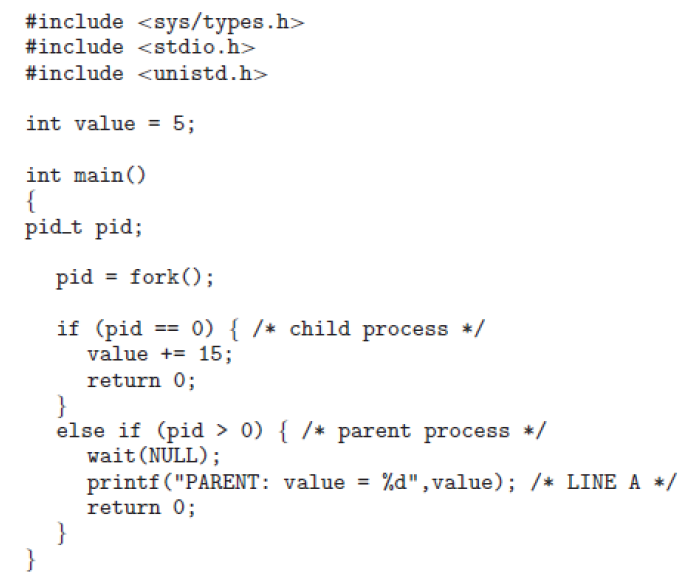
5907101012 มุขยา วงศ์ดาราพานิช

แบบฝึกหัด บทที่ 3

1. (10 คะแนน) จากโค้ดภาษาซีด้านล่างต่อไปนี้ โปรแกรมที่บรรทัด Line A ให้เอาท์พุต

เป็นค่าใด จงทดลองและอธิบายผลการทดลอง

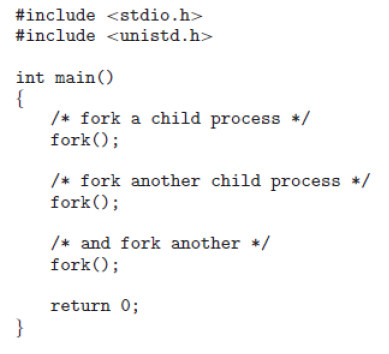


ตอบ : PARENT: value = 5

อธิบาย : ฟังชั่น fork จะสร้าง process จากตัว parent โดยระหว่างที่ process ใหม่กำลังทำงาน pid value จะกลับไปอยู่ที่ 0 จากนั้น add 15 ลงไปในตัวแปร value

และเนื่องจากเงื่อนไข pid > 0 จึงทำให้ ค่า value ใน parent = 5

2. (5 คะแนน) จากโค้ดภาษาซีต่อไปนี้ มีโปรเซสทั้งหมดกี่โปรเซส (นับรวมโปรเซสเริ่มต้นด้วย) จงอธิบาย



ตอบ : มีทั้งหมด 8 process เป็น parent 1 และ forkครั้งแรกอีก 1 เป็น 2

จากนั้น child 1 fork ก็จะมาอีก 2 รวมกับของเดิม 2 = 4

จากนั้น child 2 ทั้งสองก็ทำการ fork จนออกมาอีกอย่างละ 2 รวมกันเป็น 4

สรุป parent = 1 , child1 = 1, child2 = 2, child3 = 4

1+1+2+4 = 8

3. (5 คะแนน) เมื่อมีการสร้างโปรเซสใหม่โดยใช้คาสั่ง fork มีการแชร์สถานะ (State) ของสิ่งใดต่อไปนี้ Heap, Stack, และ Shared Memory Segment ระหว่างโปรเซสแม่กับโปรเซสลูก

ตอบ : ระหว่าง process แม่กับลูกจะมีการแชร์ shared memory segments ลงบนตัวที่พึ่งทำการ fork ใหม่

4. (5 คะแนน) จงอธิบายความหมายของ Short-Term, Medium-Term และ Long-Term Scheduling

ตอบ : Short-Term คือ ตัวจัดการคิวหรือตารางแบบสั้น โดยจะใช้วิธีเลือก process ที่รอทำงานอยู่(ready)นำมาใช้ในการประมวลผล

Medium-term คือ ตัวจัดการคิวหรือตารางระยะกลาง โดยวิธีการจัดการคือการย้าย process ออกจากหน่วยความจำ เพิ่มประสิทธิภาพในการทำงานของตัวประมวลผล โดยจะเน้นการย้าย process และ ลบ processออกจาก memory

