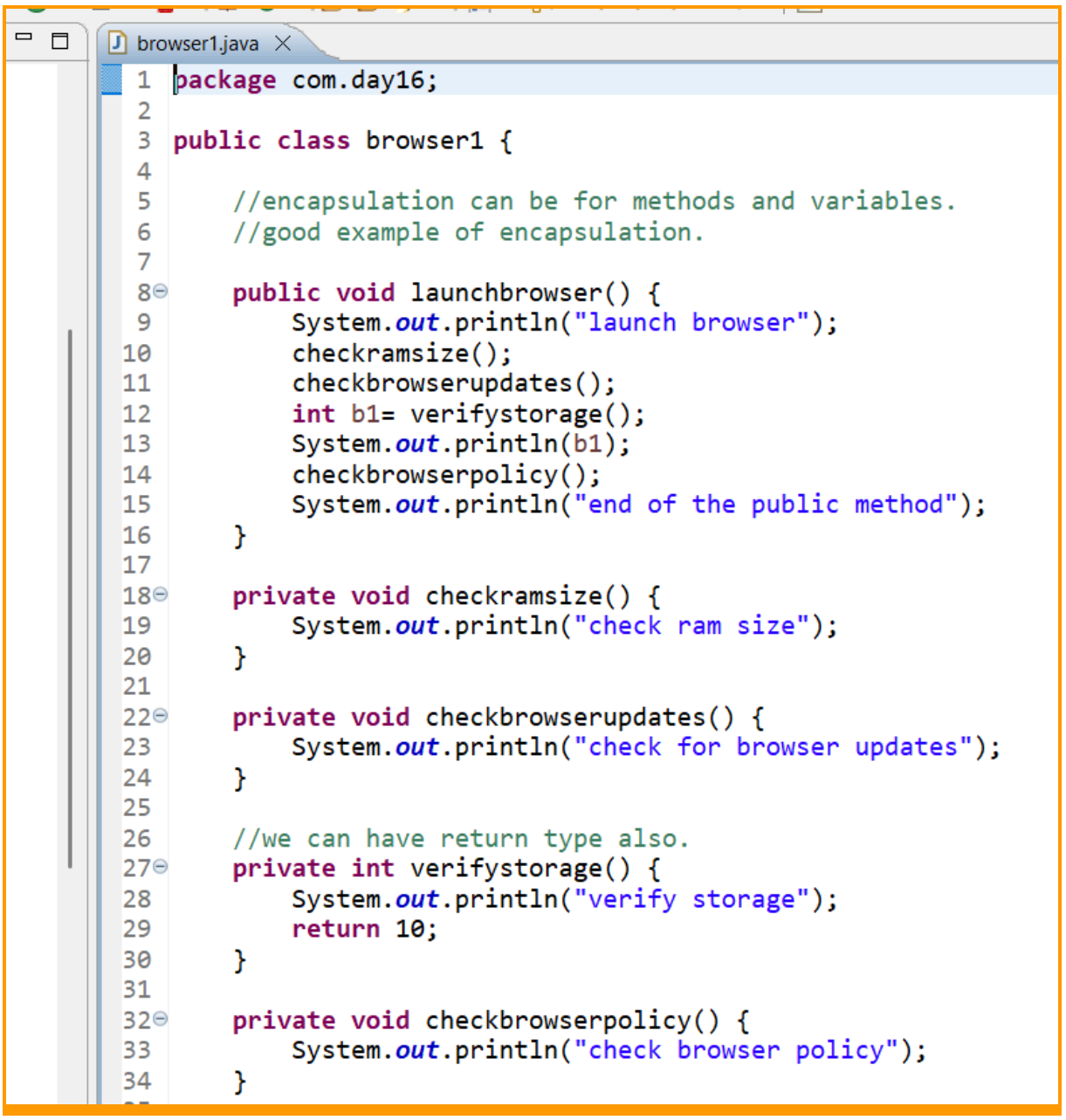
```
39
40
41         //browser:
42         Browser br = new Browser();
43         br.launchBrowser();
44
```

```
4
5⊖    public void launchBrowser() {
6         System.out.println("launching browser");
7         checkRAMSize();
8         checkBrowserUpdates();
9         checkBrowserPolicy();
10        verifyStorage();
11        System.out.println("chrome is launched");
12    }
```

Encapsulation for both methods and variables.

```
launching browser
checkRAMSize
checkBrowserUpdates
checkBrowserPolicy
verifyStorage
chrome is launched
```

paste browser1



browser1.java

```java
package com.day16;

public class browser1 {

    //encapsulation can be for methods and variables.
    //good example of encapsulation.

    public void launchbrowser() {
        System.out.println("launch browser");
        checkramsize();
        checkbrowserupdates();
        int b1= verifystorage();
        System.out.println(b1);
        checkbrowserpolicy();
        System.out.println("end of the public method");
    }

    private void checkramsize() {
        System.out.println("check ram size");
    }

    private void checkbrowserupdates() {
        System.out.println("check for browser updates");
    }

    //we can have return type also.
    private int verifystorage() {
        System.out.println("verify storage");
        return 10;
    }

    private void checkbrowserpolicy() {
        System.out.println("check browser policy");
    }
```

```
33          System.out.println("check browser policy");
34      }
35
36      public static void main(String[] args) {
37
38          browser1 b1=new browser1();
39          b1.launchbrowser();
40      }
41
42  }
43
44  //launch browser
45  //check ram size
46  //check for browser updates
47  //verify storage
48  //10
49  //check browser policy
50  //end of the public method
51
52
```

paste employee2 and test employee2

employee2.java testemployee2.java

```java
package com.day16;

public class employee2 {

    private String name;
    private int age;
    private double salary;
    private boolean isPermanent;

    //constructor with all fields.
    //constructor is also like setter.
    public employee2(String name, int age, double salary, boolean isPermanent) {
        this.name=name;
        this.age=age;
        this.salary=salary;
        this.isPermanent=isPermanent;
    }

    //setter method for name.
    public void setName(String name) {
        this.name=name;
    }

    //getter method for name.
    public String getName() {
        return this.name;
    }

    //getter for all other variables.

    public int getAge() {
        return this.age;
    }
```

```
        return this.age;
    }

    //this not mandatory for get method.
    public double getSalary() {
        return salary;
    }

    public boolean getIsPermanent() {
        return isPermanent;
    }

    //setter method for all other variables.

    public void setAge(int age) {
        this.age=age;
    }

    public void setSalary(double salary) {
        this.salary=salary;
    }

    public void setIsPermanent(boolean isPermanent) {
        this.isPermanent=isPermanent;
    }


}

```

testemployee2.java

```
package com.day16;

public class testemployee2 {

    public static void main(String[] args) {

        employee2 e1=new employee2("karan", 40, 50000.0, true);
        System.out.println(e1.getAge() + " " + e1.getName() + " " + e1.getSalary() + " " + e1.getIsPermanent());

        e1.setSalary(332443.234324);
        System.out.println(e1.getAge() + " " + e1.getName() + " " + e1.getSalary() + " " + e1.getIsPermanent());

        e1.setIsPermanent(false);
        System.out.println(e1.getAge() + " " + e1.getName() + " " + e1.getSalary() + " " + e1.getIsPermanent());
    }

}

//40 karan 50000.0 true
//40 karan 332443.234324 true
//40 karan 332443.234324 false
```

Another example-

Employee.java | TestEmp.java | Browser.java | LoginPage.java

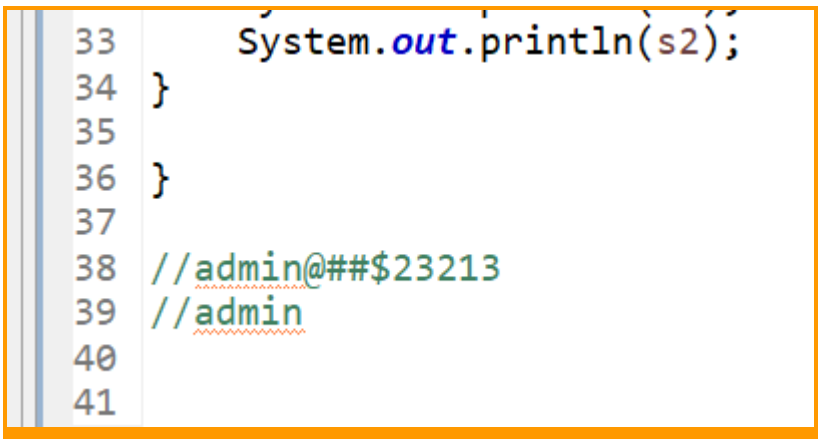
```java
package EncapsulationConcept;

public class LoginPage {

    private String userName;
    private String password;

    public String getUserName() {
        return userName;
    }

    public void setUserName(String userName) {
        this.userName = userName;
    }

    public String getPassword() {
        return password;
    }

    public void setPassword(String password) {
        this.password = password;
    }

}
```

```java
        //login:
        LoginPage lp1 = new LoginPage();
        lp1.setUserName("naveen@gmail.com");
        lp1.setPassword("naveen@123");
```

paste login1

Login1.java

```java
package com.day16;

public class Login1 {

    private String username;
    private String password;


    public String getUsername() {
        return username;
    }

    public void setUsername(String username) {
        this.username = username;
    }

    public String getPassword() {
        return password;
    }

    public void setPassword(String password) {
        this.password = password;
    }

public static void main(String[] args) {
    Login1 l1=new Login1();
    l1.setUsername("admin");
    l1.setPassword("admin@##$23213");
    String s1=l1.getPassword();
    String s2=l1.getUsername();

    System.out.println(s1);
    System.out.println(s2);
}
```



```
33      System.out.println(s2);
34  }
35
36  }
37
38  //admin@##$23213
39  //admin
40
41
```

Login1.java

```java
package com.day16;

public class Login1 {

    private String username;
    private String password;


    public String getUsername() {
        return username;
    }

    public void setUsername(String username) {
        this.username = username;
    }

    public String getPassword() {
        return password;
    }

    public void setPassword(String password) {
        this.password = password;
    }

public static void main(String[] args) {
    Login1 l1=new Login1();
    l1.setUsername("admin");
    l1.setPassword("admin@##$23213");
    String s1=l1.getPassword();
    String s2=l1.getUsername();

    System.out.println(s1);
    System.out.println(s2);
}

}

//admin@##$23213
//admin
```

Add constructor-

```
 7
 8⊖    public LoginPage(String userName, String password) {
 9        this.userName = userName;
10        this.password = password;
11    }
```

Add method-

```
29⊖    public void doLogin(String un, String pwd) {
30        System.out.println("enter username: " + un);
31        System.out.println("enter password: " + pwd);
32        System.out.println("click on login button");
33        System.out.println("user is logged in");
34    }
```

```
45        //login:
46        //POST - create credes
47        LoginPage lp1 = new LoginPage("naveen@gmail.com", "naveen@123");
48
49        lp1.doLogin(lp1.getUserName(), lp1.getPassword());
50
```

```
enter username: naveen@gmail.com
enter password: naveen@123
click on login button
user is logged in
```

Modified code-

```
27    }
28
29⊖    public void doLogin() {
30        System.out.println("enter username: " + getUserName());
31        System.out.println("enter password: " + getPassword());
32        System.out.println("click on login button");
33        System.out.println("user is logged in");
34    }
35
```

```
45        //login:
46        //POST - create credes
47        LoginPage lp1 = new LoginPage("naveen@gmail.com", "naveen@123");
48
49        lp1.doLogin();
```

```
enter username: naveen@gmail.com
enter password: naveen@123
click on login button
user is logged in
```

Get user name and password-

```
51        System.out.println(lp1.getUserName() + " " + lp1.getPassword());
```

```
user is logged in
naveen@gmail.com naveen@123
```

Set the values-

```
53        lp1.setPassword("naveen@9090");
54
55        System.out.println(lp1.getUserName() + " " + lp1.getPassword());
```

```
naveen@gmail.com naveen@123
naveen@gmail.com naveen@9090
```

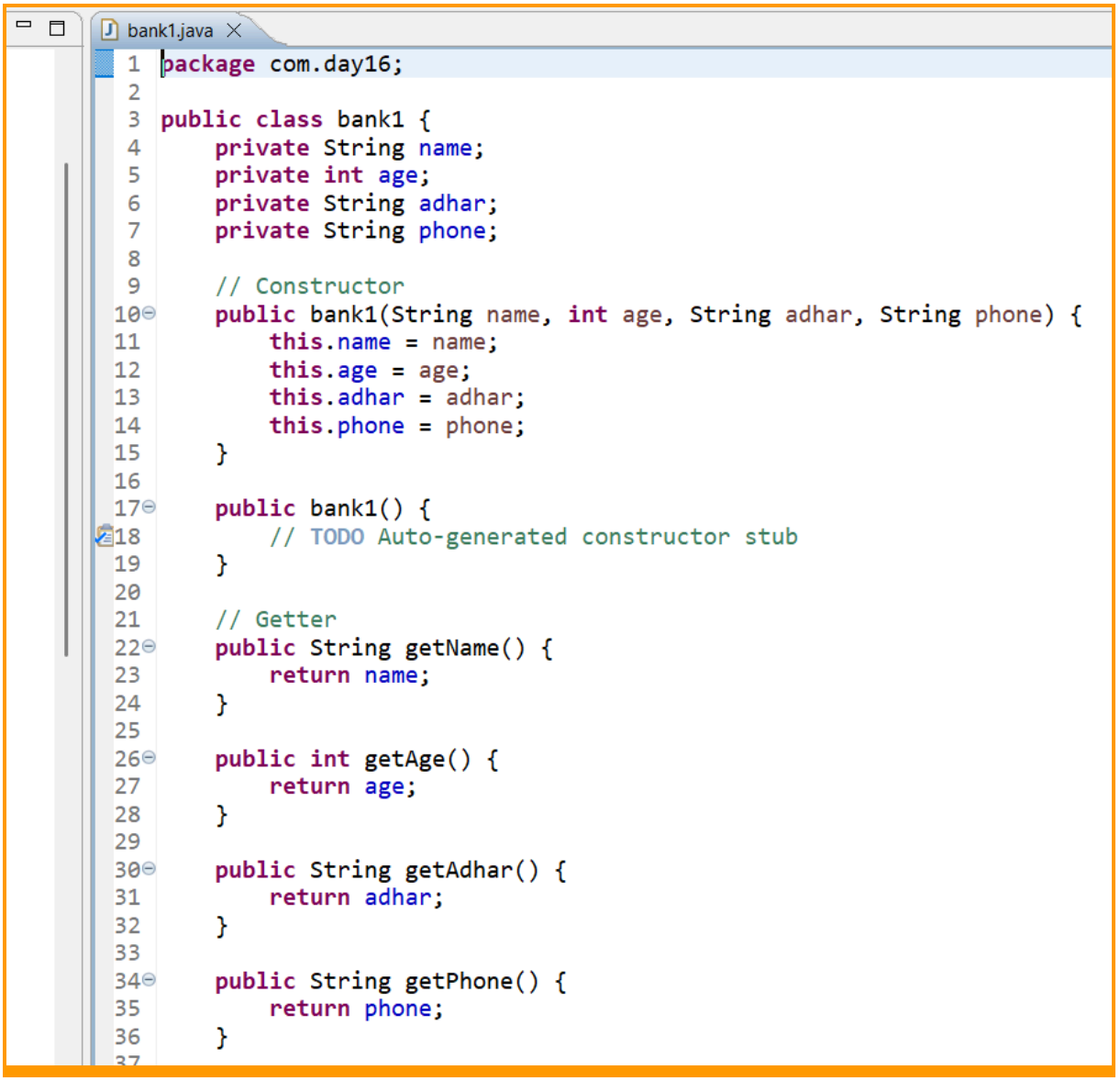
Create another object-

```
57        System.out.println("------");
58        //
59        LoginPage lp2 = new LoginPage("piyush@gmail.com", "pityus@123");
60        lp2.doLogin();
61
```

```
enter username: piyush@gmail.com
enter password: pityus@123
click on login button
user is logged in
```

