Q) Explain the usage of super keyword ?

Usage of super keyword in Java:

Instance variable: super keyword is used to refer instance variables of immediate parent class

Super constructor: super() constructor call; is used to invoke constructor of immediate parent class

Overridden Method: <methodName>is used to invoke instance method of immediate parent class (when it is overridden in the sub-class)

Note:super keyword cannot be used to refer in static context

**Q) Explain the usage of super keyword ?**

**Usage of *super* keyword in Java:**

* **Instance variable:***super* keyword is used to refer instance variables of immediate parent class
* **Super constructor:***super()* constructor call; is used to invoke constructor of immediate parent class
* **Overridden Method:***<methodName>*is used to invoke instance method of immediate parent class (when it is overridden in the sub-class)
* **Note:***super* keyword cannot be used to refer in ***static*** context

**Q) Why always immediate parent class constructor is invoked ?**

**Rules for Constructor chaining:**

* Whenever an Object is created, then its respective class constructor is invoked
* But before the execution of its own class constructor, its immediate parent class constructor is invoked
* This is repeated till root of the hierarchy is reached i.e.; until Object class

**Reason:**

* In every constructor, compiler by default inserts call to super class constructor using ***super();*** statement
* Therefore, every-time Object is created, its parent class constructor is invoked
* Programmer can also write call to ***super(); constructor*** statement explicitly, but impacts is very same much to the default one
* If this ***super() constructor*** call is present, then it must be ***1st statement*** of the constructor
* Otherwise, ***compile-time error*** will be thrown

**Q) Does inserting explicit super(); constructor call inside child class constructor results in error ?**

* There won’t be any error by inserting ***explicit super() constructor*** call
* Rather, inserting ***explicit super(); constructor*** call inside child class constructor results in invocation of parent class constructor
* Let us insert ***super(); constructor explicitly*** in the child class and see result
* Example, as shown in the below screen-capture

**Q) How to invoke parent class’s parameterized constructor from child class ?**

* As programmer can write explicit call to super class constructor using super keyword from child class constructor
* Likewise, using parameterized super(…) constructor call is also possible
* This is required, whenever programmer required to invoke parameterized constructor of the immediate parent class
* Let us see one example with its results (as shown in the below screen-capture)

**Q) How to invoke parent class’s instance variable from child class, when variable names of both parent & child class are same ?**

* To invoke parent class variable from child class, use super keyword (when names of the variable in both parent & child class are same)
* **Syntax:**super.<variable-name>
* Example, as shown in the below screen-capture

**Q) Whether it is possible to refer static variable of parent class from child class using super keyword ?**

* Although, it is very possible to access parent class’s ***static variable*** using super keyword
* But its ***usage is discouraged*** with compiler warning with message “***The static field <parent-class-name>.<static-variable-name> should be accessed in a static way***”
* As it can be accessed using class-name
* **Syntax:** <class-name>.<static-variable-name>
* Let us see one example where we are accessing static variable of parent class using super keyword

**Q) How to invoke parent class method from child class, if method is overridden in the child class ?**

* Suppose method is ***overridden***in the sub-class from parent-class
* Then to ***access overridden method*** of parent-class from child-class, use super keyword
* **Syntax:** super.<overridden-method-name>
* In the below example,***display() is overridden*** from parent class
* So to access the parent class ***display()*** method from child class display() method, we have used ***super keyword***, as shown in the below screen-capture

### **(1) What is super keyword in java?**

**Super keyword in java is a reference variable which is refers immediate parent or super class object. In other words, Super keyword is used to access the members of parent class.**

**Super keyword point to immediate parent class object. It works with object only of parent class.**

**(2) Write some points about super keyword in java?**

**There are some use of super keyword in java.**

* **Super keyword can be used to refer parent class instance variables.**
* **Super keyword can be used to invoke parent class method.**
* **Super keyword can be used to invoke parent class constructor.**

**(3) Can we access parent class variables in child class by using super keyword?**

**Yes, We can access parent class variable in child class by using java super keyword.**

**For example:**

**class Parent**

**{**

**int a = 20;**

**}**

**class Child extends Parent**

**{**

**int a = 30;**  
**void show()**  
**{**  
**System.out.println(a);//print child class value of a**  
**System.out.println(super.a);//print parent class value of a**  
**}**

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

**{**

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

**}**

**}**  
 **Output: 30**  
**20**  
 **You can see in the above example, There are same data member in both parent and child class. If you will not use super keyword in child class, it will print only child class value by default i.e 30 because priority goes to child. If you want to print both child and parent object values you have to use super keyword in sub class.**

**(4) Can we call parent class method in sub class by using super keyword.**

**Yes, We can access parent class method in child class by using super keyword.**

**For example:**

**class Parent**

**{**

**void display()**

**{**

**System.out.println("parent");**

**}**

**}**

**class Child extends Parent**

**{**

**void display()**

**{**

**System.out.println("child");**

**super.display();**

**}**

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

**{**

**Child c = new Child();**

**c.display();**

**}**

**}**  
 **Output: child**  
**parent**

**(5) Can we call parent class constructor in sub class constructor i.e constructor chaining by using super keyword?**

**Yes, We can call parent class constructor in sub class constructor by using super keyword but super keyword must be the first statement in sub class constructor.**

**Syntax:**

**class ParentDemo**

**{**

**ParentDemo()**

**{}**

**}**

**class Child extends ParentDemo**

**{**

**Child()**

**{**

**super();//must be first statement i.e in first line**

**//statement;**

**//statement;**

**}**

**psvm()**

**{}**

**}**

**(6) Can we use both "this" and "super" in constructor?**

**No, It is not possible in java we cannot use both this and super keyword together in java constructor because this and super should be the first statement in any java constructor.**

**(7) What is difference between this() and super()?**

**There are some difference between**[**java this()**](https://javatutorial95.blogspot.com/2017/02/this-keyword-in-java.html)**constructor and super() constructor.**

***Java this()***

* **It is used to access current class constructor.**
* **It can be used inside another constructor of same class.**
* **It can be used to remove ambiguity error when we have data members and local are same name.**

**For example: Remove ambiguity error**

**class Demo**

**{**

**int a;**

**int b;**

**Demo(int a, int b)**

**{**

**this.a = a;**

**this.b = b;**

**}**

**}**

**}**

***Java super()***

* **It is used to access parent class constructor from sub class.**
* **It can be used to only inside child class constructor.**
* **It doesn't remove any ambiguity error from programs.**

**(8) Can we use super keyword in static method of a sub class for calling parent class method?**

**No, We cannot use super keyword in static methods because it belongs to the immediate parent object and static belongs to the class level.**