1) What is Java?

Java is the high-level, object-oriented, secure programming language, platform-independent, high performance, Multithreaded,

2) What are the differences between C++ and Java?

Comparison Index	C++	Java
Platform-independent	C++ is platform-dependent.	Java is platform-independent.
Mainly used for	C++ is mainly used for system programming.	Java is mainly used for application programming.
Multiple inheritance	C++ supports multiple inheritance.	Java doesn't support multiple inheritance through class. It can be achieved by interfaces in java.
Operator Overloading	C++ supports operator overloading.	Java doesn't support operator overloading.

Pointers	C++ supports pointers. You can write pointer program in C++.	Java supports pointer **internally**. However, you can't write the pointer program in java. It means java has restricted pointer support in Java.
Compiler and Interpreter	C++ uses compiler only. C++ is compiled and run using the compiler which converts source code into machine code	Java uses compiler and interpreter both. Java **source code** is converted into **bytecode** at **compilation time**. The interpreter executes this **bytecode** at **runtime** and produces output. **Java is interpreted** that is why it is platform independent.
Thread Support	C++ doesn't have built-in support for threads.	Java has built-in thread support.
Object-oriented	C++ is an object-oriented language. However, in C language, single root hierarchy is not possible.	Java is also an object-oriented language. However, everything (except fundamental types) is an object in Java. It is a single root hierarchy as everything gets derived from java.lang.Object.

3) List the features of Java Programming language.

- **Platform Independent:** Java is a platform independent programming language.
- **Secured:** Java is secured because it doesn't use **explicit pointers**. Java also provides the concept of **ByteCode** and **Exception handling** which makes it **more secured**.
- **Architecture Neutral:** Java is architectural neutral as it is not dependent on the architecture. In C, the size of data types may vary according to the architecture (32 bit or 64 bit) which doesn't exist in Java.
- **Distributed:** Java is distributed because it facilitates users to create distributed applications in Java. RMI and EJB are used for creating distributed applications. This feature of Java makes us able to access files by calling the methods from any machine on the internet.
- **Dynamic:** Java is a dynamic language. It supports dynamic loading of classes. It means classes are loaded on demand.

4) What do you understand by Java virtual machine?

Java Virtual Machine is a virtual machine that enables the computer to run the Java program. The Java code is compiled by JVM to be a Bytecode which is machine independent

5) What is the difference between JDK, JRE, and JVM?

JVM

JVM is an acronym for **Java Virtual Machine**; it is an abstract machine which provides the **runtime environment in which Java bytecode can be executed**.

JVMs are available for many hardware and software platforms (so **JVM is platform dependent**).

JRE

JRE stands for **Java Runtime Environment**. It is the implementation of JVM. The Java Runtime Environment is a **set of software tools which are used for developing Java applications**. It is used to provide the runtime environment.

JDK

JDK is an acronym for **Java Development Kit**. It is a software development environment which is used to develop Java applications and applets.

8) What is the platform?

A platform is the **hardware or software environment** in which a piece of **software is executed**.

10) What gives Java its 'write once and run anywhere' nature?

The **bytecode**. Java compiler converts the Java programs into the class file (Byte Code) which is the intermediate language between source code and machine code. This bytecode is not platform specific and can be executed on any computer

12) Is Empty .java file name a valid source file name?

```
//save by .java only
class A{
```

```
public static void main(String args[]){
System.out.println("Hello java");
}
}
//compile by javac .java
//run by     java A
```

Working - will create a A.class file from the class. And can run it as well. No issue

compile it by **javac .java**
run it by **java A**

16) What is the default value of the local variables?

The local variables are not initialized to any default value,

17) What are the various access specifiers in Java?

- **Public** The classes, methods, or variables which are defined as public, **can be accessed by any class or method.**
- **Protected-** Members declared as protected are accessible within the same class, by derived classes, and within the same package.
- **Default**
  Members with no access specifier (also known as **package-private**) are accessible within the same package.
  They are **not accessible outside the package** in which the class is defined, **even by subclasses in other packages**.
- **Private** The private class, methods, or variables defined as private can be **accessed within the class only.**

18) What is the purpose of static methods and variables?

The methods or variables defined as static are shared among all the objects of the class. The static variables are stored in the class area, and we **do not need to create the object to access such variables**. Therefore, **static is used in the case**, where we need to **define variables or methods which are common to all the objects** of the class.

