CITRA AMELIA INTAN P.
2108301000A / SISTEM OPERASI - B

No.
Date

DEADLOCK.

A. Kondisi untuk mencapai Deadlock

1) **Mutual Exclusion**
   ↳ Jika suatu proses menggunakan suatu resource (sumber), maka tidak ada proses lain yang boleh menggunakan resource (sumber) tersebut.
2) **Hold and Wait**
   ↳ Pada saat suatu proses mengakses suatu resource, proses tersebut dapat meminta ijin untuk mengakses resource lain.
3) **No Preemption**
   ↳ Jika suatu proses meminta ijin untuk mengakses resource, sementara resource tidak tersedia, maka permintaan tidak dapat dibatalkan.
4) **Circular Wait**
   ↳ Jika proses P_i sedang mengakses resource R_i dan meminta ijin untuk mengakses resource R_j sementara pada saat yang bersamaan proses P_j sedang mengakses R_j dan meminta ijin untuk mengakses resource R_i.

B. Penanganan Deadlock

1) Mengabaikan Permasalahan (The Ostrich Algorithm)
   → Mengabaikan masalah yang mungkin terjadi dengan membiarkan deadlock seolah tidak terjadi apa-apa dan membiarkan deadlock tersebut mematikan program. Dengan begitu The Ostrich Algorithm mengasumsikan bahwa tidak ada masalah lebih efektif daripada untuk memungkinkan masalah itu terjadi dibandingkan upaya pencegahannya.
2) Deteksi dan Pemulihan (Recovery)
   → Deteksi digunakan pada sistem yang mengijinkan terjadinya deadlock dengan memeriksa apakah terjadi deadlock dan menentukan proses dan sumber daya yang terlibat deadlock secara presisi. Begitu telah dapat ditentukan, sistem dipulihkan dari deadlok dengan metode Pemulihan sehingga beroperasi kembali, bebas dari deadlock. Mungkin dapat menyelesaikan eksekusi dan membebaskan sumber dayanya dari proses deadlock.
3) Pencegahan, dengan meniadakan salah satu dari empat kondisi Deadlock
   → Pencegahan merupakan solusi yang bersih dipandang dari sudut tercegahnya deadlok dengan pengkondisian sistem agar menghilangkan kemungkin terjadinya deadlock.

-> Jika suatu proses yang membawa beberapa sumber daya meminta resource lain yang tidak dapat segera dipenuhi, maka semua resource yang sedang dibawa proses tersebut harus dibebaskan.
-> Proses yang sedang dalam keadaan menunggu, resource yang dibawanya ditunda dan ditambahkan pada daftar resource.
-> Proses akan direstart hanya jika dapat memperoleh resource yang lama dan resource baru yang diminta.

4) Pengalokasian Sumber Daya (Resource) yang Efisien
-> Situasi ketika Resource dialokasikan pada penggunaan nilai tertinggi. Tidak ada alternatif untuk menggunakan resource lebih lanjut tanpa membuat yang lain lebih buruk.