

IMPORTANT INTERVIEW QUESTIONS

1. Why is Java a platform independent language?

Ans: Java language was developed so that it does not depend on any hardware or software because the compiler compiles the code and then converts it to platform-independent byte code which can be run on multiple systems.

The only condition to run that byte code is for the machine to have a runtime environment (JRE) installed in it.

2. Can java be said to be the complete object-oriented programming language?

Ans: Java is not a pure object-oriented programming language, because it has direct access to primitive data types. And these primitive data types don't directly belong to the Integer classes.

3. Can you differentiate between Heap and Stack Memory with respect to java? And how does java utilize these memories?

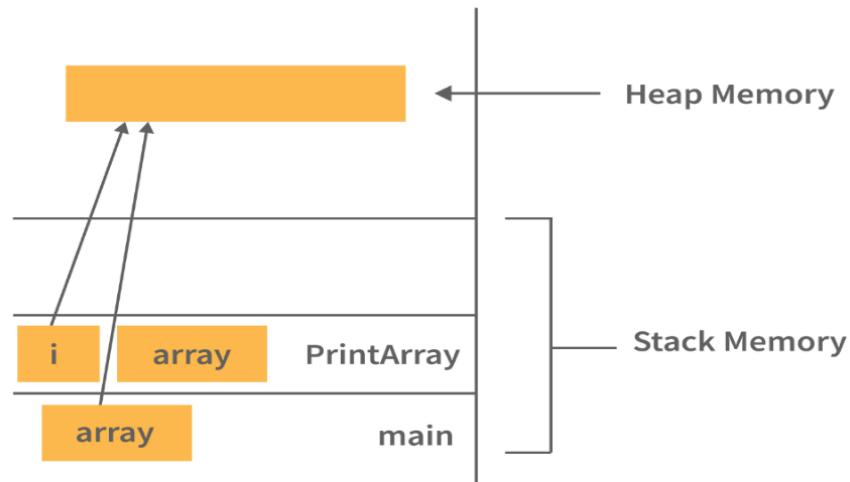
Ans: Stack memory is the portion of memory that was assigned to every individual program. And it was fixed. On the other hand, Heap memory is the portion that was not allocated to the java program but it will be available for use by the java program when it is required, mostly during the runtime of the program.

Java Utilizes this memory as -

- When we write a java program then all the variables, methods, etc are stored in the stack memory.
- And when we create any object in the java program then that object was created in the heap memory. And it was referenced from the stack memory.

Example- Consider the below java program:

```
class Main {  
    public void printArray(int[] array){  
        for(int i : array)  
            System.out.println(i);  
    }  
    public static void main(String args[]) {  
        int[] array = new int[10];  
        printArray(array);  
    }  
}
```



Main and PrintArray is the method that will be available in the stack area and as well as the variables declared that will also be in the stack area.

And the Object (Integer Array of size 10) we have created, will be available in the Heap area because that space will be allocated to the program during runtime.

4. How is Java different from C++?

Ans: Some of the basic differences between Java and C++ are:

- C++ is only a compiled language, whereas Java is compiled as well as an interpreted language.
- Java programs are machine-independent whereas a c++ program can run only in the machine in which it is compiled.
- C++ allows users to use pointers in the program. Whereas java doesn't allow it. Java internally uses pointers.
- C++ supports the concept of Multiple inheritances whereas Java doesn't support this. And it is due to avoiding the complexity of name ambiguity that causes the diamond problem.

5. Pointers are used in C/ C++. Why does Java not make use of pointers?

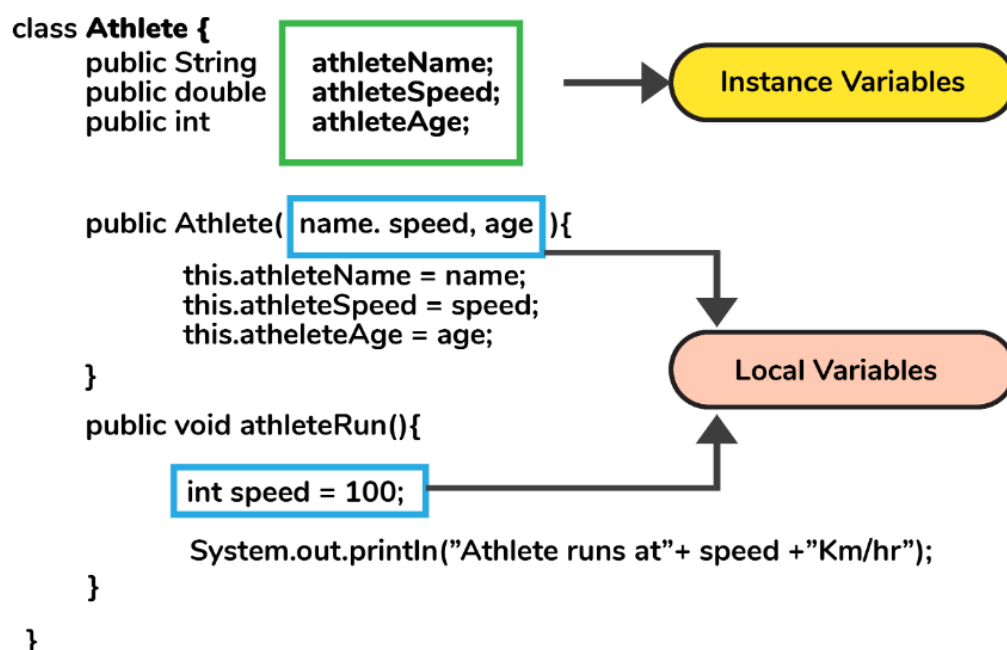
Ans: Pointers are quite complicated and unsafe to use by beginner programmers. Java focuses on code simplicity, and the usage of pointers can make it challenging. Pointer utilization can also cause potential errors. Moreover, security is also compromised if pointers are used because the users can directly access memory with the help of pointers.

