Alokasi Memori

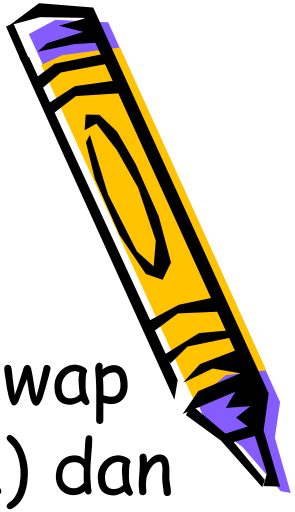
Heri Kurniawan

OS-Gasal 2009/2010

Tujuan Pembelajaran

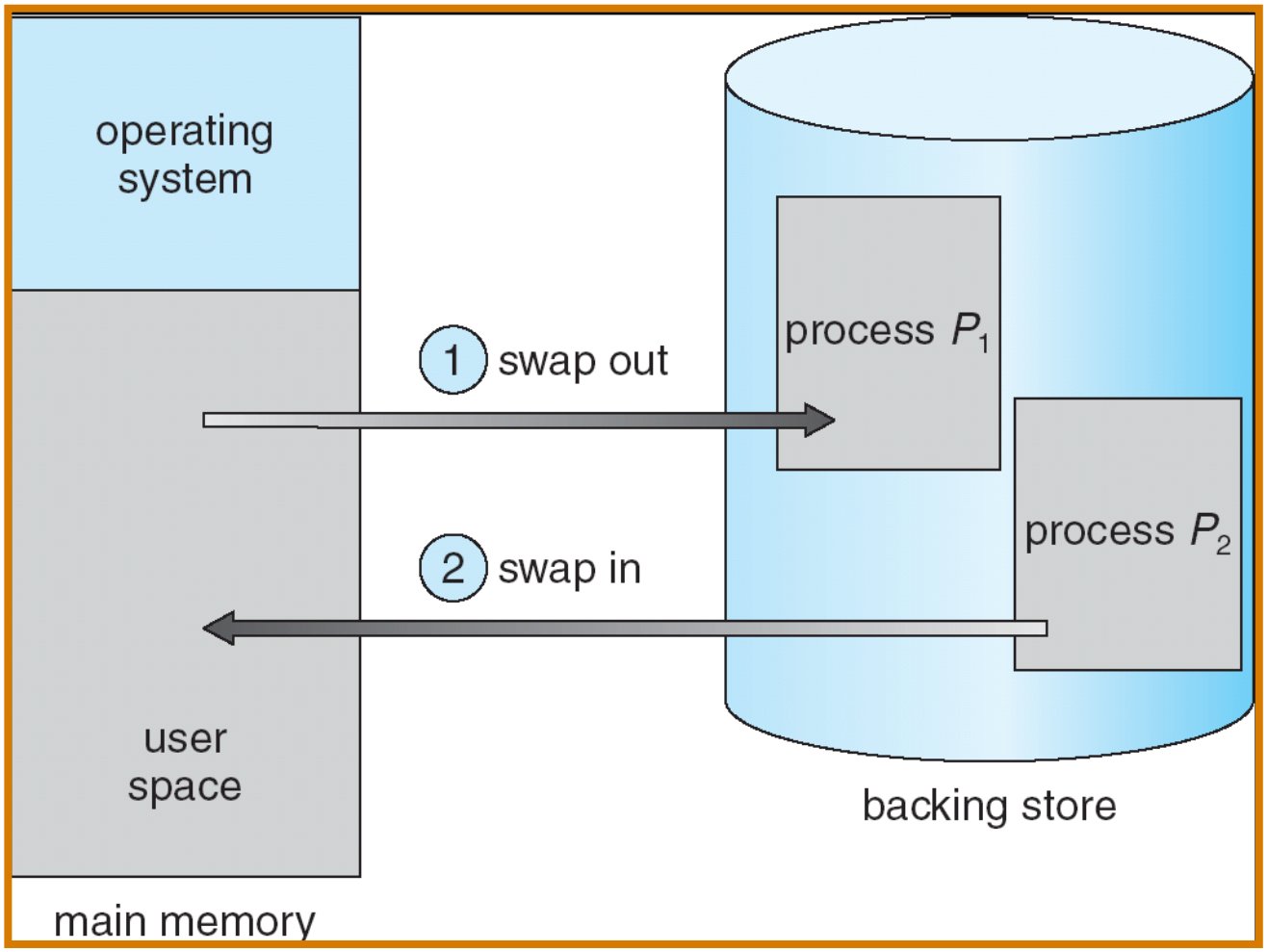
- Memahami proses swapping
- Memahami proses alokasi memori berurutan (Contiguous Memori Allocation)

