Thread class

Method of thread class:-

Method   **return type**

- ❖ **public void run():** is used to perform action for a thread.
- ❖ **public void start():** starts the execution of the thread.JVM calls the run() method on the thread.
- ❖ **public void sleep(long miliseconds):** Causes the currently executing thread to sleep (temporarily cease execution) for the specified number of milliseconds.
- ❖ **public void join():** *waits for a thread to die.*
- ❖ **public void join(long miliseconds):** *waits for a thread to die for the specified miliseconds.*
- ❖ **public int getPriority():** *returns the priority of the thread.*
- ❖ **public int setPriority(int priority):** *changes the priority of the thread.*
- ❖ **public String getName():** *returns the name of the thread.*
- ❖ **public void setName(String name):** *changes the name of the thread.*
- ❖ **public Thread currentThread():** *returns the reference of currently executing thread.*
- ❖ **public int getId():** *returns the id of the thread.*
- ❖ **public Thread.State getState():** *returns the state of the thread.*
- ❖ **public boolean isAlive():** *tests if the thread is alive.*
- ❖ **public void yield():** *causes the currently executing thread object to temporarily pause and allow other threads to execute.*
- ❖ **public void suspend():** *is used to suspend the thread(depricated).*
- ❖ **public void resume():** *is used to resume the suspended thread(depricated).*
- ❖ **public void stop():** *is used to stop the thread(depricated).*
- ❖ **public boolean isDaemon():** *tests if the thread is a daemon thread.*
- ❖ **public void setDaemon(boolean b):** *marks the thread as daemon or user thread.*

- ❖ *public void interrupt():* *interrupts the thread.*
- ❖ *public boolean isInterrupted():* *tests if the thread has been interrupted.*
- ❖ *public static boolean interrupted():* *tests if the current thread has been interrupted.*

Multithread:

Note:- **Run is a special method of thread class which is invoke help of start method.**

Example:-

```
class India extends Thread
{
public void run()
{
for(int i=0;i<5;i++)
{
if(i==3)
{
System.out.println("india relinquist control here");
yield();
}
System.out.println("in India");
}
}
```

```java
}
class china extends Thread
{
public void run()
{
for(int i=0;i<5;i++)
{
try
{
sleep(1000);
}
catch(Exception e)
{
System.out.println(e);
}
if(i==3)
{
stop();
}
System.out.println("in China");
}
}
```

```java
}
class Lanka extends Thread
{
public void run()
{
for(int i=0;i<5;i++)
{
System.out.println("in Lanka");
}
}
}
class thread2
{
public  static void main(String arg[])
{
India a=new India();
Lanka d=new Lanka();
china c=new china();
a.start();
d.start();
c.start();
}
```

}

We will run this program, our answer different in all time

Example:-

```
class India extends Thread

{

India(String n)

{

super(n);

}

public void run()

{

System.out.println("In India");

}

}

class thread3

{

public static void main(String arg[])

{
```

```java
India i=new India("My india");

System.out.println(i.getName());

i.start();

}

}
```

Output:-

```
My india

In india
```

Example:-

```java
class thread4

{

public static void main(String arg[])

{

Thread t=Thread.currentThread();

t.setName("Pradeep");

System.out.println(t.getName());

try

{

    t.sleep(1000);

}
```

```java
        catch(Exception e)

        {

            System.out.println(e);

        }

        System.out.println("End of program");

        }

        }
```

Output:-

```
Pradeep

//after 1000 time sleep then run

End of program
```

Example:-

```java
        class Gwalior

        {

        public void show()

        {

        System.out.println("In gwalior");

        }

        }

        class india extends Gwalior implements Runnable

        {
```

```java
public void run()

{

System.out.println("india");

}

}

class thread5

{

public static  void main(String arg[])

{

india e=new india();

Thread t=new Thread();

t.start();

e.show();

}

}
```

Output:-

In Gwalior