รูปภาพประกอบด้วย สัญลักษณ์, ยอด, เครื่องหมาย, มงกุฎ

คำอธิบายที่สร้างโดยอัตโนมัติ

**ใบงานที่ 5**

**เรื่อง** Threads Creation and Execution

**เสนอ**

**อาจารย์ปิยพล ยืนยงสถาวร**

**จัดทำโดย**

**นายอธิศ สุนทโรดม**

**รหัส 65543206086-2**

**ใบงานนี้เป็นส่วนหนึ่งของวิชาระบบปฏิบัติการ (ENGCE125)**

**หลักสูตรวิศกรมมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์**

**มหาวิทยาลัยเทคโนโลยีราชมงคลล้านนา เชียงใหม่**

**ประจำภาคเรียนที่ 2 ปีการศึกษา 2566**

**จุดประสงค์การสอน**

- เพื่อให้มีความรู้ความเข้าใจการสร้าง Creating Threads

- เพื่อให้มีความรู้ความเข้าใจ Terminating Thread Execution

- เพื่อให้มีความรู้ความเข้าใจ Passing Arguments To Threads

- เพื่อให้มีความรู้ความเข้าใจ Thread Identifiers

- เพื่อให้มีความรู้ความเข้าใจ Joining Threads

- เพื่อให้มีความรู้ความเข้าใจ Detaching / Undetaching Threads

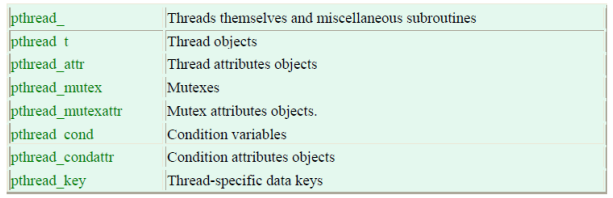
**อุปกรณ์ที่ใช้งาน**

- คอมพิวเตอร์ 1 เครื่อง พร้อมติดตั้ง Virtualbox

- CentOS เวอร์ชัน 7 หรือสูงกว่าที่ติดตั้งบน Virtualbox

**ทฤษฎี**

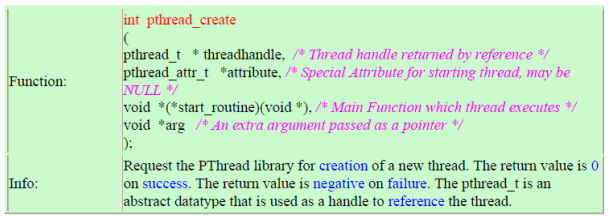
**การตั้งชื่อ: ข้อกำหนดทั้งหมดเริ่มต้นด้วย pthread\_ฟังก์ชั่นการจัดการ**

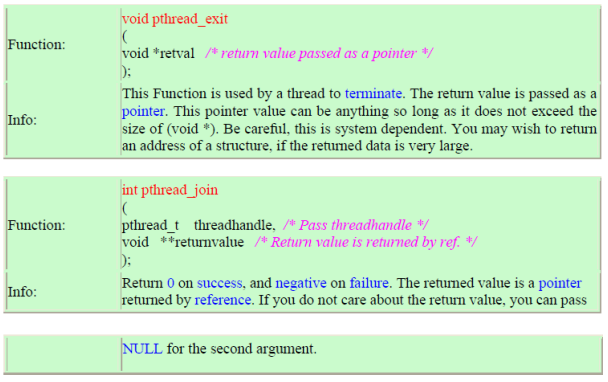


**ฟังก์ชั่นการจัดการ**

ฟังก์ชัน pthread\_create ถูกใช้เพื่อสร้างเธรดใหม่ และฟังก์ชัน pthread\_exit ใช้เพื่อยุติเธรดนั้นเอง

การดำเนินการนี้จะรอการสิ้นสุดของเธรดอื่นโดยใช้ฟังก์ชัน pthread\_join





เริ่มทำงาน

รวมไลบรารี pthread.h:

#รวม <pthread.h>

ประกาศตัวแปรประเภท pthread\_t

pthread\_t the\_thread

เมื่อคุณคอมไพล์ไฟล์ lpthread ไปยังตัวเชื่อมโยง:

