JAVA THREAD 3

030523313 - Network programming

Asst. Prof. Dr. Choopan Rattanapoka

ปัญหาที่ 1 : ในการใช้งาน Thread

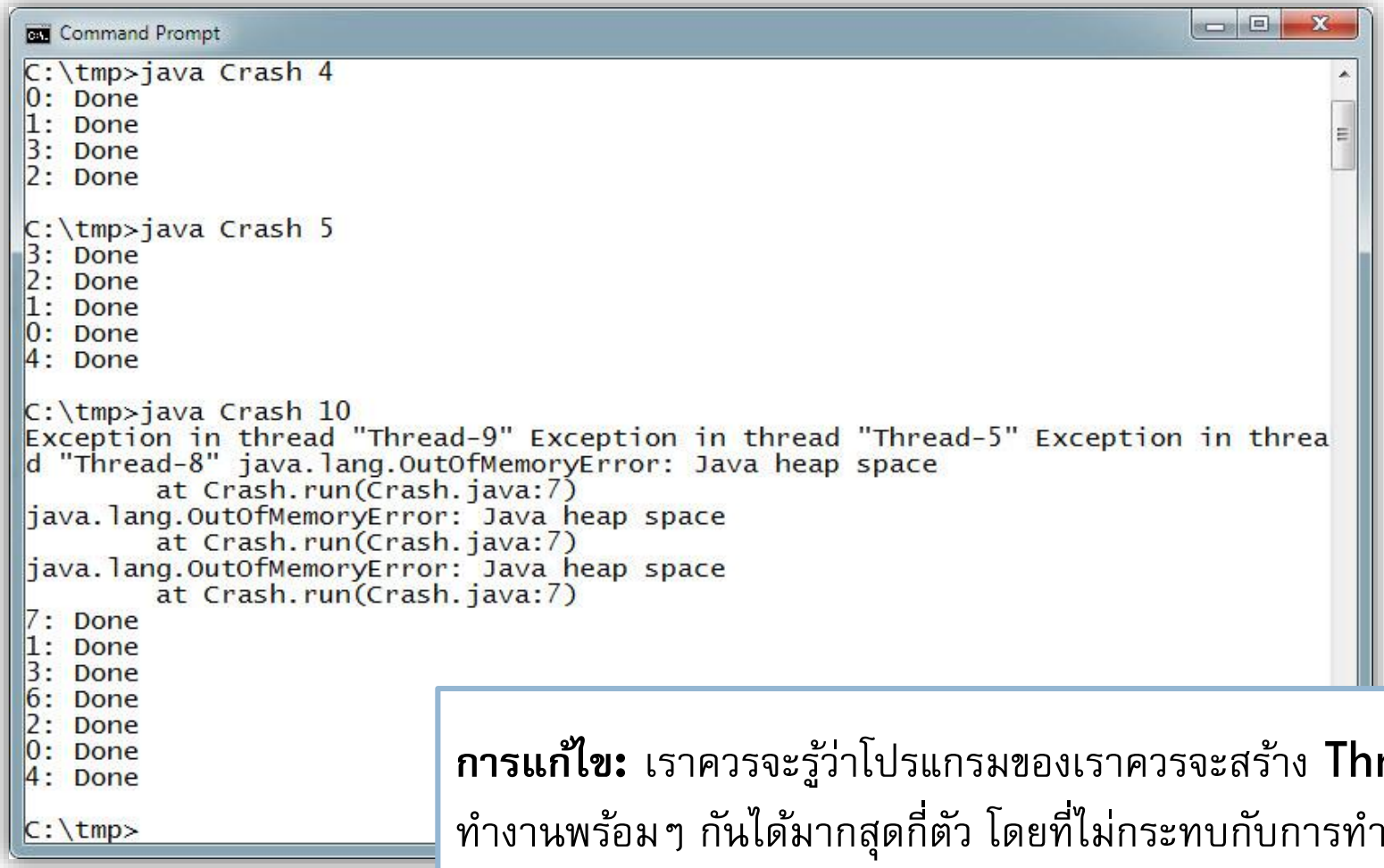
- ในการทำงานจริงกับ Thread สำหรับ Application ทั่วไป และServer Application การสร้าง Thread โดยไม่มีการจำกัดจำนวนอาจจะทำให้เครื่องคอมพิวเตอร์ที่เรียกใช้งานโปรแกรมมีปัญหา

