For Multiple Files, Custom Library and File Read/Write, use our new - Advanced Java IDE

```java
import java.util.*;
public class Main
{
    public static void main(String args[])
    {
        Scanner s=new Scanner(System.in);
        int num=s.nextInt()
        int num1=s.nextInt();
        int num2=num+num1;
        System.out.println(num2);
    }
}
```

Execute Mode, Version, Inputs & Arguments

JDK 11.0.4

Interactive

Stdin Inputs

4
4

CommandLine Arguments

Execute

Result

CPU Time: sec(s), Memory: kilobyte(s)

```
/Main.java:7: error: ';' expected
        int num=s.nextInt()
                           ^
1 error
```

```java
import java.util.*;
public class Main{
    public static void main(String args[]){
        Scanner s=new Scanner(System.in);
        int sum=0;
        int num=s.nextInt();
        int num1=s.nextInt();
        for(i=0;i<num;i++)
        {
            sum=sum+num1;
        }
        System.out.println(sum);
    }
}
```

Execute Mode, Version, Inputs & Arguments

JDK 11.0.4

Interactive

Stdin Inputs

3
4

CommandLine Arguments

Execute

esult

PU Time: sec(s), Memory: kilobyte(s)

```
/Main.java:10: error: cannot find symbol
        for(i=0;i<num;i++)
            ^
  symbol:   variable i
  location: class Main
```

For Multiple Files, Custom Library and File Read/Write, use our new - Advanced Java IDE

```java
import java.util.*;
public class Main
{
    public static void main(String args[])
    {
        Scanner s=new Scanner(System.in);
        int arr[]=new int[]{1,2,3,4,5};
        System.out.print(arr[7]);
    }
}
```

Execute Mode, Version, Inputs & Arguments

Execute

Result

CPU Time: 0.24 sec(s), Memory: 34744 kilobyte(s)

```
Exception in thread "main" java.lang.ArrayIndexOutOfBoundsException: Index 7 out of bounds for length 5
    at Main.main(Main.java:8)
```

Multithreading

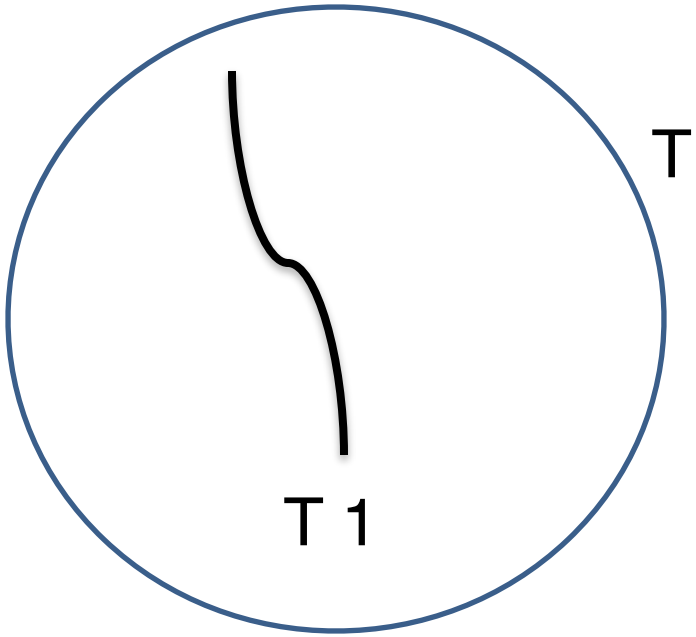
Why do we use threads ?