Thus, a certain level of abstraction is furnished by not including pointers in Java. Moreover, the usage of pointers can make the procedure of garbage collection quite slow and erroneous. Java makes use of references as these cannot be manipulated, unlike pointers.

6. What do you understand by an instance variable and a local variable?

Ans: Instance variables are those variables that are accessible by all the methods in the class. They are declared outside the methods and inside the class. These variables describe the properties of an object and remain bound to it at any cost. All the objects of the class will have their copy of the variables for utilization. If any modification is done on these variables, then only that instance will be impacted by it, and all other class instances continue to remain unaffected.

Local variables are those variables present within a block, function, or constructor and can be accessed only inside them. The utilization of the variable is restricted to the block scope. Whenever a local variable is declared inside a method, the other class methods don't have any knowledge about the local variable.



7. What do you mean by data encapsulation?

Ans: Data Encapsulation is an Object-Oriented Programming concept of hiding the data attributes and their behaviours in a single unit. It helps developers to follow modularity while developing software by ensuring that each object is independent of other objects by having its own methods, attributes, and functionalities. It is used for the security of the private properties of an object and hence serves the purpose of data hiding.

8. Tell us something about JIT compiler.

Ans: JIT stands for Just-In-Time and it is used for improving the performance during run time. It does the task of compiling parts of byte code having similar functionality at the same time thereby reducing the amount of compilation time for the code to run.

The compiler is nothing but a translator of source code to machine-executable code. But what is special about the JIT compiler? Let us see how it works:

- First, the Java source code (.java) conversion to byte code (.class) occurs with the help of the javac compiler.
- Then, the .class files are loaded at run time by JVM and with the help of an interpreter, these are converted to machine understandable code.
- JIT compiler is a part of JVM. When the JIT compiler is enabled, the JVM analyzes the method calls in the .class files and compiles them to get more efficient and native code. It also ensures that the prioritized method calls are optimized.
- Once the above step is done, the JVM executes the optimized code directly instead of interpreting the code again. This increases the performance and speed of the execution.

9. How is an infinite loop declared in Java?

Ans: Infinite loops are those loops that run infinitely without any breaking conditions. Some examples of consciously declaring infinite loop is:

Using For Loop:

```
for (;;) {  
    // Business logic  
    // Any break logic  
}
```

Using while loop:

```
while(true){  
    // Business logic  
    // Any break logic  
}
```

Using do-while loop:

```
do {  
    // Business logic  
    // Any break logic  
}while(true);
```

10. Why is the main method static in Java?

Ans: The main method is always static because static members are those methods that belong to the classes, not to an individual object. So, if the main method will not be static then for every object, it is available. And that is not acceptable by JVM. JVM calls the main method based on the class name itself. Not by creating the object.

Because there must be only 1 main method in the java program as the execution starts from the main method. So, for this reason the main method is static.

11. What is a ClassLoader?

Ans: Java ClassLoader is the program that belongs to JRE (Java Runtime Environment). The task of ClassLoader is to load the required classes and interfaces to the JVM when required.

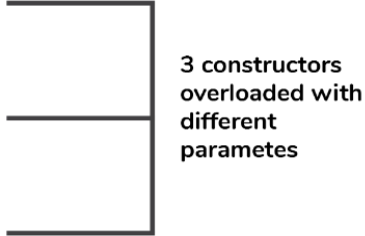
Example- To get input from the console, we require the Scanner class. And the Scanner class is loaded by the ClassLoader.

12. Briefly explain the concept of constructor overloading.

Ans: Constructor overloading is the process of creating multiple constructors in the class consisting of the same name with a difference in the constructor parameters. Depending upon the number of parameters and their corresponding types, distinguishing of the different types of constructors is done by the compiler.