Static variables are shared among all instances of a class, and static methods can be called without creating an instance.

makes your program more memory efficient (

```
public class Counter {
    // Static variable to store the total count across all instances
    private static int totalCount = 0;
    // Instance variable to store the count for each object
    private int count;
    // Constructor to initialize the count for each object
    public Counter() {
        // Increment the total count and assign it to the instance count
        totalCount++;
        count = totalCount;
    }
    // Static method to get the total count across all instances
    public static int getTotalCount() {
        return totalCount;
    }
    // Instance method to get the count for a specific object
    public int getCount() {
        return count;
    }
    public static void main(String[] args) {
        // Creating instances of Counter
        Counter c1 = new Counter();
        Counter c2 = new Counter();
        Counter c3 = new Counter();
        // Accessing static method to get the total count
        System.out.println("Total Count: " + Counter.getTotalCount());   -3
        // Accessing instance methods to get counts for each object
        System.out.println("Count for c1: " + c1.getCount());   - 1      assign current TotCount to Count)
        System.out.println("Count for c2: " + c2.getCount());   - 2
        System.out.println("Count for c3: " + c3.getCount());   - 3
    }
}
```

19) What are the advantages of Packages in Java?

- Packages avoid the name clashes.
- The Package provides easier access control.
- It is easier to locate the related classes.

Core Java - OOPs Concepts: Initial OOPs Interview Questions

24) What is an object?

In Java, Object is an **instance** of the class. The object of a class can be created by using the **new** keyword

27) What is the constructor?

It is invoked when the class is instantiated, and the memory is allocated for the object. Every time, an object is created using the **new** keyword, the default constructor of the class is called. The name of the constructor must be similar to the class name. The constructor must not have an explicit return type.

28) How many types of constructors are used in Java?

- **Default Constructor**
- **Parameterized Constructor**

29) What is the purpose of a default constructor?

The purpose of the default constructor is to assign the default value to the objects.

30) Does constructor return any value?

Ans: yes, The constructor implicitly returns the current instance of the class

32) Can you make a constructor final?

No

When a variable is declared as `final`, its value cannot be changed once assigned.

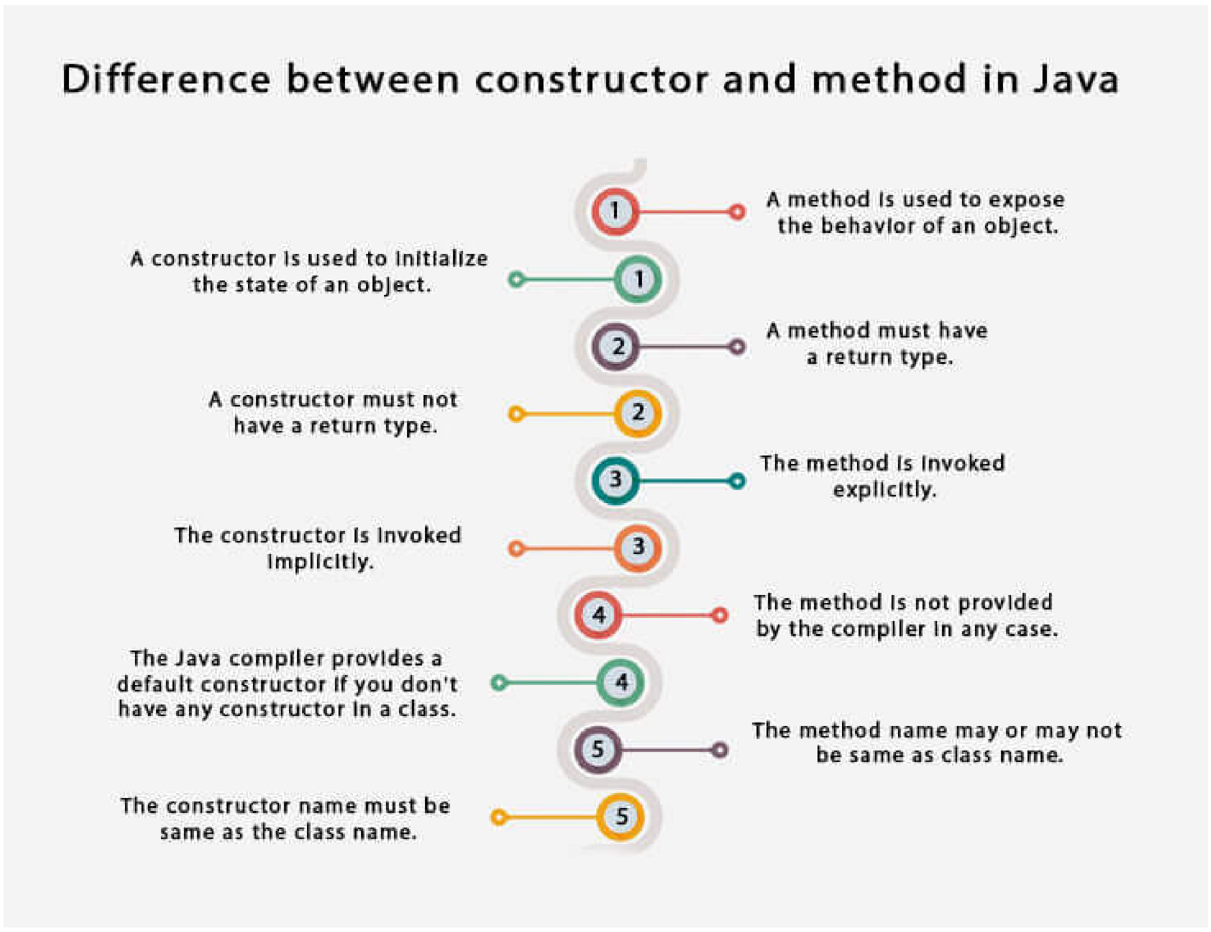
33) Can we overload the constructors?