- multithreading is a specialized form of multitasking
- two distinct types of multitasking
  - Process – based
  - Thread-based

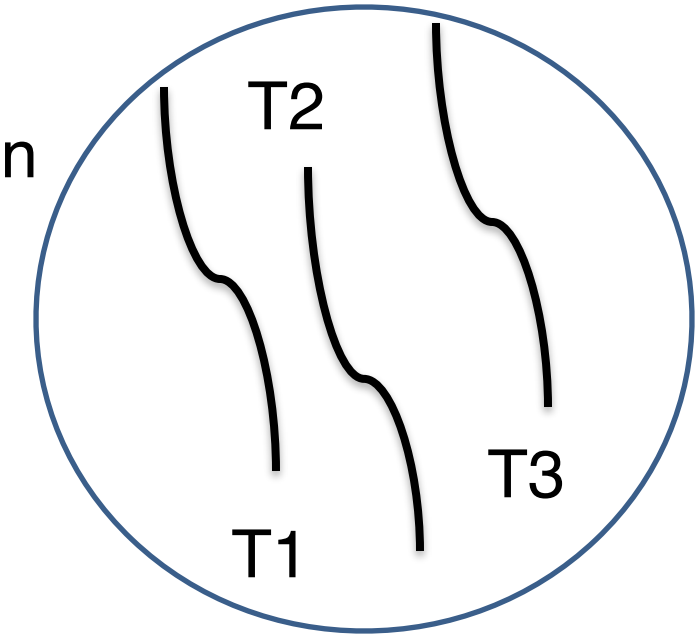
Thread Process

Single thread process

Multi thread process



Threads of execution

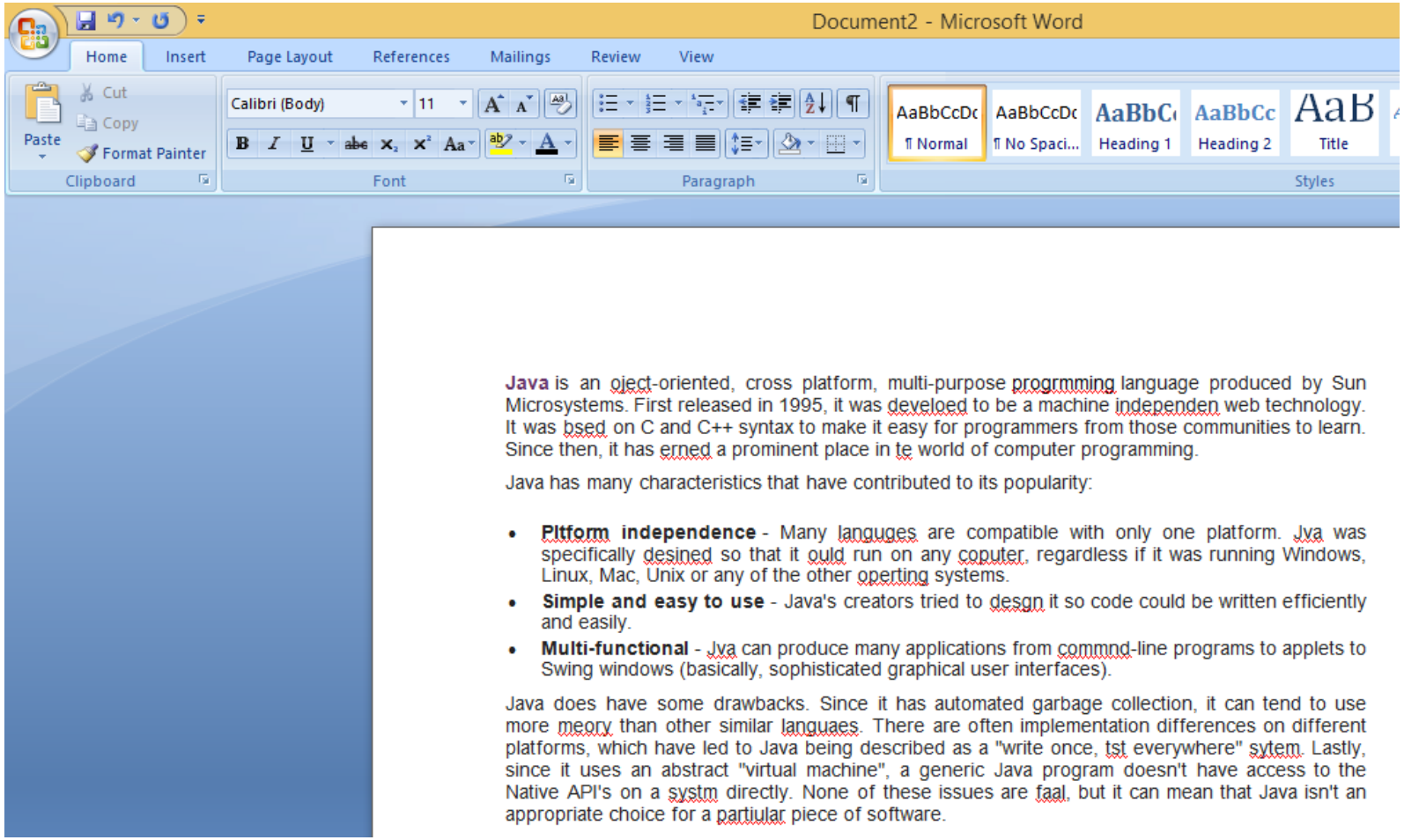


Single instruction stream

Multi instruction stream

Real time example:

Real time application:



MS word

↓

Typing [Spell Check]

