**Coding**

1. **Using Thread.Sleep() Method For Main Thread**

**// Java Program for sleeping the main thread.**

import java.io.\*;

import java.lang.Thread;

class GFG {

public static void main(String[] args){

// we can also use throws keyword followed by exception name for throwing the exception

try {

for (int i = 0; i < 5; i++) {

// it will sleep the main thread for 1 sec ,each time the for loop runs

Thread.sleep(1000);

System.out.println(i); // printing the value of the variable

}

}

catch (Exception e) { // catching the exception

System.out.println(e);

}

}

}

**Output**

0

1

2

3

4

**2. IllegalArguementException When Sleep Time Is Negative**

**// Java Program for showing how exception can occur if we pass the negative timeout value.**

import java.io.\*;

import java.lang.Thread;

class GFG {

public static void main(String[] args)

{

// we can also use throws keyword followed by exception name for throwing the exception

try {

for (int i = 0; i < 5; i++) {

// this will throw the

// IllegalArgumentException

Thread.sleep(-100);

// printing the value of the variable

System.out.println(i);

}

}

catch (Exception e) {

// catching the exception

System.out.println(e);

}

}

}

**Output**

java.lang.IllegalArgumentException: timeout value is negative

1. **Code snippet for yield method**

public class JavaYieldExp extends Thread {

public void run() {

for (int i=0; i<3 ; i++)

System.out.println(Thread.currentThread().getName() + " in control"); }

public static void main(String[]args) {

JavaYieldExp t1 = new JavaYieldExp();

JavaYieldExp t2 = new JavaYieldExp();

// this will call run() method

t1.start();

t2.start();

for (int i=0; i<3; i++) {

// Control passes to child thread

t1.yield();

System.out.println(Thread.currentThread().getName() + " in control");

} } }

**Output:**

main in control

main in control

main in control

Thread-0 in control

Thread-0 in control

Thread-0 in control

Thread-1 in control

Thread-1 in control

Thread-1 in control

1. **Example of currentThread() method**

class TestMultiNaming2 extends Thread

{

public void run()

{

System.out.println(Thread.currentThread().getName());

}

public static void main(String args[])

{

TestMultiNaming2 t1=new TestMultiNaming2();

TestMultiNaming2 t2=new TestMultiNaming2();

t1.start();

t2.start();

}

}

**Output:**

Thread-0

Thread-1

1. **Example of naming a thread : Using setName() Method**

class TestMultiNaming1 extends Thread{

public void run(){

System.out.println("running...");

}

public static void main(String args[])

{

TestMultiNaming1 t1=new TestMultiNaming1();

TestMultiNaming1 t2=new TestMultiNaming1();

System.out.println("Name of t1:"+t1.getName());

System.out.println("Name of t2:"+t2.getName());

t1.start();

t2.start();

t1.setName("Sonoo Jaiswal");

System.out.println("After changing name of t1:"+t1.getName());

}

}

Output:

Name of t1:Thread-0

Name of t2:Thread-1

After changing name of t1:Sonoo Jaiswal

running...

running...