Swapping



- Sebuah proses dapat di swap sementara (swap out) ke sebuah memori sekunder (harddisk) dan kemudian dikembalikan ke memori untuk dieksekusi (swap in).
- **Roll out, roll in** – varian swapping dalam *priority -based scheduling algorithms*; Proses yang mempunyai prioritas paling rendah diswap sehingga proses yang mempunyai prioritas tinggi dapat dimasukkan ke memori dan dieksekusi

Skema proses swap



Swapping

- Memori sekunder harus dapat menampung duplikasi dari *image memory* dari semua proses user dan harus dapat diakses langsung
- *Swap Time*
  - Besar proses user =10 Mb
  - Kecepatan transfer disk – memory =40mb/detik
  - Kec. Transfer Proses = 10mb/40mb=0.25 detik
  - Asumsi :
    - tidak ada *headseek* dan rata-rata *latency* = 0.008 detik
  - Transfer sekali = (0.25+0.008)=0.258 detik
  - Transfer swap in + swap out = 0.258 + 0.258 = 0.516detik

Swapping

- Mayoritas waktu yang habis dalam proses *swap* adalah waktu transfer. Total waktu transfer harus proporsional terhadap jumlah memori yang di*swap*.
- Proses memberikan informasi kebutuhan memory kepada sistem operasi melalui *system call* (*release memory* dan *request memory*)
- Batasan swapping
  - Swap proses yang sedang *idle*
  - Jangan swap proses yang sedang menunggu operasi I/O