Long-Term คือ ตัวจัดการคิวหรือตารางระยะยาว โดยจะเลือก process จาก queue และโหลดลงไปใน memory เพื่อทำการ execution

5. (5 คะแนน) จงอธิบายสิ่งที่ Kernel ทำเมื่อเกิด Context-Switch ระหว่างโปรเซส

ตอบ : อันดับแรกขึ้นอยู่กับ OS แต่ละตัวก็จะทำงานแตกต่างกันออกไป แต่ไอเดีย ประมาณว่าเป็นการ switch cpu จาก context ใน processนึง ไปยัง context ของ process อื่น

โดยขั้นตอนในการ switch จะต้องมีการ save->switch->load เพื่อทำการประมวลผลอีกครั้ง

6. (5 คะแนน) โปรเซส init ในระบบปฏิบัติการ Unix หรือ Linux คืออะไร มีหน้าที่อย่างไร จงอธิบายพอสังเขป

ตอบ : init หรือ initialization ใน unix หรือ linux จะเกิดขึ้นครั้งแรกตอนทำการ booting computer system โดยจะทำหน้าที่เป็น backgroup process จนกว่าจะทำการปิดเครื่อง

7. (5 คะแนน) Zombie Process และ Orphan Process คืออะไร เกิดขึ้นได้เมื่อใด จงอธิบาย

ตอบ : Zombie process เกิดขึ้นเมื่อ parent process ไม่ใช้ระบบ wait system call หลังจาก child process ตาย เพื่อให้ออกจากการประมวลผล ทำให้เกิดการรัน child process ค้างไว้ไม่จบสิ้น

Orphan Process เกิดขึ้นเมื่อ child process ยังทำงานอยู่ในขณะที่ parent process ตายแล้วสาเหตุมาจาก parent process ไม่รอการทำงานของ child process ทำการ exit ก่อนนั่นเอง

8. (30 คะแนน) จากเรื่อง Interprocess Communication จงเขียนโปรแกรมเพื่อคานวณ Collatz Conjecture จากอินพุต n ที่มีค่าบวกใดๆ โดย n ตัวใหม่คานวณได้ดังต่อไปนี้ จนกว่า n จะมีค่าเป็น 1

n = 3\*n + 1 if n is odd

n = n/2 if n is even

เช่น ถ้าเริ่มต้น n = 35 ผลลัพธ์ที่ได้คือ 35, 106, 53, 160, 80, 40, 20, 10, 5, 16, 8, 4, 2, 1

ให้เขียนโปรแกรมโดยใช้ภาษาซี และใช้ POSIX Shared Memory ในโปรเซสหลักมีขั้นตอนดังนี้

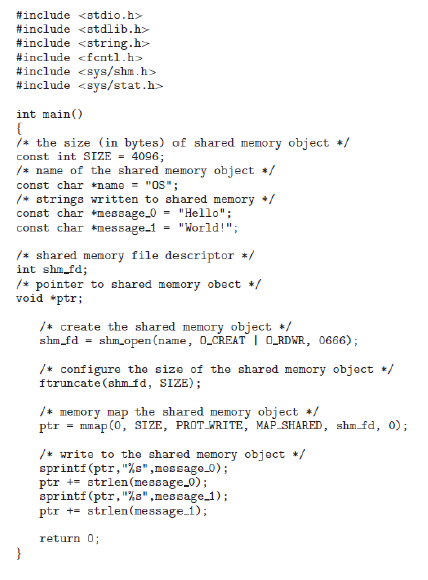
1. สร้าง Shared-Memory Object (shm open(), ftruncate(), and mmap())

