**Super Keyword-**

It is used to refer immediate parent class object, method and constructor.

When we create the instance of child class, an instance of parent class is created implicitly which we can referred by super keyword.

**Usage of Super keyword:**

1. To refer immediate parent class global variable
2. To invoke immediate parent class method
3. To invoke immediate parent class constructor -> super()

Why?

# **Program-1** for use of immediate parent class objects.

**public** **class** Parent {

**int** x= 20;

}

**class** Child **extends** Parent {

**int** x = 25;

**public** **void** test() {

**int** x= 30;

//Scenario-1

Parent p = **new** Parent();

System.***out***.println("Parent class x variable=" + p. x);

//Scenario-2

System.***out***.println("Immediate super class of child class x variable" + **super**. x);

}

}

**public** **class** TestMain {

**public** **static** **void** main(String[] args) {

Child c= **new** Child();

c.test();

}

}

Output-Parent class x variable=20

Immediate super class of child class x variable20

In Scenario 1, we are calling the x variable of parent class for that purpose we loading the whole class that is not good programmer approach. For use of single variable, we should go for super keyword in java.

In Scenario 2, we are trying to print the x variable of immediate super class by using the super keyword that is the best approach because we are not wasting the memory here.

**# Program 2**- program for use of immediate super class method.

**class** Parent {

**void** test() {

System.***out***.println("Parent class method.");

}

}

**class** Child **extends** Parent {

**void** test() {

**super**.test();

}

}

**public** **class** TestMain {

**public** **static** **void** main(String[] args) {

Child c= **new** Child();

c.test();

}

}

Output-

Parent class method.

# **Program-3** to use of immediate super class constructor-

**class** Parent {

Parent() {

System.***out***.println("Parent class constructor.");

}

}

**class** Child **extends** Parent {

Child() {

**super**();

}

}

**public** **class** TestMain {

**public** **static** **void** main(String[] args) {

Child c= **new** Child();