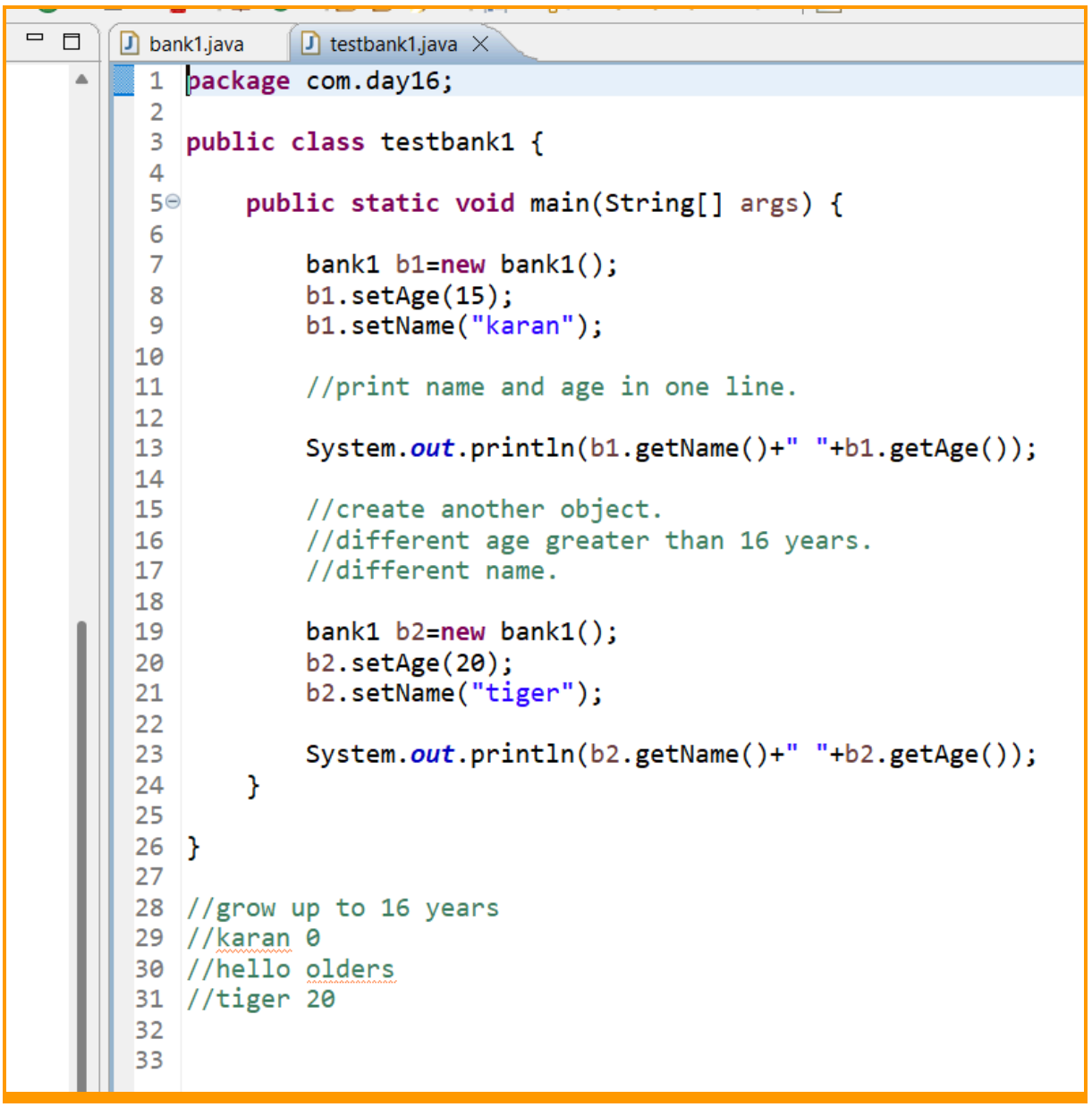
Another code-

paste bank1


bank1.java
1 package com.day16;
2
3 public class bank1 {
4 private String name;
5 private int age;
6 private String adhar;
7 private String phone;
8
9 // Constructor
10 public bank1(String name, int age, String adhar, String phone) {
11 this.name = name;
12 this.age = age;
13 this.adhar = adhar;
14 this.phone = phone;
15 }
16
17 public bank1() {
18 // TODO Auto-generated constructor stub
19 }
20
21 // Getter
22 public String getName() {
23 return name;
24 }
25
26 public int getAge() {
27 return age;
28 }
29
30 public String getAdhar() {
31 return adhar;
32 }
33
34 public String getPhone() {
35 return phone;
36 }

```
35          return phone;
36      }
37
38      // Setter
39      public void setName(String name) {
40          this.name = name;
41      }
42
43      //we can write any damn code inside getters and setters.
44      public void setAge(int age) {
45          if(age>=16) {
46              this.age=age;
47              System.out.println("hello olders");
48          }
49          else {
50              System.out.println("grow up to 16 years");
51          }
52      }
53
54      public void setAdhar(String adhar) {
55          this.adhar = adhar;
56      }
57
58      public void setPhone(String phone) {
59          this.phone = phone;
60      }
61  }
62
```

paste testbank1



Conditions in getters and setters allowed-

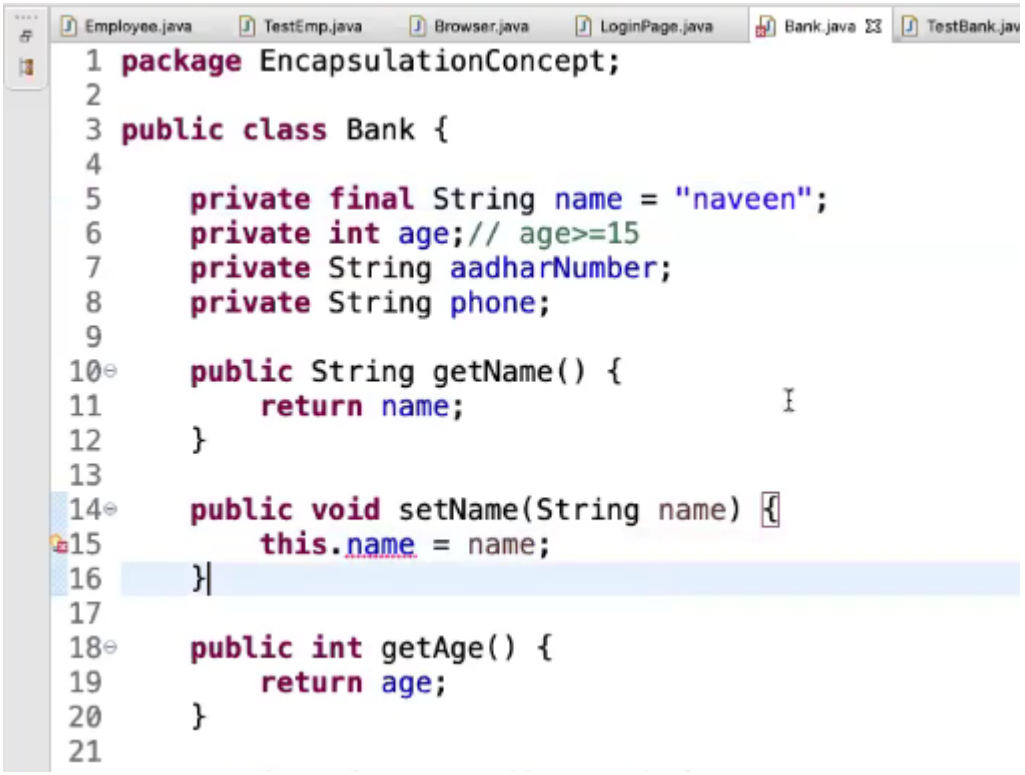
```
21
22    public void setAge(int age) {
23        if (age >= 15) {
24            this.age = age;
25        }
26    }
```

Change age to 16- covered above.

Note-

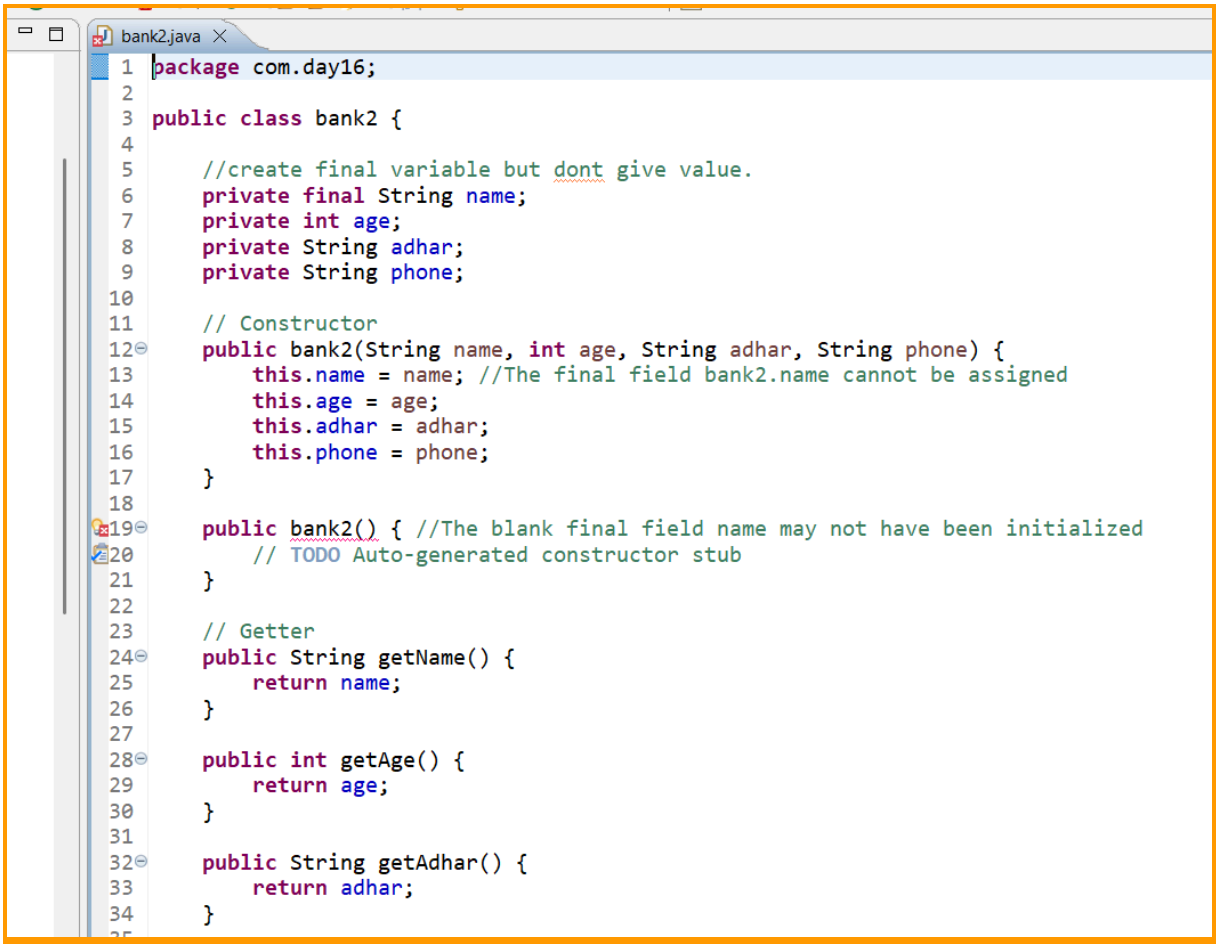
Keep all initial checks in setter.

Final variable no relation to encapsulation-



//The final field bank2.name cannot be assigned

paste bank2



bank2.java

```java
package com.day16;

public class bank2 {

    //create final variable but dont give value.
    private final String name;
    private int age;
    private String adhar;
    private String phone;

    // Constructor
    public bank2(String name, int age, String adhar, String phone) {
        this.name = name; //The final field bank2.name cannot be assigned
        this.age = age;
        this.adhar = adhar;
        this.phone = phone;
    }

    public bank2() { //The blank final field name may not have been initialized
        // TODO Auto-generated constructor stub
    }

    // Getter
    public String getName() {
        return name;
    }

    public int getAge() {
        return age;
    }

    public String getAdhar() {
        return adhar;
    }
```

