นาย อนุชิต ศุภวาลย์ 5907101003

**แบบฝึกหัด บทที่ 4**

**1. (5 คะแนน) เรื่อง User-Level Thread และ Kernel-Level Thread**

* **จงอธิบายความแตกต่างระหว่าง User-Level Thread และ Kernel-Level Thread**

- user-level threads ระบบปฏิบัติการจะวางเธรดทั้งหมดของ Multithread process ไว้ใน Execution context เพียงตัวเดียว และ kernel-level threads ระบบปฏิบัติการจะวางเธรดแต่ละตัวไว้ใน Execution context ของตัวเอง

* **สถานการณ์ใดที่เหมาะสมกับการใช้ User-Level Thread และ Kernel-Level Thread**

- User-Level Thread เหมาะกับ Many-to-One แต่ Kernel-Level Thread เหมาะกับ One-to-One

* **สถานการณ์ใดที่เหมาะสมกับการใช้ Kernel-Level Thread**

- เหมาะกับ One-to-One

**2.** **(5 คะแนน)** **สถานการณ์ใดที่ Multithreaded Solution ที่ใช้ Multiple Kernel Thread ทำให้ระบบมีประสิทธิภาพดีกว่าการใช้ Single-Threaded Solution บนระบบที่มีโพรเซสเซอร์เดียว จงยกตัวอย่างพร้อมคำอธิบาย**

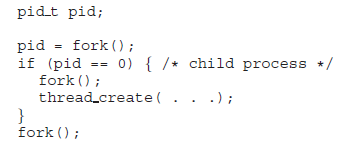
- ไม่ว่า Thread ไหนจะไม่ทำงาน แต่ Thread อื่นๆจะสามารถทำงานต่อไปได้ แต่ถ้าเป็น Single Thread จะไม่สามารถ Request ได้

**3. (5 คะแนน)** **สิ่งใดต่อไปนี้ Register Values , Heap Memory , Global Variables , Stack Memory ที่ถูกแชร์ระหว่าง Multithreaded Process**

- Global Variables คือ ตัวแปรที่กำหนดหรือประกาศไว้นอกฟังก์ชันใด ๆ ทุกฟังก์ชัน สามารถนำตัวแปรประเภท Global ไปใช้ได้ทุกฟังก์ชัน

- Heap Memory เป็นพื้นที่สำหรับการเรียกใช้แบบ dynamic ตรงข้ามกับการใช้งานแบบ stack  หรือเรียกอีกอย่างหนึ่งว่า “free store “ ผู้ใช้เป็นผู้ใช้ จองใช้งานเองไม่ได้ใช้งานแบบ อัตโนมัติ โดย function

**4. (5 คะแนน) จากส่วนของโค้ดต่อไปนี้**



* **มีโปรเซสที่สร้างขึ้นกี่โปรเซส**

- 6 โปรเซส

* **มี Thread ที่สร้างขึ้นกี่ Thread**

- 8 Thread

**5. (5 คะแนน) Thread Pool คืออะไร จงอธิบาย**

- Thread เป็นการทำงานของโปรแกรมที่สามารถทำงานแบบ concurrence (แบบขนาน) โดยแต่ละ thread จะทำงานไปพร้อมกัน โดยปกติการทำงานของโปรแกรมนั้นจะทำงานทีละบรรทัด และต้องรอให้บรรทัดก่อนหน้าเสร็จก่อน โปรแกรมจึงจะประมวลผลในบรรทัดต่อไป การใช้ thread นั้นสามารถทำให้โปรแกรมทำงานพร้อมกันได้ในแต่ละ thread ที่ถูกสร้างขึ้นภายในโปรแกรม ตัวอย่างการใช้งาน มักจะพบในโปรแกรมที่สามารถแบ่งงานกันทำได้ หรือโปรแกรม server และ client โดยที่ server จะสร้าง thread ขึ้นมาเพื่อจัดการกับแต่ละ client ที่เชื่อมต่อเข้ามา และใช้กับการพัฒนาเกม เป็นต้น

