Basic Java Interview Preparation For Freshers In 2022

@nileshmali

@webiyor

1) **What is minimum requirement to run the java program from jvm jdk and jre** ?
   - JDK, because it include JRE and some other files too.

2) What is the constructor in java?

a. A constructor initializes an object when it is created. It has the same name as its class is syntactically similar to a method

Important point about constructor

1. Constructor name must be the same name as your class name.
2. Constructor is used only to initialize your object.
3. Constructor is called at the time of object creation (i.e suppose your class name is Car and you create its object say " Car obj = new Car() " as soon as the above statement gets executed constructor gets called automatically
4. Constructor cannot be inherited.

Constructor can be of 3 types

a. Simple constructor
b. Parameterized constructor
c. Copy Constructor

3) Does constructor return any value?

- Constructor doesn't return any value as it does not have any return type.

4) Can we make a constructor final?

- Final means no changes be made once initialized so to do use final keyword with constructor could cause a problem.
- The constructor cannot be final because the constructor cannot be inherited and in constructor values are initialized to the variable so by this variables changes every time.
- A constructor can't inherited so can't be overridden, so there is no use of making constructor final.

5) What is static in java?

- Static is a non-access modifier in java which is applicable for the following

a. Blocks
b. Variables
c. Method
d. Nested classes

6) Why main method is static in java?

- Public static void main(String[] args)
  {
  …..
  }
- Main method is static in java because main() is called by the JVM befor any objects are made.
- Since it is static it can be directly called by the class.

7) Can we override a static method?

- No, we cannot override a static method java.
- If a derived class defined a static method with same signature as static method in base class, the method in the derived class hides the method in the base class.

8) Inheritance in java?

- The process by which one class acquires the properties (data members) and functionalities (Method) of another class is called inheritance.
- An object which acquired the properties of the another object is called inheritance properties means code & data / method & fields.

Type of Inheritance.

a. Single Inheritance
b. Multilevel Inheritance
c. Hierarchical Inheritance
d. Hybrid Inheritance

9) Why multiple inheritance is not possible java?

- Java does not support multiple inheritance.

- Suppose C is the child class extending from both parent class A and parent class B with same method defined in them. Then child class cannot understand which class method to call. So there is a confusion here which leads to ambiguity and leads to compile time error.
- This is the reason java does not support multiple inheritance
- Java does not support multiple inheritance. For this java has interface

10) Difference between abstract class and inheritance

	Abstraction Class	Interface
1.	A class which is declared with the abstract keyword is knows as an abstract class in java	The interface in java is a mechanism to achieve abstraction, It can't have a method body
2.	A abstract class can have abstract and non-abstract method	Interface can have only abstract methods
3.	Abstract class can have final, non-final, static and not-static variable	Interface has only static and final variable
4.	Abstract class may contain non-final variables	Variable declared in a java interface are by default final
5.	Abstract class can be extended using keyword "extends"	Interface can be implemented using keyword "implements"
6.	A java abstract class can have class members like private, protected, etc	Member of a java interface are public by default
7.	Abstract class can extend another java class and implement multiple java interface	Interface can extend another java interface only

8.	Example :
	public abstract class Car
	{
	public abstract void Wheel();
	}
		Example :
		public interface Aeroplane
		{
		void Wheel();
		}

---------- JAVA ----------

11) Why java is not 100 % object-oriented?

- Because of the primitive data type namely
- Boolean, byte, char, int, float, double, short, long

12) What is java?

- Java is high-level programming language and is platform independent
- Java collection of object it was develop by sun microsystem
- There are lots of application, website and games that develop using java

13) Why java is platform independent?

- Java is platform-independent because is does not depend on any type of platform hence,
- Java is platform-independent in java program are compiled into byte code and the byte code is platform independent.

14) Different between JDK, JRE and JVM

- JDK : Java development kit
- JDK : is combination of JVM and JRE
- JDK = JRE + JVM

- JRE : Java Runtime endowment
- Inside the JRE is development tools compiler
- Also inside the JRE. JVM and set of libraries eg. Rt jar etc

- JVM : Java virtual machine
- It is called virtual machine because it does not physically exit
- It is a specification that provides a runtime environment in which bytecode can be executed

15) String[] args

- Java main method accept a single argument of type string array.
- This is also called of java command line argument

16) Class

- Class is a collection of object
- It is just a template or blueprint
- Class does not occupy space

17) Object

- Object is an instance of class
- Object is real world entity
- Object occupy space

18) Explain JDK, JRE and JVM?