Yes, the constructors can be overloaded by changing the number of arguments accepted by the constructor or by changing the data type of the parameters.

34) What do you understand by copy constructor in Java?

```java
//Java program to initialize the values from one object to another
class Student6{
  int id;
  String name;
  //constructor to initialize integer and string
  Student6(int i,String n){
  id = i;
  name = n;
  }
  //constructor to initialize another object
  Student6(Student6 s){
  id = s.id;
  name =s.name;
  }
  void display(){System.out.println(id+" "+name);}

  public static void main(String args[]){
  Student6 s1 = new Student6(111,"Karan");
  Student6 s2 = new Student6(s1);
  s1.display();
  s2.display();
 }
```


Difference between constructor and method in Java
1 A method is used to expose the behavior of an object.
1 A constructor is used to initialize the state of an object.
2 A method must have a return type.
2 A constructor must not have a return type.
3 The method is invoked explicitly.
3 The constructor is invoked implicitly.
4 The method is not provided by the compiler in any case.
4 The Java compiler provides a default constructor if you don't have any constructor in a class.
5 The method name may or may not be same as class name.
5 The constructor name must be same as the class name.

In Java, when you provide a parameterized constructor in a class, the default (no-argument) constructor is not automatically generated if you haven't explicitly defined one. However, if you don't define any constructor at all, Java will automatically provide a default constructor for you.

42) Why is the main method static?

Because the object is not required to call the static method. If we make the main method non-static, JVM will have to create its object first and then call main() method which will lead to the extra memory allocation

44) What is the static block?

Static block is used to initialize the static data member. It is executed before the main method, at the time of classloading.

47) What is the difference between static (class) method and instance method?

static or class method	instance method
1)A method that is declared as static is known as the static method.	A method that is not declared as static is known as the instance method.
2)We don't need to create the objects to call the static methods.	The object is required to call the instance methods.
3)Non-static (instance) members cannot be accessed in the static context (static method, static block, and static nested class) directly.	Static and non-static variables both can be accessed in instance methods.

Difference between abstract class and interface

Abstract class	Interface

1) Abstract class can **have abstract and non-abstract** methods.	Interface can have **only abstract** methods.
2) Abstract class **doesn't support multiple inheritance**.	Interface **supports multiple inheritance**.
3) Abstract class **can have final, non-final, static and non-static variables**.	Interface has **only static and final variables**.
4) Abstract class **can provide the implementation of interface**.	Interface **can't provide the implementation of abstract class**.
8) A Java **abstract class** can have class members like private, protected, etc.	Members of a Java interface are public by default.

51) What is **this** keyword in java?

The **this** keyword is a reference variable that **refers to the current object.**

60) Why is multiple inheritance not supported in java?

To **reduce the complexity and simplify the language**, multiple inheritance is not supported in java.

61) What is aggregation? (Has a)

Aggregation can be defined as the relationship between two classes where the aggregate class contains a reference to the class it owns.