Thread แต่ละตัวจะใช้จองเนื้อที่หน่วยความจำ 50 MB

จำลองให้ทำงาน 2 วินาที

```
public class Crash implements Runnable {
    int id;

    public Crash(int id) { this.id = id; }

    public void run() {
        int buffer[] = new int[50000000];
        try {
            Thread.sleep(2000);
        }   catch(Exception e) { }
        System.out.println(id + ": Done");
    }

    public static void main(String[] args) {
        int n = Integer.parseInt(args[0]);
        for(int i = 0; i < n; i++) {
            Crash c = new Crash(i);
            Thread t = new Thread(c);
            t.start();
        }
    }
}
```

ปัญหาที่ 1: เรียกใช้งานโปรแกรม



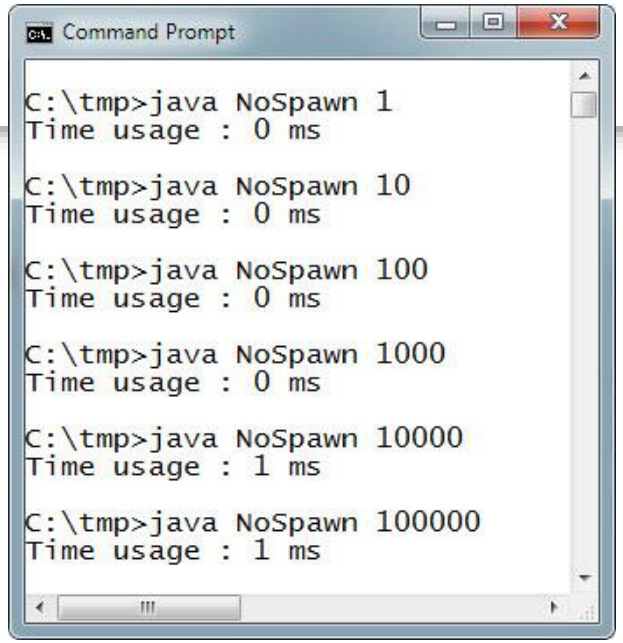
การแก้ไข: เราควรจะรู้ว่าโปรแกรมของเราควรจะสร้าง Thread ให้ทำงานพร้อมๆ กันได้มากสุดกี่ตัว โดยที่ไม่กระทบกับการทำงานมากนัก

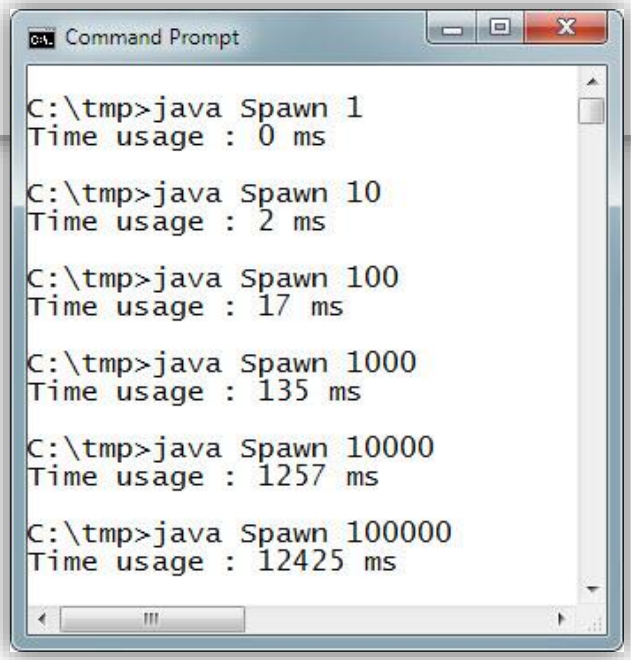
ปัญหาที่ 2 : ในการใช้งาน Thread

```
public class Spawn implements Runnable {
    public void run() { }

    public static void main(String[] args) {
        int n = Integer.parseInt(args[0]);
        long startTime = System.currentTimeMillis();
        for(int i = 0; i < n; i++) {
            Spawn s = new Spawn();
            Thread t = new Thread(s);
            t.start();
            try {
                t.join();
            } catch(Exception e) { }
        }
        long stopTime = System.currentTimeMillis();
        System.out.println("Time usage : " + (stopTime - startTime) + " ms");
    }
}
```

