The **join()** method:

The `java.lang.Thread` class provides the `join()` method which allows one thread to wait until another thread completes its execution. In simpler words, it means it waits for the other thread to die.

```java
public class J1 extends Thread {
    public void run() {
        for (int i = 1; i <= 5; i++) {
            try {
                Thread.sleep(500);
            } catch (Exception e) {
                System.out.println(e);
            }
            System.out.println(i);
        }
    }
    public static void main(String args[]) {
        J1 t1 = new J1();
        J1 t2 = new J1();
        J1 t3 = new J1();
        t1.start();
        try {
            t1.join();
        } catch (Exception e) {
            System.out.println(e);
        }
        t2.start();
        t3.start();
    }
}
```

`t1` is first started and joined hence `t1` task will be executed completely, then only `t2` and `t3` will be executed.
Join method has a void type and throws InterruptedException. Joining threads in Java has three methods namely:

1. join()
2. join(long millis)
3. join(long millis, int nanos)

**Can we start a Thread twice in Java?**

No, once a thread is started, it can never be started again. Doing so will throw an `IllegalThreadStateException`.

```java
public class M1 extends Thread{
    public void run(){
        System.out.println("running thread..");
    }
    public static void main(String args[]){
        M1 t1=new M1();
        t1.start();
        t1.start();
    }
}
```

**Exercise:**

1. Why is join method used in thread? What are its benefits? Explain.
2. Can we start a thread twice? What will happen then?