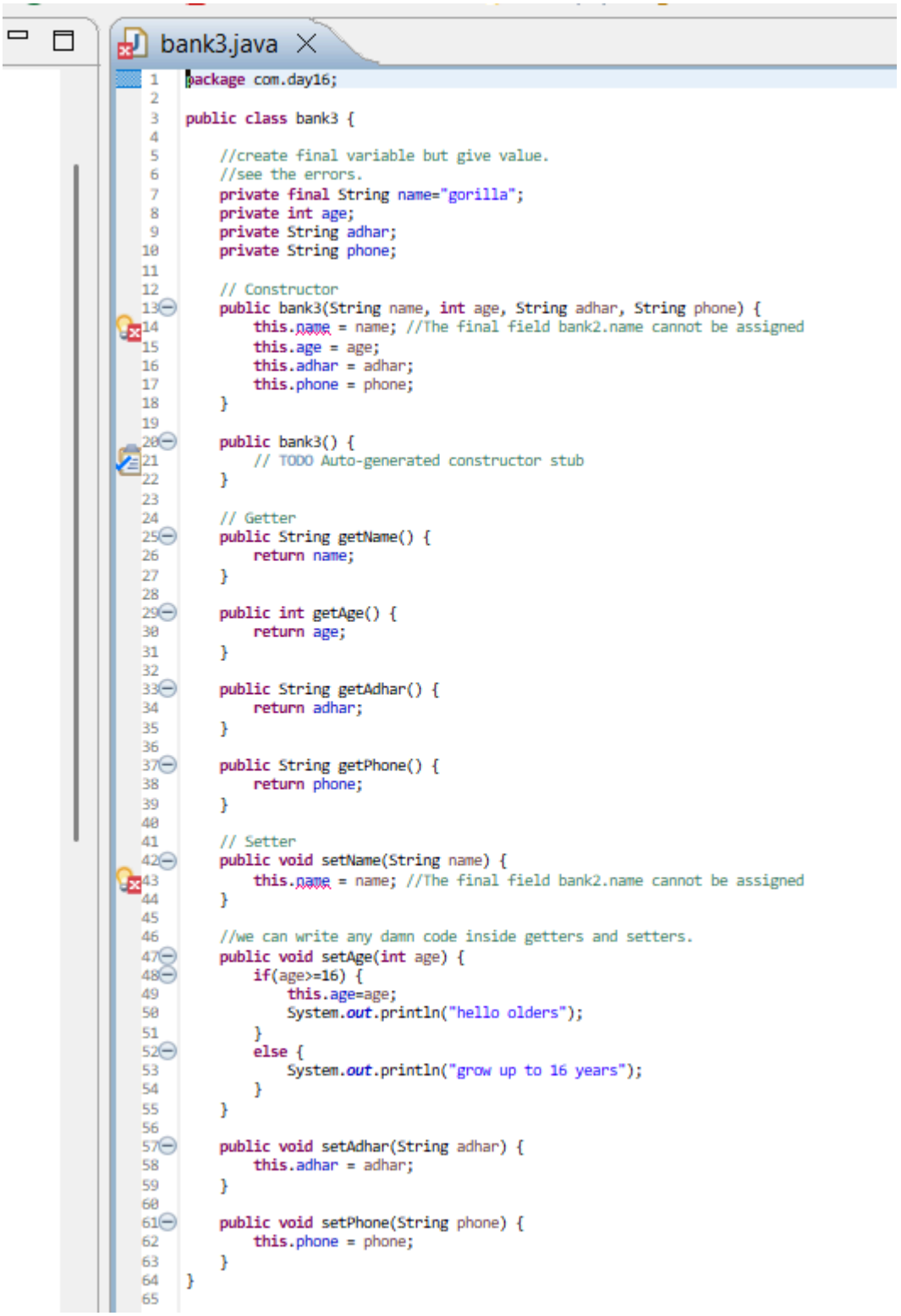
```
33          return adhar;
34      }
35
36      public String getPhone() {
37          return phone;
38      }
39
40      // Setter
41      public void setName(String name) {
42          this.name = name; //The final field bank2.name cannot be assigned
43      }
44
45      //we can write any damn code inside getters and setters.
46      public void setAge(int age) {
47          if(age>=16) {
48              this.age=age;
49              System.out.println("hello olders");
50          }
51          else {
52              System.out.println("grow up to 16 years");
53          }
54      }
55
56      public void setAdhar(String adhar) {
57          this.adhar = adhar;
58      }
59
60      public void setPhone(String phone) {
61          this.phone = phone;
62      }
63  }
64
```

paste bank3

bank2.java bank3.java

```java
package com.day16;

public class bank3 {

    //create final variable but give value.
    //see the errors.
    private final String name="gorilla";
    private int age;
    private String adhar;
    private String phone;

    // Constructor
    public bank3(String name, int age, String adhar, String phone) {
        this.name = name; //The final field bank2.name cannot be assigned
        this.age = age;
        this.adhar = adhar;
        this.phone = phone;
    }

    public bank3() {
        // TODO Auto-generated constructor stub
    }

    // Getter
    public String getName() {
        return name;
    }

    public int getAge() {
        return age;
    }

    public String getAdhar() {
        return adhar;
    }
```

```
34          return adhar;
35      }
36
37      public String getPhone() {
38          return phone;
39      }
40
41      // Setter
42      public void setName(String name) {
43          this.name = name; //The final field bank2.name cannot be assigned
44      }
45
46      //we can write any damn code inside getters and setters.
47      public void setAge(int age) {
48          if(age>=16) {
49              this.age=age;
50              System.out.println("hello olders");
51          }
52          else {
53              System.out.println("grow up to 16 years");
54          }
55      }
56
57      public void setAdhar(String adhar) {
58          this.adhar = adhar;
59      }
60
61      public void setPhone(String phone) {
62          this.phone = phone;
63      }
64  }
65
```



bank3.java

```java
package com.day16;

public class bank3 {

    //create final variable but give value.
    //see the errors.
    private final String name="gorilla";
    private int age;
    private String adhar;
    private String phone;

    // Constructor
    public bank3(String name, int age, String adhar, String phone) {
        this.name = name; //The final field bank2.name cannot be assigned
        this.age = age;
        this.adhar = adhar;
        this.phone = phone;
    }

    public bank3() {
        // TODO Auto-generated constructor stub
    }

    // Getter
    public String getName() {
        return name;
    }

    public int getAge() {
        return age;
    }

    public String getAdhar() {
        return adhar;
    }

    public String getPhone() {
        return phone;
    }

    // Setter
    public void setName(String name) {
        this.name = name; //The final field bank2.name cannot be assigned
    }

    //we can write any damn code inside getters and setters.
    public void setAge(int age) {
        if(age>=16) {
            this.age=age;
            System.out.println("hello olders");
        }
        else {
            System.out.println("grow up to 16 years");
        }
    }

    public void setAdhar(String adhar) {
        this.adhar = adhar;
    }

    public void setPhone(String phone) {
        this.phone = phone;
    }
}
```

paste bank4

bank4.java

```java
package com.day16;

public class bank4 {

    private final String name="gorilla";
    private int age;
    private String adhar;
    private String phone;

    // Constructor
    public bank4(int age, String adhar, String phone) {
        this.age = age;

        //same setter condition can be set in constructor also.
        //sysout doesnt come in constructor.
        //constructor only for initialising and setting values not returning anything.
        //because constructor doesnt have return type.
        if(age>=16) {
            this.age=age;
            sysout.println("hello olders");
        }
        else {
            sysout.println("grow up to 16 years");
        }
        this.adhar = adhar;
        this.phone = phone;
    }

    public bank4() {
        // TODO Auto-generated constructor stub
    }

    // Getter
    public String getName() {
        return name;
    }
```

```java
    public String getName() {
        return name;
    }

    public int getAge() {
        return age;
    }

    public String getAdhar() {
        return adhar;
    }

    public String getPhone() {
        return phone;
    }

    // Setter
    public void setName(String name) {
        this.name = name; //The final field bank2.name cannot be assigned
    }
```

```
    public void setName(String name) {
        this.name = name; //The final field bank2.name cannot be assigned
    }

    //we can write any damn code inside getters and setters.
    public void setAge(int age) {
        if(age>=16) {
            this.age=age;
            System.out.println("hello olders");
        }
        else {
            System.out.println("grow up to 16 years");
        }
    }

    public void setAdhar(String adhar) {
        this.adhar = adhar;
    }

    public void setPhone(String phone) {
        this.phone = phone;
    }
}

```

paste bank 5

bank5.java

```java
package com.day16;

public class bank5 {

    private final String name="gorilla";
    private int age;
    private String adhar;
    private String phone;

    // Constructor
    public bank5(int age, String adhar, String phone) {
        this.age = age;

        if(age>=16) {
            this.age=age;
        }
        else {
            this.age=10;
        }
        this.adhar = adhar;
        this.phone = phone;
    }

    public bank5() {
        // TODO Auto-generated constructor stub
    }

    // Getter
    public String getName() {
        return name;
    }

    public int getAge() {
        return age;
    }
```

```
34          return age;
35      }
36
37      public String getAdhar() {
38          return adhar;
39      }
40
41      public String getPhone() {
42          return phone;
43      }
44
45      // Setter
46      public void setName(String name) {
47          this.name = name; //The final field bank2.name cannot be assigned
48      }
49
50      //we can write any damn code inside getters and setters.
51      public void setAge(int age) {
52          if(age>=16) {
53              this.age=age;
54              System.out.println("hello olders");
55          }
56          else {
57              System.out.println("grow up to 16 years");
58          }
59      }
60
61      public void setAdhar(String adhar) {
62          this.adhar = adhar;
63      }
64
65      public void setPhone(String phone) {
66          this.phone = phone;
67      }
68  }
69
```

paste bank6

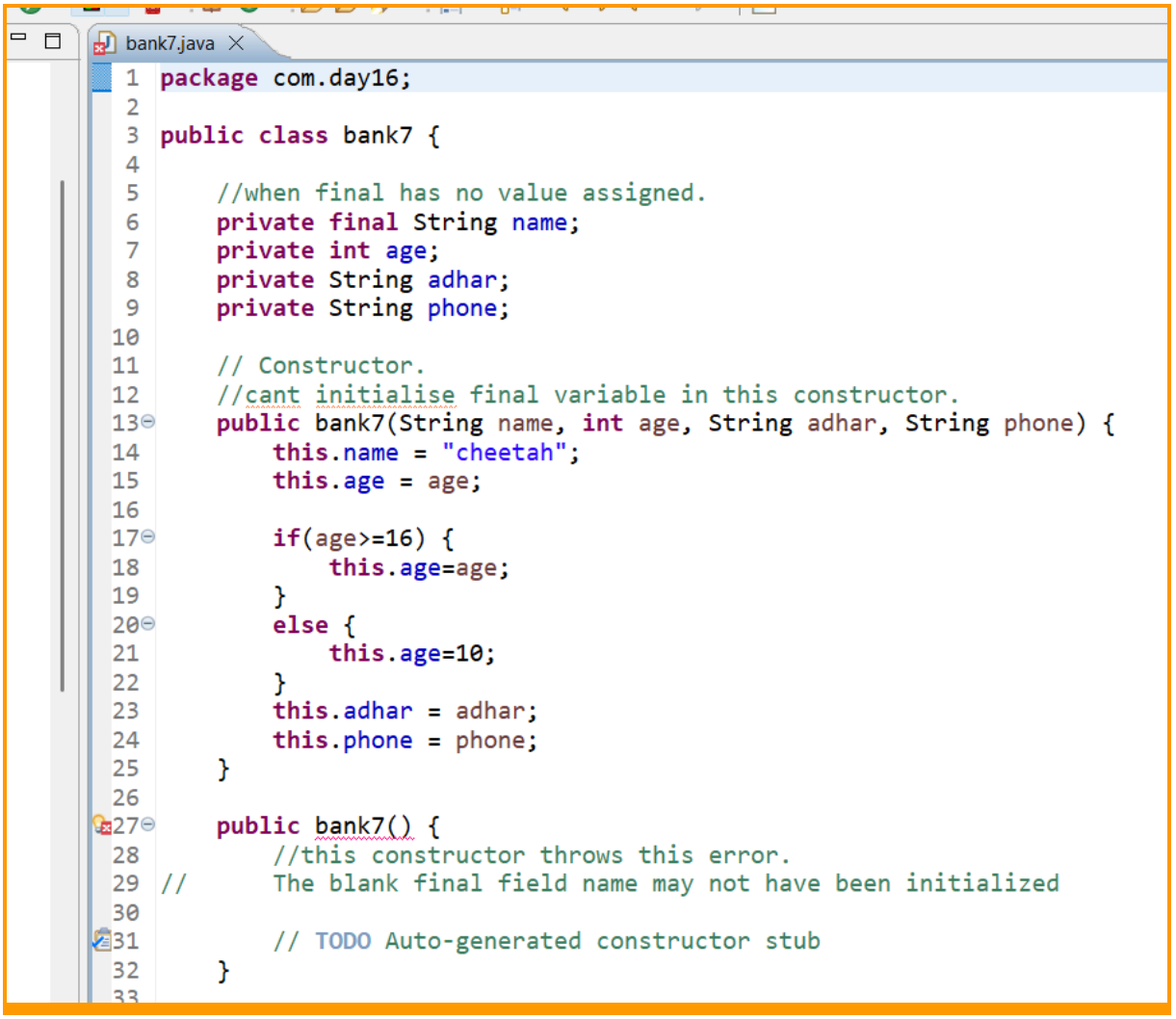
bank6.java

```java
package com.day16;

public class bank6 {

    //when final has value assigned.
    private final String name="gorilla";
    private int age;
    private String adhar;
    private String phone;

    // Constructor
    public bank6(int age, String adhar, String phone) {
        this.age = age;

        if(age>=16) {
            this.age=age;
        }
        else {
            this.age=10;
        }
        this.adhar = adhar;
        this.phone = phone;
    }

    public bank6() {
        // TODO Auto-generated constructor stub
    }

    // Getter
    public String getName() {
        return name;
    }

    public int getAge() {
        return age;
    }
```

```
    public int getAge() {
        return age;
    }

    public String getAdhar() {
        return adhar;
    }

    public String getPhone() {
        return phone;
    }

    //we can write any damn code inside getters and setters.
    public void setAge(int age) {
        if(age>=16) {
            this.age=age;
            System.out.println("hello olders");
        }
        else {
            System.out.println("grow up to 16 years");
        }
    }

    public void setAdhar(String adhar) {
        this.adhar = adhar;
    }

    public void setPhone(String phone) {
        this.phone = phone;
    }
```

```
        this.phone = phone;
    }

    public static void main(String[] args) {

        bank6 obj = new bank6(20, "adhar", "phone");
        System.out.println(obj.getName());
        System.out.println(obj.getAge());
        System.out.println(obj.getAdhar());
        System.out.println(obj.getPhone());
    }
}

//gorilla
//20
//adhar
//phone
```

paste bank7


bank7.java
1 package com.day16;
2
3 public class bank7 {
4
5 //when final has no value assigned.
6 private final String name;
7 private int age;
8 private String adhar;
9 private String phone;
10
11 // Constructor.
12 //cant initialise final variable in this constructor.
13 public bank7(String name, int age, String adhar, String phone) {
14 this.name = "cheetah";
15 this.age = age;
16
17 if(age>=16) {
18 this.age=age;
19 }
20 else {
21 this.age=10;
22 }
23 this.adhar = adhar;
24 this.phone = phone;
25 }
26
27 public bank7() {
28 //this constructor throws this error.
29 // The blank final field name may not have been initialized
30
31 // TODO Auto-generated constructor stub
32 }
33

```
        // TODO Auto-generated constructor stub
    }

    // Getter
    public String getName() {
        return name;
    }

    public int getAge() {
        return age;
    }

    public String getAdhar() {
        return adhar;
    }

    public String getPhone() {
        return phone;
    }

    //we can write any damn code inside getters and setters.
    public void setAge(int age) {
        if(age>=16) {
            this.age=age;
            System.out.println("hello olders");
        }
        else {
            System.out.println("grow up to 16 years");
        }
    }
```

```
        }
    }

    public void setAdhar(String adhar) {
        this.adhar = adhar;
    }

    public void setPhone(String phone) {
        this.phone = phone;
    }

    public static void main(String[] args) {

        bank7 obj = new bank7("manking", 20, "adhar", "phone");
        System.out.println(obj.getName());
        System.out.println(obj.getAge());
        System.out.println(obj.getAdhar());
        System.out.println(obj.getPhone());
    }
}
```

