Universitas Esa Unggul Kampus Harapan Indah	**MODUL 10** **CCS 210 SISTEM OPERASI**	
Judul	MANAJEMEN MEMORY **PENCATATAN PEMAKAIAN MEMORY**	
Penyusun	**Distribusi**	**Perkuliahan**
Nixon Erzed	**FASILKOM** **UNIVERSITAS ESA UNGGUL**	Pertemuan OL – 11

Materi	• Teknik Pencatatan • Peta Bit • Senarai Berkait • Algoritma Alokasi

PENCATATAN PEMAKAIAN MEMORI

Dasar untuk pendekatan dalam pengelolaan ruang memory :

- ruang-ruang alamat harus dapat diidentifikasi : terpakai atau tidak terpakai
- indentifikasi berdasarkan per alamat → informasi identifikasi sangat besar
- harus didefinisikan satuan ruang sebagai basis indentifikasi
  misal : 1 unit ruang = 100K alamat

Dasar pencatatan : peta bit

Sistem menyediakan bit biner untuk mengidentifikasi status unit memory :

- Bit bernilai 0 → unit kosong
- Bit bernilai 1 → unit teralokasi

Implementasi pencatatan untuk kebutuhan alokasi-dealokasi :

- Penggunaan peta bit secara langsung
- Menggorganisasikannya dengan suatu senarai linear (linear linked list)
- Menggorganisasikannya dengan suatu pohon (system Buddy)

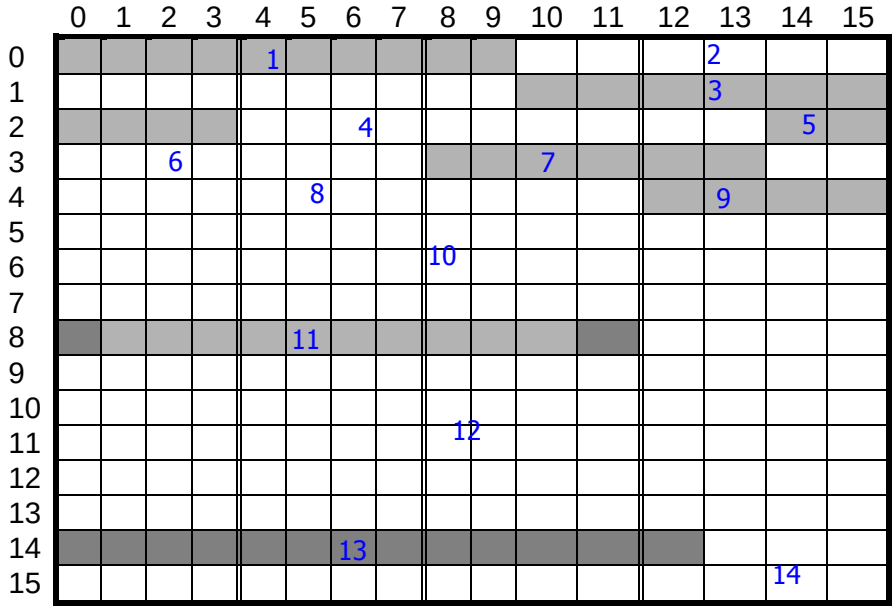
1. Pencatatan dengan peta bit

Setiap unit lokasi memori berkorespondensi dengan sebuah bit pada peta bit.

Sebagai contoh adalah sebagai berikut :

1111 1111 1100 0000 0000 0000 0011
1111 1111 0000 0000 0011 0000 0000
1111 1100 0000 0000 0000 1111 0000
0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000

