Java Interview Question and Answers

➢ **Topic –** Java Constructor:

Constructor Interview Questions in Java

---

1. What is a constructor in Java?
Ans: A constructor is a block of code, similar to a method that is used to initialize the state of an object (i.e. instance variable) in a class through a new operator. It is automatically called and executed at the time of object creation by JVM.

2. What is the main objective of a constructor in java?
Or, Why do we need a constructor in a class as a member?

Ans: The main objective/purpose of a constructor in java is to initialize instance variables in a class (or to set the initial state of an object).

3. When a constructor is called/invoked in Java?
Ans: The constructor of a class is called every time an object is created with a new keyword. For example, in the below code, two objects of class are created with the new keyword, therefore, constructor is called two times.

```
public class Test {

Test() {

      System.out.println("Inside constructor");

}

public static void main(String[ ] args){

   Test t1 = new Test();

   Test t2 = new Test();

  }
}
```

4. Is it possible for a class to have multiple constructors in java?
Ans: Yes, a class can have multiple constructors with different parameters. Which constructor gets called for object creation depends on the arguments passed while constructing different objects.

5. Does a constructor return any value?
Ans: The constructor can not have any return type even void also because if there is a return type then JVM would consider as a method, not a constructor.

6. Can a constructor be marked with the final keyword?
Ans: No, a constructor cannot be marked with the final keyword.

7. Is it possible to inherit a constructor?
Ans: No, a constructor cannot be inherited in java.

---

8. Can we use this() and super() inside the constructor?
Ans: No, this() and super() cannot be used together inside the constructor.

9. Will the below code compile successfully? If not, why?

```
class Test {

Test() {



}

public void display() {

   Test();

  }

}
```

Ans: On compilation of the above code, compile-time error will generate: The method Test() is undefined for the type Test. This is because it is illegal to invoke a constructor like this.

10. What are possible access modifiers that can be marked for a constructor?

Ans: The possible access modifiers for a constructor is as follows:

a) **private**: The constructor marked with private can be accessed only from its class.
b) **protected**: The constructor declared with protected access modifier is accessible from any class which resides in the same package.
c) **public**: The public constructor is accessible from any class within or outside the package.

11. Is it possible to invoke a constructor of a class more than once for an object?

Ans: No, it is not possible to call a constructor of a class more than once for an object. It is invoked only once per object at the time of object creation.

---

12. Does a constructor of the class get called, before or after creating an object?

Ans: A constructor gets called concurrently when the object creation is going on. JVM first allots memory space for the object in the heap and then executes the constructor to initialize instance variables. By the time object creation is completed, the execution of a constructor is completed.

13. How many types of constructors are in Java?

Ans: There are two types of constructors in java that are as follows:

- Default constructor (Non-parameterized constructor)
- Parameterized constructor

14. What is a default constructor?

Ans: A constructor that takes no parameter is called default constructor or parameterized constructor. For example:

```
A a = new A(); // It will create an object of class A by invoking default constructor.
```

15. What is a parameterized constructor?

Ans: A constructor that contains one or more parameters is called parameterized constructor. The parameterized constructor allows us to initialize different values to distinct objects.

16. What is the main purpose of default constructor in java?

Ans: The main purpose of default constructor is to initialize default values (null or zero value) to the objects.

Java compiler creates a default constructor at compile time only if there is no constructor in a class. After constructing the default constructor, default values are initialized to objects.

---

17. Is it necessary to define a constructor as the same name as the name of class?

Ans: Yes, a constructor must have the same name as that of class name. If the name of constructor is different, Java compiler will treat it as a normal method.

18. Give an example that proves the definition of constructor.

Ans:

```
public class Test
{
   String name;
   int age;
 Test() // constructor function.
 {
    name = "John";
    age = 20;
   }
}
```

19. Write a program to define a constructor and pass data through parameters.

Ans:

```
public class Person
```

```
{
    String name;
    int age;
Person(String n)
{
    name = n;
    age = 25;
  }
}
```

20. Will the below code compile successfully? If yes, what will be the output of the following program?

```
public class Test
{
 Test() {
       System.out.println("Calling default constructor");
 }
 Test(Test test) {
      System.out.println("Calling parameterized constructor");
 }
public static void main(String[] args)
{
     Test t = new Test(new Test());
 }

}
```

Ans: Yes, the above code will be compiled successfully. The output of the program is as follows:

- Calling default constructor
- Calling parameterized constructor

21. Is it possible to call a constructor from another constructor if multiple constructors are defined in a class?

Ans: If multiple constructors are defined inside a class, it is possible to call a constructor from another constructor using this keyword.

---

22. What will be the output of the following program?

```
public class Test

{

 Test(Object object) {

      System.out.println("Hello");

 }

 Test(Test test) {

      System.out.println("World");

 }

public static void main(String[] args)

{

    Test t = new Test(null);

 }

}
```

Ans: Output: world.

23. What is the use of constructor in java?
Ans: The use of constructor in java is:

a) to assign the default value of instance variables.
b) to execute a particular code at the time of object creation, we can write them inside the constructor.

24. Can we declare a constructor as private?
Ans: Yes, we can declare a constructor with a private access modifier. It is mainly done not to allow users to create an object of class from outside of the class. Basically, we use a *private constructor* in a singleton design pattern.

