TECHPROED

PROFESSIONAL TECHNOLOGY EDUCATION

WELCOME TO TECHPROED JAVA TUTORIAL

Testi baslatmak icin asagidaki adımları takip ediniz

Go to www.socrative.com

Click on Login

Click on Student Login

***Room Name*: A L P T E K I N 3 5 2 3**

Kayıtta kullandığınız ismi tam olarak yazınız

Time: 11 Minutes

TECHPROED
PROFESSIONAL TECHNOLOGY EDUCATION

Inheritance'da Constructors Çağırma Kuralları

1) parent constructor önce çalışır

```
public class Animal {
    public Animal() {
        System.out.println("Parent constructor önce çalışır");
    }
}


public class Zebra extends Animal {
    public Zebra() {
        System.out.println("Child constructor sonra çalışır");
    }
}
```

```
public class Animal {
    public Animal() {
        System.out.println("Parent constructor önce çalışır");
}


public class Zebra extends Animal {
    public Zebra() {
        super();
        System.out.println("Child constructor sonra çalışır");
    }
}
```

Note: Parent Class'dan Default constructor çağırmak icin super() keyword kullanılsa da olur kullanılmasa da olur.

2) Eğer uygun parent constructor yoksa, Compile Time Error alırsınız.

```
public class Animal {
    public Animal(int age) {
        System.out.println("Parent constructor with parameter");
    }
}



public class Zebra extends Animal {
    public Zebra() {
        super();
        System.out.println("Child constructor");
    }
}
```

Does not compile

```
public class Animal {
    public Animal(int age) {
        System.out.println("Parent constructor with parameter");
}



public class Zebra extends Animal {
    public Zebra() {
        super(5);
        System.out.println("Child constructor");
    }
}
```

Compiles

3) super() keyword parent constructor, this() keyword in class constructor çağırmak icin kullanılır

Output nedir?

```
public class Animal{

    public Animal(){
        System.out.println("Parent Constructor");
    }

}



class Mammal extends Animal{

    public Mammal(int age){
        super();
        System.out.println("Child Constructor called by this()");
    }

    public Mammal(){
        this( age: 11);
        System.out.println("Child Constructor");
    }

    public static void main(String args[]){
        Mammal mammal = new Mammal();
    }
}
```

TECHPROED
PROFESSIONAL TECHNOLOGY EDUCATION

3) super() ve this() keywords ilk satırda yer almalıdır ve sadece bir kere kullanılmalıdır.

```
public class Animal {
    public Animal(int age) {
        System.out.println("Parent constructor runs first");
    }
}


public class Zebra extends Animal {
    public Zebra() {
        System.out.println("Child constructor runs at the end");
        super();
    }
}
```

Compile Time Error

```
public class Animal {
    public Animal() {
        System.out.println("Parent constructor runs first");
    }
}


public class Zebra extends Animal {
    public Zebra() {
        super();
        System.out.println("Child constructor runs at the end");
    }
}
```

Compile olur

Note: Solda super() keyword ilk satırda değil Compile Time Error verir.

Note: super() ve this() keywords aynı constructor'ın icinde aynı zamanda var olamaz.

Aşağıdaki 3 class tamamıyla aynıdır !!!

1)
```
public class Donkey {
}
```

Bu class'in constructor'i yok, fakat Java default constructor oluşturur
```
public Donkey() {
 }
```

2)
```
public class Donkey {
    public Donkey() {
    }
}
```

3)
```
public class Donkey {
    public Donkey() {
        super();
    }
}
```

super(); Object class'dan default constructor çağırır.
```
public Donkey() {
}
```

Calling Inherited Class Members

Note: “super” keyword parent class’dan variable çağırmak icin kullanılır.
“this” keyword iced bulunulan class’dan variable çağırmak icin kullanılır.