paste bank8

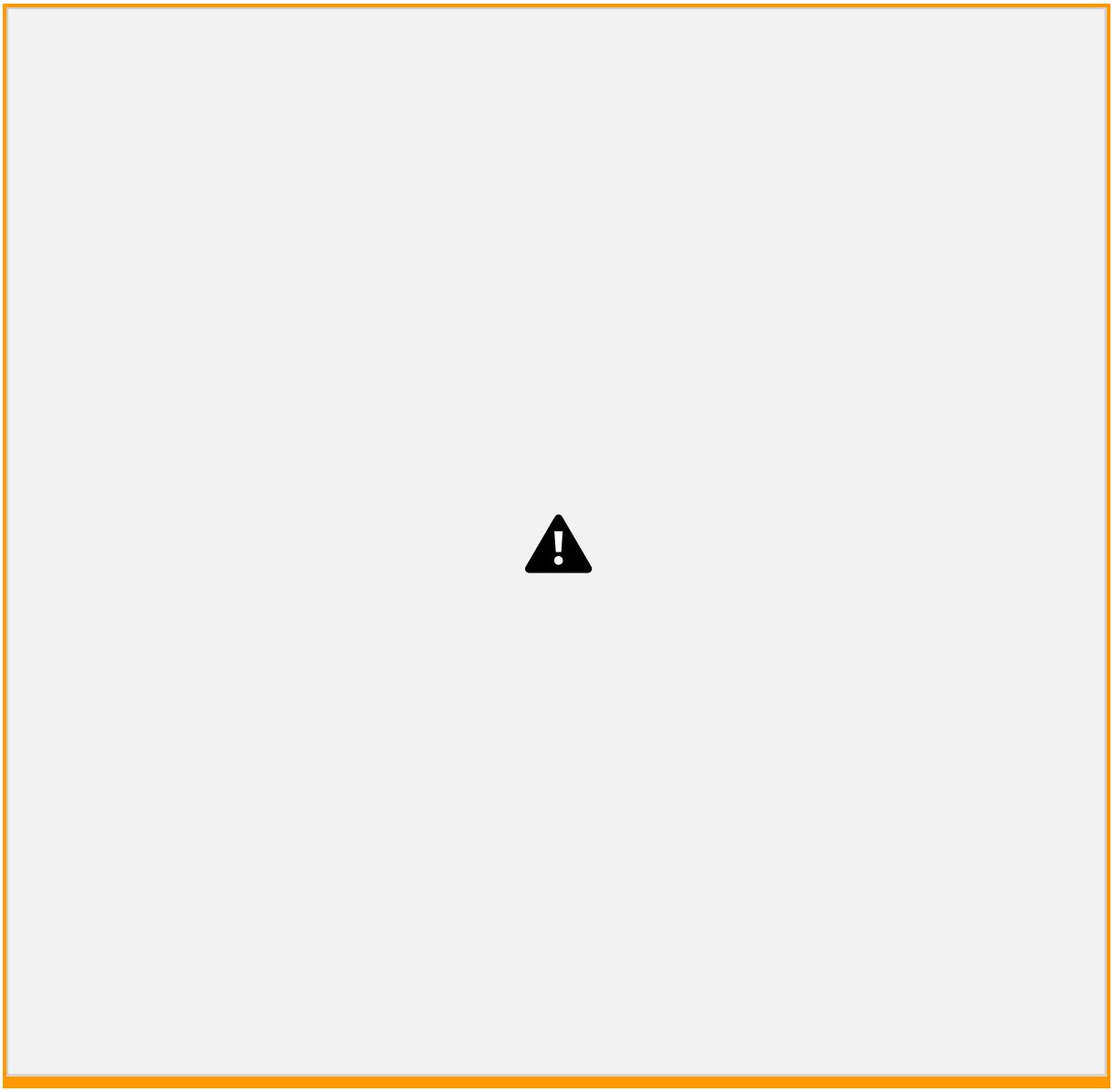
bank8.java

```java
package com.day16;

public class bank8 {

    //when final has no value assigned.
    private final String name;
    private int age;
    private String adhar;
    private String phone;

    // Constructor.

    public bank8(String name, int age, String adhar, String phone) {
        this.name = "stone cold";
        this.age = age;

        if(age>=16) {
            this.age=age;
        }
        else {
            this.age=10;
        }
        this.adhar = adhar;
        this.phone = phone;
    }

    // Getter
    public String getName() {
        return name;
    }

    public int getAge() {
        return age;
    }
```

```
        return age;
    }

    public String getAdhar() {
        return adhar;
    }

    public String getPhone() {
        return phone;
    }

    //we can write any damn code inside getters and setters.
    public void setAge(int age) {
        if(age>=16) {
            this.age=age;
            System.out.println("hello olders");
        }
        else {
            System.out.println("grow up to 16 years");
        }
    }

    public void setAdhar(String adhar) {
        this.adhar = adhar;
    }

    public void setPhone(String phone) {
        this.phone = phone;
    }
```

```
        this.phone = phone;
    }

    public static void main(String[] args) {

        bank8 obj = new bank8(getna, "adhar", "phone"); //we cannot use getname method directly.
        //first we need to create the object then only can access.
        System.out.println(obj.getName());
        System.out.println(obj.getAge());
        System.out.println(obj.getAdhar());
        System.out.println(obj.getPhone());
    }
}

//gorilla
//20
//adhar
//phone
```

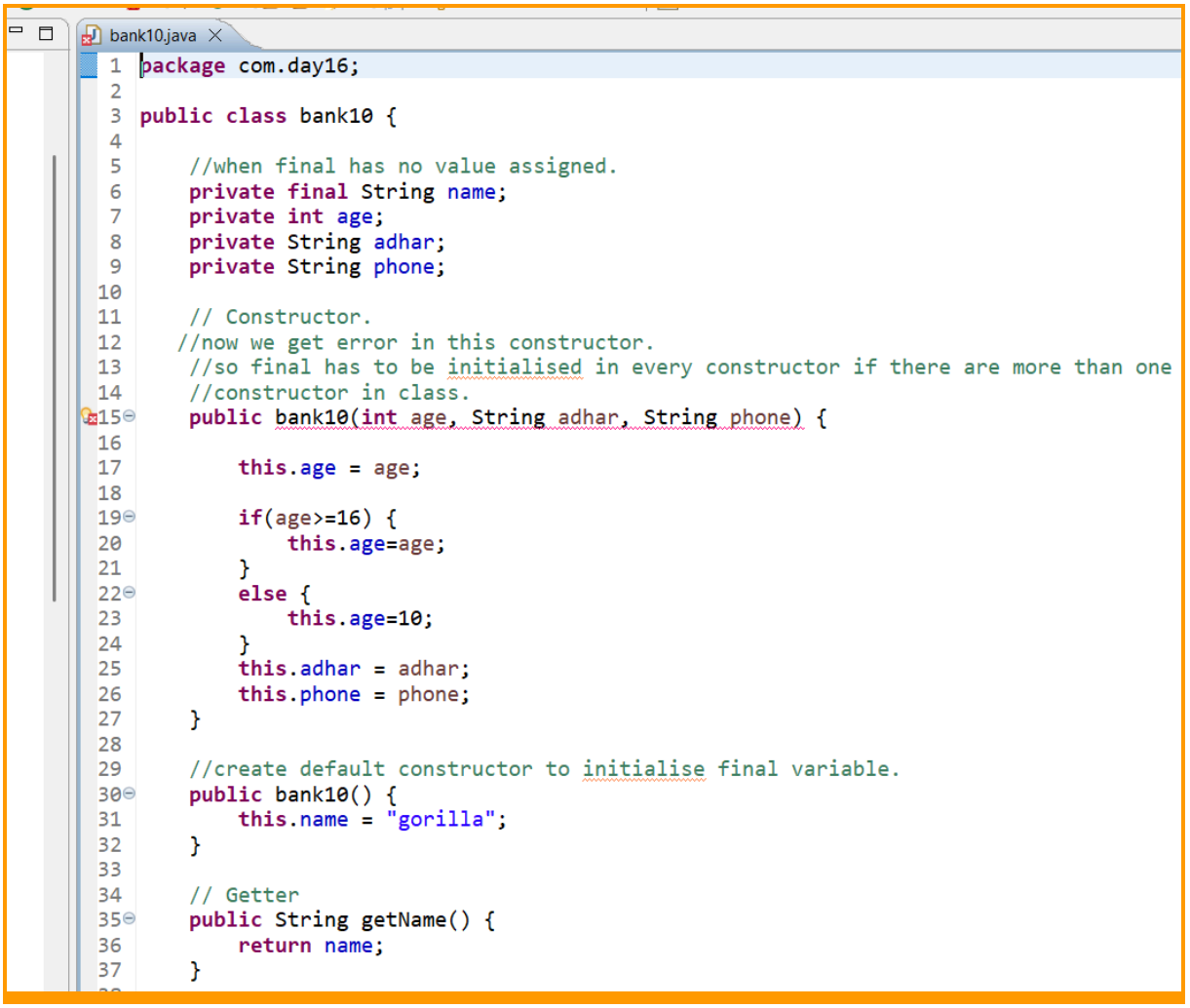
paste bank9



```
36          return adhar;
37      }
38
39      public String getPhone() {
40          return phone;
41      }
42
43      //we can write any damn code inside getters and setters.
44      public void setAge(int age) {
45          if(age>=16) {
46              this.age=age;
47              System.out.println("hello olders");
48          }
49          else {
50              System.out.println("grow up to 16 years");
51          }
52      }
53
54      public void setAdhar(String adhar) {
55          this.adhar = adhar;
56      }
57
58      public void setPhone(String phone) {
59          this.phone = phone;
60      }
61
62      public static void main(String[] args) {
```

```
61
62      public static void main(String[] args) {
63
64          //allows to change value of final but doesnt reflect in output.
65          //in output can see original value only.
66          bank9 obj = new bank9("manking", 20, "adhar", "phone");
67          System.out.println(obj.getName());
68          System.out.println(obj.getAge());
69          System.out.println(obj.getAdhar());
70          System.out.println(obj.getPhone());
71      }
72  }
73
74  //cheetah
75  //20
76  //adhar
77  //phone
78
79
```

paste bank10



bank10.java

```java
package com.day16;

public class bank10 {

    //when final has no value assigned.
    private final String name;
    private int age;
    private String adhar;
    private String phone;

    // Constructor.
   //now we get error in this constructor.
    //so final has to be initialised in every constructor if there are more than one
    //constructor in class.
    public bank10(int age, String adhar, String phone) {

        this.age = age;

        if(age>=16) {
            this.age=age;
        }
        else {
            this.age=10;
        }
        this.adhar = adhar;
        this.phone = phone;
    }

    //create default constructor to initialise final variable.
    public bank10() {
        this.name = "gorilla";
    }

    // Getter
    public String getName() {
        return name;
    }
```

```
36          return name;
37      }
38
39      public int getAge() {
40          return age;
41      }
42
43      public String getAdhar() {
44          return adhar;
45      }
46
47      public String getPhone() {
48          return phone;
49      }
50
51      //we can write any damn code inside getters and setters.
52      public void setAge(int age) {
53          if(age>=16) {
54              this.age=age;
55              System.out.println("hello olders");
56          }
57          else {
58              System.out.println("grow up to 16 years");
59          }
60      }
61
62      public void setAdhar(String adhar) {
63          this.adhar = adhar;
64      }
```

```
63          this.adhar = adhar;
64      }
65
66      public void setPhone(String phone) {
67          this.phone = phone;
68      }
69
70      public static void main(String[] args) {
71
72          bank10 obj = new bank10(10, "adhar", "phone");
73          System.out.println(obj.getName());
74          System.out.println(obj.getAge());
75          System.out.println(obj.getAdhar());
76          System.out.println(obj.getPhone());
77      }
78  }
79
80  //gorilla
81  //20
82  //adhar
83  //phone
84
85
```

paste bank11

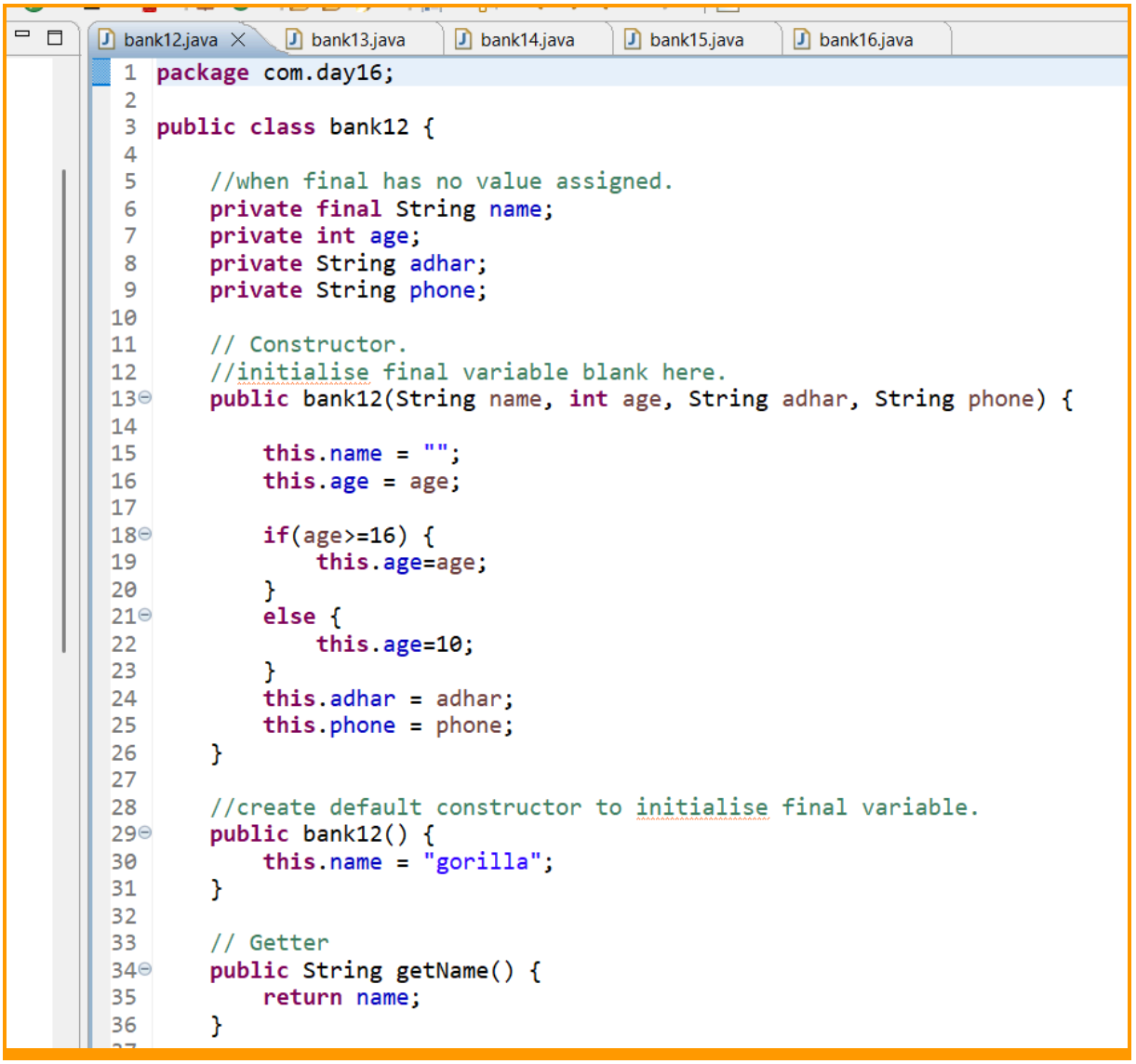
bank11.java

```java
package com.day16;

public class bank11 {

    //when final has no value assigned.
    private final String name;
    private int age;
    private String adhar;
    private String phone;

    // Constructor.
    //initialise final variable blank here.
    public bank11(int age, String adhar, String phone) {

        this.name = "";
        this.age = age;

        if(age>=16) {
            this.age=age;
        }
        else {
            this.age=10;
        }
        this.adhar = adhar;
        this.phone = phone;
    }

    //create default constructor to initialise final variable.
    public bank11() {
        this.name = "gorilla";
    }

    // Getter
    public String getName() {
        return name;
    }
```

```
35          return name;
36      }
37
38      public int getAge() {
39          return age;
40      }
41
42      public String getAdhar() {
43          return adhar;
44      }
45
46      public String getPhone() {
47          return phone;
48      }
49
50      //we can write any damn code inside getters and setters.
51      public void setAge(int age) {
52          if(age>=16) {
53              this.age=age;
54              System.out.println("hello olders");
55          }
56          else {
57              System.out.println("grow up to 16 years");
58          }
59      }
60
61      public void setAdhar(String adhar) {
62          this.adhar = adhar;
63      }
64
65      public void setPhone(String phone) {
66          this.phone = phone;
67      }
```

```
66          this.phone = phone;
67      }
68
69      public static void main(String[] args) {
70
71          //create object of empty constructor.
72          bank11 obj1 = new bank11();
73          System.out.println(obj1.getName());
74
75          bank11 obj = new bank11(10, "adhar", "phone");
76          System.out.println(obj.getName()); //blank
77          System.out.println(obj.getAge());
78          System.out.println(obj.getAdhar());
79          System.out.println(obj.getPhone());
80      }
81  }
82
83  //gorilla
84  //
85  //10
86  //adhar
87  //phone
88
89
```

