

เมื่อโปรแกรมเรียก `fork()` ในระบบ Unix จะเกิดอะไรขึ้น

- ☐ a. Orphan Process
- ☐ b. Parent Process
- ☐ c. Thread
- ☐ d. Zombie Process
- ☐ e. Child Process

วิธีการ IPC แบบใดที่ หลาย process เข้าถึง memory เดียวกันได้โดยตรง

- ☐ a. Pipes
- ☐ b. Sockets
- ☐ c. Shared Memory
- ☐ d. FIFOs
- ☐ e. Message Passing

สถานะ Running ของ Process หมายถึงอะไร

- ☐ a. Process รอ parent เรียก wait()
- ☐ b. Process ถูก terminate แล้ว
- ☐ c. Process กำลังถูกประมวลผลโดย CPU
- ☐ d. Process ถูกสร้างแต่ยังไม่รัน
- ☐ e. Process รอ I/O

สถานะ Waiting ของ Process หมายถึงอะไร

- ☐ a. Process รอ event หรือ I/O เสร็จ
- ☐ b. Process กำลังถูกประมวลผลโดย CPU
- ☐ c. Process ถูก terminate แล้ว
- ☐ d. Process รอ CPU
- ☐ e. Process ถูกสร้างแต่ยังไม่รัน

---

สถานะ Zombie เกิดขึ้นเมื่อใด

- ☐ a. Process กำลังรอ CPU
- ☐ b. Process รอ I/O
- ☐ c. Process ถูก init/systemd รับเลี้ยง
- ☐ d. Process เสร็จสิ้นแต่ parent ยังไม่เรียก wait()
- ☐ e. Process กำลังถูกประมวลผล

อะไรคือ หน้าที่หลักของ Process Control Block (PCB)

- ☐ a. เก็บข้อมูลทั้งหมดของ process เช่น PID, state, CPU registers, memory info
- ☐ b. เป็น stack ของ process
- ☐ c. จัดการการสื่อสารระหว่าง process
- ☐ d. เก็บข้อมูลเกี่ยวกับ memory allocation
- ☐ e. เก็บข้อมูลเกี่ยวกับ CPU scheduling

Process Control Block (PCB) ใช้สำหรับอะไรใน CPU scheduling

- ☐ a. เก็บข้อความจาก message passing
- ☐ b. แสดงผลลัพธ์ของ process
- ☐ c. แสดงสถานะของ process เพื่อให้ scheduler เลือก process ถัดไป
- ☐ d. สร้าง child process โดยใช้ fork()
- ☐ e. เก็บข้อมูลเกี่ยวกับ disk I/O

ข้อใดไม่ใช่ผลกระทบของ Context Switching

- ☐ a. ทำให้ process ที่ทำงานอยู่ต้องรอ
- ☐ b. ทำให้ process ที่รอ I/O เสร็จเร็วขึ้น
- ☐ c. ต้องเก็บและโหลดค่าของ CPU registers
- ☐ d. เพิ่ม overhead ของ CPU
- ☐ e. ต้องจัดการ PCB ของ process

ใน Message Passing แบบ Synchronous สิ่งใดเกิดขึ้นหลังส่งข้อความ

- ☐ a. Sender เข้า Ready Queue
- ☐ b. Sender ต้องเข้าถึง shared memory
- ☐ c. Sender ถูก terminate
- ☐ d. Sender ต้องรอจนกว่า receiver รับข้อความ
- ☐ e. Sender ทำงานต่อทันที



ข้อใดเป็น ข้อดีของ Shared Memory เมื่อเทียบกับ Message Passing

- ☐ a. ต้องรอการตอบรับเสมอ
- ☐ b. ง่ายต่อการโปรแกรม
- ☐ c. ส่งข้อมูลขนาดใหญ่ได้เร็วกว่า
- ☐ d. ใช้ได้กับระบบที่ไม่ใช่ Unix
- ☐ e. ปลอดภัยกว่า