↓

Threads

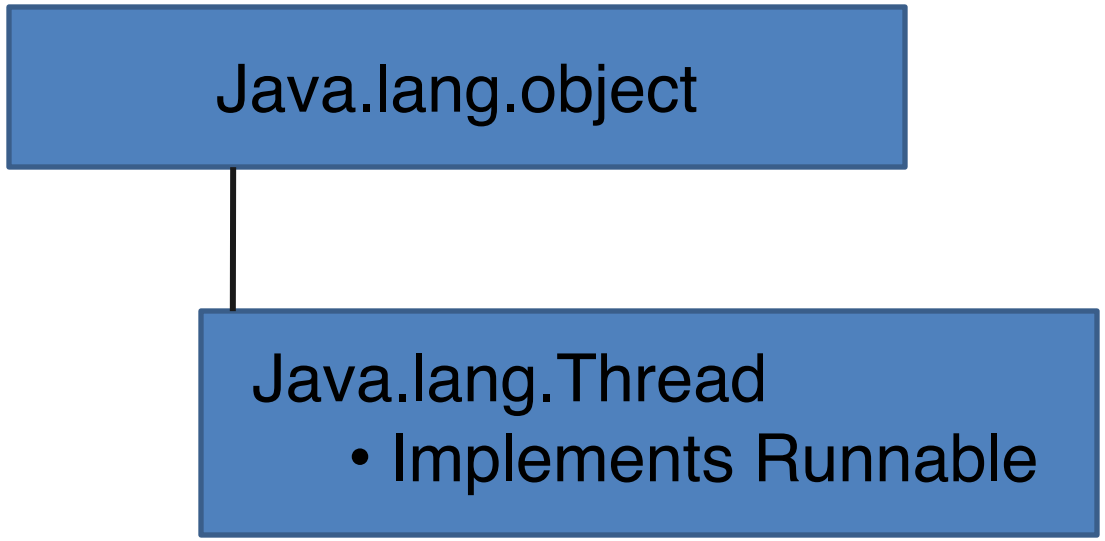
↓

Unit of a process

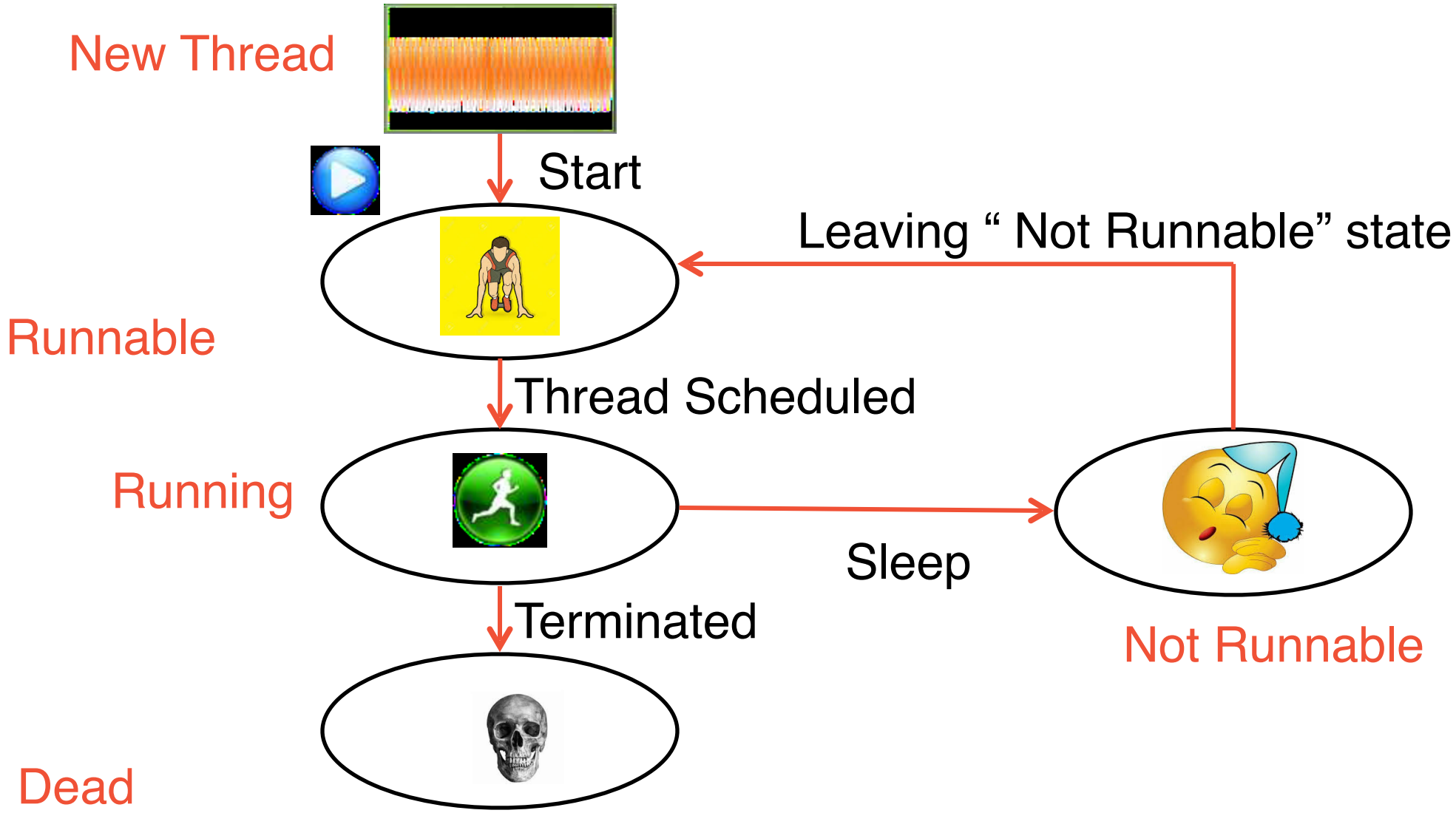
Thread class

Class - java.lang.Thread

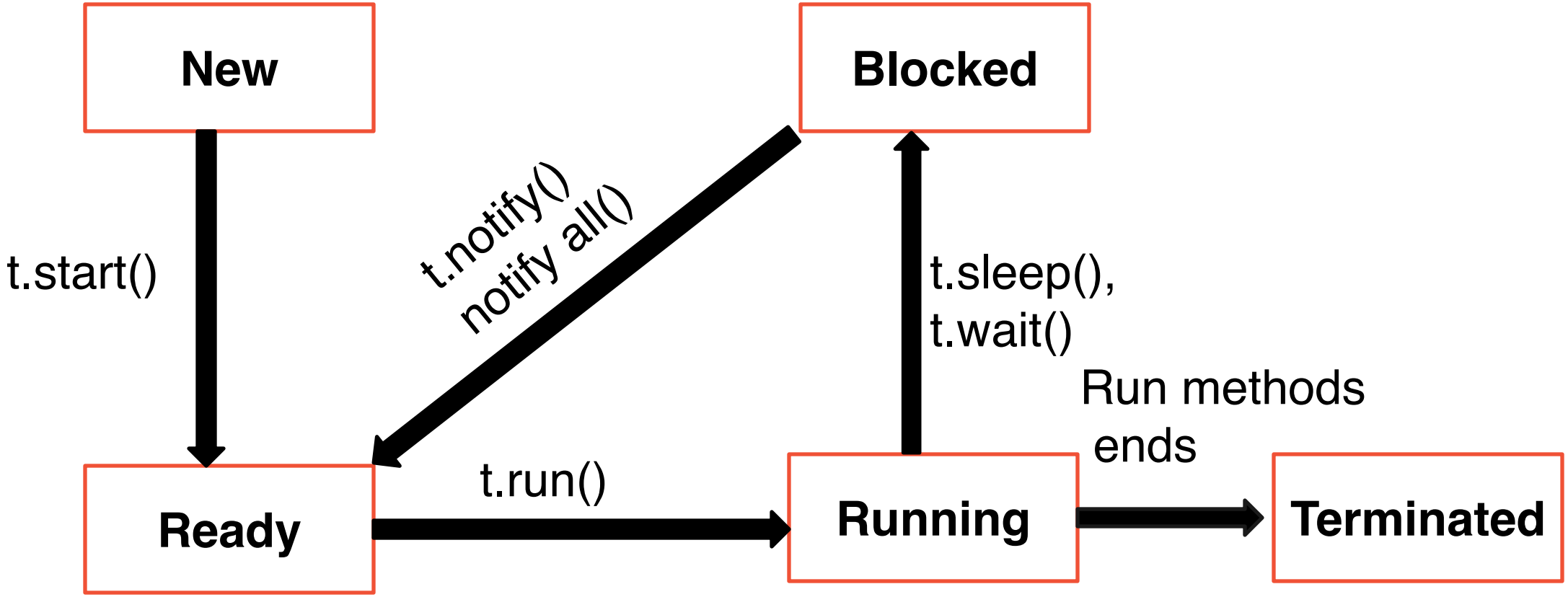
Thread Class Hierarchy:



Life cycle of a Thread



Thread - Methods



main thread

When a Java program starts executing:

- The main thread begins running
- The main thread is immediately created when **main()** commences execution

```java
//Predict the Output
import java.lang.Thread;          //Thread package
public class Main
{
    public static void main(String args[])
    {
        Thread t = Thread.currentThread();
        System.out.println("Current Thread :" + t);
        try
        {
            Thread.sleep(1);
        }
        catch (InterruptedException e)
        {
            System.out.println("Main Thread Interrupted");
        }
    }
}
```

Creating thread

Thread creating by extending the thread class

- java.lang.Thread

This can be achieved in any of the following two ways :

