**Lab Sheet 4**.**1** Create a single thread by extending Thread class

### Solution:

Step 1: Start

Step 2: Create a class MyThread that extends builtin class Thread

Step 3: Override run() to define work of thread. Use Thread.CurrentThread() to print the thread is running currently

Step 4: Create object of MyThread

Step 5: Assign a name to thread using setName(“Name”)

Step 6: Call start() to start the thread

Step 7: Stop

class MyThread extends Thread {

// run() method to perform action for thread.

public void run()

{

int a= 10;

int b=12;

int result = a+b;

System.out.println(Thread.currentThread()+" started running..");

System.out.println("Sum of two numbers is: "+ result);

System.out.println(Thread.currentThread()+" completed..");

}

}

public class TestThread {

public static void main( String args[] )

{

System.out.println(Thread.currentThread()+" started");

// Creating instance of the class extend Thread class

MyThread t = new MyThread();

t.setName("first");

//calling start method to execute the run() method of the Thread class

t.start();

System.out.println(Thread.currentThread()+" completed");

}

}

**Lab Sheet 4.2:** Create 3 threads 1st, 2nd and 3rd to print numbers 5 to 1 concurrently by extending Thread Class.

Requirement:

* Override run() to print 5 to 1 using for loop
* Use sleep() for switching the context to other threads
* Use setName() to set the name of Thread or Use constructor Thread() to set the name of Thread

## Sol:

## class MyThread extends Thread {

## String name;

## MyThread (String name){

## setName(name); or // super(name);

## this.name=name;

## System.out.println( "A New thread: " + name + "is created\n" );

## }

## public void run() {

## try {

## for(int j = 5; j > 0; j--) {

## System.out.println(name + ": " + j);

## Thread.sleep(1000);

## }

## }catch (InterruptedException e) {

## System.out.println(name + " thread Interrupted");

## }

## System.out.println(name + " thread exiting.");

## }

## }

## public class TestMultiThread {

## public static void main(String args[]) {

## MyThread t1=new MyThread(“one”);

## MyThread t2=new MyThread(“two”);

## MyThread t3=new MyThread(“three”);

## t1.start();

## t2.start();

## t3.start();

## try {

## Thread.sleep(8000);

## } catch (InterruptedException excetion) {

## System.out.println("Inturruption occurs in Main Thread");

## }

## System.out.println("We are exiting from Main Thread");

## }

**Lab Sheet 4.3:**Create 3 threads 1st, 2nd and 3rd to print factorial of three different numbers concurrently by extending Thread Class.

Requirement:

* Override run() to print factorial using for loop
* Use sleep() for switching the context to other threads
* Use constructor Thread() to set the name of Thread
* Demonstrate join() and isAlive() method

## Sol:

class MyThread extends Thread {

String name;

int number;

long fact=1;

MyThread (int number,String name){

super(name); //calling Thread()

this.number=number;

this.name=name;

System.out.println( "A New thread: " + name + " is created\n" );

}

public void run() {

try {

for(int i = 1; i <= number; i++) {

System.out.println(name + " calculating factorial");

fact=fact\*i;

Thread.sleep(1000);

}

}catch (InterruptedException e) {

System.out.println(name + " thread Interrupted");

}

System.out.println(name + " calculated factorial "+fact);

} }

public class TestMultiThread {

public static void main(String args[]) {

MyThread t1=new MyThread(5,"one");

MyThread t2=new MyThread(4,"two");

MyThread t=new MyThread(3,"three");

t1.start();

t2.start();

t3.start();

System.out.println("1st Alive : "+t1.isAlive());

System.out.println("2nd Alive : "+t2.isAlive());

System.out.println("3rd Alive : "+t3.isAlive());

try {

t1.join();

t2.join();

t3.join();

System.out.println("1st Alive : "+t1.isAlive());

System.out.println("2nd Alive : "+t2.isAlive());

System.out.println("3rd Alive : "+t3.isAlive());

} catch (InterruptedException excetion) {

System.out.println("Inturruption occurs in Main Thread");

}

System.out.println("We are exiting from Main Thread");

}

}