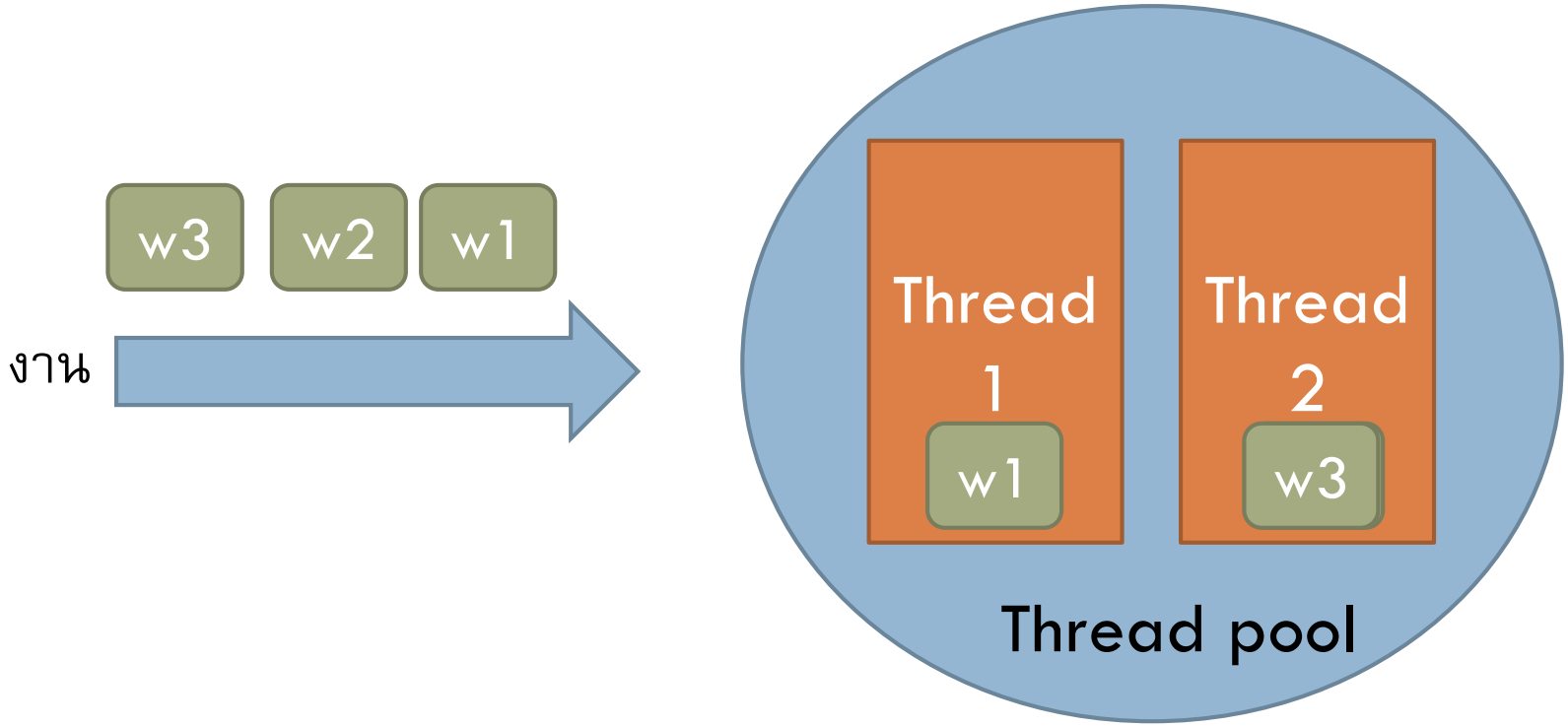
ปัญหาที่ 2: เรียกใช้งานโปรแกรม

```java
public class NoSpawn  {

    public static void main(String[] args) {
        int n = Integer.parseInt(args[0]);
        long startTime = System.currentTimeMillis();
        for(int i = 0; i < n; i++) { }
        long stopTime = System.currentTimeMillis();
        System.out.println("Time usage : " + (stopTime - startTime) + " ms");
    }
}
```