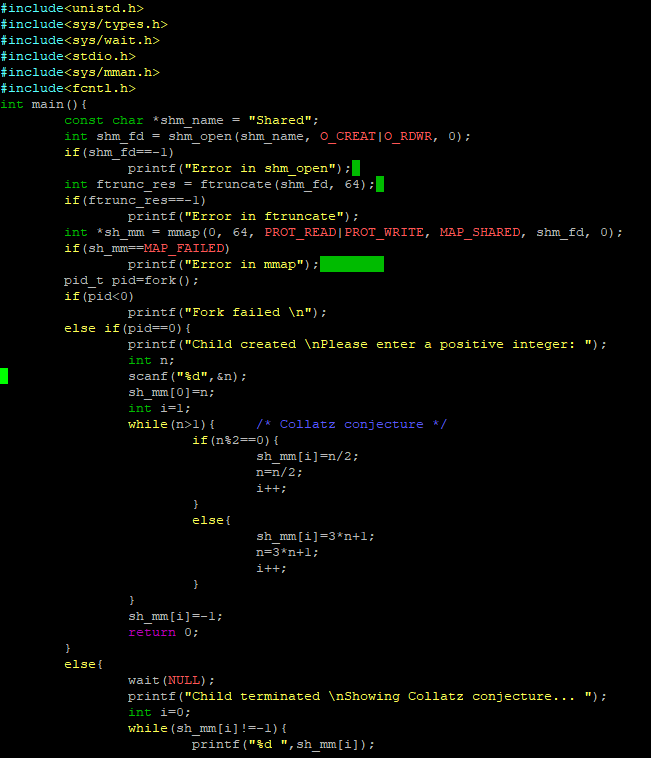
2. สร้างโปรเซสลูกเพื่อคานวณ n โดยที่โปรเซสลูกเขียนค่า n ที่คานวณได้ในแต่ละรอบลงใน Shared-Memory Object

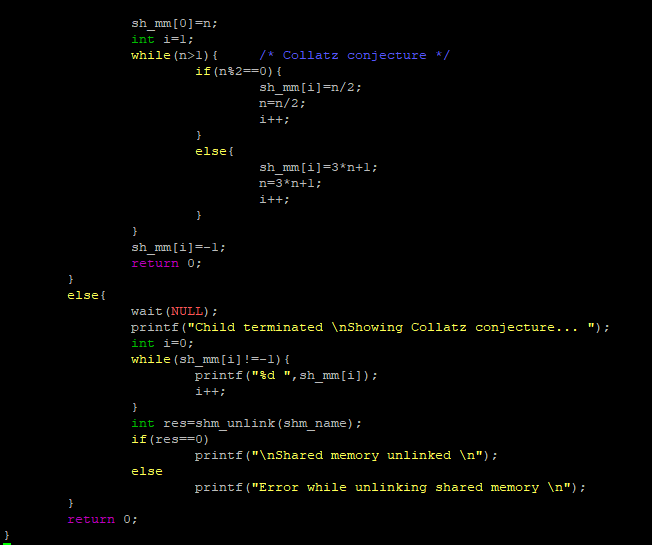
3. รอจนโปรเซสลูกเสร็จ แล้วแสดงผลข้อมูลจาก Shared-Memory Object ออกทางจอภาพ

4. ลบ Shared-Memory Object

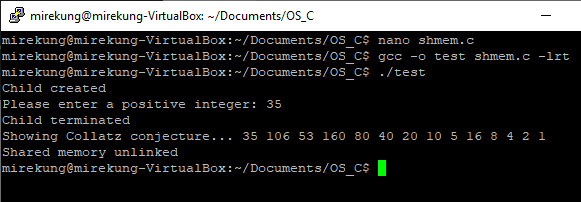
ตัวอย่างของการใช้ Shared-Memory Object





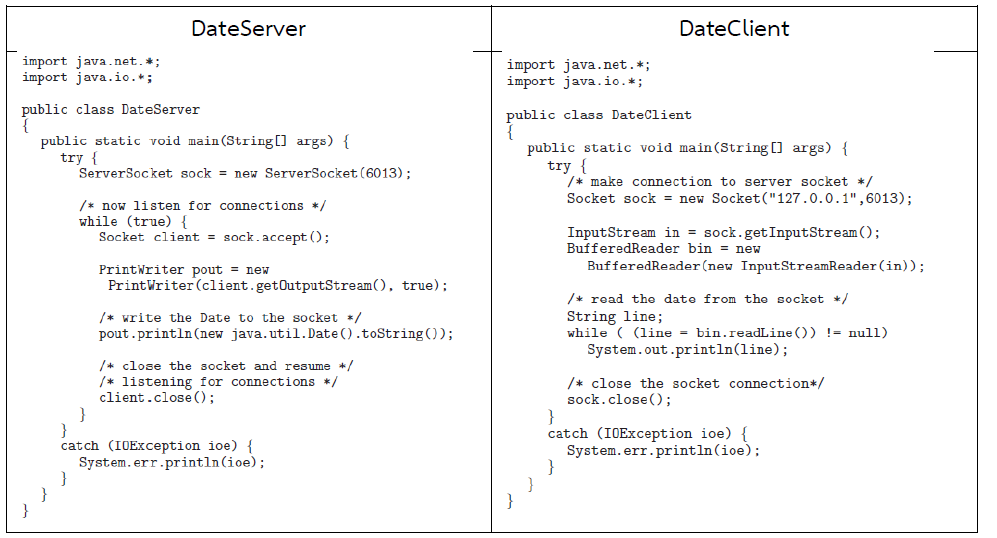


(ตัวอย่าง code)