**gcc thread.c -o เธรด -lpthread**

ขั้นตอนก่อนหน้านี้จะถูกสร้างขึ้นภายในกระบวนการ เมื่อสร้างขึ้นแล้วมันเป็นความรู้สึก คงเหมือนกับนักบัญชีของคุณนะสร้างเธรดอื่น โปรดทราบว่า "เธรดเริ่มต้น" มีอยู่ตามค่าเริ่มต้นและเป็นเธรดการยุติหลักเธรดการดำเนินการ

**int pthread\_attr\_destroy (pthread\_attr\_t \*attr) pthread\_attr\_destroy**

ทำลายอ็อบเจ็กต์แอ็ตทริบิวต์ที่ชี้ไปที่ attr และเผยแพร่ทรัพยากรทั้งหมดที่เกี่ยวข้องกับ attr ยังคงไม่ได้กำหนดไว้ และคุณไม่จำเป็นต้องใช้อีกครั้งเมื่อเรียกใช้ฟังก์ชัน pthreads จนกระทั่ง เริ่มต้นใหม่แล้ว

**int pthread\_attr\_setattr (pthread\_attr\_t \*obj, ค่า int)**

ตั้งค่า attr ในค่าในแอตทริบิวต์ ตัวชี้ไปที่ obj ดูรายการไฟที่มีอยู่ด้านล่างเป็นไปได้และค่าที่สามารถนำมาใช้เกี่ยวกับความสำเร็จของฟังก์ชัน...เหล่านี้คือ 0

**int pthread\_attr\_getattr (const pthread\_attr\_t \*obj, int \*ค่า)**

คงการตั้งค่าปัจจุบันของ attr ใน obj ไว้เป็นสมุนไพร ค่าฟังก์ชันเหล่านี้จะกลับเป็น 0

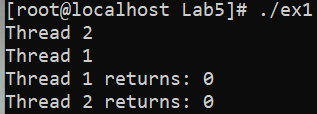
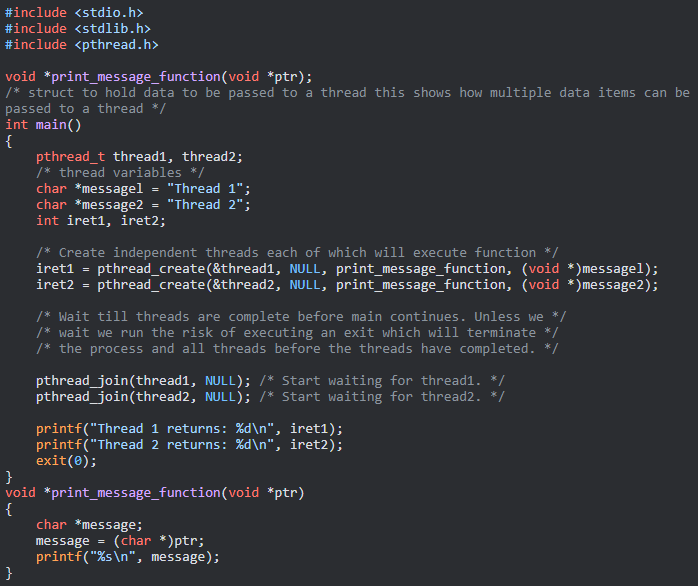
**การทดลองที่ 5**

**วิธีการทดลอง**

1. เขียนโปรแกรมตามโค้ดตัวอย่างบนระบบปฏิบัติการ CentOS และแสดงผลทดลอง
2. อธิบายการทำงานของโปรแกรม
3. สรุปผลการทดลอง
4. ส่งงานใน Microsoft Teams เป็นไฟล์ PDF

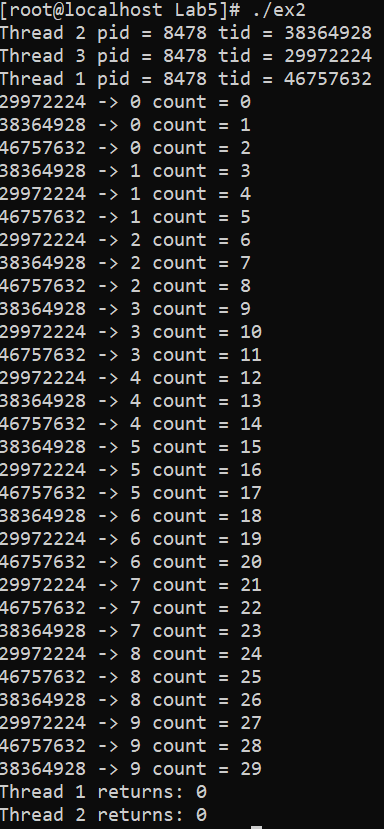
**Example: Pthread Creation and Termination**

