Nama: Ellexia Leonie G.
NPM: 2108 3010027
Kelas: Sistem Operasi - A

No.
Date . .

deadlock

Kondisi untuk mencapai deadlock

1. Mutual exclusion (mutual exclusion conditional)
   - Dua atau lebih resource tidak dapat dibagikan (hanya satu proses yang dapat digunakan pada satu waktu).

2. Kondisi genggam dan tunggu (hold and wait)
   - Sebuah proses menahan setidaknya satu sumber daya dan menunggu sumber daya.
   - Pada saat suatu proses mengakses suatu resources, proses tersebut dapat meminta izin untuk mengakses resource lain.

3. Kondisi non-preemption (non-preemption condition)
   - Sebuah resource tidak dapat diambil dari sebuah proses kecuali proses tersebut melepaskan resource tersebut.
   - Jika suatu proses meminta izin untuk mengakses resource, sementara resource tidak tersedia, maka permintaan tidak dapat dibatalkan.

4. Kondisi menunggu secara sirkuler (circuler wait condition)
   - Satu set proses menunggu satu sama lain dalam bentuk melingkar.
   - Jika proses P_i sedang mengakses resource R_i, dan meminta izin untuk mengakses resource R_j, dan pada saat bersamaan proses P_j sedang mengakses R_j dan minta izin untuk mengakses resource R_i

No.

Date

Penanganan Deadlock

1. Mengabaikan permasalahan (The Ostrich Algorithm)
   Penghindaran deadlock membutuhkan informasi tentang sumber daya yang mana yang akan suatu proses meminta, dan berapa lama akan digunakan. Dengan informasi tersebut dapat diputuskan apakah suatu proses harus menunggu / tidak.
   The Ostrich Algorithm → berpura-pura bahwa tidak ada masalah apapun.
   Maksudnya : saat terjadi deadlock, maka seolah-olah deadlock tidak terjadi dan membiarkan secara otomatis memahkan program. Sehingga proses yang memalankan proses melalui operator harus menunggu pada waktu tertentu dan mencoba lagi.

2. Deteksi dan Pemulihan (Recovery)
   - Mendeteksi jika terjadi deadlock pada suatu proses, dengan mencari sistem mana yang terlibat di dalamnya.
   - Setelah diketahui sistem mana yang terlibat, maka diadakan proses untuk memperbaiki dan mengadikan sistem berjalan kembali.

3. Pencegahan, dengan meniadakan salah satu dari 4 kondisi deadlock
   - kondisi untuk mengatasi deadlock dengan cara menyakinkan bahwa paling sedikit satu dari kondisi deadlock tidak terjadi.
     a) mutual exclusion → buat resource shareable
     b) hold and wait → melepas resource pada saat request
     c) no preemption → melepas resource pada saat waiting
     d) circular wait → request berurutan

4. Pengalokasian sumber daya yang efisien
   Sistem dapat mengalokasikan resource untuk tiap proses dalam

No.

Date

urutan yang tepat tanpa terjadinya deadlock Biasanya menggunakan algoritma grat alokasi. Algoritma ini bekerja dengan mendeteksi perputaran dalam sistem. Jika tidak ada perputaran dalam grat, sistem berada dalam status aman, tetapi jika perputaran ditemukan maka sistem dalam status tidak aman.