paste bank12



bank12.java | bank13.java | bank14.java | bank15.java | bank16.java

```java
package com.day16;

public class bank12 {

    //when final has no value assigned.
    private final String name;
    private int age;
    private String adhar;
    private String phone;

    // Constructor.
    //initialise final variable blank here.
    public bank12(String name, int age, String adhar, String phone) {

        this.name = "";
        this.age = age;

        if(age>=16) {
            this.age=age;
        }
        else {
            this.age=10;
        }
        this.adhar = adhar;
        this.phone = phone;
    }

    //create default constructor to initialise final variable.
    public bank12() {
        this.name = "gorilla";
    }

    // Getter
    public String getName() {
        return name;
    }
```

```
35          return name;
36      }
37
38      public int getAge() {
39          return age;
40      }
41
42      public String getAdhar() {
43          return adhar;
44      }
45
46      public String getPhone() {
47          return phone;
48      }
49
50      //we can write any damn code inside getters and setters.
51      public void setAge(int age) {
52          if(age>=16) {
53              this.age=age;
54              System.out.println("hello olders");
55          }
56          else {
57              System.out.println("grow up to 16 years");
58          }
59      }
60
61      public void setAdhar(String adhar) {
62          this.adhar = adhar;
63      }
```

```
62          this.adhar = adhar;
63      }
64
65      public void setPhone(String phone) {
66          this.phone = phone;
67      }
68
69      public static void main(String[] args) {
70
71          //create object of empty constructor.
72          bank12 obj1 = new bank12();
73          System.out.println(obj1.getName());
74
75          //still we get blank for this final here.
76          bank12 obj = new bank12(obj1.getName(),10, "adhar", "phone");
77          System.out.println(obj.getName()); //blank
78          System.out.println(obj.getAge());
79          System.out.println(obj.getAdhar());
80          System.out.println(obj.getPhone());
81      }
82  }
```

```
81      }
82  }
83
84  //gorilla
85  //
86  //10
87  //adhar
88  //phone
89
90
91
```

paste bank13

bank13.java | bank14.java | bank15.java | bank16.java

```java
package com.day16;

public class bank13 {

    //when final has no value assigned.
    private final String name;
    private int age;
    private String adhar;
    private String phone;

    // Constructor.
    //initialise final variable blank here.
    public bank13(String name, int age, String adhar, String phone) {

        this.name = "";
        this.age = age;

        if(age>=16) {
            this.age=age;
        }
        else {
            this.age=10;
        }
        this.adhar = adhar;
        this.phone = phone;
    }

    //create default constructor to initialise final variable.
    public bank13() {
        this.name = "gorilla";
    }

    // Getter
    public String getName() {
        return name;
    }
```

```
35          return name;
36      }
37
38      public int getAge() {
39          return age;
40      }
41
42      public String getAdhar() {
43          return adhar;
44      }
45
46      public String getPhone() {
47          return phone;
48      }
49
50      //we can write any damn code inside getters and setters.
51      public void setAge(int age) {
52          if(age>=16) {
53              this.age=age;
54              System.out.println("hello olders");
55          }
56          else {
57              System.out.println("grow up to 16 years");
58          }
59      }
```

```
            System.out.println("grow up to 16 years");
        }
    }

    public void setAdhar(String adhar) {
        this.adhar = adhar;
    }

    public void setPhone(String phone) {
        this.phone = phone;
    }

    public static void main(String[] args) {

        //create object of empty constructor.
        bank13 obj1 = new bank13();
        System.out.println(obj1.getName());
        String s1=obj1.getName();

        //still we get blank as this constructor is initialising it to blank.
        bank13 obj = new bank13(s1,10, "adhar", "phone");
        System.out.println(obj.getName()); //blank
        System.out.println(obj.getAge());
        System.out.println(obj.getAdhar());
        System.out.println(obj.getPhone());
    }
}
```

```
        }
}

//gorilla
//
//10
//adhar
//phone
```

paste bank14

bank14.java | bank15.java | bank16.java

```java
package com.day16;

public class bank14 {

    //when final has no value assigned.
    private final String name;
    private int age;
    private String adhar;
    private String phone;

    // Constructor.

    public bank14(int age, String adhar, String phone) {

        this.name = "";
        this.age = age;

        if(age>=16) {
            this.age=age;
        }
        else {
            this.age=10;
        }
        this.adhar = adhar;
        this.phone = phone;
    }

    //create default constructor to initialise final variable.
    public bank14() {
        this.name = "gorilla";
    }

    // Getter
    public String getName() {
        return name;
    }
```

```
35          return name;
36      }
37
38      public int getAge() {
39          return age;
40      }
41
42      public String getAdhar() {
43          return adhar;
44      }
45
46      public String getPhone() {
47          return phone;
48      }
49
50      //we can write any damn code inside getters and setters.
51      public void setAge(int age) {
52          if(age>=16) {
53              this.age=age;
54              System.out.println("hello olders");
55          }
56          else {
57              System.out.println("grow up to 16 years");
58          }
59      }
```

```
58          }
59      }
60
61      public void setAdhar(String adhar) {
62          this.adhar = adhar;
63      }
64
65      public void setPhone(String phone) {
66          this.phone = phone;
67      }
68
69      public static void main(String[] args) {
70
71          //create object of empty constructor.
72          bank14 obj1 = new bank14();
73          System.out.println(obj1.getName());
74          String s1=obj1.getName();
75          System.out.println(s1);
```

```
74          String s1=obj1.getName();
75          System.out.println(s1);
76
77          //still we get blank as this constructor is initialising it to blank.
78          bank14 obj = new bank14(10, "adhar", "phone");
79          System.out.println(obj.getName()); //blank
80          System.out.println(obj.getAge());
81          System.out.println(obj.getAdhar());
82          System.out.println(obj.getPhone());
83      }
84  }
85
86  //gorilla
87  //gorilla
88  //
89  //10
90  //adhar
91  //phone
92
93
94
95
96
```

paste bank15

bank15.java bank16.java

```java
package com.day16;

public class bank15 {

    //when final has no value assigned.
    private final String name;
    private int age;
    private String adhar;
    private String phone;

    // Constructor.
    //give value to final variable here also.
    public bank15(int age, String adhar, String phone) {

        this.name = "lion king";
        this.age = age;

        if(age>=16) {
            this.age=age;
        }
        else {
            this.age=10;
        }
        this.adhar = adhar;
        this.phone = phone;
    }

    //create default constructor to initialise final variable.
    public bank15() {
        this.name = "gorilla";
    }

    // Getter
    public String getName() {
        return name;
    }
```

```
35          return name;
36      }
37
38      public int getAge() {
39          return age;
40      }
41
42      public String getAdhar() {
43          return adhar;
44      }
45
46      public String getPhone() {
47          return phone;
48      }
49
50      //we can write any damn code inside getters and setters.
51      public void setAge(int age) {
52          if(age>=16) {
53              this.age=age;
54              System.out.println("hello olders");
55          }
56          else {
57              System.out.println("grow up to 16 years");
58          }
59      }
60
61      public void setAdhar(String adhar) {
62          this.adhar = adhar;
63      }
```

```
62          this.adhar = adhar;
63      }
64
65      public void setPhone(String phone) {
66          this.phone = phone;
67      }
68
69      public static void main(String[] args) {
70
71          //create object of empty constructor.
72          bank15 obj1 = new bank15();
73          System.out.println(obj1.getName());
74          String s1=obj1.getName();
75          System.out.println(s1);
76
77
78          bank15 obj = new bank15(10, "adhar", "phone");
79          System.out.println(obj.getName());
80          System.out.println(obj.getAge());
81          System.out.println(obj.getAdhar());
82          System.out.println(obj.getPhone());
83      }
84  }
```

```
83      }
84  }
85
86  //gorilla
87  //gorilla
88  //lion king
89  //10
90  //adhar
91  //phone
92
93
94
95
96
97
```

paste bank16

bank16.java

```java
package com.day16;

public class bank16 {

    //when final has no value assigned.
    private final String name;
    private int age;
    private String adhar;
    private String phone;

    // Constructor.
    //give value to final variable here also.
    //give as parameter also.
    public bank16(String name, int age, String adhar, String phone) {

        this.name = "lion king";
        this.age = age;

        if(age>=16) {
            this.age=age;
        }
        else {
            this.age=10;
        }
        this.adhar = adhar;
        this.phone = phone;
    }

    //create default constructor to initialise final variable.
    public bank16() {
        this.name = "gorilla";
    }
```

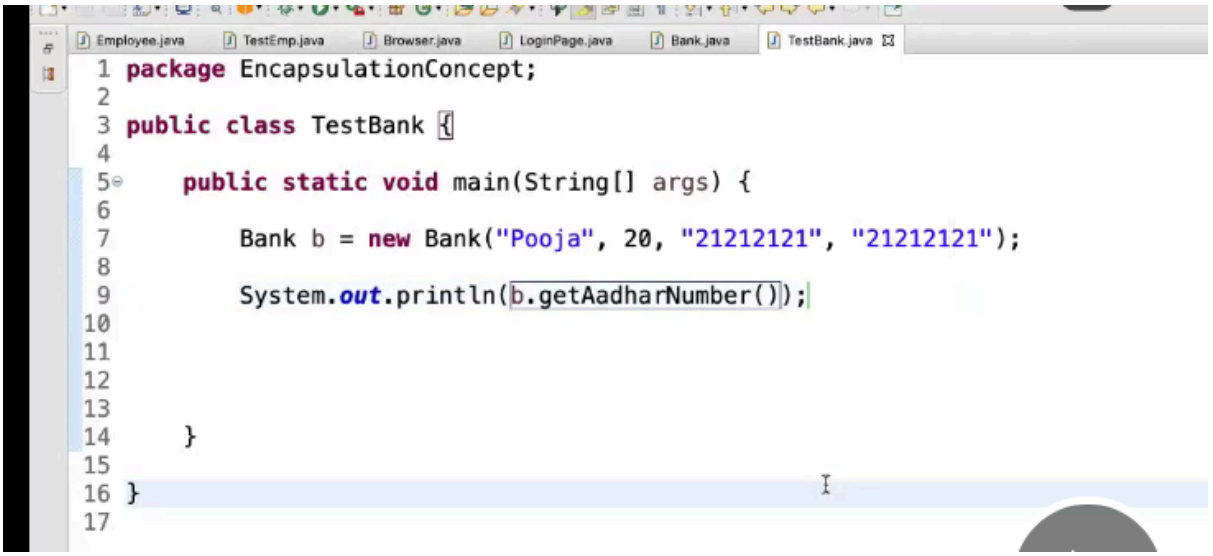
```
31          this.name = "gorilla";
32      }
33
34      // Getter
35      public String getName() {
36          return name;
37      }
38
39      public int getAge() {
40          return age;
41      }
42
43      public String getAdhar() {
44          return adhar;
45      }
46
47      public String getPhone() {
48          return phone;
49      }
50
51      //we can write any damn code inside getters and setters.
52      public void setAge(int age) {
53          if(age>=16) {
54              this.age=age;
55              System.out.println("hello olders");
56          }
57          else {
58              System.out.println("grow up to 16 years");
59          }
60      }
```

```
            System.out.println( grow up to 18 years );
        }
    }

    public void setAdhar(String adhar) {
        this.adhar = adhar;
    }

    public void setPhone(String phone) {
        this.phone = phone;
    }

    public static void main(String[] args) {

        //create object of empty constructor.
        bank16 obj1 = new bank16();
        System.out.println(obj1.getName());
        String s1=obj1.getName();
        System.out.println(s1);

        //cant change final value from object creation step.
        bank16 obj = new bank16("vince macmahon", 10, "adhar", "phone");
        System.out.println(obj.getName());
        System.out.println(obj.getAge());
        System.out.println(obj.getAdhar());
        System.out.println(obj.getPhone());
    }
}
```

```
        }
    }

    //gorilla
    //gorilla
    //lion king
    //10
    //adhar
    //phone
```

Same condition can be set in constructor also-

```
 9
10    public Bank(String name, int age, String aadharNumber, String phone) {
11        this.name = name;
12
13        if (age >= 15) {
14            this.age = age;
15        }
16
17        this.aadharNumber = aadharNumber;
18        this.phone = phone;
19    }
```