Constructor Overloading

```
class Hospital {  
    int variable1, variable2;  
    double variable3;  
  
    public Hospital(int doctors, int nurses) {  
        variable1 = doctors;  
        variable2 = nurses;  
    }  
    public Hospital(int doctors) {  
        variable1 = doctors;  
    }  
    public Hospital(double salaries) {  
        variable3 = salaries;  
    }  
}
```



3 constructors overloaded with different parameters

13. Can the main method be Overloaded?

Ans: Yes, it is possible to overload the main method. We can create as many overloaded main methods we want. However, JVM has a predefined calling method that JVM will only call the main method with the definition of -

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

Consider the below code snippets:

```
class Main {  
    public static void main(String args[]) {  
        System.out.println(" Main Method");  
    }  
    public static void main(int[] args){  
        System.out.println("Overloaded Integer array Main Method");  
    }  
    public static void main(char[] args){  
        System.out.println("Overloaded Character array Main Method");  
    }  
    public static void main(double[] args){  
        System.out.println("Overloaded Double array Main Method");  
    }  
    public static void main(float args){  
        System.out.println("Overloaded float Main Method");  
    }  
}
```

14. What are shallow copy and deep copy in java?

Ans: To copy the object's data, we have several methods like deep copy and shallow copy.

Example -

```
class Rectangle{
    int length = 5;
    int breadth = 3;
}
```

Object for this Rectangle class –

```
Rectangle obj1 = new Rectangle();
```

Shallow copy - The shallow copy only creates a new reference and points to the same object.

Example - For Shallow copy, we can do this by -

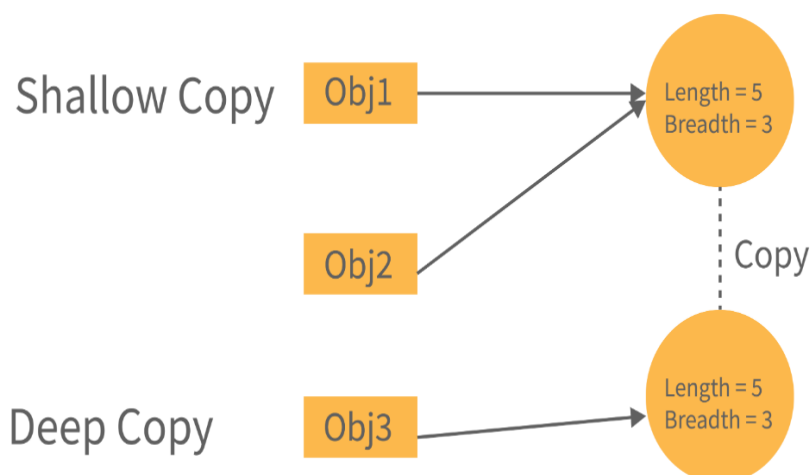
```
Rectangle obj2 = obj1;
```

Now by doing this what will happen is the new reference is created with the name obj2 and that will point to the same memory location.

Deep Copy - In a deep copy, we create a new object and copy the old object value to the new object.

Example -

```
Rectangle obj3 = new Rectangle();
Obj3.length = obj1.length;
Obj3.breadth = obj1.breadth;
```



Now, consider the following example code:

```
class Rectangle {
    int length = 5;
    int breadth = 3;
}
public class Main {
    public static void main(String[] args) {
        Rectangle obj1 = new Rectangle();
        //Shallow Copy
        Rectangle obj2 = obj1;

        //Deep Copy
        Rectangle obj3 = new Rectangle();
        obj3.length = obj1.length;
        obj3.breadth = obj1.breadth;

        System.out.println(" Before Changing the value of object 1,
                           the object2 will be - ");
        System.out.println(" Object2 Length = "+obj2.length+",
                           Object2 Breadth = "+obj2.breadth);

        System.out.println(" Before Changing the value of object 1,
                           the object3 will be - ");
        System.out.println(" Object3 Length = "+obj3.length+",
                           Object3 Breadth = "+obj3.breadth);

        //Changing the values for object1.
        obj1.length = 10;
        obj1.breadth = 20;

        System.out.println("\n After Changing the value of object
                           1, the object2 will be - ");
        System.out.println(" Object2 Length = "+obj2.length+",
                           Object2 Breadth = "+obj2.breadth);

        System.out.println(" Before Changing the value of object 1,
                           the object3 will be - ");
        System.out.println(" Object3 Length = "+obj3.length+",
                           Object3 Breadth = "+obj3.breadth);
    }
}
```

Output:

Before Changing the value of object 1, the object2 will be -

Object2 Length = 5, Object2 Breadth = 3

Before Changing the value of object 1, the object3 will be -

Object3 Length = 5, Object3 Breadth = 3

After Changing the value of object 1, the object2 will be -

Object2 Length = 10, Object2 Breadth = 20

Before Changing the value of object 1, the object3 will be -

Object3 Length = 5, Object3 Breadth = 3

15. What is the main objective of garbage collection?

Ans: The main objective of this process is to free up the memory space occupied by the unnecessary and unreachable objects during the Java program execution by deleting those unreachable objects.

This ensures that the memory resource is used efficiently, but it provides no guarantee that there would be sufficient memory for the program execution.

Reference:

- <https://www.interviewbit.com/java-interview-questions/>

*****END*****