**Nama : Herman Setiawan**

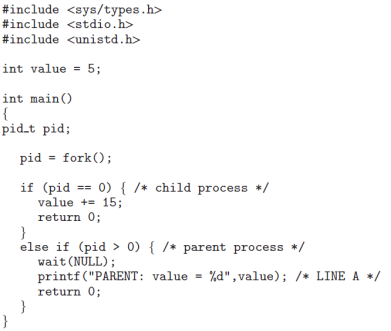
**NIM : 2009076043**

**Mata Kuliah : Sistem Operasi**

**Ujian Tengah Semester**

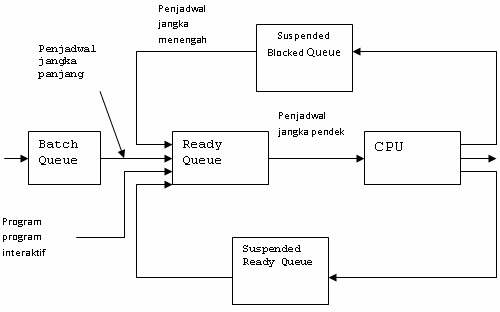
Soal

1. Jika diketahui 6 antrian proses (A, B, C, D, E, F) dengan waktu kedatangan secara bersamaan yaitu: 0. Lama eksekusi tiap – tiap antrian proses secara berurutan 1, 3, 7, 5, 5, 3. Hitunglah Turn Arround Time (TA) dengan menggunakan Teknik penjadwalan proses:
2. First In First Out (F I F O)
3. Shortest Job First (S J F)
4. Round Robin jika diketahui Quantum = 2
5. Dalam Penjadwalan proses terdapat tiga macam penjadwalan, sebutkan dan jelaskan disertai gambar!
6. Sumber daya apa yang digunakan saat thread dibuat? Bagaimana mereka berbeda dari yang digunakan Ketika suatu proses dibuat?
7. Output apa yang akan ditampilkan pada LINE A? Jelaskan!

****

**JAWABAN**

**2.**



Terdapat tiga tipe Penjadwal, yaitu:

**a. Penjadwal jangka pendek (short term scheduller)**

* Bertugas menjadwalkan alokasi pemroses diantara proses-proses ready di memori utama
* Sasarannya untuk memaksimalkan kinerja untuk memenuhi kriteria yang diharapkan

**b. Penjadwal jangka menengah (medium term scheduller)**

* Untuk menangani proses-proses swapping, proses-proses yang mempunyai critical condition kecil, maka saat itu sebagai proses yang tertunda, begitu kondisi yang membuatnya tertunda hilang, proses dimasukkan kembali ke memori utama dan ready.
* Swapping adalah kegiatan pemindahan proses yang tertunda dari memori utama ke memori sekunder.

**c. Penjadwal jangka panjang (long term scheduller)**

Bekerja terhadap antrian batch dan memilih batch berikutnya yang harus dieksekusi. Batch biasanya adalah proses-proses dengan penggunaan daya yang intensif (cpu time, memori, perangkat I/O)

**3.** Saat thread dibuat, beberapa sumber daya yang digunakan adalah:

1. Memori: Thread membutuhkan alokasi memori untuk menyimpan variabel-variabel lokal, tumpukan (stack), dan status register yang terkait dengan eksekusi thread tersebut. Setiap thread memiliki tumpukan sendiri yang digunakan untuk menyimpan data yang berkaitan dengan eksekusi thread, seperti variabel lokal dan pemanggilan fungsi.
2. Konteks Register: Setiap thread memiliki konteks register yang mencakup nilai-nilai register seperti program counter, register umum, register status, dan register lainnya. Konteks register ini digunakan untuk melacak posisi eksekusi thread dan menyimpan informasi yang diperlukan untuk melanjutkan eksekusi thread setelah jeda atau beralih ke thread lain.
3. Sumber daya bersama: Ketika thread dibuat, mereka dapat mengakses dan menggunakan sumber daya bersama yang dimiliki oleh proses utama. Ini termasuk sumber daya seperti file yang terbuka, soket jaringan, koneksi ke database, dan sebagainya. Thread-thread dalam satu proses dapat berbagi dan bekerja secara bersama-sama pada sumber daya ini.

Perbedaan dengan sumber daya yang digunakan saat proses dibuat adalah sebagai berikut:

1. Ruang Alamat: Setiap proses memiliki ruang alamat virtualnya sendiri. Ini berarti setiap proses memiliki area memori yang terisolasi dan tidak dapat diakses oleh proses lain. Ketika proses dibuat, ruang alamat baru dialokasikan untuknya, termasuk kode program, data, tumpukan, dan heap. Dalam kasus thread, mereka berbagi ruang alamat yang sama dengan proses utama, sehingga dapat saling berbagi data dan variabel.
2. Identitas Proses: Setiap proses memiliki identitas unik dalam sistem operasi, yaitu nomor identifikasi proses (Process ID atau PID). PID digunakan untuk mengidentifikasi dan mengendalikan proses dalam sistem. Sebaliknya, thread tidak memiliki identitas unik mereka sendiri. Mereka beroperasi di dalam konteks proses utama dan diidentifikasi oleh alamat memori yang terkait dengan thread tersebut.
3. Sumber daya Terkait Proses: Saat proses dibuat, sistem operasi mengalokasikan sumber daya tertentu untuk proses tersebut, seperti ruang memori, deskriptor file, area heap, dan sumber daya sistem lainnya. Proses memiliki entitas yang terpisah dan independen dalam sistem operasi. Namun, ketika thread dibuat, sumber daya-sumber daya ini dibagi dan digunakan bersama oleh semua thread dalam proses. Hal ini memungkinkan thread untuk berbagi data dan melakukan komunikasi yang lebih efisien.

**4.** Pada coding tersebut, output yang akan ditampilkan pada LINE A adalah sebagaiberikut:

Jika ‘pid == 0’, berarti ini adalah bagian child process. Pada bagian ini, nilai ‘value, akan diubah dengan menambahkannya sebesar 15. Namun, tidak ada perintah untuk mencetak nilai ‘value’ di child process ini.

Jika ‘pid > 0’, berarti ini adalah bagian dari parent process. Pada bagian ini, program akan menunggu child process selesia dengan menggunakan ‘wait(NULL)’. Setelah itu, program akan mencetak pesan “PARENT:value = [nilai value]” di mana [nilai value] akan menggambarkan nilai ‘value’ pada saat itu.

Jadi, output pada LINE A akan menampilkan pesan “PARENT:value = [nilai value]” yang mencerminkan nilai ‘value’ pada parent process setelah child process selesai