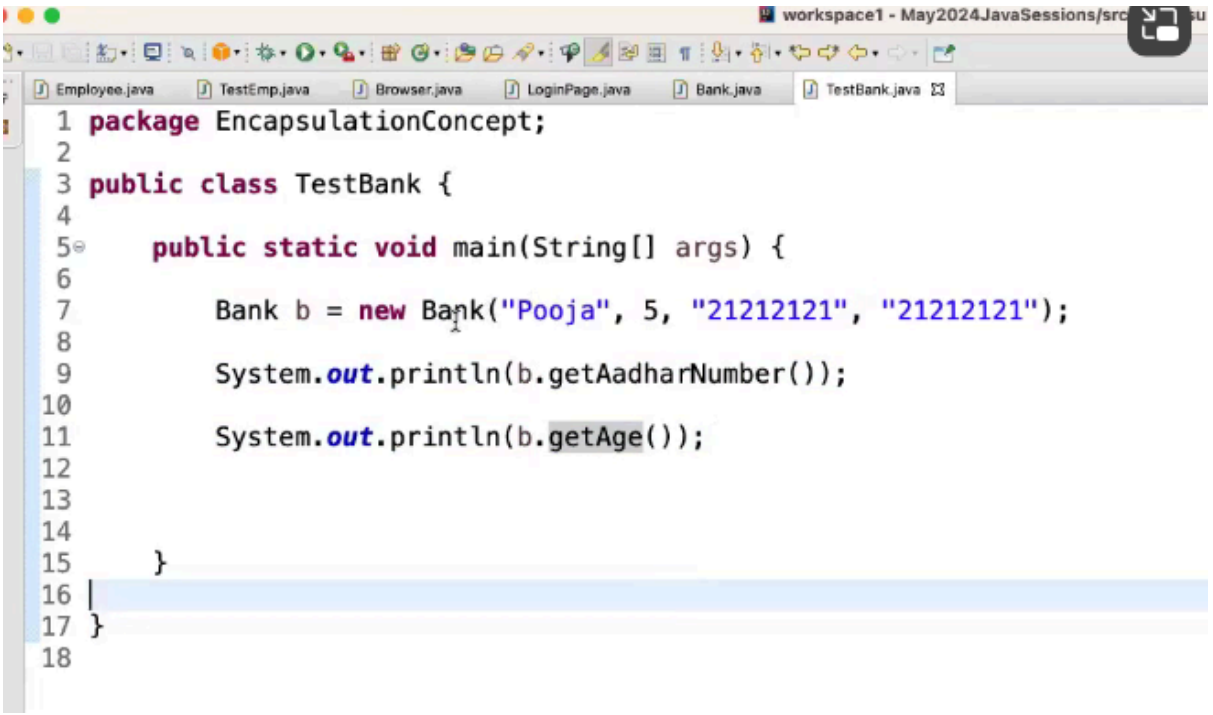
Employee.java | TestEmp.java | Browser.java | LoginPage.java | Bank.java | TestBank.java

```
 1 package EncapsulationConcept;
 2
 3 public class TestBank {
 4
 5     public static void main(String[] args) {
 6
 7         Bank b = new Bank("Pooja", 20, "21212121", "21212121");
 8
 9         System.out.println(b.getAadharNumber());
10
11
12
13
14     }
15
16 }
17
```

Console | Prob

<terminated> TestBank [J

```
21212121
```



paste bank17 and test bank17

bank17.java

```java
package com.day16;

public class bank17 {

    private String name;
    private int age;
    private String adhar;
    private String phone;

    // Constructor
    public bank17(String name, int age, String adhar, String phone) {

        this.name=name;

        this.age = age;

        //same setter condition can be set in constructor also.
        if(age>=16) {
            this.age=age;
        }
        else {
            this.age=10;
        }
        this.adhar = adhar;
        this.phone = phone;
    }

    public bank17() {
        // TODO Auto-generated constructor stub
    }

    // Getter
    public String getName() {
        return name;
    }
```

```
34          return name;
35      }
36
37      public int getAge() {
38          return age;
39      }
40
41      public String getAdhar() {
42          return adhar;
43      }
44
45      public String getPhone() {
46          return phone;
47      }
48
49      // Setter
50      public void setName(String name) {
51          this.name = name;
52      }
53
54      //we can write any damn code inside getters and setters.
55  //    public void setAge(int age) {
56  //      if(age>=16) {
57  //          this.age=age;
58  //          System.out.println("hello olders");
59  //      }
60  //      else {
61  //          System.out.println("grow up to 16 years");
62  //      }
63  //    }
```

```
62  //        }
63  //    }
64
65      public void setAdhar(String adhar) {
66          this.adhar = adhar;
67      }
68
69      public void setPhone(String phone) {
70          this.phone = phone;
71      }
72  }
73
```

testbank17.java

```
1  package com.day16;
2
3  public class testbank17 {
4
5      public static void main(String[] args) {
6
7          bank17 obj = new bank17("manking", 20, "adhar", "phone");
8          System.out.println(obj.getName());
9          System.out.println(obj.getAge());
10         System.out.println(obj.getAdhar());
11         System.out.println(obj.getPhone());
12
13         //create another object with different values.
14
15         bank17 obj1 = new bank17("vince macmahon", 10, "adhar", "phone");
16         System.out.println(obj1.getName());
17         System.out.println(obj1.getAge());
18         System.out.println(obj1.getAdhar());
19         System.out.println(obj1.getPhone());
20     }
21
22 }
23
24
25 //manking
26 //20
27 //adhar
28 //phone
29
30 //vince macmahon
31 //10
32 //adhar
33 //phone
34
```

Actual business logic looks like this-

See the difference between the ones written in setter and constructor and here.

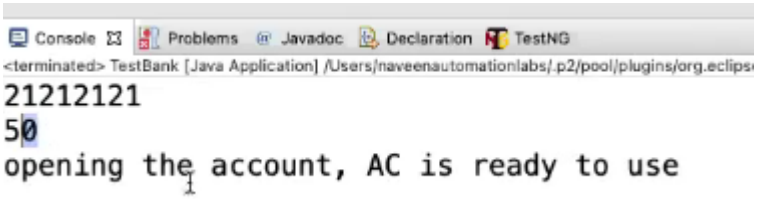
```
57      //buss logic method:
58      public void openAccount() {
59          if(this.age>=15) {
60              System.out.println("opening the account");
61              //...
62              //..
63          }
64          else {
65              System.out.println("age is less than 15");
66          }
67      }
68
69  }
```

```
1  package EncapsulationConcept;
2
3  public class TestBank {
4
5      public static void main(String[] args) {
6
7          Bank b = new Bank("Pooja", 5, "21212121", "21212121");
8
9          System.out.println(b.getAadharNumber());
10
11         System.out.println(b.getAge());
12
13         b.openAccount();
14
15
```

```
Console  Problems  Javadoc  Decl
<terminated> TestBank [Java Application] /Users/navee
21212121
0
age is less than 15
```

Valid age-

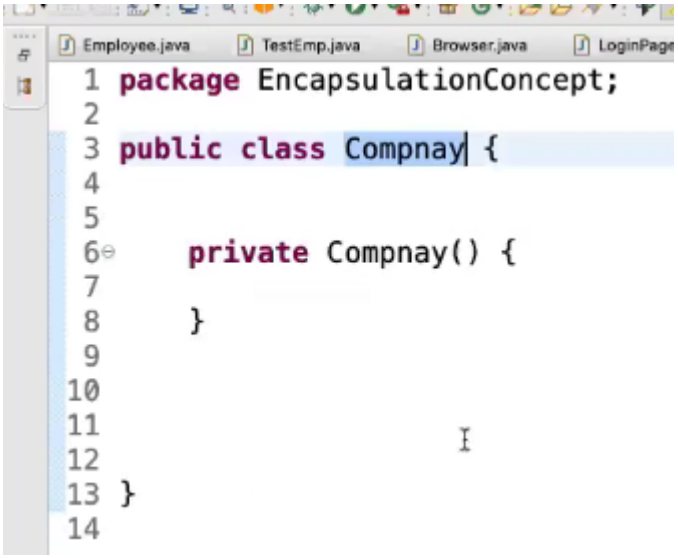
```
5      public static void main(String[] args) {
6
7          Bank b = new Bank("Pooja", 50, "21212121", "21212121");
8
```



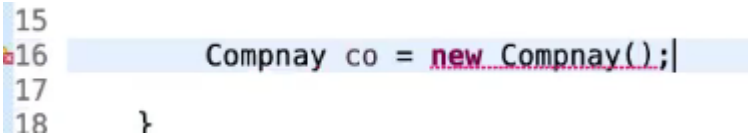
Private constructor-

Allowed.

But we cannot create object of class.



Error.



//The constructor privateconst2() is not visible

when we try to access from another class.

paste privateconst1

privateconst1.java

```java
package com.day16;

public class privateconst1 {

    private privateconst1() {
        System.out.println("this is private constructor.");
    }

    //in same class private works.
    public static void main(String[] args) {
        privateconst1 obj = new privateconst1();
    }
}

//this is private constructor.
```

We have methods but constructor is private-

Employee.java | TestEmp.java | Browser.java | LoginPage.java

```java
package EncapsulationConcept;

public class Compnay {


    private Compnay() {

    }


    public static void getRevenue() {
        System.out.println("get rev");
    }


}
```

```java
        Compnay.getRevenue();
```

We create the method as static so can be accessed with class name.

Constructor for dealing with objects not class names.

get rev

paste company1 and testcompany1

company1.java

```java
package com.day16;

public class company1 {

    //private constructor.
    private company1() {
        System.out.println("private constructor.");
    }

    //how to access methods and variables which are inside private constructor class.
    //create everything as static so we dont have to create objects.
    public static int i=10;

    public static void getdetails(String name) {
        System.out.println("hello " + name);
    }

}
```

testcompany1.java

```java
package com.day16;

public class testcompany1 {

    public static void main(String[] args) {

        company1.getdetails("karan");
        int v=company1.i;
        System.out.println(v);
    }

}


//hello karan
//10
```

paste company2

company2.java

```java
package com.day16;

public class company2 {

    //private constructor.
    private company2() {
        System.out.println("private constructor.");
    }

    //how to access methods and variables which are inside private constructor class.
    public static int i=10;

    public static void getdetails(String name) {
        System.out.println("hello " + name);
    }

    public static void main(String[] args) {

        company2.getdetails("karan");
        int v=company2.i;
        System.out.println(v);
    }

}

//hello karan
//10

```

paste company3, testcompany3

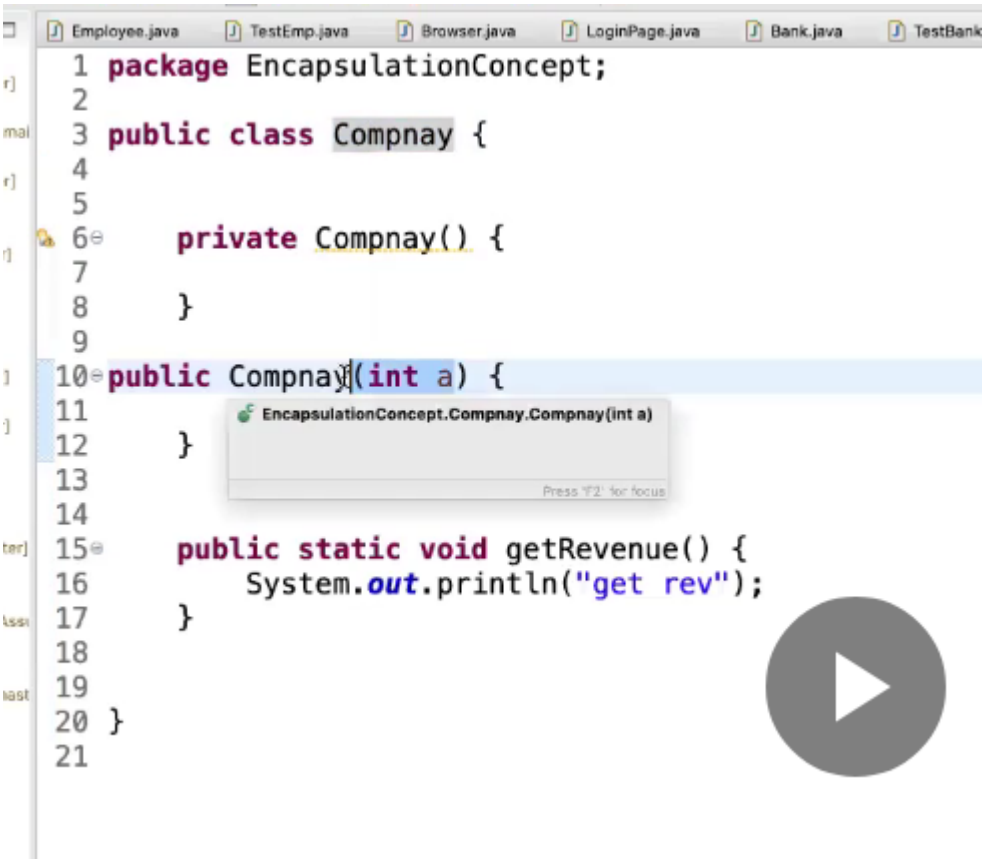
company3.java

```java
package com.day16;

public class company3 {

    //private constructor.
    private company3() {
        System.out.println("private constructor.");
    }

    //how to access methods and variables which are inside private constructor class.
    public static int i=10;

    public static void getdetails(String name) {
        System.out.println("hello " + name);
    }

}
```

testcompany3.java

```java
package com.day16;

public class testcompany3 {

    public static void main(String[] args) {

        //cant create object of private constructor class.
        company3 c1=new company3(); //The constructor company3() is not visible
        c1.getdetails("karan");
        int v1=c1.i;
        System.out.println(v1);
    }

}
```

We can have public and private constructor in class-



paste company4 and test company4-

company4.java

```java
package com.day16;

public class company4 {

    //private constructor.
    private company4(int a) {
        System.out.println("private constructor." + a);
    }

    //public constructor also present.
    public company4(String c) {
        System.out.println("this is public constructor" + c);
    }

    //how to access methods and variables which are inside private constructor class.
    //create everything as static so we dont have to create objects.
    public static int i=10;

    public static void getdetails(String name) {
        System.out.println("hello " + name);
    }

}
```

testcompany4.java

```java
package com.day16;

public class testcompany4 {

    public static void main(String[] args) {

        company4 c2=new company4("lion");
        c2.getdetails("karan");
        //warning - The static method getdetails(String) from the type company4 should be accessed in a static way
        int v2=c2.i;
        //warning - The static field company4.i should be accessed in a static way
        System.out.println(v2);

        //change the static value.
        int s=company4.i=20;
        System.out.println(s);

        company4.getdetails("giraffe");
    }

}

//this is public constructorlion
//hello karan
//10
//20
//hello giraffe
```

paste company5-

company5.java

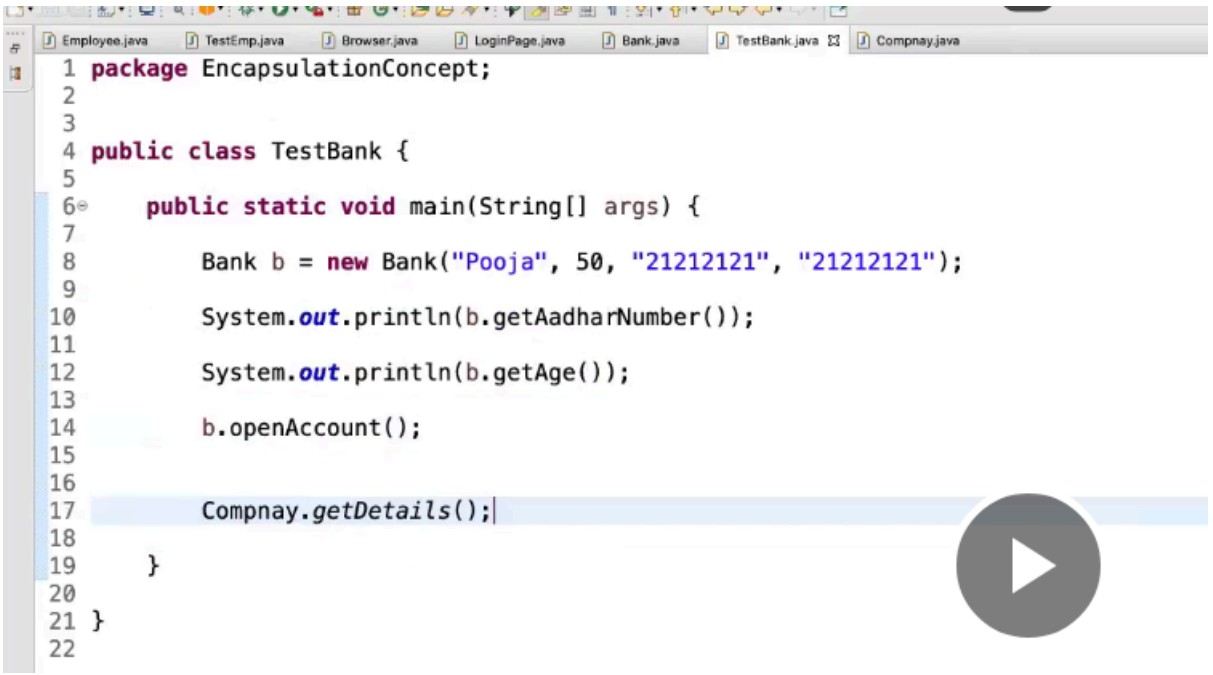
```java
package com.day16;

public class company5 {

    //private constructor.
    private company5(int a) {
        System.out.println("private constructor." + a);
    }

    //public constructor also present.
    //Duplicate method company5(int) in type company5
    //we cannot have two constructor with same number of parameters.
    public company5(int c) {
        System.out.println("this is public constructor" + c);
    }

    public static int i=10;

    public static void getdetails(String name) {
        System.out.println("hello " + name);
    }

}
```