Address.java

```java
public class Address {
String city,state,country;

public Address(String city, String state, String country) {
    this.city = city;
    this.state = state;
    this.country = country;
}


}
```

```java
public class Emp {
int id;
String name;
Address address;

public Emp(int id, String name,Address address) {
    this.id = id;
    this.name = name;
    this.address=address;
}

void display(){
System.out.println(id+" "+name);
System.out.println(address.city+" "+address.state+" "+address.country);
}

public static void main(String[] args) {
Address address1=new Address("gzb","UP","india");
Address address2=new Address("gno","UP","india");

Emp e=new Emp(111,"varun",address1);
Emp e2=new Emp(112,"arun",address2);
```

62) What is composition?

we can say that composition is the particular case of aggregation which represents a stronger relationship between two objects. Example: A class contains students. A student cannot exist without a class. There exists composition between class and students.

63) What is the difference between aggregation and composition?

Aggregation represents the weak relationship whereas composition represents the strong relationship. For example, the bike has an indicator (aggregation), but the bike has an engine (composition).

64) Why does Java not support pointers?

The pointer is a variable that refers to the memory address. They are not used in Java because they are unsafe(unsecured) and complex to understand.

67) What are the main uses of the super keyword?

super can be used to refer to the immediate parent class instance variable.
super can be used to invoke the immediate parent class method.

68) What are the differences between this and super keyword?

The super keyword always points to the parent class contexts whereas this keyword always points to the current class context.

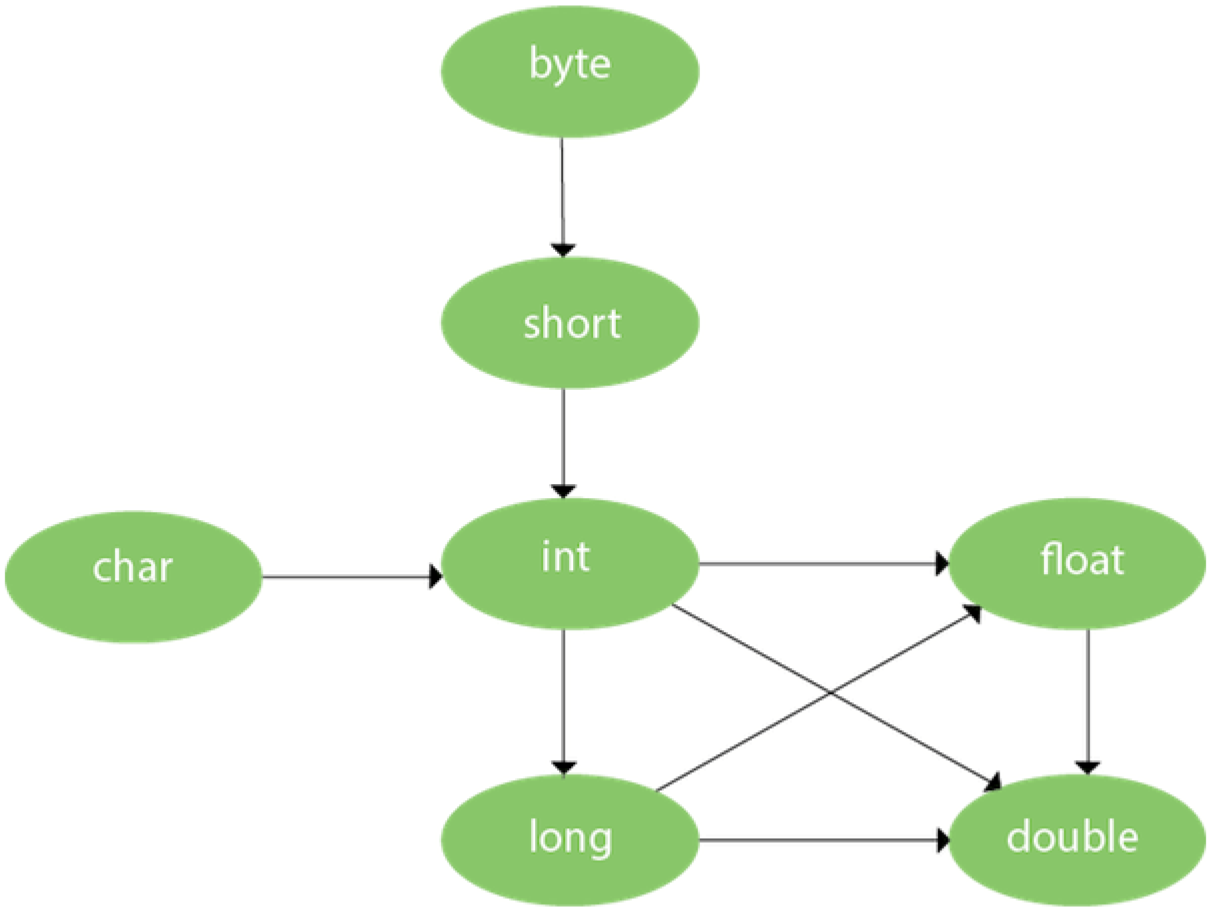
72) What is method overloading?

