**1.what is abstract class in java?**

A class which is declared as abstract is known as an **abstract class**. It can have abstract and non-abstract methods. It needs to be extended and its method implemented. It cannot be instantiated.

Java abstract class is a class that can not be initiated by itself, it needs to be subclassed by another class to use its properties.

An abstract class is declared using the “abstract” keyword in its class definition.

**Abstract class:** is a restricted class that cannot be used to create objects (to access it, it must be inherited from another class

Eg:

**abstract** **class** Shape {

**public** **abstract** **void** draw();

}

**class** Circle **extends** Shape {

@Override

**public** **void** draw() {

System.out.println("Drawing a circle");

}

}

**class** Rectangle **extends** Shape {

@Override

**public** **void** draw() {

System.out.println("Drawing a rectangle");

}

}

**public** **class** AbstractClass{

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

Shape circle = **new** Circle();

Shape rectangle = **new** Rectangle();

circle.draw(); // Output: Drawing a circle

rectangle.draw(); // Output: Drawing a rectangle

}

}

**2. Inheritance in java**

**3:Types of Inheritances:**

On the basis of class, there can be three types of inheritance in java: single, multilevel and hierarchical.

In java programming, multiple and hybrid inheritance is supported through interface only.

**4. why multiple Inheritance is not allowed in java**

Multiple Inheritance is a feature of an object-oriented concept, where a class can inherit properties of more than one parent class. The problem occurs when there exist methods with the same signature in both the superclasses and subclass. On calling the method, the compiler cannot determine which class method to be called and even on calling which class method gets the priority.

**5.multithreading**

**Multithreading in**[Java](https://www.javatpoint.com/java-tutorial) is a process of executing multiple threads simultaneously.

A thread is a lightweight sub-process, the smallest unit of processing. Multiprocessing and multithreading, both are used to achieve multitasking.

However, we use multithreading than multiprocessing because threads use a shared memory area. They don't allocate separate memory area so saves memory, and context-switching between the threads takes less time than process.

 Multithreading can also handle multiple requests from the same user.

* **Multitasking** is a computer's ability to execute two or more concurrent programs. Multithreading makes multitasking possible when it breaks programs into smaller, executable threads. Each thread has the programming elements needed to execute the main program, and the computer executes each thread one at a time.
* **Multiprocessing** uses more than one CPU to speed up overall processing and supports multitasking.

Java Multithreading is mostly used in games, animation, etc.

**6.life cycle of thread?**

In Java, a thread always exists in any one of the following states. These states are:

1. New
2. Active
3. Blocked / Waiting
4. Timed Waiting
5. Terminated

**7.** **can we start the thread twice?**

No. After starting a thread, it can never be started again. If you does so, an IllegalThreadStateException is thrown. In such case, thread will run once but for second time, it will throw exception.

**8.what are the methods in object class**

No. After starting a thread, it can never be started again. If you does so, an IllegalThreadStateException is thrown. In such case, thread will run once but for second time, it will throw exception.

**9.class I want to print class name and package name how will u do?**

Eg:

class HelloWorld {

public static void main(String[] args) {

HelloWorld main = new HelloWorld();

String className = main.getClass().getName();

System.out.println(className);

}

}

10.if I create a object of object class and if want to call the clone method is that possible?

No