If method is also private-

Employee.java TestEmp.java Browser.java LoginPage.java Bank.java

```java
package EncapsulationConcept;

public class Compnay {

    private Compnay() {

    }

    private static void getRevenue() {
        System.out.println("get rev");
    }

    public static void getDetails() {
        getRevenue();
    }

}
```

Create public method and encapsulate private inside it.



paste company6 and testcompany6-

company6.java

```java
package com.day16;

public class company6 {

    //private constructor.
    private company6(int a) {
        System.out.println("private constructor." + a);
    }

    //private method.
    private void getdetails(String name) {
        System.out.println("get details method." + name);
    }

    private void getaddress() {
        System.out.println("getaddress method");
    }

    private static void getphonenumber() {
        System.out.println("getphonenumber method");
    }

    private static void getlocation(String location) {
        System.out.println("getlocation method " +location);
    }

//one public method to call all private methods.
    //encapsulation concept.
    public static void getalldetails(String name, String location) {
        //how to access private constructor inside public static method.
        company6 c1=new company6(10);
        c1.getdetails(name);
        c1.getaddress();
        getphonenumber();
        getlocation(location);
    }
```

```java
        getlocation(location);
    }

}
```

testcompany6.java

```java
package com.day16;

public class testcompany6 {

    public static void main(String[] args) {

        company6.getalldetails("karan", "giraffe");
    }

}

//private constructor.10
//get details method.karan
//getaddress method
//getphonenumber method
//getlocation method giraffe
```

Static variable can be accessed inside constructor-

```java
    // private class variables
    private String name;
    private int age;
    private double salary;
    private boolean isPerm;
    static int wheels = 4;

    //public const.. is also like setter
    public Employee(String name, int age, double salary, boolean isPerm) {
        this.name = name;
        this.age = age;
        this.salary = salary;
        this.isPerm = isPerm;
        Employee.wheels = 5;
    }
```

But mandatory to give some value to it, else error.

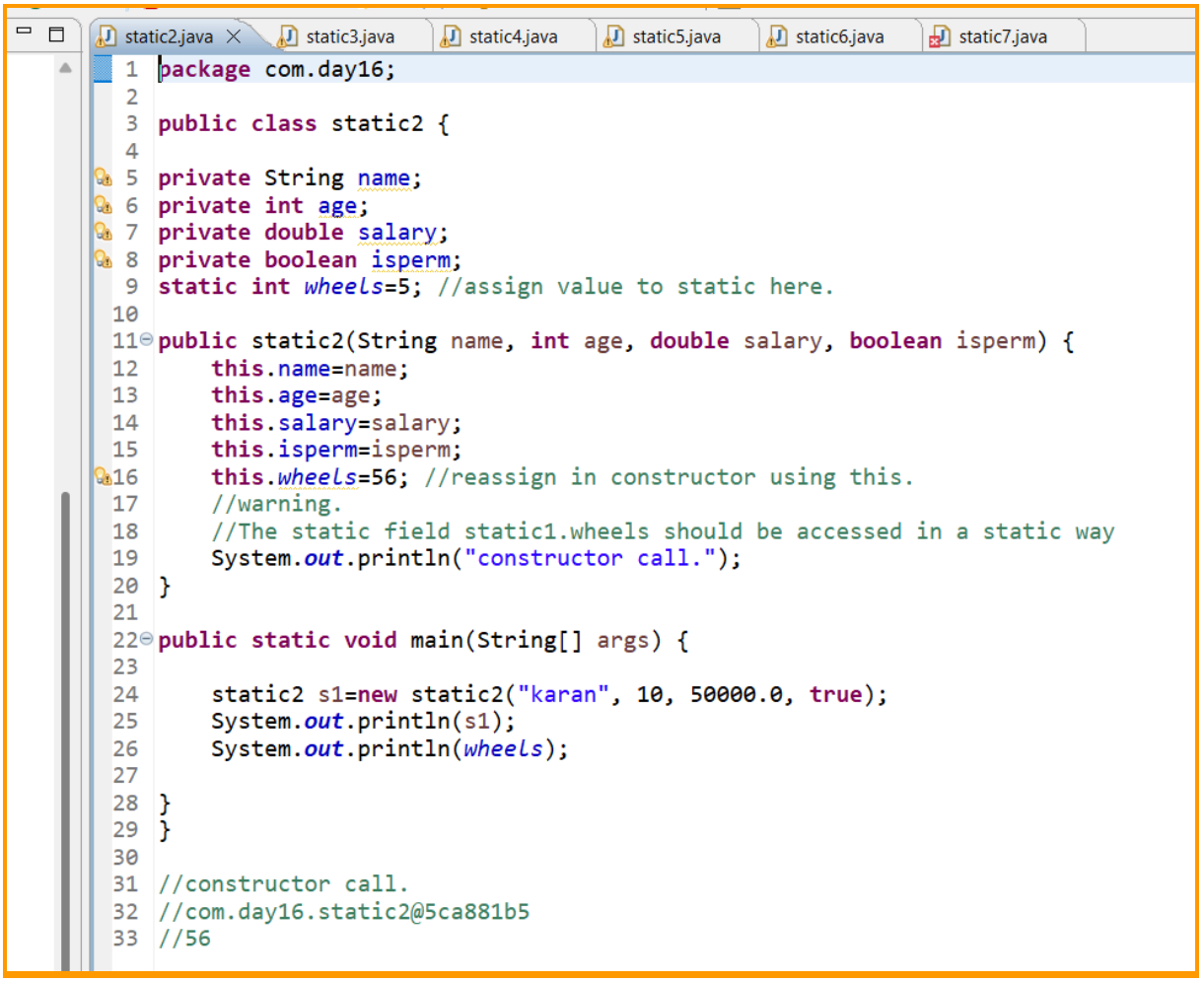
//Syntax error, insert "VariableDeclarators" to complete LocalVariableDeclaration

paste static 1 to 7-

different combinations tried with static.

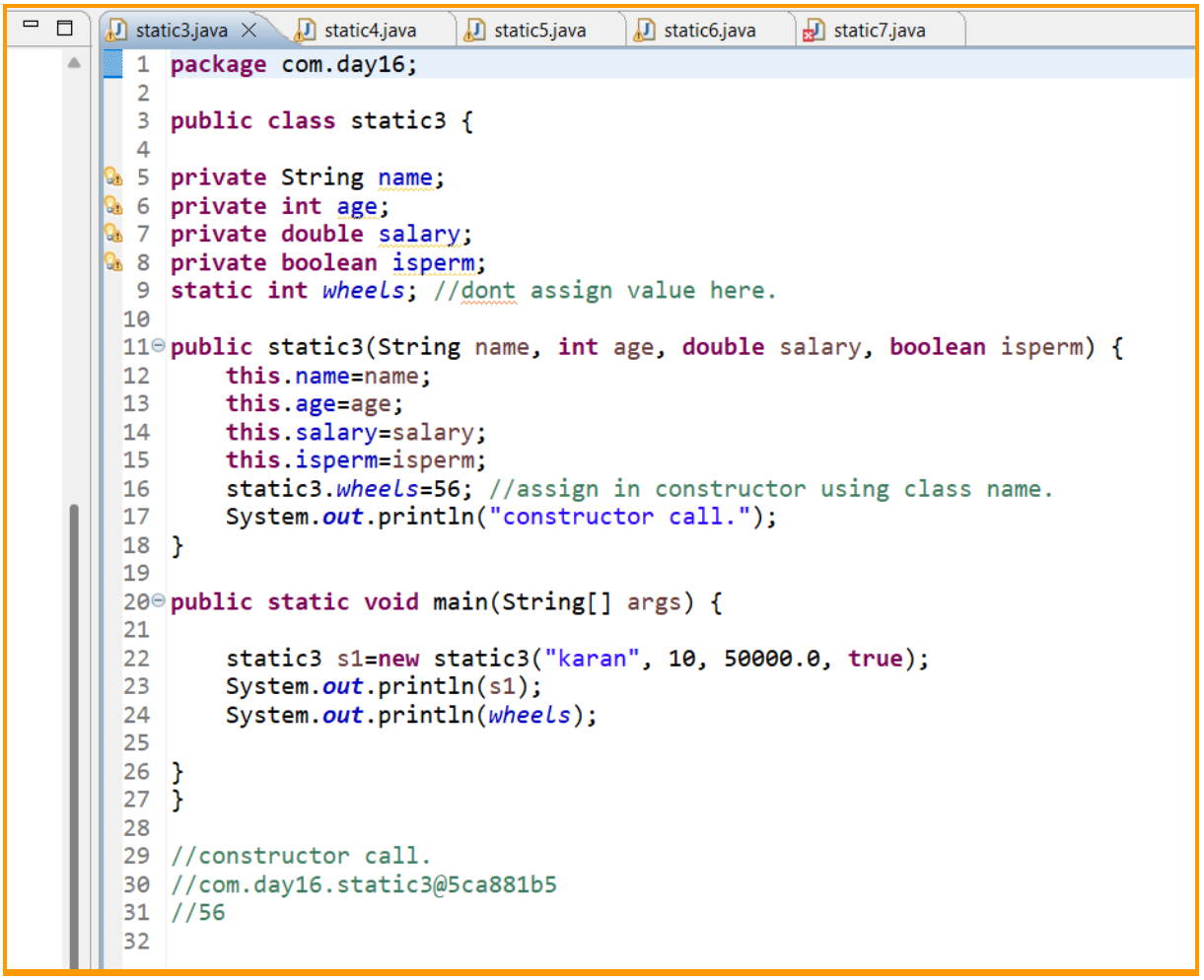
static1.java | static2.java | static3.java | static4.java | static5.java | static6.java | static7

```java
package com.day16;

public class static1 {

private String name;
private int age;
private double salary;
private boolean isperm;
static int wheels; //we dont assign values to static here.

public static1(String name, int age, double salary, boolean isperm) {
    this.name=name;
    this.age=age;
    this.salary=salary;
    this.isperm=isperm;
    this.wheels=56; //we assign values to static here using this.
    //warning.
    //The static field static1.wheels should be accessed in a static way
    System.out.println("constructor call.");
}

public static void main(String[] args) {

    static1 s1=new static1("karan", 10, 50000.0, true);
    System.out.println(s1);
    System.out.println(wheels);

}
}

//constructor call.
//com.day16.static1@5ca881b5
//56
```



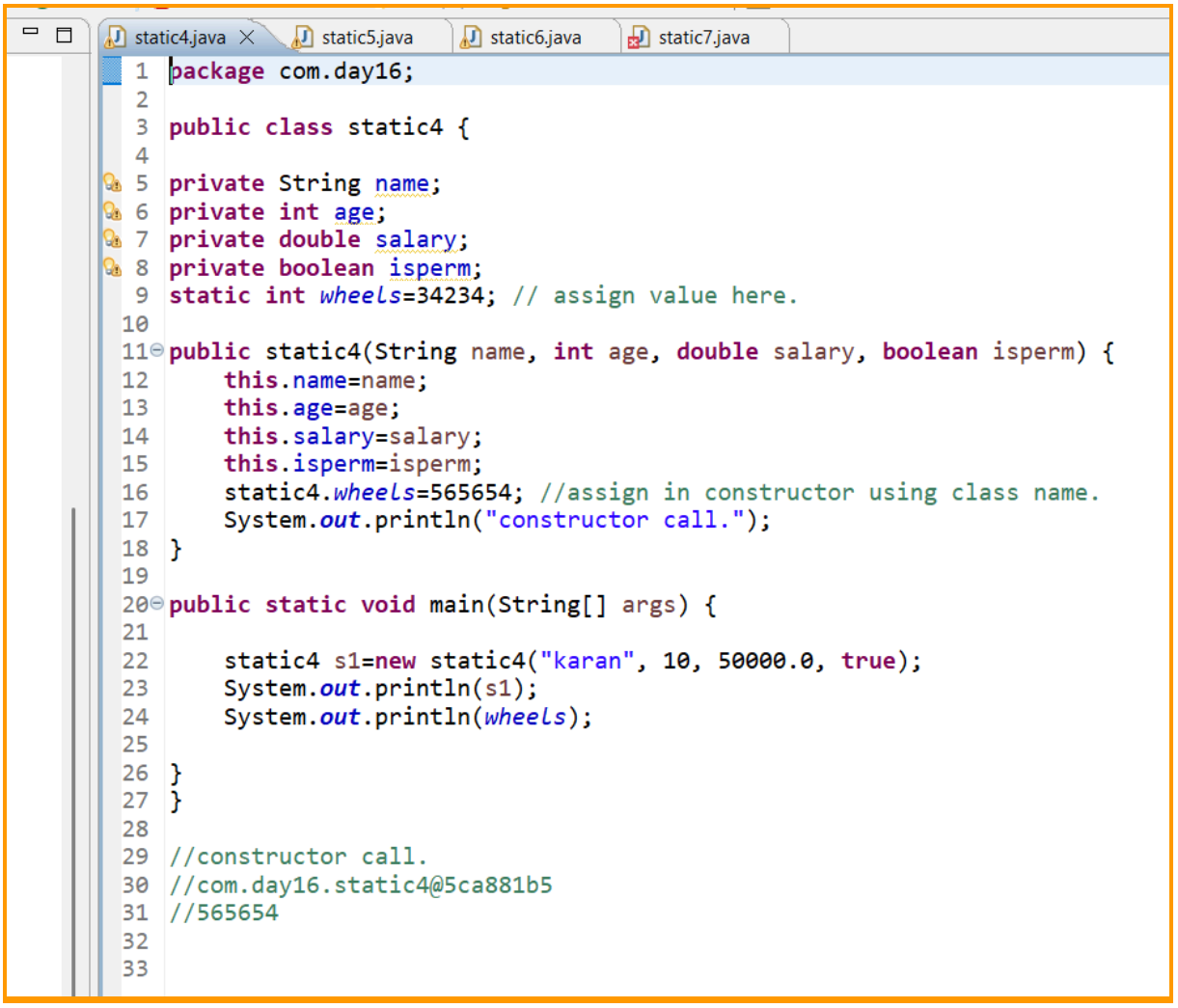
static2.java | static3.java | static4.java | static5.java | static6.java | static7.java