Method overloading is the polymorphism technique which allows us to create multiple methods with the same name but different signature. We can achieve method overloading in two ways.

- By Changing the number of arguments
- By Changing the data type of arguments

Method overloading increases the readability of the program.

76) What is method overloading with type promotion?



78) What is method overriding:

If a subclass provides a specific implementation of a method that is already provided by its parent class, it is known as Method Overriding. It is used for runtime polymorphism and to implement the interface methods.

Rules for Method overriding

- The method must have the same name as in the parent class.
- The method must have the same signature as in the parent class.

82) Difference between method Overloading and Overriding.

Method Overloading	**Method Overriding**
1) Method overloading increases the readability of the program.	Method overriding provides the specific implementation of the method that is already provided by its superclass.
2) Method overloading occurs within the class.	Method overriding occurs in two classes that have IS-A relationship between them.
3) In this case, the parameters must be different.	In this case, the parameters must be the same.

87) Can you have virtual functions in Java?

Yes, all functions in Java are virtual by default.

88) What is covariant return type?

Now, since java5, it is possible to override any method by changing the return type if the return type of the subclass overriding method is subclass type. It is known as covariant return type.

89) What is the output of the following Java program?

```
class Base
```

```
{
   public void baseMethod()
   {
      System.out.println("BaseMethod called ...");
   }
}
class Derived extends Base
{
   public void baseMethod()
   {
      System.out.println("Derived method called ...");
   }
}
public class Test
{
   public static void main (String args[])
   {
      Base b = new Derived();
      b.baseMethod();
   }
}
```

Output

```
Derived method called ...
```

90) What is the final variable?

In Java, the final variable is used to restrict the user from updating it. If we initialize the final variable, we can't change its value.

91) What is the final method?

If we change any method to a final method, we can't override it.

92) What is the final class?

If we make any class final, we can't inherit it into any of the subclasses

93) What is the final blank variable?

A final variable, not initialized at the time of declaration, is known as the final blank variable. We can't initialize the final blank variable directly. Instead, we have to initialize it by using the class constructor

101) What is the difference between compile-time polymorphism and runtime polymorphism?

compile-time polymorphism - overloading	**Runtime polymorphism -** overridden
It is also known as static binding, early binding, or **overloading**.	It is also known as dynamic binding, late binding, **overriding**, or dynamic method dispatch.
It provides **fast execution** because the type of an object is determined at compile-time.	It provides **slower execution** as compare to compile-time because the type of an object is determined at run-time.

Compile-time polymorphism provides less flexibility because all the things are resolved at **compile-time**.	Run-time polymorphism provides more flexibility because all the things are resolved at **runtime**.

Override

```
class Bike{
 int speedlimit=90;
}
class Honda3 extends Bike{
 int speedlimit=150;
 public static void main(String args[]){
  Bike obj=new Honda3();
  System.out.println(obj.speedlimit);//90  cant override variables
  }
```

104) What is the difference between static binding and dynamic binding?

In case of the static binding, the type of the object is determined at compile-time whereas, in the dynamic binding, the type of the object is determined at runtime.

107) What is the abstraction?

Abstraction is a process of hiding the implementation details and showing only functionality to the user.

two ways to achieve the abstraction.

- Abstract Class
- Interface

108) What is the difference between abstraction and encapsulation?

Abstraction hides the implementation details whereas encapsulation wraps code and data into a single unit.

121) How to make a read-only class in Java?

A class can be made read-only by making all of the fields private.The read-only class will have only getter methods

122) How to make a write-only class in Java?

A class can be made write-only by making all of the fields private. The write-only class will have only setter method

123) What are the advantages of Encapsulation in Java?

- By providing only the setter or getter method, you can make the class read-only or write-only. In other words, you can skip the getter or setter methods.
- It is a way to achieve data hiding in Java because other class will not be able to access the data through the private data members.
- The encapsulate class is easy to test. So, it is better for unit testing.

125) What are the advantages of defining packages in Java?

By defining packages, we can avoid the name conflicts between the same class names defined in different packages

132) What is Exception Handling?