9. (15 คะแนน) จากเรื่อง Interprocess Communication จงทดลองโปรแกรมภาษาจาวา

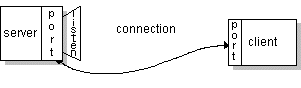
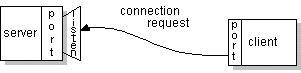
ต่อไปนี้ (หมายเหตุ ให้รัน DateServer ก่อน แล้วจึงค่อยรัน DateClient)



* 1. a. โค้ดตัวอย่างข้างต้น มี Interprocess Communication ด้วยวิธีการใด
  2. ตอบ : แบบ Socket
  3. b. การรันโปรแกรมได้ผลลัพธ์อะไร จงอธิบายพฤติกรรมของโปรแกรมอย่าง ละเอียด
  4. ตอบ : ผลลัพธ์เป็นวันที่ ณ เวลาและวันที่รัน จากคำสั่ง java.util.Date()toString()
  5. DateServer เป็นตัวกำหนด port ในการสื่อสารในที่นี้คือ port 6013 โดยจะกำหนดให้สื่อภายนอกหรือ Client เชื่อมต่อเข้ามา คล้ายกับการเชื่อมต่อ http ที่ฝั่ง clientจะต้องเชื่อมต่อเข้ามาที่ port 80
  6. ฝั่ง DateServer จะทำการรีเทิร์นค่าที่เป็นวันที่ และเวลา ณ ช่วงที่ Client เชื่อมต่อเข้ามานั่นเอง หลังจากส่งแล้วก็จะทำการจบการเชื่อมต่อกับ Client ด้วยคำสั่ง client.close();

ฝั่ง DateClient ทำการเชื่อมต่อไปยัง port ปลายทางด้วย port 6013 โดยให้แสดงผลข้อมูลใน ip 127.0.0.1

จากนั้นทำการตั้งตัวแปรขึ้นมารับข้อมูลที่ได้จากฝั่ง DateServer และทำการ Readlineข้อมูลที่ได้รับมา จากนั้นทำการปิด port สื่อสารด้วย sock.close();

* 1. c. จงอธิบายความหมายของ Socket
  2. ตอบ : โดยปกติเซิฟเวอร์ที่ทำงานอยู่ในคอมพิวเตอร์นั้นจะมี socket ซึ่งแต่ละ socket จะถูกกำหนดให้เป็น port number หมายเลขที่กำหนด โดย server จะทำหน้าที่แค่รอฟัง connection request จากฝั่ง client เพื่อ accept connection request และทำการ provide service ของ server ให้ฝั่ง client
  3.  
  4. d. หากรัน DateClient ก่อนการรัน DateServer ได้ผลลัพธ์อย่างไร
  5. ตอบ : เกิด error ทางฝั่ง DateClient เนื่องจากพยายามเชื่อมต่อกับ port ที่ไม่มีอยู่

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 : คือ Threads ที่ implement โดย user โดยไม่ยุ่งเกี่ยวกับ kernel ฝั่ง kernel จะมอง user threads เป็น single-threaded processes, User - Level Threads มีขนาดเล็ก และเร็วกว่า Kernel level thread โดยจะถูกแสดงออกในรูปแบบของ program counter, stack, register, process control block

พูดง่ายๆคือ ง่ายต่อการสร้าง จัดการง่าย ไม่ต้องรอ kernel mode privileges ในการทำ thread switching

ข้อเสียคือไม่สามารถทำ multithreaded application และ process ทั้งหมดจะถูก block หากมีการใช้ blocking operation

Kernel-Level Thread : คือ threads จัดการ operating system โดยตรง และจัดการโดยตัว kernel เอง เพราะฉะนั้นประสิทธิภาพจะดี แต่จะช้ากว่า User - Level Threads

พูดง่ายๆคือ Multiple threads ในprocessเดียวกันสามารถได้รับการ scheduled บน processors ใน kernel-level threads , kernel routines สามารถเป็น multithreaded ได้เช่นกัน