- extending the **Thread class**
- implementing the **Runnable interface**

```
//Predict the Output
import java.util.*;
class Test extends Thread
{
    public void run()
    {
        System.out.println("thread is running...");
    }
}
public class Main
{
    public static void main(String args[])
    {
        Test create=new Test();  //Object creation
        create.start();
    }
}
```

```java
//Predict the Output
import java.util.*;
class Test extends Thread
{
    public void run()
    {
        System.out.println("Process Error");
    }
}
public class Main
{
    public static void main(String args[])
    {
        Test create=new Test();
        create.start();
        create.run();
        create.run();
    }
}
```

```java
//Predict the Output
import java.util.*;
class Test extends Thread
{
    public void run()
    {
        System.out.println(" thread passed ");
    }
}
public class Main
{
    public static void main(String args[])
    {
        Test create=new Test();
        create.start();
        create.start();
    }
}
```

```java
//Predict the Output
import java.util.*;
public class Main
{
    public static void main(String args[])
    {
        Thread t = Thread.currentThread( );
        System.out.println("Current Thread :" + t);
        t.setName("Void");
        System.out.println("Current thread :" +t);
        try
        {
            Thread.sleep(1);
        }
        catch (InterruptedException e)
        {
            System.out.println("Main Thread Interrupted");
        }
    }
}
```

```java
//Predict the Output
import java.lang.Thread;
class example extends Thread {
    public void run() {
        try{
            System.out.println (Thread.currentThread());
        }
        catch (Exception e) {
            System.out.println ("Exception is caught");
        }
    }
}
public class Main{
    public static void main(String[] args) {
        int n = 4;
        for (int i=0; i<4; i++)
        {
            example object = new example();
            object.start();
        }
    }
}
```

```
//Predict the Output
import java.lang.Thread;
import java.util.Scanner;
class example extends Thread
{
    public void run()
    {
        try
        {
            System.out.println (Thread.currentThread());
        }
        catch (Exception e)
        {
            System.out.println ("Exception is caught");
        }
    }
}
```

```
1  public class Main
2  {
3      public static void main(String[] args)
4      {
5          Scanner s=new Scanner(System.in);
6          int num=s.nextInt();
7          for (int i=0; i<num; i++)
8          {
9              example object = new example();
10             object.start();
11         }
12     }
13 }
14
15
16
17
18
19
20
21
22
```

```java
//Predict the Output
import java.lang.Thread;
import java.util.Scanner;
class example extends Thread
{
    public void run()
    {
        try
        {
            System.out.println (Thread.currentThread());
        }
        catch (Exception e)
        {
            System.out.println ("Exception is caught");
        }
    }
}
```

```
1  public class Main
2  {
3      public static void main(String[] args)
4      {
5          Scanner s=new Scanner(System.in);
6          int num=s.nextInt();
7          for (int i=0; i<num; i++)
8          {
9              example object = new example();
10             object.start();
11             object.setName("Priya");
12             object.setPriority(6);
13         }
14     }
15 }
16
17
18
19
20
21
22
```

```java
//Predict the Output
import java.lang.Thread;
import java.util.Scanner;
class example extends Thread
{
    public void run()
    {
        try
        {
            System.out.println (Thread.currentThread());
        }

        catch (Exception e)
        {
            System.out.println ("Exception is caught");
        }
    }
}
```

```
1  public class Main
2  {
3      public static void main(String[] args)