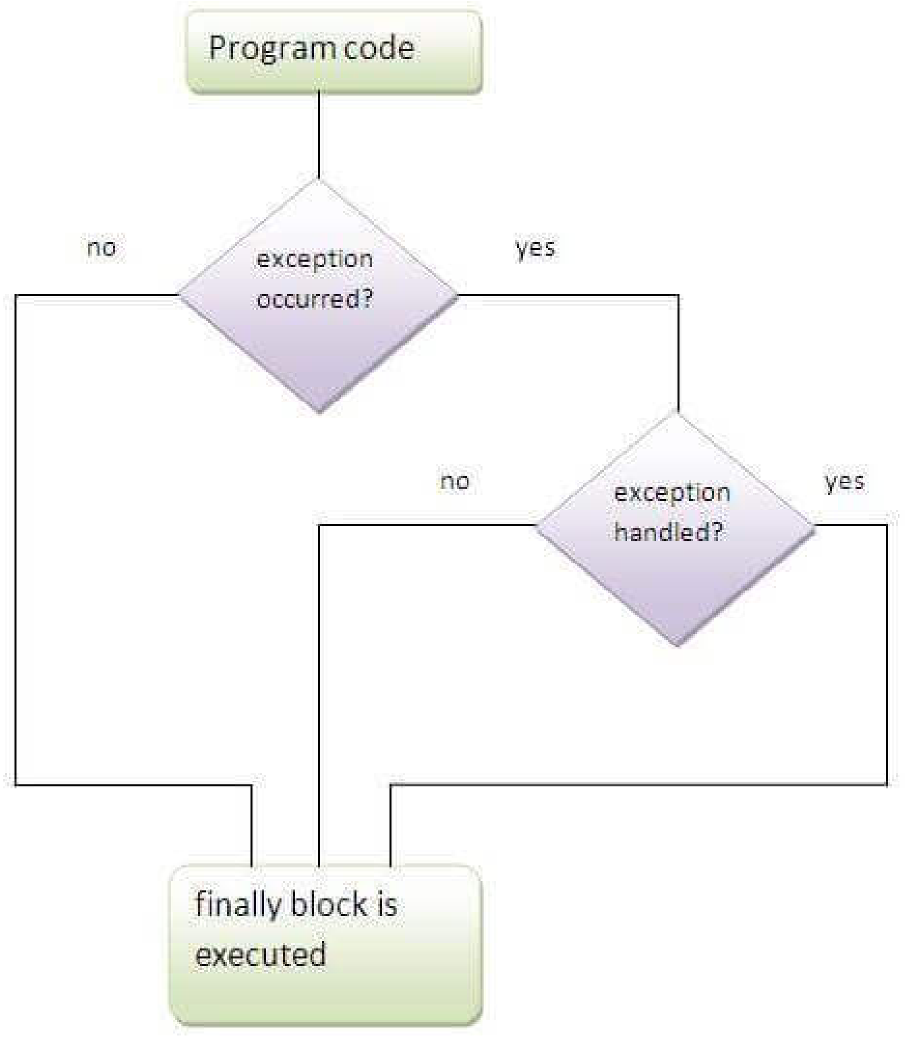
Exception Handling is a mechanism that is used to handle runtime error

136) Is it necessary that each try block must be followed by a catch block?

It is not necessary that each try block must be followed by a catch block. It should be followed by either a catch block OR a finally block.

138) What is finally block?

The "finally" block is executed whether an exception is handled or not. In other words, we can say that finally block is the block which is always executed



179) What is Garbage Collection?

Garbage collection is the process of removing unused objects from the memory to free up space and make this space available for Java Virtual Machine

186) What is the difference between final, finally and finalize?

final	**finally**	**finalize**
The final class can't be inherited, final method can't be overridden, and final variable value can't be changed.	Finally is used to place important code, it will be executed whether an **exception** is handled or not.	finalize is a method in Java that is called by the garbage collector before an object is reclaimed (destroyed) to perform any necessary cleanup operations.
Final is a keyword.	Finally is a block.	Finalize is a method.

207) Give a brief description of Java socket programming?

Java Socket programming is used for communication between the **applications running** on **different JRE.**

208) What is Socket?

A socket is simply an endpoint for communications between the machines. It provides the connection mechanism to connect the two computers using TCP.

221) What is object cloning?

The object cloning is a way to create an exact copy of an object. The clone() method of the Object class is used to clone an object.

227) What is a singleton class?

Singleton class is the class which can not be instantiated more than once. To make a class singleton, we either make its constructor private