**6. (20 คะแนน) จงหาตัวอย่างของโปรแกรมที่เขียนด้วย Java Thread และ OpenMP แสดงโค้ดพร้อมผลลัพธ์ของการรันที่ได้ และอธิบายพฤติกรรมของโปรแกรม**

**Java Thread**

public class ThreadExample {

public static void main(String[] args){

for (int i = 0; i < 4; i++) {

Thread t1 = new Thread(new HelloMessage(i + 1));

t1.start();

}

}

}

class HelloMessage implements Runnable {

public int threadNum;

public HelloMessage(int threadNum) {

this.threadNum = threadNum;

}

@Override

public void run() {

System.out.println("Hello from thread " + threadNum);

}

}

ผลลัพธ์

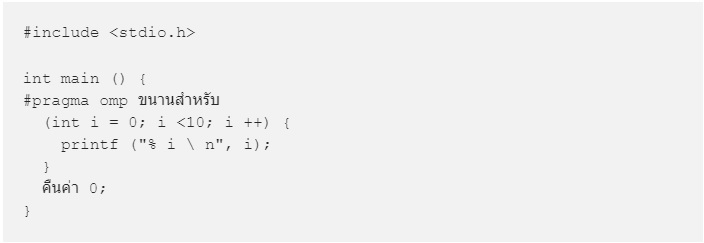
Hello from thread 1

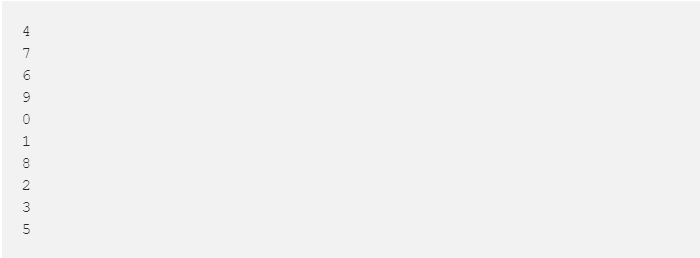
Hello from thread 4

Hello from thread 3

Hello from thread 2

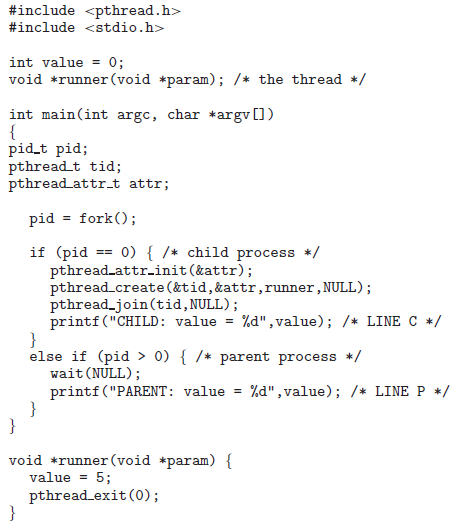
ในตัวอย่างเป็นโปรแกรมในการแสดงผลข้อความสวัสดี โดยคลาส HelloMessage นั้น Implement มาจาก Interfaces Runnable และในคลาสนี้มีตัวแปร threadNum สำหรับเก็บหมายเลขของ Thread ในการทำงานของ Thread นั้นเราได้ใช้คำสั่ง For loop สำหรับสร้าง Thread 4 Thread และกำหนดหมายเลขให้กับ Thread โดยเริ่มจาก 0 ในการทำงานนั้น Thread แต่ละ Thread จะทำการแสดงข้อความ Hello ตามด้วยหมายเลข Thread ของมันเอง ซึ่งในการแสดงผลข้อความในแต่ละ Thread นี้เกิดขึ้นพร้อมๆ เพราะว่าการทำงานของ Thread นั้นเป็นแบบคู่ขนานกันไป และนี่เป็นผลลัพธืการทำงานของโปรแกรม คุณจะเห็นว่าลำดับการทำงานของ Thread นั้นไม่เรียงจาก 1 ไป 4 เพราะว่า Thread นั้นทำงานแบบ Parallelism