**ข้อ 1. อธิบายคําสั่งที่กําหนดให้ต่อไปนี้ พร้อมกับทําการสร้าง Thread ขึ้นมาจากโปรแกรมคําสั่งที่กําหนดให้**



โปรแกรมจัดการเธรด โปรแกรมนี้สร้างเธรดขึ้นมาสองเธรด โดยแต่ละเธรดจะทำงานซ้ำๆ กัน โดยการพิมพ์ข้อความ "Thread 1" และ "Thread 2" ตามลำดับโดยเริ่มต้น #include <stdio.h>, #include<stdlib.h>, #include <pthread.h>คำสั่งเหล่านี้ใช้ในการนำเข้าไลบรารีที่จำเป็นสำหรับการทำงานของโปรแกรม void \*print\_message\_function(void \*ptr) ฟังก์ชันนี้ทำหน้าที่เป็นฟังก์ชันทำงาน (runnable function) ของเธรด ฟังก์ชันนี้จะพิมพ์ข้อความที่ส่งเข้ามาเป็นอาร์กิวเมนต์ pthread\_t thread1, thread2; ตัวแปรเหล่านี้ใช้เพื่อเก็บข้อมูลของเธรด

**ข้อ 2. นำ Example: pthread1.c มาทำการ เพิ่ม ฟังก์ชั่นการทำงาน เข้าไป ใน pthread1.c**

ได้ทำการเพิ่มฟังก์ชัน print\_message\_funtion() เพื่อรับอาร์กิวเมนต์เป็นตัวแปร message ซึ่งเป็นตัวชี้ไปยังข้อความที่จะพิมพ์ ฟังก์ชันนี้จะเริ่มทำงานโดยพิมพ์ข้อความที่ส่งเข้ามาเป็นอาร์กิวเมนต์ จากนั้นจะวนซ้ำไปเรื่อยๆ โดยพิมพ์ข้อความ "Thread 1" และ "Thread 2" สลับกัน

ขั้นตอนการทำงานของฟังก์ชันprint\_message\_functio() มีดังนี้ ดึงข้อความที่จะพิมพ์ออกมาจากตัวแปร message พิมพ์ข้อความที่ดึงออกมาวนซ้ำไปเรื่อยๆ โดยพิมพ์ข้อความ "Thread 1" และ "Thread 2" สลับกันเมื่อฟังก์ชันนี้ทำงานเสร็จสิ้น จะคืนค่า NULL ให้กับฟังก์ชัน pthread\_create() ในภาพประกอบ ฟังก์ชัน print\_message\_function() ถูกเรียกใช้จากฟังก์ชัน main() โดยส่งข้อความ "Thread 1" และ "Thread 2" เป็นอาร์กิวเมนต์ ส่งผลให้เธรดทั้งสองจะทำงานซ้ำๆ กัน โดยพิมพ์ข้อความ "Thread 1" และ "Thread 2" สลับกัน

**สรุปผลการทดลอง**

เพื่อจัดการเธรด ผลการทดลองพบว่าสามารถสร้างเธรดขึ้นมาได้สำเร็จ โดยเธรดแต่ละเธรดจะทำงานซ้ำๆ กัน โดยพิมพ์ข้อความ "Thread 1" หรือ "Thread 2" ตามลำดับจากผลการทดลองพบว่า การสร้างเธรดโดยใช้ไลบรารี pthread สามารถทำได้อย่างง่ายดาย โดยเพียงแค่นำเข้าไลบรารี pthread ประกาศตัวแปรเพื่อเก็บข้อมูลของเธรด สร้างฟังก์ชันทำงาน (runnable function) ของเธรด และเรียกใช้ฟังก์ชัน pthread\_create() เพื่อสร้างเธรดนอกจากนี้ ผลการทดลองยังแสดงให้เห็นว่าเธรดแต่ละเธรดจะทำงานซ้ำๆ กัน โดยทำงานแยกจากกัน ไม่ขึ้นต่อกันส่งผลให้สามารถทำงานหลายอย่างพร้อมกันได้