Write a program to implement getName(), getName(), getPriority(), setPriority(),isAlive(), yield in thread.

class MyThread extends Thread {

 // Constructor

 public MyThread(String name) {

 super(name); // Setting thread name using Thread class constructor

 }

 @Override

 public void run() {

 // Code that will run in the thread

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

 // Check if the thread is alive

 if (Thread.currentThread().isAlive()) {

 System.out.println(Thread.currentThread().getName() + " is alive.");

 }

 // Yield to allow other threads to run

 Thread.yield();

 // Print the current thread priority

 System.out.println(Thread.currentThread().getName() + "'s priority: " + Thread.currentThread().getPriority());

 }

}

public class ThreadExample {

 public static void main(String[] args) {

 // Create two threads with different priorities and names

 MyThread t1 = new MyThread("Thread 1");

 MyThread t2 = new MyThread("Thread 2");

 // Set different priorities for the threads

 t1.setPriority(Thread.MIN\_PRIORITY);

 t2.setPriority(Thread.MAX\_PRIORITY);

 // Start both threads

 t1.start();

 t2.start();

 // Print the names and priorities of the threads

 System.out.println("Thread 1 Name: " + t1.getName());

 System.out.println("Thread 1 Priority: " + t1.getPriority());

 System.out.println("Thread 2 Name: " + t2.getName());

 System.out.println("Thread 2 Priority: " + t2.getPriority());

 // Wait for threads to finish

 try {

 t1.join();

 t2.join();

 } catch (InterruptedException e) {

 e.printStackTrace();

 }

 // Final check if the threads are still alive

 System.out.println("Is Thread 1 alive? " + t1.isAlive());

 System.out.println("Is Thread 2 alive? " + t2.isAlive());

 }

}