4      {
5          Scanner s=new Scanner(System.in);
6          int num=s.nextInt();
7          String L1=s.next();
8          int num2=s.nextInt();
9          for (int i=0; i<num; i++)
10         {
11             example object = new example();
12             object.start();
13             object.setName(L1);
14             object.setPriority(num2);
15         }
16     }
17 }
18
19
20
21
22
```

```java
//Predict the Output
import java.util.*;
class example extends Thread
{
    public void run()
    {
        try
        {
            System.out.println (Thread.currentThread());
            System.out.println (Thread.currentThread().getId());
        }
        catch (Exception e)
        {
            System.out.println ("Exception is caught");
        }
    }
}
```

```
1  public class Main
2  {
3      public static void main(String[] args)
4      {
5          Scanner s=new Scanner(System.in);
6          int num=s.nextInt();
7          for (int i=0; i<num; i++)
8          {
9              example object = new example();
10             object.start();
11         }
12     }
13 }
14
15
16
17
18
19
20
21
22
```

```java
//Predict the Output
import java.util.*;
class Test {
    public static void m1() {
        System.out.println("Hello Visitors");
    }
}
class Create extends Test
{
    public void run() {
        System.out.println("Child Thread");
    }
}
public class Main{
    public static void main(String[] args) {
        Create t = new Create();
        t.m1();
        Thread t1 = new Thread(t);
        t1.start();
        System.out.println("Main thread");
    }
}
```

```java
//Predict the Output
import java.util.*;
class Test {
    public static void m1() {
        System.out.println("Hello Visitors");
    }
}
class Create extends Test implements Runnable
{
    public void run() {
        System.out.println("Child Thread");
    }
}
public class Main{
    public static void main(String[] args) {
        Create t = new Create();
        t.m1();
        Thread t1 = new Thread(t);
        t1.start();
        System.out.println("Main thread");
    }
}
```

```java
//Predict the Output
import java.lang.Runnable;
class Methods implements Runnable
{
    public void run() {
        try{
            System.out.println (Thread.currentThread());
        }
        catch (Exception e) {
            System.out.println ("Exception is caught");
        }
    }
}
public class Main {
    public static void main(String[] args) {
        int num = 4;
        for (int i=0; i<4; i++) {
            Thread object = new Thread(new Methods());
            object.start();
        }
    }
}
```

```java
//Predict the Output
import java.lang.Runnable;
public class Main
{
    public static void main(String[] args)
    {
        System.out.println(Thread.currentThread());
        System.out.println("Creating Runnable Instance");
        Runnable runnable = new Runnable()
        {
            @Override
            public void run()
            {
                System.out.println(Thread.currentThread().getName());
            }
        }
        System.out.println("Creating a Thread Instance");
        Thread thread = new Thread(runnable);
        System.out.println("Launching a Thread");
        thread.start();
    }
}
```

```java
//Predict the Output
import java.lang.Runnable;
public class Main
{
    public static void main(String[] args)
    {
        System.out.println(Thread.currentThread());
        System.out.println("Creating Runnable Instance");
        Runnable runnable = new Runnable()
        {
            @Override
            public void run()
            {
                System.out.println(Thread.currentThread().getName());
            }
        };
        System.out.println("Creating a Thread Instance");
        Thread thread = new Thread(runnable);
        System.out.println("Launching a Thread");
        thread.start();
    }
}
```