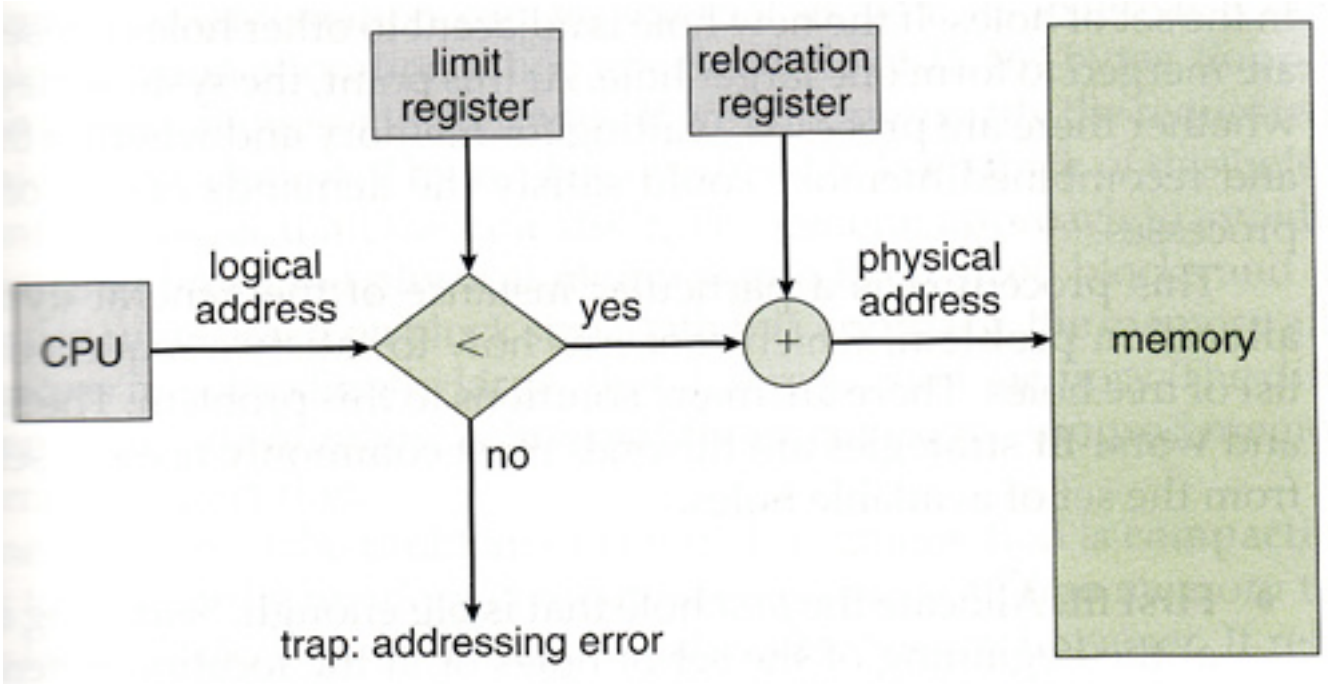
Swapping

- Model swapping yang berbeda dapat terjadi pada beberapa sistem operasi, seperti UNIX, Linux dan Windows

Pemetaan memori dan proteksi

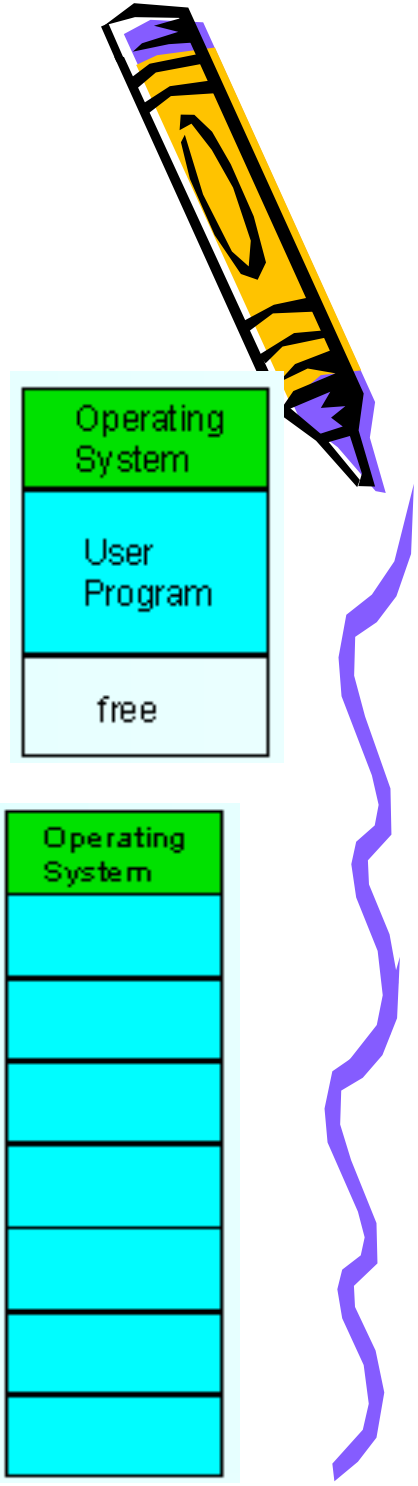
- Register Relokasi (*relocation register)* berisikan nilai terkecil alamat fisik (*physical address*)
- Register Limit *(limit register)* berisikan range alamat logika (*logical address*), setiap alamat logika harus lebih kecil dari nilai register limit.
- MMU melakukan pemetaan alamat logika
  - Jika nilai alamat logika >= nilai pada register limit, maka terjadi trap
  - Jika nilai alamat logika < nilai pada register limit, maka akses user diperbolehkan