**OpenMP**

ผลลัพธ์

ตัวเลขไม่เป็นระเบียบเนื่องจากการวนซ้ำแต่ละครั้งในลูปจะดำเนินการในเวลาที่แตกต่างกันเล็กน้อยในแบบคู่ขนาน

**7. (10 คะแนน) ทดลองโค้ดภาษาซีต่อไปนี้ ผลลัพธ์ที่ได้จาก Line C และ Line P เป็นอย่างไร**



* เมื่อเกิดการทำงานของ child process จะมีการทำงานของ runner และจะทำให้ value มีค่าเท่ากับ 5 แต่เมื่อจบการทำงาน value จะมีค่า = 0

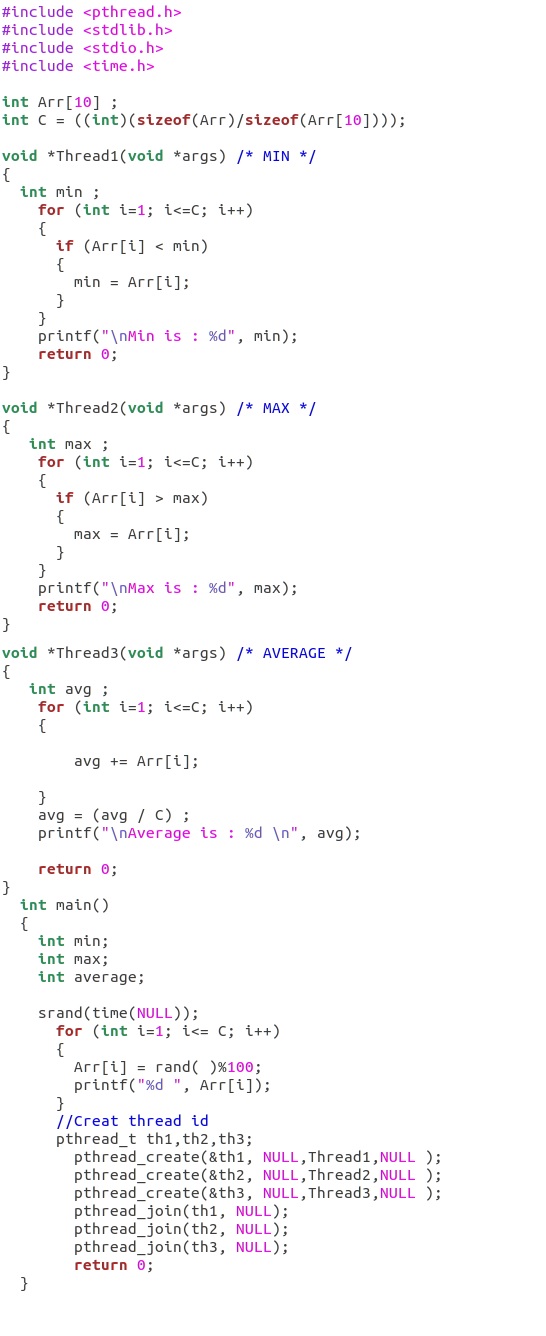
**8. (30 คะแนน) จงเขียนโปรแกรมแบบ Multithread โดยทีการทำงานดังนี้**

* **Main Thread สร้างเลขจำนวนเต็มแบบสุ่ม 10 ค่า แล้วแสดงออกทางจอภาพ**
* **Main Thread เตรียมตัวแปร min , max และ avg สำหรับเก็บค่าต่ำสุด , สูงสุด , และค่าเฉลี่ยตามลำดับ**
* **สร้าง Thread ย่อย 3 Thread มีหน้าที่ดังนี้ จากเลขจำนวนเต็มที่สุ่มมาจาก Main Thread**

1. **Thread#1: ค้นหาค่าต่ำสุด เก็บลงใน min**
2. **Thread#2: ค้นหาค่าสูงสุด เก็บลงใน max**
3. **Thread#3: ค้นหาค่าเฉลี่ย เก็บลงใน avg**

* **Main Thread แสดงผลลัพธ์ของ min , max , avg ออกทางจอภาพ**

Code



ผลลัพธ์