JDK	JRE	JVM
It stands for Java Development Kit.	It stands for Java Runtime Environment.	It stands for Java Virtual Machine.
It is the tool necessary to compile, document and package Java programs.	JRE refers to a runtime environment in which Java bytecode can be executed.	It is an abstract machine. It is a specification that provides a run-time environment in which Java bytecode can be executed.
It contains JRE + development tools.	It's an implementation of the JVM which physically exists.	JVM follows three notations: Specification, **Implementation,** and **Run time Instance**.

19) Explain public static void main(String args[]) in Java.

main() in Java is the entry point for any Java program. It is always written as **public static void main(String[] args)**.

- **public**: Public is an access modifier, which is used to specify who can access this method. Public means that this Method will be accessible by any Class.
- **static**: It is a keyword in java which identifies it is class-based. main() is made static in Java so that it can be accessed without creating the instance of a Class. In case, main is not made static then the compiler will throw an error as **main**() is called by the JVM before any objects are made and only static methods can be directly invoked via the class.
- **void**: It is the return type of the method. Void defines the method which will not return any value.
- **main**: It is the name of the method which is searched by JVM as a starting point for an application with a particular signature only. It is the method where the main execution occurs.
- **String args[]**: It is the parameter passed to the main method.

20) What are wrapper classes in Java?

- Wrapper classes convert the Java primitives into the reference types (objects). Every primitive data type has a class dedicated to it. These are known as wrapper classes because they “wrap” the primitive data type into an object of that class. Refer to the below image which displays different primitive type, wrapper class and constructor argument.

21) Difference between Heap and Stack Memory in java

- Stack :
    - Stack memory is the portion of memory that was assigned to every individual program. And it was fixed.
    - When we write a java program then all the variables, methods, etc are stored in the stack memory.
- Heap :
    - 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.
    - 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.

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

- 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.

22) What do you understand by an instance variable and a local variable?

- **Instance variables**: are those variables that are accessible by all the methods in the class.

  Example:

```
class Athlete {
        public String athleteName;
        public double athleteSpeed;
        public int athleteAge;
}
```

-
- **Local variables:** are those variables present within a block, function, or constructor and can be accessed only inside them.

  Example:

```
public void athlete() {
    String athleteName;
    double athleteSpeed;
    int athleteAge;
}
```

23) Tell us something about JIT compiler.

- 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.

24) Comment on method overloading and overriding by citing relevant examples.

- **Method overloading:**

```
class OverloadingHelp {
  public int findarea (int l, int b) {
        int var1;
        var1 = l * b;
        return var1;
  }
  public int findarea (int l, int b, int h) {
        int var2;
        var2 = l * b * h;
        return var2;
  }
}
```

- Both the functions have the same name but differ in the number of arguments. The first method calculates the area of the rectangle, whereas the second method calculates the area of a cuboid.

- **Method overriding:**

```
class HumanBeing {
	public int walk (int distance, int time) {
		int speed = distance / time;
		return speed;
	}
}
class Athlete extends HumanBeing {
	public int walk(int distance, int time) {
		int speed = distance / time;
		speed = speed * 2;
		return speed;
	}
}
```

- Both class methods have the name walk and the same parameters, distance, and time. If the derived class method is called, then the base class method walk gets overridden by that of the derived class

25) Explain the use of final keyword in variable, method and class.

In Java, the final keyword is used as defining something as constant /final and represents the non-access modifier.

- **final variable:**
  - When a variable is declared as final in Java, the value can't be modified once it has been assigned.
  - If any value has not been assigned to that variable, then it can be assigned only by the constructor of the class.

- **final method:**

- A method declared as final cannot be overridden by its children's classes.
- A constructor cannot be marked as final because whenever a class is inherited, the constructors are not inherited. Hence, marking it final doesn't make sense. Java throws compilation error saying

- **final class:**
  - No classes can be inherited from the class declared as final. But that final class can extend other classes for its usage

26) Do final, finally and finalize keywords have the same function?

All three keywords have their own utility while programming.