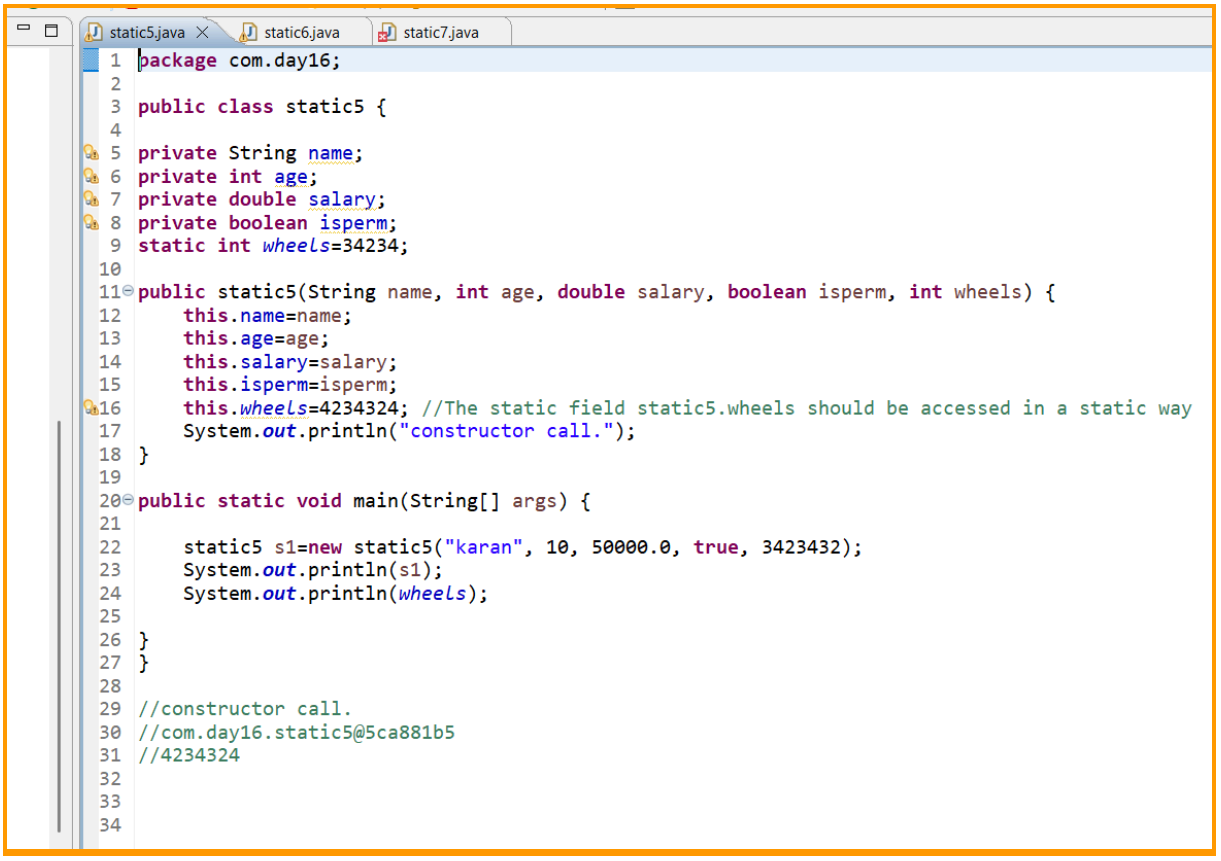
```java
package com.day16;

public class static2 {

private String name;
private int age;
private double salary;
private boolean isperm;
static int wheels=5; //assign value to static here.

public static2(String name, int age, double salary, boolean isperm) {
    this.name=name;
    this.age=age;
    this.salary=salary;
    this.isperm=isperm;
    this.wheels=56; //reassign in constructor using this.
    //warning.
    //The static field static1.wheels should be accessed in a static way
    System.out.println("constructor call.");
}

public static void main(String[] args) {

    static2 s1=new static2("karan", 10, 50000.0, true);
    System.out.println(s1);
    System.out.println(wheels);

}
}

//constructor call.
//com.day16.static2@5ca881b5
//56
```



static3.java | static4.java | static5.java | static6.java | static7.java

```java
package com.day16;

public class static3 {

private String name;
private int age;
private double salary;
private boolean isperm;
static int wheels; //dont assign value here.

public static3(String name, int age, double salary, boolean isperm) {
    this.name=name;
    this.age=age;
    this.salary=salary;
    this.isperm=isperm;
    static3.wheels=56; //assign in constructor using class name.
    System.out.println("constructor call.");
}

public static void main(String[] args) {

    static3 s1=new static3("karan", 10, 50000.0, true);
    System.out.println(s1);
    System.out.println(wheels);

}
}

//constructor call.
//com.day16.static3@5ca881b5
//56

```


static4.java
static5.java
static6.java
static7.java
1 package com.day16;
2
3 public class static4 {
4
5 private String name;
6 private int age;
7 private double salary;
8 private boolean isperm;
9 static int wheels=34234; // assign value here.
10
11 public static4(String name, int age, double salary, boolean isperm) {
12 this.name=name;
13 this.age=age;
14 this.salary=salary;
15 this.isperm=isperm;
16 static4.wheels=565654; //assign in constructor using class name.
17 System.out.println("constructor call.");
18 }
19
20 public static void main(String[] args) {
21
22 static4 s1=new static4("karan", 10, 50000.0, true);
23 System.out.println(s1);
24 System.out.println(wheels);
25
26 }
27 }
28
29 //constructor call.
30 //com.day16.static4@5ca881b5
31 //565654
32
33



static5.java | static6.java | static7.java

```java
package com.day16;

public class static5 {

private String name;
private int age;
private double salary;
private boolean isperm;
static int wheels=34234;

public static5(String name, int age, double salary, boolean isperm, int wheels) {
    this.name=name;
    this.age=age;
    this.salary=salary;
    this.isperm=isperm;
    this.wheels=4234324; //The static field static5.wheels should be accessed in a static way
    System.out.println("constructor call.");
}

public static void main(String[] args) {

    static5 s1=new static5("karan", 10, 50000.0, true, 3423432);
    System.out.println(s1);
    System.out.println(wheels);

}
}

//constructor call.
//com.day16.static5@5ca881b5
//4234324
```

static6.java

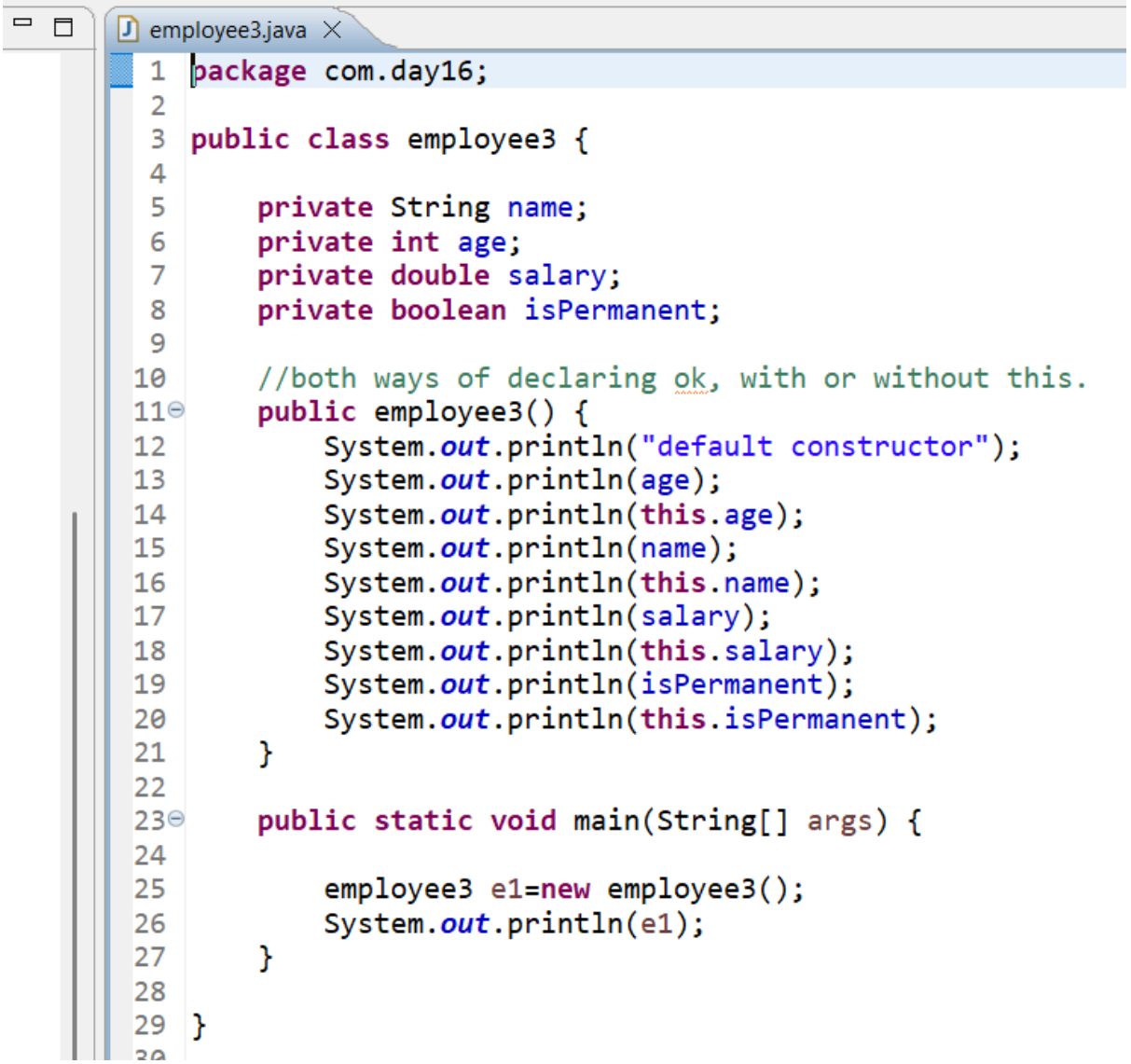
```java
package com.day16;

public class static6 {

private String name;
private int age;
private double salary;
private boolean isperm;
static int wheels=34234;

//what ever is in constructor that is the last value of static so it gets printed.

public static6(String name, int age, double salary, boolean isperm, int wheels) {
    this.name=name;
    this.age=age;
    this.salary=salary;
    this.isperm=isperm;
    static6.wheels=4234324;
    System.out.println("constructor call.");
}

public static void main(String[] args) {

    static6 s1=new static6("karan", 10, 50000.0, true, 3423432);
    System.out.println(s1);
    System.out.println(wheels);

}
}

//constructor call.
//com.day16.static6@5ca881b5
//4234324
```

static7.java

```java
package com.day16;

public class static7 {

private String name;
private int age;
private double salary;
private boolean isperm;
static int wheels=34234;

public static7(String name, int age, double salary, boolean isperm {
    this.name=name;
    this.age=age;
    this.salary=salary;
    this.isperm=isperm;
    static7.wheels; //Syntax error, insert "VariableDeclarators" to complete LocalVariableDeclaration
    System.out.println("constructor call.");
}

public static void main(String[] args) {

    static7 s1=new static7("karan", 10, 50000.0, true);
    System.out.println(s1);
    System.out.println(wheels);

}
}
```

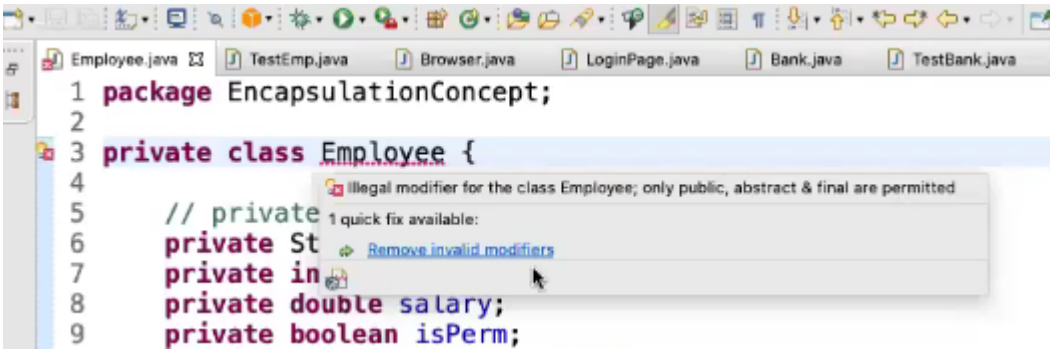
To get the default values use default constructor-1.37.52

paste employee3



```
28
29  }
30
31  //default constructor
32  //0
33  //0
34  //null
35  //null
36  //0.0
37  //0.0
38  //false
39  //false
40  //com.day16.employee3@5ca881b5
41
42
```

Private class not allowed-



for static variable no need to assign values.

default value will be taken for that variable's data type.

paste static8-

static8.java

```java
package com.day16;

public class static8 {

    private String name;
    private int age;
    private double salary;
    private boolean isPermanent;
    private static int wheels;
    private static String type;
    private static boolean flag;
    private static double number;

    //both ways of declaring ok, with or without this.
    //print the static variables default value.
    public static8() {
        System.out.println("default constructor");
        System.out.println(age);
        System.out.println(this.age);
        System.out.println(name);
        System.out.println(this.name);
        System.out.println(salary);
        System.out.println(this.salary);
        System.out.println(isPermanent);
        System.out.println(this.isPermanent);
        System.out.println(wheels);
        System.out.println(this.wheels);//The static field static8.wheels should be accessed in a static way
        System.out.println(type);
        System.out.println(this.type);//The static field static8.type should be accessed in a static way
        System.out.println(flag);
        System.out.println(this.flag);//The static field static8.flag should be accessed in a static way
        System.out.println(number);
        System.out.println(this.number);//The static field static8.number should be accessed in a static way
    }
```

```java
        System.out.println(this.number);//The static field static8.number should be accessed in a static way
    }

    public static void main(String[] args) {

        static8 e1=new static8();
        System.out.println(e1);
        System.out.println(wheels);
        System.out.println(type);
        System.out.println(flag);
        System.out.println(number);
    }
```

```
43          System.out.println(number);
44      }
45
46  }
47
48  //default constructor
49  //0
50  //0
51  //null
52  //null
53  //0.0
54  //0.0
55  //false
56  //false
57  //0
58  //0
59  //null
60  //null
61  //false
62  //false
63  //0.0
64  //0.0
65  //com.day16.static8@5ca881b5
66  //0
67  //null
68  //false
69  //0.0
70
```