| **"this" keyword in Java** | **"super" keyword in Java** |
| --- | --- |
| 1. "this" is an implicit reference variable keyword used to represent the current class. | 1. "super" is an implicit reference variable keyword used to represent the immediate parent class. |
| 2. “this” is to invoke methods of the current class. | 2. “super” is used to invoke methods of the immediate parent class. |
| 3. “this” is used to invoke a constructor of the current class. | 3. “super” is used to invoke a constructor of the immediate parent class. |
| 4. “this” refers to the instance and static variables of the current class. | 4. “super” refers to the instance and static variables of the immediate parent class. |
| 5. "this" can be used to return and pass as an argument in the context of a current class object. | 5. "super" can be used to return and pass as an argument in the context of an immediate parent class object. |

**Difference Between this and super Keyword in Java**

**Similarities Between “this” and “super” Keyword in Java**

1. **Both "this" and "super" are non-static, so they can't be used in static areas.** It means that we cannot use both the keywords in the main function in Java.

**Example 1**

public class Main {

int a = 50;

public static void main(String[] args) {

System.out.println(this.a);

}

}

**Output:**

Main.java:4: error: non-static variable this cannot be referenced from a static context

System.out.println(this.a);

**Explanation:**

In the above example, we are referencing the non-static variable "this" in the static context. This led to a compiler error.

1. **Both “super” and “this” keywords in Java can be used in constructor chaining to call another constructor.** “this()” calls the no-argument constructor of the current class, and “super()” calls the no-argument constructor of the parent class.

**Example 2**

*// declaring parent class*

class Parent{

Parent(){

System.out.println("Parent class no argument constructor");

}

Parent(String s){

System.out.println("Parent class parameterized "+s);

}

}

*// Child class extends parent class*

class Child extends Parent{

Child(){

*// referring current class parameterized constructor*

this("constructor.");

System.out.println("Child class no-argument constructor.");

}

Child(String s){

*// referring parent class parameterized constructor*

super("constructor.");

System.out.println("Child class parameterized "+s);

}

}

public class Main {

public static void main(String[] args) {

*// instance of child class*

Child obj = new Child();

}

**Output:**

Parent class parameterized constructor.

Child class parameterized constructor.

Child class no-argument constructor.

**Explanation:**

In the above example, we first forward a call from the child class no-argument constructor to the child class parameterized constructor using the **"this"** keyword. We are forwarding a call to the parent class parameterized constructor using the "super" keyword from a child class parameterized constructor.

1. **“this” and “super” must be the first statement if used inside the constructor.** This means we cannot call both statements in a single constructor.