**VVV เข้าใจว่าอาจารย์เขียนคำถามแยกกันเป็น User-level Thread 1ข้อ และ Kernel-Level Thread 1ข้อ VVV**

• สถานการณ์ใดที่เหมาะสมกับการใช้ User-Level Thread และ Kernel-Level Thread

ควรใช้เมื่อไม่มี OS เข้ามาสนับสนุนการทำงานของthread และthreadที่ถูกสร้างขึ้นก็ไม่จำเป็นต้องรันกับ APIของระบบปฏิบัติการใดอันหนึ่ง เพราะฉะนั้นสามารถ portable ไปยัง OS อื่นๆได้นั่นเอง

• สถานการณ์ใดที่เหมาะสมกับการใช้ Kernel-Level Thread

เมื่อต้องการให้ kernel ส่ง threads ของ processors ไปยัง processors อื่นๆ ในเวลาเดียวกัน ทำให้ช่วยปรับปรุงระสิทธิภาพของการประมวลผลแบบขนาน

2. (5 คะแนน) สถานการณ์ใดที่ Multithreaded Solution ที่ใช้ Multiple Kernel Threads ทาให้ระบบมีประสิทธิภาพดีกว่าการใช้ Single-Threaded Solution บนระบบที่มีโพรเซสเซอร์เดียว จงยกตัวอย่างพร้อมคาอธิบาย

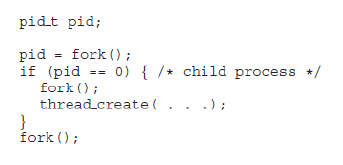
เมื่อต้องการความถูกต้องสูงในการทำงาน เนื่องจาก multithread kernel threads เมื่อมี kernel thread ใดๆพังหรือทำงานผิดพลาด จะมี kernel thread อื่นคอยรองรับการทำงานอยู่ แต่ถ้าทำงานแบบ single thread นั้น เมื่อ kernel threads เกิดผิดพลาด การทำงานก็จะหยุดไปเลย

ตัวอย่างเช่นกันพัฒนาซอฟแวร์ที่มีการทำงานพร้อมๆกันหลายๆคำสั่ง และต้องการความแม่นยำสูง

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

ตอบ : Heap Memory, Global Variables

4. (5 คะแนน) จากส่วนของโค้ดต่อไปนี



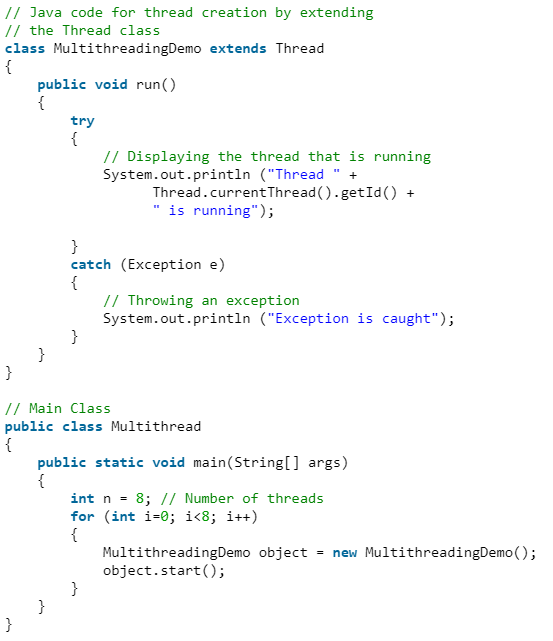
* 1. • มีโปรเซสที่สร้างขึ้นกี่โปรเซส
  2. ตอบ : 6 Process
  3. จากตัว parent 2 process และใน if 4 process
  4. • มี Thread ที่สร้างขึ้นกี่ Thread
  5. ตอบ : 8 Thread
  6. จากใน if ที่มี 4 process ทำการสร้าง thread ขึ้น 4\*2 = 8

5. (5 คะแนน) Thread Pool คืออะไร จงอธิบาย

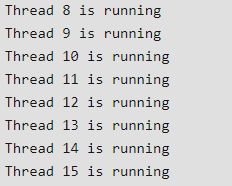
ThreadPool เป็น Class ใน Libary .NET ใช้สำหรับทำงานกับ Multi-thread โดยเฉพาะและยังจัดการกับทำงานจำพวกmethod และยังเป็นการทำงานแบบ Asynchronous ThreadPool นั้น ทำงานเป็น Background thread ดังนั้น Application สามารถหยุดการทำงานได้ทันทีถ้า Foreground Thread อื่นๆ สิ้นสุดการทำงานลง