Untuk status memory sebagai berikut



Jika 1 unit = 1 MB → maka kapasitas Memory = 256 MB

2. Pencatatan memakai senarai berakait.

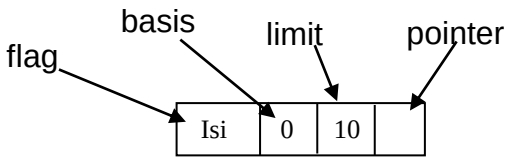
- Dengan 1 senarai
- Dengan 2 senarai

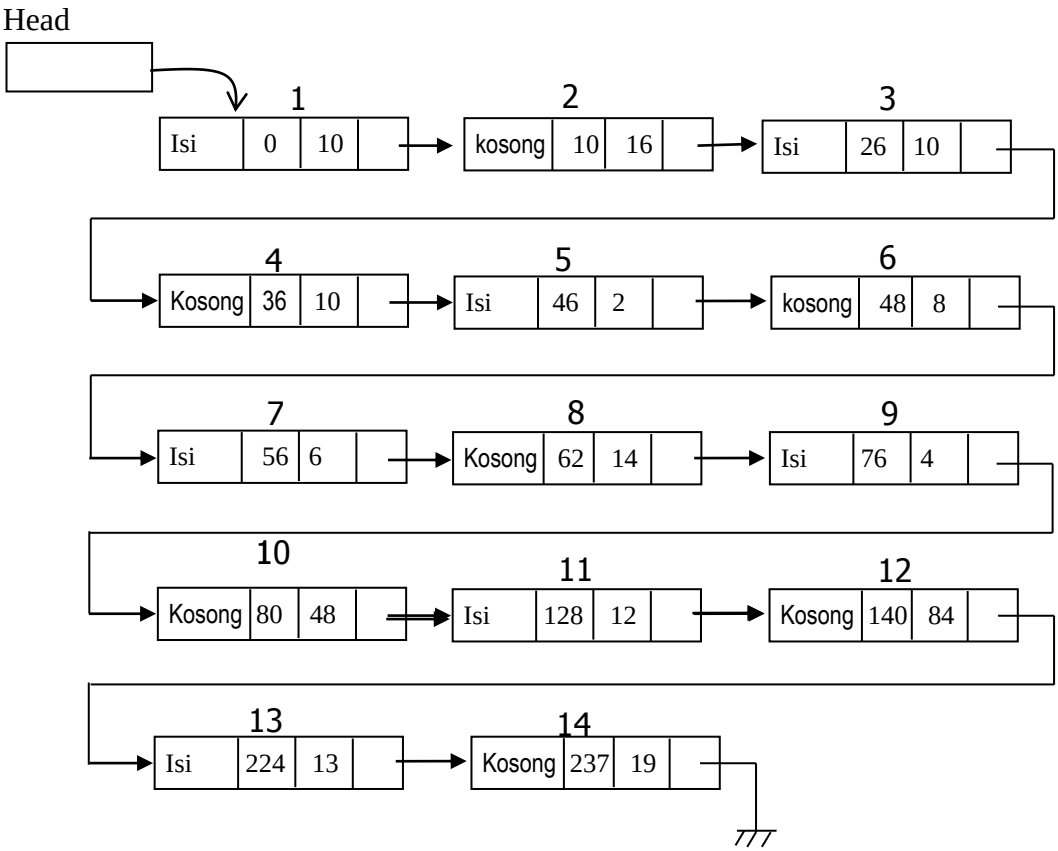
Setiap blok memori yang dialokasikan untuk sebuah proses dan yang bebas dicatat pada simpul senarai.

DENGAN 1 SENARAI

Struktur Simpul : record dengan 4 field

- → flag → type [isi, kosong]
- → basis → nomor/alamat unit awal pada blok memory
- → limit/ofset → panjang blok → jumlah unit pada blok kontigue
- → pointer ke simpul berikut (node suksesor)





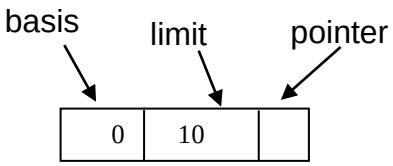
DENGAN 2 SENARAI

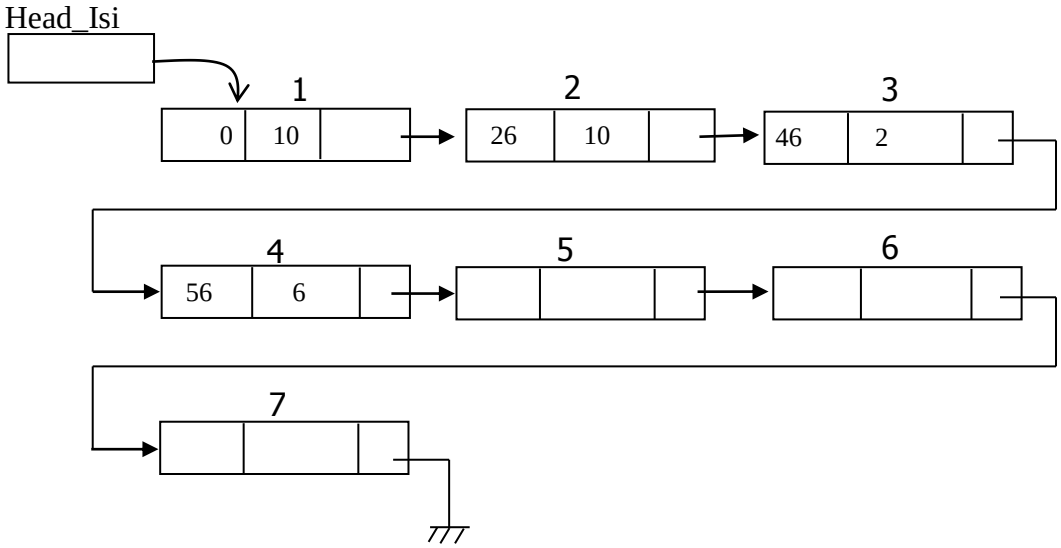
Membedakan senarai untuk :

