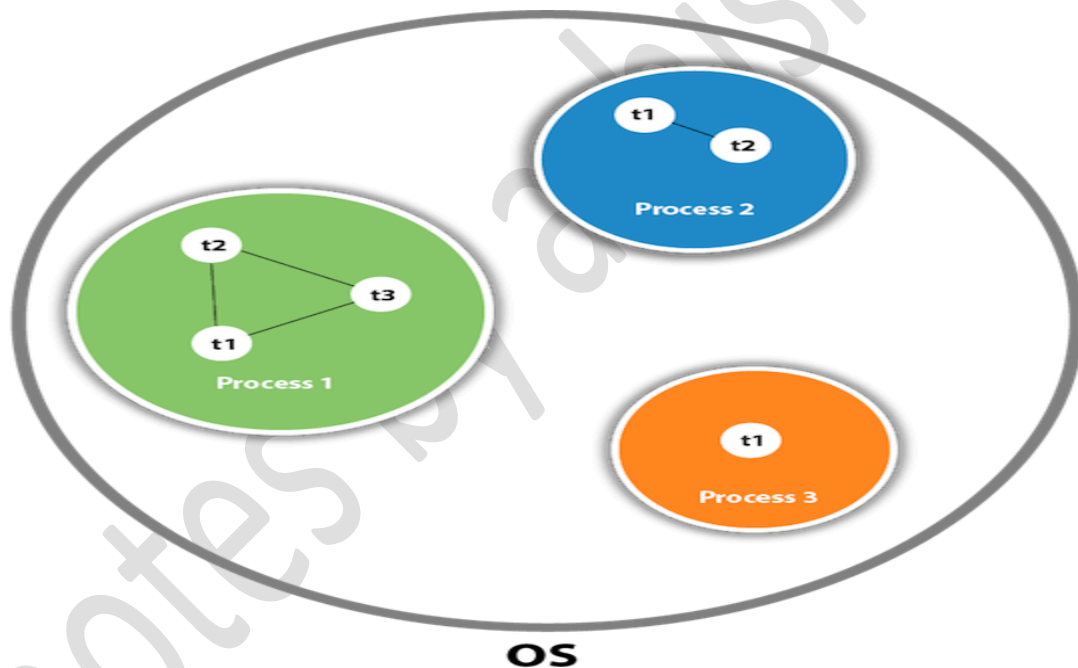


Multithreading in Java:

- Multithreading is a Java feature that allows concurrent execution of two or more parts of a program for maximum utilization of CPU.
- Each part of such program is called a thread. So, threads are light-weight processes within a process.



Multitasking

Multitasking is a process of executing multiple tasks simultaneously. We use multitasking to utilize the CPU. Multitasking can be achieved in two ways:

- Process-based Multitasking (Multiprocessing)
- Thread-based Multitasking (Multithreading)

Process-based Multitasking (Multiprocessing)

- Each process has an address in memory. In other words, each process allocates a separate memory area.
- A process is heavyweight.
- Cost of communication between the process is high.
- Switching from one process to another requires some time for saving and loading registers, memory maps, updating lists, etc.

Thread-based Multitasking (Multithreading):

- Threads share the same address space.
- A thread is lightweight.
- Cost of communication between the thread is low.