Proteksi alamat Hardware dengan menggunakan register base dan limit



Alokasi Memori

- **Single partition**
  - Sistem operasi menempati alamat memori paling bawah
  - Satu proses user menempati area memori diatas sistem operasi
- **Multiple Partition**
  - Terdapat banyak proses dalam memori
  - SO menyimpan informasi ketersediaan kapasitas memori
  - Tipe partisi :
    - Fixed → besar partisi tetap
    - Variable/dynamic → besar partisi tergantung besar proses



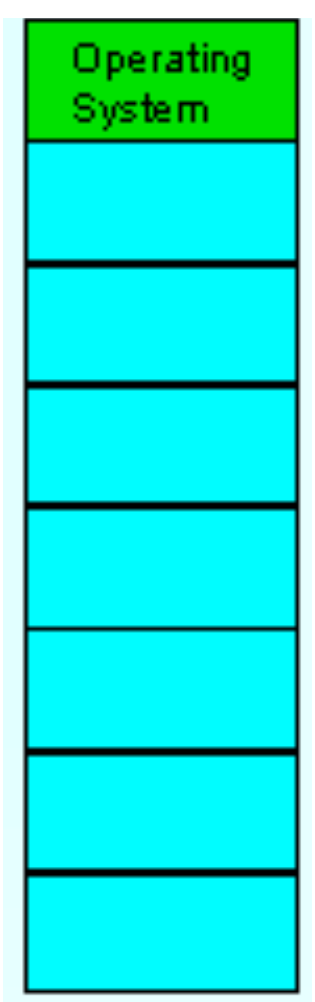
Alokasi Memori

- Fixed Partition
- Berdasarkan besar partisi, fixed partition terbagi menjadi dua tipe
  - Besar partisi sama
    - Proses kecil dapat menempati ruang memori yang besar → fragmentasi internal besar
  - Besar partisi berbeda
    - Mempunyai fragmentasi internal lebih kecil dibanding partisi yang besarannya sama
    - Antrian
      - Single input queue
      - Multiple input queue

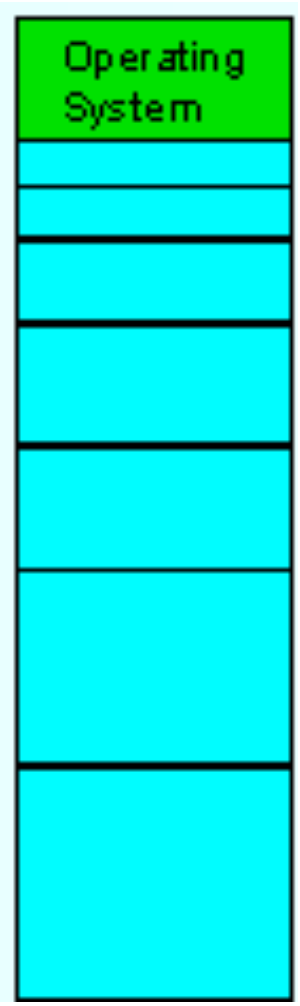
Alokasi Memori (Fixed Partition)

- Antrian
  - Single input queue
    - Proses dimasukkan kedalam besaran partisi paling kecil
      - Misalnya proses A=13 M, hanya bisa memasuki partisi yang mempunyai besaran maks 16 M
    - Fragmentasi internal lebih kecil dibanding multiple input queue
  - Multiple input queue
    - Proses mencari ke seluruh partisi, besaran partisi paling kecil untuk dapat dimasuki
    - Derajat multiprogramming lebih tinggi dibanding single input queue