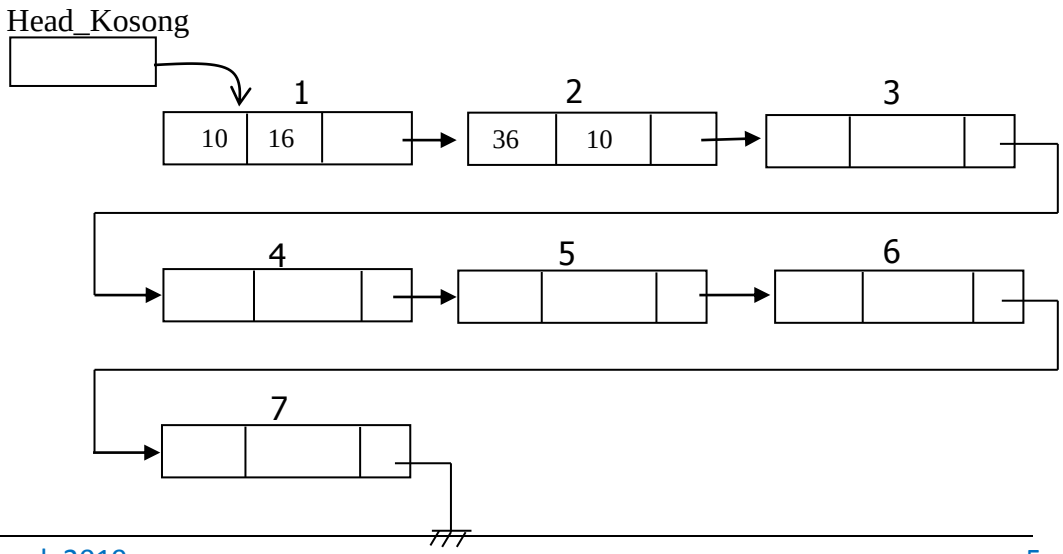
- rantai/list simpul kosong
- rantai/list simpul isi

Struktur Simpul : record dengan 3 field

- basis → nomor/alamat unit awal pada blok memory
- limit/ofset → panjang blok → jumlah unit pada blok kontigue
- pointer ke simpul berikut (node suksesor)







STRATEGI ALOKASI MEMORI :

Terdapat beragam strategi pengalokasian proses ke memori. Alokasi harus mencari sekumpulan blok memori yang ukurannya cukup memuat proses yang akan diload.

Algoritma pengalokasian memori antara lain :

1. First fit algorithm
2. Next fit algorithm
3. Best fit algorithm
4. Worst fit algorithm

→ implementasi 2 senarai :

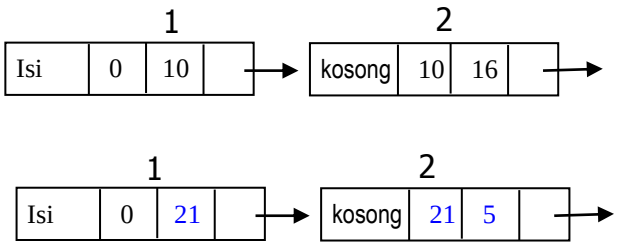
5. Quick fit algorithm

First fit algorithm

Manajer memori akan menscan sampai menemukan lubang besar yang mencukupi penempatan proses. Jika lubang lebih besar akan dibagi menjadi 2 bagian, partisi pertama sekebutuhan proses dan bagian berikutnya adalah lubang kosong. Lubang secara otomatis menjadi satu partisi saja jika berukuran tepat sama dengan kebutuhan.