MCQ

```
//Predict the output
class Test implements Runnable
{
    public void run()
    {
        System.out.println("Method");
    }
}
public class Main
{
    public static void main(String[] args)
    {
        Thread create = new Thread();
        create.start();
        System.out.println("Main");
    }
}
```

Question 1

A) Main ✓

B) Method

C) Main
Method

D) Compilation error

```
//How many threads will be created for the following code?
class Test extends Thread
{
    public void run()
    {
        System.out.println("Run");
    }
}
public class Main
{
    public static void main(String[] args)
    {
        Test t = new Test();
        t.run();
    }
}
```

Question 2

A) Depend upon system

B) One thread created ✓

C) Two thread created

D) Compilation error

```
//Predict the output
class One extends Thread
{
        public void run()
        {
                for(int i=0; i<2; i++){
                        System.out.print(i);
                }
        }
}
public class Test
{
        public static void main(String args[])
        {
                Test t = new Test();
                t.call(new One());
        }
        public void call(One o)
        {
                o.start();
        }
}
```

Question 3

A) 0 0

B) 0 1 2

C) 01 ✓

D) Compilation error

```
1  //Predict the output
2  public class Test implements Runnable
3  {
4      public void run()
5      {
6          System.out.print("go");
7      }
8      public static void main(String arg[])
9      {
10         Thread t = new Thread(new Test());
11         t.run();
12         t.run();
13     }
14 }
15
16
17
18
19
20
21
22
```

Question 4

A) gogogo

B) gogo ✓

C) go

D) Compilation error

```
//Predict the output
public class Test extends Thread
{
    public static void main(String argv[])
    {
        Test t = new Test();
        t.run();
        t.start();
    }
    public void run()
    {
        System.out.println("run-test");
    }
}
```

Question 5

A) No output

B) run-test

C) run-test
run-test

D) Compilation error

THANK YOU