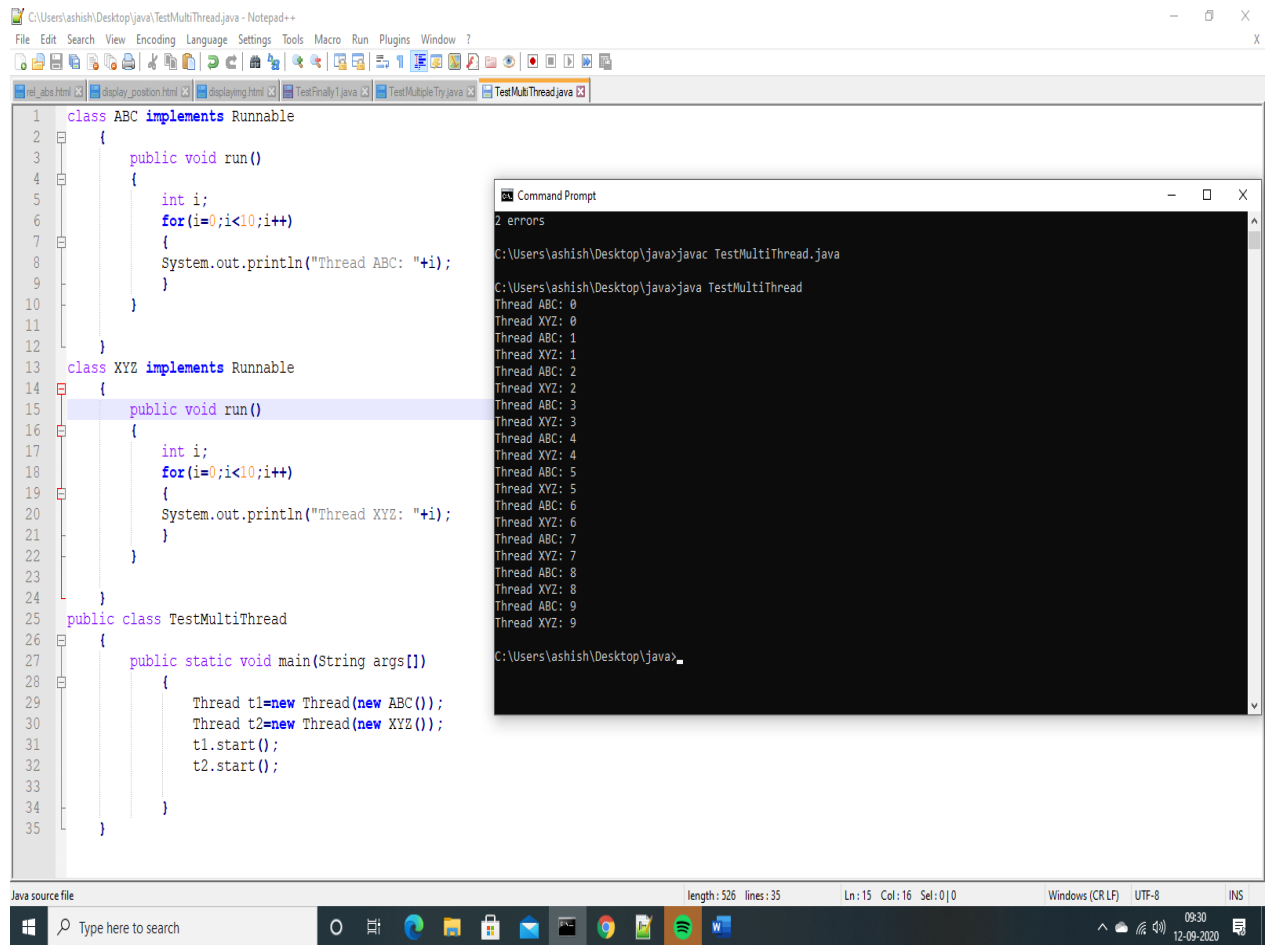
Threads can be created by using two mechanisms:

1. Implementing the Runnable Interface
2. Extending the Thread class

1. Implementing the Runnable Interface:

- We create a new class which implements `java.lang.Runnable` interface and override `run()` method.
- Then we instantiate a Thread object and call `start()` method on this object.

Example:



The screenshot shows a Notepad++ editor with a Java file named `TestMultiThread.java`. The code defines two classes, `ABC` and `XYZ`, both implementing the `Runnable` interface. Each class has a `run()` method that prints its name and a counter from 0 to 9. The `TestMultiThread` class contains a `main` method that creates and starts two threads, `t1` (of type `ABC`) and `t2` (of type `XYZ`).

Overlaid on the editor is a Command Prompt window showing the output of the program. It displays the command `javac TestMultiThread.java` and `java TestMultiThread`, followed by the interleaved output of the two threads: `Thread ABC: 0` through `Thread ABC: 9` and `Thread XYZ: 0` through `Thread XYZ: 9`.

```
1 class ABC implements Runnable
2 {
3     public void run()
4     {
5         int i;
6         for(i=0;i<10;i++)
7         {
8             System.out.println("Thread ABC: "+i);
9         }
10    }
11
12    class XYZ implements Runnable
13    {
14        public void run()
15        {
16            int i;
17            for(i=0;i<10;i++)
18            {
19                System.out.println("Thread XYZ: "+i);
20            }
21        }
22    }
23
24    public class TestMultiThread
25    {
26        {
27            public static void main(String args[])
28            {
29                Thread t1=new Thread(new ABC());
30                Thread t2=new Thread(new XYZ());
31                t1.start();
32                t2.start();
33            }
34        }
35    }
```

```
2 errors
C:\Users\ashish\Desktop\java>javac TestMultiThread.java
C:\Users\ashish\Desktop\java>java TestMultiThread
Thread ABC: 0
Thread XYZ: 0
Thread ABC: 1
Thread XYZ: 1
Thread ABC: 2
Thread XYZ: 2
Thread ABC: 3
Thread XYZ: 3
Thread ABC: 4
Thread XYZ: 4
Thread ABC: 5
Thread XYZ: 5
Thread ABC: 6
Thread XYZ: 6
Thread ABC: 7
Thread XYZ: 7
Thread ABC: 8
Thread XYZ: 8
Thread ABC: 9
Thread XYZ: 9
C:\Users\ashish\Desktop\java>
```