5-5-2020

Method Overloading in java

09 Monday — 7th week 040/325 — February

Many methods with same name in a class is called method overloading but with different signatures.

Example:-

```
class demo
{
    void m1(int i)
    {
    System.out.println("i=" + i);
    }
    void m1(int j, int k)
    {
    System.out.println("j+k=" +(j+k));
    }
}
class mdemo{
public static void main(String args[])
    {
    demo d = new demo();
    d.m1(20);      // i=20
    d.m1(20,30);   // j+k=50
    }
}
```

In above example we save it with mdemo.java

Compile & Run it through Command Prompt.

```
javac mdemo.java
java mdemo
```

At calling time, arguments are matched with the definition then gets executed.

Output:

```
i=20
j+k=50
```

Example 2:

```
class overloaddemo
{
void sum_m1(int i)
{
System.out.println("First function:" + i);
}
void sum_m1(int i, int j)
{
System.out.println("Second function:" + (i+j));
}
void sum_m1(int i, int j, int k)
{
System.out.println("Third function:" + (i+j+k));
}
public static void main(String args[])
{
overloaddemo obj = new overloaddemo();
obj.sum_m1(5);    // first version called
obj.sum_m1(20,5);    // second version called
obj.sum_m1(5,7,15);    // third version called
}
}
```

Output will be:-

First function: 5
Second function: 25
Third function: 27

Inheritance using Java

7th week
042-323

11 February
Wednesday

→ Inheritance is the mechanism of deriving new class from old one. Old class is known as Superclass. Baseclass

→ New class is known as Subclass (child class)

→ Java does not support multiple inheritance.

→ Subclass inherits all instance variables and methods defined by superclass and it also adds its own unique elements.

Benefits of Java inheritance :-

(i) Reusability of code.

(ii) Code Sharing.

(iii) Consistency in using an interface.

```
class A
  { static int i=20;
    A(){
System.out.println("value of i in Parent class:"+i);
    }
  }
class B extends A
  {
    B()
    { super(); // call the A class constructor
      System.out.println("child class"+i);
    }
  }
class C extends B
  {
```

Notes

```
C(){ super(); //call the B class Constructor
    System.out.println("Child 2 class :" + i);
}
```

12 Thursday

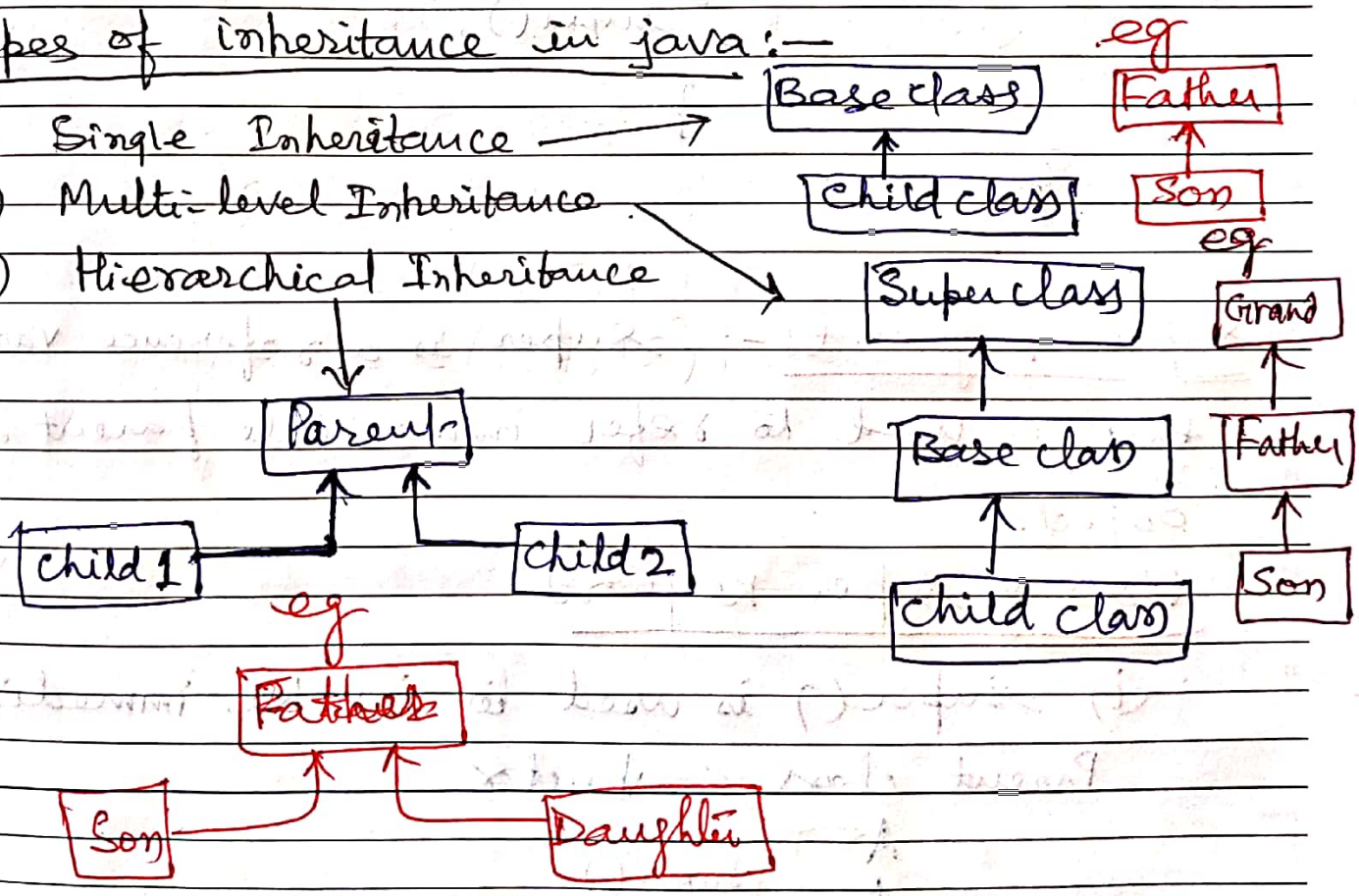
```
class inheritancedemo
{
    public static void main(String a[])
    {
        C obj = new C();
    }
}
```