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

ตัวอย่าง Java Thread



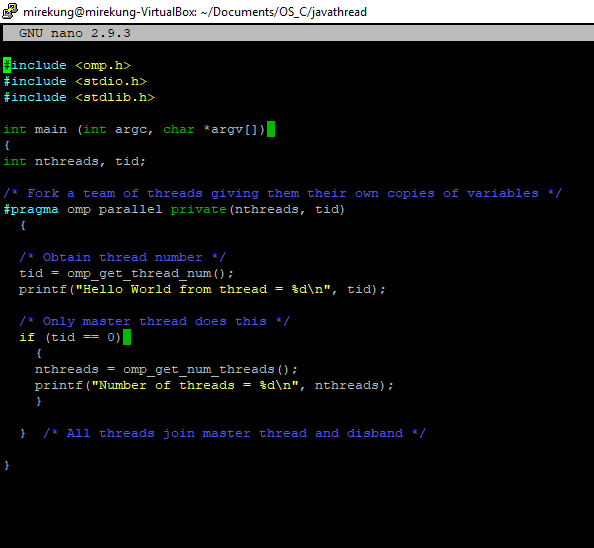
Output

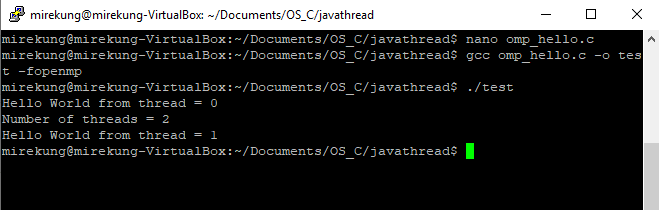


พฤติกรรมของโปรแกรมคือ อันดับแรกเรามาพูดถึง method run() ก่อน ซึ่งทำการ extends มาจาก Thread โดยทำการดักจับ error ด้วยการเช็คว่ามี thread runningอยู่ไหม ถ้าไม่มีก็จะ thow exception

ส่วนใน main method จะทำการสมมุติ thread = 8 คือมี 8 thread และผลการรันก็เป็นอย่างที่เห็นคือการเรียก method run() มาใช้โดยให้ค่า thread = 8

ตัวอย่าง OpenMP



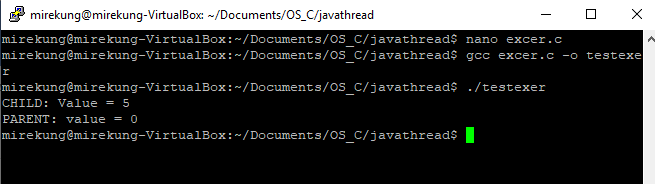


ซึ่งถือว่าการรันถูกต้อง เพราะในตัวของ VM ที่ทำการทดลองนั้น ได้ทำการตั้งprocessorไว้แค่สองตัว

โดยเริ่มต้นโปรแกรมจะ Generate ID ขึ้นมาและทำการเก็บลงตัวแปรชื่อ tid หลังจากนั้นนำค่าใน tid มาปริ้นด้วย line “hello world from thread”

ถ้า tid ==0 นั่นหมายความว่าเป็น master thread เพราะฉะนั้นจะมีการทำพฤติกรรมต่อไปคือให้ nthreads = จำนวน thread ที่มีเก็บไว้ในตัวแปร nthreads และทำการปริ้นออกมา

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



8. (30 คะแนน) จงเขียนโปรแกรมแบบ Multithread โดยมีการทางานดังนี้

• Main Thread สร้างเลขจานวนเต็มแบบสุ่ม 10 ค่า แล้วแสดงออกทางจอภาพ

• Main Thread เตรียมตัวแปร min, max, และ avg สาหรับเก็บค่าต่าสุด, สูงสุด, และค่าเฉลี่ยตามลาดับ

• สร้าง Thread ย่อย 3 Thread มีหน้าที่ดังนี้ จากเลขจานวนเต็มที่สุ่มมาจาก Main Thread

i. Thread #1: ค้นหาค่าต่าสุด เก็บลงใน min

ii. Thread #2: ค้นหาค่าสูงสุด เก็บลงใน max

iii. Thread #3: คานวณค่าเฉลี่ย เก็บลงใน avg

• Main Thread แสดงผลลัพธ์ของ min, max, avg ออกทางจอภาพ