Note: Esasında “this” keyword parent class’dan variable çağırmak için de kullanılabilir; fakat tavsiye edilmez. Çünkü, child ve parent class’larda aynı isimli iki variable varsa, “this” parent class’dan variable çağıramaz.

```java
public class Animal {

    protected int num1 = 10;
    public String name = "Ali";
    private int num2 = 11;

    public int getNum2(){
        return num2;
    }

    public void setNum2(int num2){
        this.num2 = num2;
    }
}
```

```java
public class Mammal extends Animal {

    protected int num3 = 12;
    public String name2 = "Veli";
    private int num4 = 13;

    public Mammal(){
        System.out.println(this.num1);
        System.out.println(super.num1);

        System.out.println(this.getNum2());
        System.out.println(super.getNum2());

        this.setNum2(23);
        System.out.println(this.getNum2());
        System.out.println(super.getNum2());

        super.setNum2(33);
        System.out.println(this.getNum2());
        System.out.println(super.getNum2());

        System.out.println(this.num3);
        System.out.println(this.num4);

        System.out.println(this.name);
        System.out.println(super.name);

        System.out.println(this.name2);
    }

    public static void main(String args[]){
        Mammal mammal = new Mammal();
    }
}
```

Output nedir?

```java
public class Animal{

    public Animal(){
        System.out.println("Parent Constructor");
    }

}



class Mammal extends Animal{

    public Mammal(int age){
        super();
        System.out.println("Child Constructor called by this()");
    }

    public Mammal(){
        this( age: 11);
        System.out.println("Child Constructor");
    }

    public static void main(String args[]){
        Mammal mammal = new Mammal( age: 15);
    }
}
```

Output nedir?

Output of following Java Program?

```
class Base {
    public void show() {
        System.out.println("Base::show() called");
    }
}

class Derived extends Base {
    public void show() {
        System.out.println("Derived::show() called");
    }
}

public class Main {
    public static void main(String[] args) {
        Base b = new Derived();;
        b.show();
    }
}
```

Output nedir?

```
class Test01 {

    public void myMethod(int i, int j) {
        System.out.println(i + j);
    }
}

class Test extends Test01 {

    public void myMethod(int i, int j) {
        System.out.println(i * j);
    }
}

class Main {

    public static void main(String args[]){
         Test obj = new Test();
         obj.myMethod(3, 4);
     }
}
```

Output nedir?

```
class Test01 {

    public void myMethod(int i, int j) {
        System.out.println(i + j);
    }

    public void yourMethod(int i, int j) {
        System.out.println(i * j);
    }
}

class Test extends Test01 {

    public void myMethod(int i, int j) {
        System.out.println(i - j);
    }
}

class Main {

    public static void main(String args[]){
         Test obj = new Test();
         obj.myMethod(12, 4);
         obj.yourMethod(5, 10);
     }
}
```

Output nedir?

Output of following Java program?

```
class Base {
    public void Print() {
        System.out.println("Base");
    }
}

class Derived extends Base {
    public void Print() {
        System.out.println("Derived");
    }
}

class Main{
    public static void DoPrint( Base o ) {
        o.Print();
    }
    public static void main(String[] args) {
        Base x = new Base();
        Base y = new Derived();
        Derived z = new Derived();
        DoPrint(x);
        DoPrint(y);
        DoPrint(z);
    }
}
```

Output nedir?

```
class Test01 {

    public int i;
    public int j;

    Test01() {
        i = 1;
        j = 2;
    }
}

class Test extends Test01 {
    int i = 12;
    int j = 13;

    Test(){
         super();
     }
}

class Main {
    public static void main(String args[]){
         Test obj = new Test();
         System.out.println(obj.i + " " + obj.j);
     }
}
```

Output nedir?

```
class Test01 {

    public int i;
    public int j;

    Test01() {
        i = 1;
        j = 2;
    }
}

class Test extends Test01 {
    int a;

    Test(){
         super();
     }
}

class Main {
    public static void main(String args[]){
         Test obj = new Test();
         System.out.println(obj.i + " " + obj.j);
     }
}
```

Output nedir?

```
class Test01 {

    public int i;
    public int j;

    Test01() {
        i = 1;
        j = 2;
    }
}

class Test extends Test01 {
    int i = 12;

    Test(){
         super();
     }
}

class Main {
    public static void main(String args[]){
         Test obj = new Test();
         System.out.println(obj.i + " " + obj.j);
     }
}
```