output will be:-

value of i in Parent class : 20
child class 10
child 2 class : 10

Types of inheritance in java:-

(i) Single Inheritance →
(ii) Multi-level Inheritance →
(iii) Hierarchical Inheritance



Notes

extends → It is a keyword used for extending (deriving) a class from a base class.

Syntax:-

```
class parent1 {
    static int i=10;
    parent1() {
    }
}
class child1 extends parent1
{
    child1() {
        super();
    }
}
```

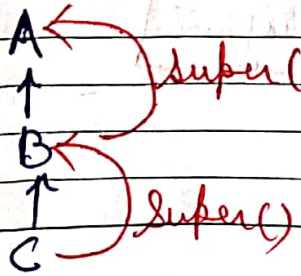
Base class

Child class

Super keyword:- Super is a reference variable that is used to refer immediate parent class object.

Uses of Super keyword

(i) Super() is used to invoke immediate Parent class Constructor



Notes

(ii) super is used to invoke immediate parent class method.

Eg:- super.method();

calls method() from its base class.

(iii) super is used to refer immediate parent class variable also.

Eg:- super.i;

accesses i variable of its base class

Complete Example:- calling parent class variable

```
class Parent {
        int i=10;
    }

class Child extends Parent
    {
        int i=20;
        void m1()
        {
    System.out.println("Value of i in child:"+i);
    System.out.println("Value of i in Parent:"+super.i);
        }
}
    class supertest{
    public static void main(String args[])
    {
        Child c1 = new Child();
        c1.m1();
    }
}
```

output is:-

Value of i in child: 20
Value of i in Parent: 10

15 Sunday

Notes

8th week
047-318

16 February
Monday

Complete Example:- calling Parent class Constructor using super keyword.

Any child class constructor's first line will be super(); supplied by the java virtual machine (JVM) whether we have written it explicitly or not.

Ex:-
```
class super1
{
    super1()
    {
    System.out.println("Super class Constructor.");
    }
}
class child2 extends super1
{
    child2()
    { //super();
    System.out.println("Child class Constructor.");
    }
}
class supertest
{
    public static void main(String args[])
    {
        child2 obj = new child2();
    }
}
```

Whether we write it or not base class constructor gets called first.

Save it with name Supertest.java

Compile & Run it :-

Output is :- Super class Constructor.
Child class Constructor.

Notes

FEBRUARY

S	M	T	W	T	F	S
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28

MARCH

S	M	T	W	T	F	S
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31				

8th week
061-317

17
Tuesday

Complete example :- Accessing parent class variable by using super.

```
class super1{
        int i=10;
        void m1() {
        System.out.println("Super class Method");
                }
                }
class child2 , extends super1
  {
        int i=20;
        void m1()
        {
System.out.println("value of i:"+super.i);
super.m1();
        }
  }
class supertest {
public static void main(String args[])
  {
        child2 obj = new child2();
        obj.m1();
  }
}
```

Output will be :-
Value of i: 10
Super class Method.