25. When does Java compiler define the default constructor?
Ans: Java compiler defines a default constructor only if there is no explicit constructor declared by the programmer.

26. Why a constructor defined by Java compiler is always called as default constructor?
Ans: A constructor defined by Java compiler is always called as default constructor because it obtains all its default properties from its class. They are:

a) Its access modifier is same as its class access modifier.
b) Its name is same as class name.
c) It has no parameters and logic.

27. What is constructor overloading in java?
Ans: *Constructor overloading* is a technique in which a class can have more than one constructor having the same name but different parameter lists. A parameter list consists of order and types of arguments.

28. How does Java compiler differentiate among multiple constructors in a class?
Ans: Java compiler differentiates multiple constructors based on the number of parameter lists and their types. Therefore, the signature of each constructor must be different.

29. What is the use of overloading constructor in java?
Ans: Overloading constructors can be used for performing different tasks based on different data.

30. What is the output of the below code snippet?

```
public class Test

{

 Test(int i) {

      System.out.println("Hello");

 }

 Test(long l) {

      System.out.println("World");

 }

public static void main(String[] args)

{

     Test t = new Test(20);

 }

}
```

Ans: Output: Hello.

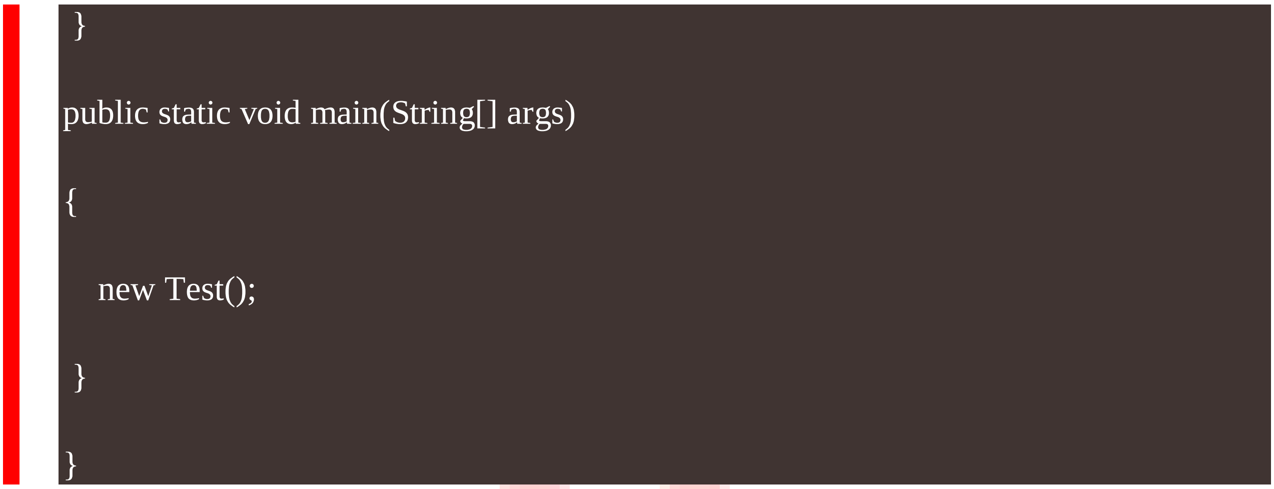
31. What will the output of the following program?

```
public class Test

{

 Test(int i) {

      System.out.println("Hello");

 }

 Test(Test test) {
      System.out.println("World");

 }

public static void main(String[] args)

{

    Test t = new Test(new Test(20));

 }

}
```

Ans: Output: Hello, World.

32. Will the below code snippet compile successfully? If yes, what will be the output of the following program?

```
public class Test

{

  Test() {

      this(20);

      System.out.println("One");

   }

  Test(int i) {

      this(null);

      System.out.println("Two");

    }

  Test(Test test) {

       System.out.println("Three");
```



```
 }

public static void main(String[] args)

{

   new Test();

 }

}
```

Ans: Output: Three, Two, and One.

---

33. What are the advantages of constructor overloading in Java?

Ans: There are several advantages of using constructor overloading in a program. They are as follows:

- Java constructor overloading helps to achieve static polymorphism.
- The main advantage of constructor overloading is to allow an instance of a class to be initialized in various ways.
- It allows defining of the multiple constructors of a class with different signatures.
- It helps to perform different tasks for different purposes.

34. Can we have more than one constructor with same signature in a class?

Ans: No, we cannot have more than one constructor with same signature in a class.

35. What is constructor chaining in java?

Ans: *Constructor chaining in Java* is a technique of calling one constructor from within another constructor by using this and super keywords.

36. Why do we use/need constructor chaining?

Ans: Constructor chaining can be used when we want to perform multiple tasks in a separate constructor for each task and make their order by chaining. It is useful to make the program more readable and easy to understand for everyone.

37. How to call one constructor from another constructor in Java?

Ans: Using this(), we can call the current class constructor within the “same class”.

Using super(), we can call the superclass constructor from the “base class”.

---

38. What is a copy constructor?
Ans: A constructor which is used to copy the data of one object to another object of the same class type is called copy constructor in Java.

39. Can we create an object of class within the same class if a constructor is marked with private?
Ans: Yes, we can create an object of class within the same class if constructor is marked with private but not outside the class.

40. Can class be extended when a constructor is declared private?
Ans: No.