Final: If any restriction is required for classes, variables, or methods, the final keyword comes in handy. Inheritance of a final class and overriding of a final method is restricted by the use of the final keyword. The variable value becomes fixed after incorporating the final keyword. Example:

```
final int a=100;
a = 0;  // error
```

The second statement will throw an error.

Finally: It is the block present in a program where all the codes written inside it get executed irrespective of handling of exceptions. Example:

```
try {
        int variable = 5;
}
catch (Exception exception) {
        System.out.println("Exception occurred");
}
finally {
        System.out.println("Execution of finally block");
}
```

Finalize: Prior to the garbage collection of an object, the finalize method is called so that the clean-up activity is implemented. Example:

```
public static void main(String[] args) {
        String example = new String("InterviewBit");
        example = null;
        System.gc(); // Garbage collector called
}
public void finalize() {
          // Finalize called
}
```

27) When can you use super keyword?

- The super keyword is used to access hidden fields and overridden methods or attributes of the parent class.
- Following are the cases when this keyword can be used:
  - Accessing data members of parent class when the member names of the class and its child subclasses are same.
  - To call the default and parameterized constructor of the parent class inside the child class.
  - Accessing the parent class methods when the child classes have overridden them.
- The following example demonstrates all 3 cases when a super keyword is used.

```
public class Parent{
    protected int num = 1;
    Parent(){
        System.out.println("Parent class default constructor.");
    }
    Parent(String x){
        System.out.println("Parent class parameterised constructor.");
    }
    public void foo(){
        System.out.println("Parent class foo!");
    }
}
public class Child extends Parent{
    private int num = 2;
    Child(){
        System.out.println("Child class default Constructor");
        super("Call Parent");    // to call parameterised constructor.
        super();    // to call default parent constructor
    }
    void printNum(){
        System.out.println(num);
        System.out.println(super.num); //prints the value of num of parent class
    }
    @Override
    public void foo(){
        System.out.println("Parent class foo!");
        super.foo();    //Calls foo method of Parent class inside the Overriden foo method of Child class.}}
```

28) Can the static methods be overloaded?

Yes! There can be two or more static methods in a class with the same name but differing input parameters.

29) Can the static methods be overridden?

- No! Declaration of static methods having the same signature can be done in the subclass but run time polymorphism can not take place in such cases.
- Overriding or dynamic polymorphism occurs during the runtime, but the static methods are loaded and looked up at the compile time statically. Hence, these methods cant be overridden

30) Difference between static methods, static variables, and static classes in java.

- **Static Methods and Static variables** are those methods and variables that belong to the class of the java program, not to the object of the class. This gets memory where the class is loaded. And these can directly be called with the help of class names.
    - For example - We have used mathematical functions in the java program like - max(), min(), sqrt(), pow(), etc. And if we notice that, then we will find that we call it directly with the class name. Like - Math.max(), Math.min(), etc. So that is a static method. And Similarly static variables we have used like (length) for the array to get the length. So that is the static method.
- **Static classes** - A class in the java program cannot be static except if it is the inner class. If it is an inner static class, then it exactly works like other static members of the class.

31) What is the main objective of garbage collection?

- 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.

32) How would you differentiate between a String, StringBuffer, and a StringBuilder?

Parameter	String	StringBuffer	StringBuilder
Storage	String Pool	Heap	Heap
Mutability	Immutable	Mutable	Mutable
Thread Safe	Not used in a threaded environment	Used in a multi-threaded environment	Used in a single-threaded environment
Performance	Slow	Slower than StringBuilder but faster than String	Faster than StringBuffer
Syntax	String var =“Edureka”; String var=new String(“Edureka”);	StringBuffer var = new StringBuffer("Edureka");	StringBuilder var = new StringBuilder ("Edureka");

---------- OOPS ----------

33) What is encapsulation?

- **1)** Data hiding: **Encapsulation is the process of hiding unwanted information, such as restricting access to any member of an object.**

  2) Data binding: **Encapsulation is the process of binding the data members and the methods together as a whole, as a class**

34) What is Polymorphism?

- Polymorphism is composed of two words - “poly” which means “many”, and “morph” which means x`“shapes”. Therefore Polymorphism refers to something that has many shapes

Thank You

LinkedIn – Nilesh Mali

YouTube - Webiyor