Misal : dimiliki proses dengan size = 10,5 MB ➔ 11 unit

- Alokasikan unit dari simpul 2
- Update simpul 1 dan 2

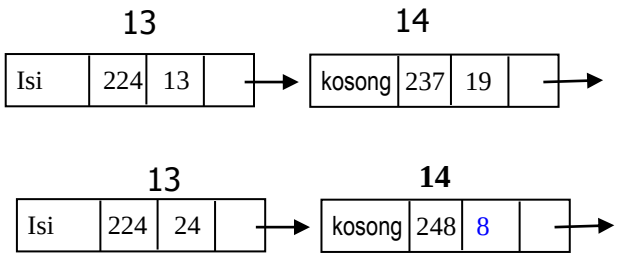


Next fit algorithm

Mirip dengan first fit algorithm, tapi dimulai dari posisi terakhir area memori. Diimplementasikan dengan cara pemanggilan rekursif

Misal : dimiliki proses dengan size = 10,5 MB ➔ 11 unit

- Alokasikan unit dari simpul 14
- Update simpul 13 dan 14

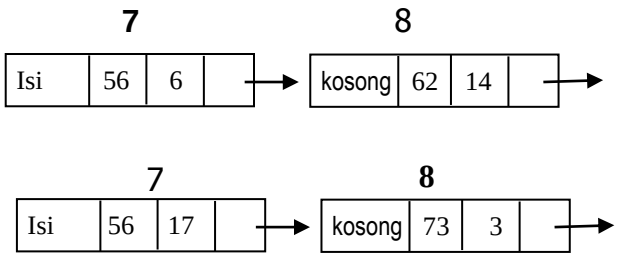


Best fit algorithm

Algoritma ini mencari sampai akhir dan memilih lubang paling kecil yang dapat memuat proses.

Misal : dimiliki proses dengan size = 10,5 MB ➔ 11 unit

- Alokasikan unit dari simpul 8
- Update simpul 7 dan 8



Worst fit algorithm

→ Best fit memperbesar resiko fragmentasi eksternal

Algoritma ini mencari lubang terbesar yang dapat memuat proses, sehingga diharapkkan lubang sisa masih cukup besar untuk menampung proses yang lain

Misal : dimiliki proses dengan size = 10,5 MB ➔ 11 unit

- Alokasikan unit dari simpul 12
- Update simpul 11 dan 12

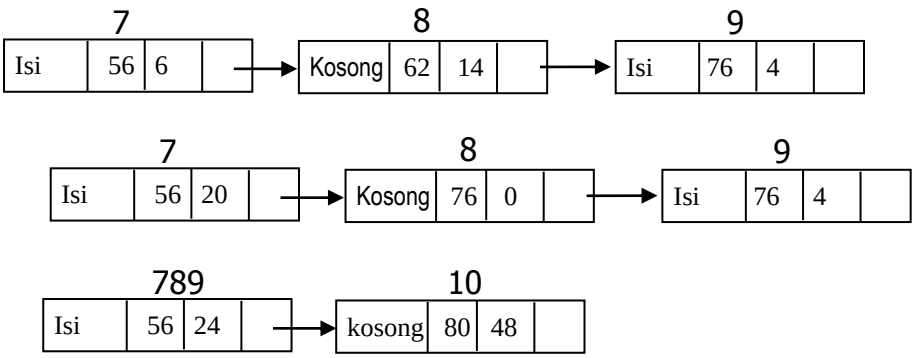
Catatan khusus :

jika ukuran proses = ukuran blok kosong

→ terjadi penggabungan simpul

misal pakai algoritma Best Fit dan butuh 14 unit

maka simpul 8 menghasilkan sisa 0 → simpul 7,8,9 digabung jadi 1



Quick fit algorithm

Algoritma ini mengelola 2 senarai, senarai untuk proses dan senarai untuk lubang. Jika terdapat proses masuk, maka senarai lubang akan discan untuk menemukan lubang yang paling sesuai, dan sebuah simpul baru disisipkan pada senarai proses.

Varian dari algoritma ini adalah dengan mengelola beberapa senarai lubang memori dengan beragam ukuran yang paling sering diminta. Algoritma ini sangat cepat dalam alokasi memory tapi sulit dalam melakukan dealokasi.

Dealokasi

Lakukan update pada simpul2 yang sesuai dengan lokasi proses yang akan di-dealokasi.

Misal :

Sebuah proses yang berukuran 2,5 MB dan berada diunit 133 → completion
Proses akan di-dealokasi

Simpul 11 harus diubah menjadi 3 simpul
→ disisipkan 2 simpul baru ke senarai

pada simpul-simpul senarai. Senarai disort berdasarkan urutan alamat blok.