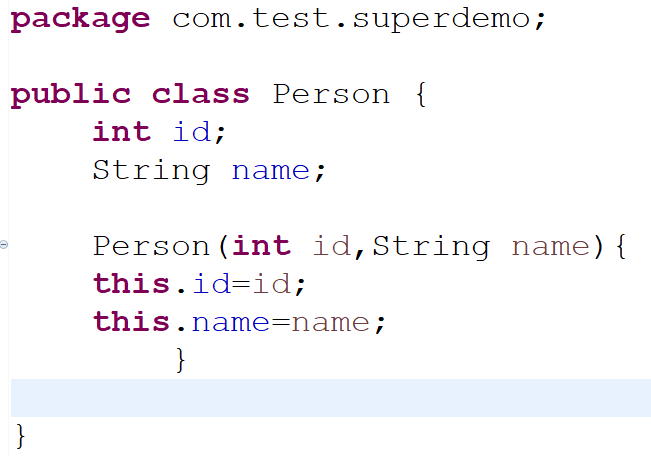
}

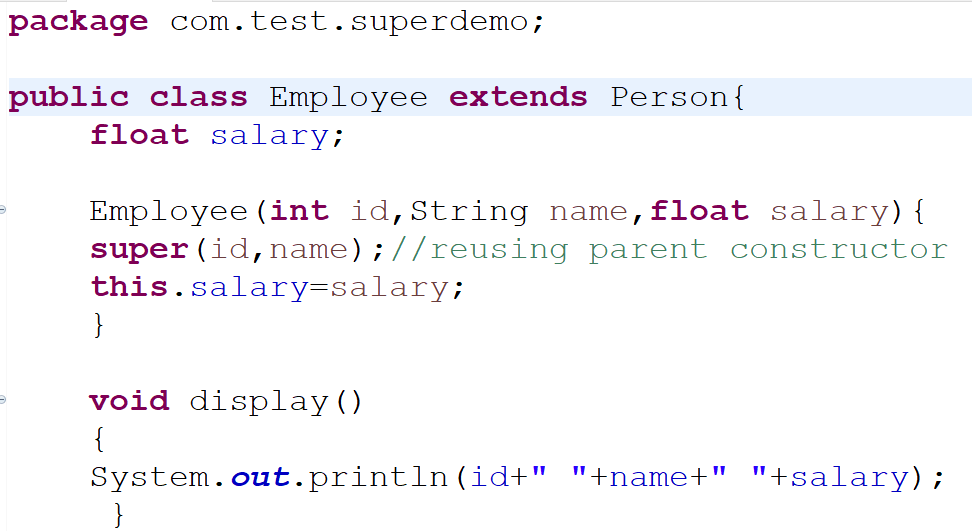
}

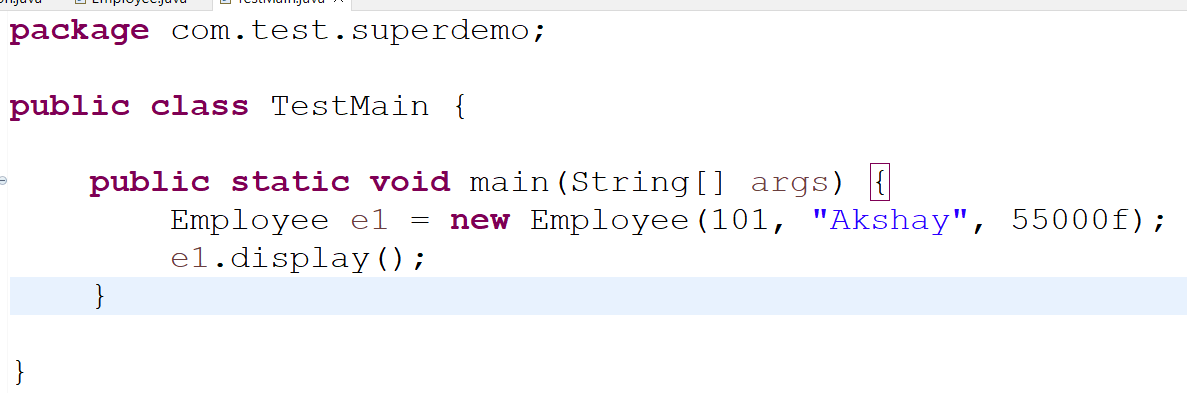
Output-

Parent class constructor.

Real Time Example:







Solution:



**This Keyword-**

It is used to refer current class variables, method and constructor.

Note- this keyword is not used in static context

Usage of “this” keywords:

1. To current class global variable
2. To invoke current class method(implicitly)
3. To invoke current class constructor -> this()
4. To pass arguments in method call
5. To pass argument in constructor
6. To return the current class instance from method

Why?

**# Program-1** use of current class variable.

**class** Parent {

**int** x = 20;

}

**class** Child **extends** Parent {

**int** x = 25;

**void** test() {

**int** x = 30;

Child c = **new** Child();

//In Scenario 1

System.***out***.println("By creating objects=" + c.x);

//In Scenario 2

System.***out***.println("By using this keyword=" + **this**.x);

}

}

**public** **class** TestMain {

**public** **static** **void** main(String[] args) {

Child c= **new** Child();

c.test();

}

}

Output-

By creating objects=25

By using this keyword=25

In Scenario 1, we are calling the x variable of child class for that purpose we loading the whole class that is not good programmer approach. For use of single variable, we should go for this keyword in java.

In Scenario 2, we are trying to calling the x variable of current class by using the keyword that is the best approach because we are not wasting the memory here.

**# Program 2**-

2.1 program for use of current class method.

**class** Parent {

**void** test() {

System.***out***.println("Parent class method.");

}

}

**class** Child **extends** Parent {

**void** test() {

System.***out***.println("Child class method");

}

**void** demo() {

**this**.test();

}

}

**public** **class** TestMain {

**public** **static** **void** main(String[] args) {

Child c= **new** Child();

c.demo();

}

}

2.2

Output-

Child class method.

# **Program-3.1** to use of current class constructor-

**package** com.test;

**class** Test {

**public** Test() {

System.***out***.println("Test constructor called using" + " this keyword..");

}

**public** Test(**int** x) {

**this**(); // invoke the constructor of current class

}

**public** **static** **void** main(String[] args) {

Test test = **new** Test(10);

}

}

Output-

Test constructor called using this keyword..

# **Program-3.2** to use of current class constructor-

**public** **class** Example {

Example() {

**this**("Java");

System.***out***.println("Inside Constructor without parameter");

}

Example(String str) {

System.***out***.println("Inside Constructor with parameter" + str);

}

**public** **static** **void** main(String[] args) {

Example obj = **new** Example();

}

}

Output-

Inside Constructor with String parameter as Java

Inside Constructor without parameter