การแก้ไขปัญหา ด้วย Thread Pool

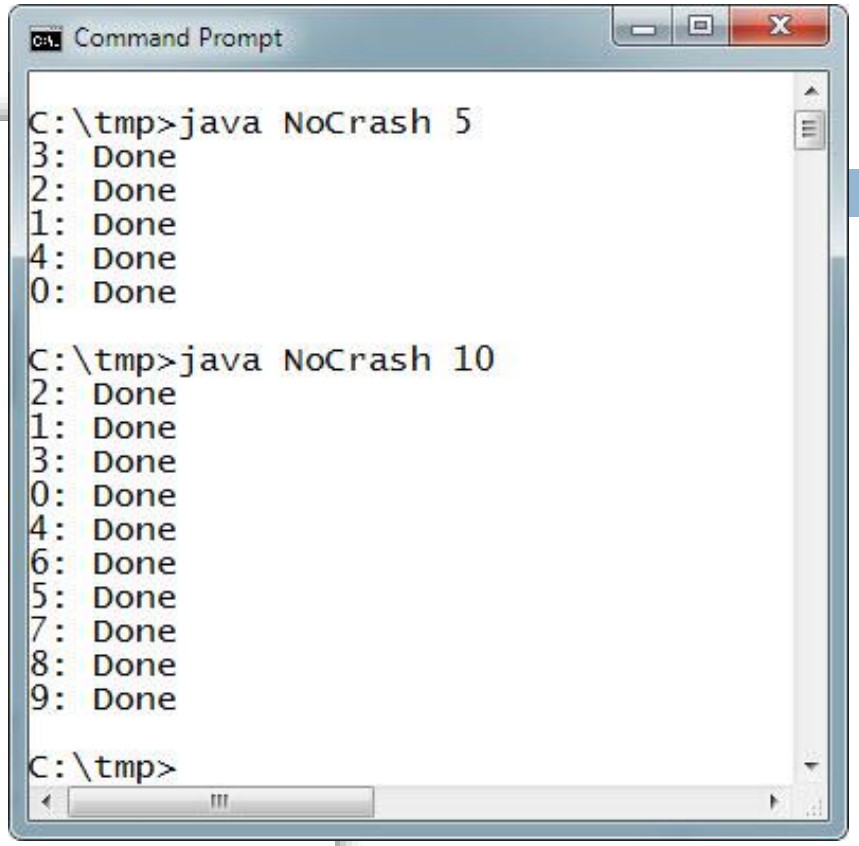


Java ใจดีมี Class มาให้ใช้งาน

- มี 2 class ที่ทำงานร่วมกันเพื่อทำงานในรูปแบบของ Thread Pool
  - **java.util.concurrent.Executors**
    - เป็น class ที่มี static method สำหรับสร้าง ExecutorService
    - เมธอด newFixedThreadPool(int nThread) สำหรับสร้าง ThreadPool ที่มี thread ภายในรอการทำงานจำนวน nThread ตัว
    - เมธอด newSingleThreadExecutor() สำหรับสร้าง Threadpool ที่มี Thread เดียว
  - **java.util.concurrent.ExecutorService**
    - เป็น class ที่ทำหน้าที่เป็น ThreadPool
    - เมธอด execute(Runnable r) ใช้สำหรับเอางาน r เข้าไปเข้าคิวรอการทำงาน
    - เมธอด shutdown() เมื่อต้องการหยุดการให้บริการ
    - เมธอด isTerminated() ตรวจสอบว่า ExecutorService ได้ปิดการทำงานเสร็จแล้วหรือไม่

แก้ไขปัญหาที่ 1

```java
import java.util.concurrent.*;

public class NoCrash implements Runnable {
    int id;

    public NoCrash(int id) { this.id = id; }

    public void run() {
        int buffer[] = new int[50000000];
        try {
            Thread.sleep(2000);
        }   catch(Exception e) { }
        System.out.println(id + ": Done");
    }

    public static void main(String[] args) {
        int n = Integer.parseInt(args[0]);
        ExecutorService es = Executors.newFixedThreadPool(5);
        for(int i = 0; i < n; i++) {
            NoCrash c = new NoCrash(i);
            es.execute(c);
        }
        es.shutdown();
    }
}
```



แก้ไขปัญหาที่ 2

```java
import java.util.concurrent.*;

public class SpawnThreadPool implements Runnable {
    public void run() { }

    public static void main(String[] args) {
        int n = Integer.parseInt(args[0]);
        long startTime = System.currentTimeMillis();

        ExecutorService executor = Executors.newSingleThreadExecutor();
        for(int i = 0; i < n; i++) {
            SpawnThreadPool s = new SpawnThreadPool();
            executor.execute(s);
        }
        executor.shutdown();
        while(!executor.isTerminated()) {}
        long stopTime = System.currentTimeMillis();
        System.out.println("Time usage : " + (stopTime - startTime) + " ms");
    }
}
```

ผลการรันเปรียบเทียบ