Fixed Partition

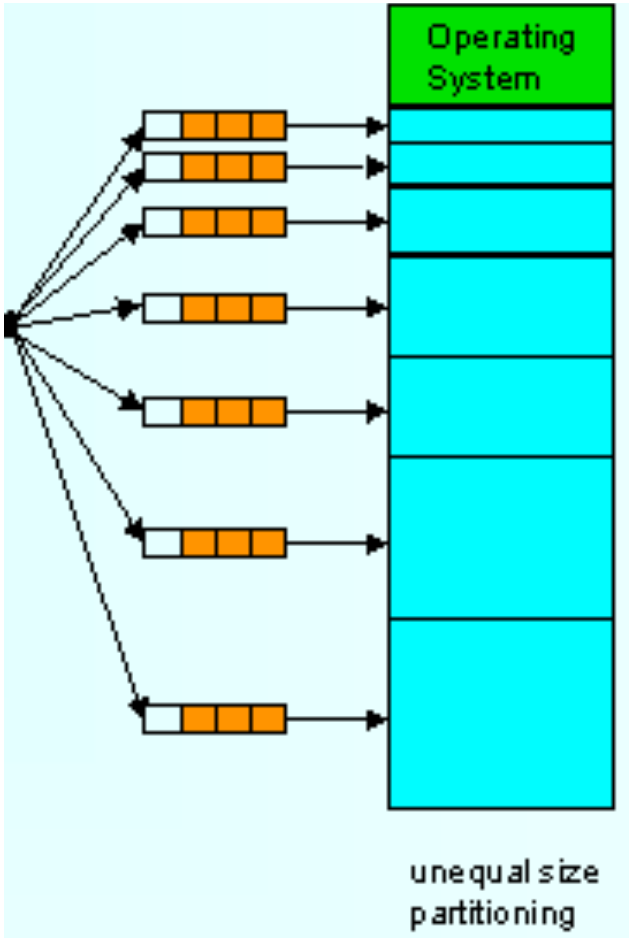


Besar partisi sama

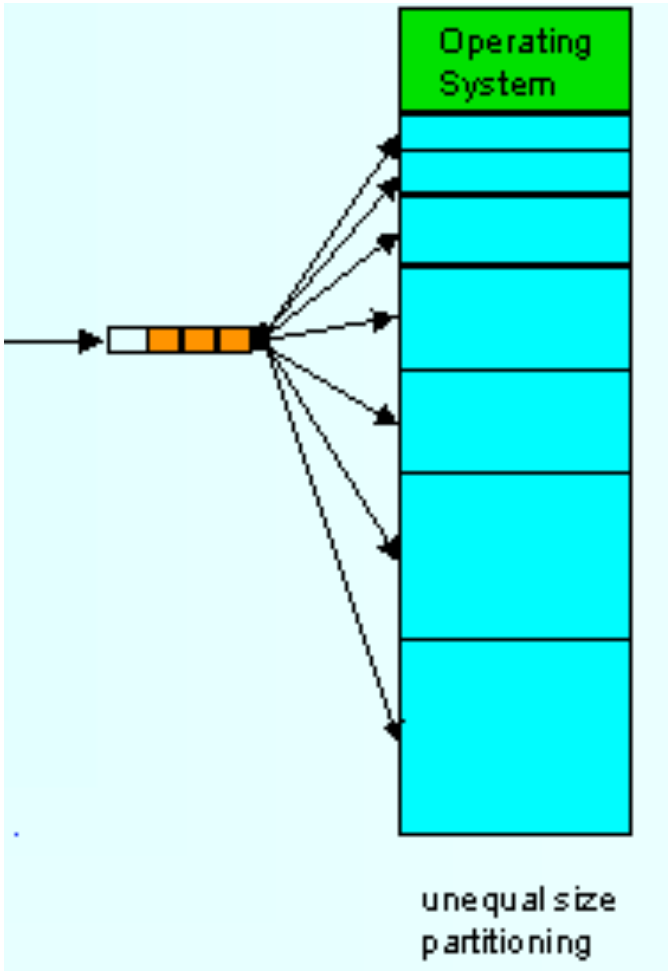


Besar partisi beda

Fixed Partion



Multiple input queue



Single input queue

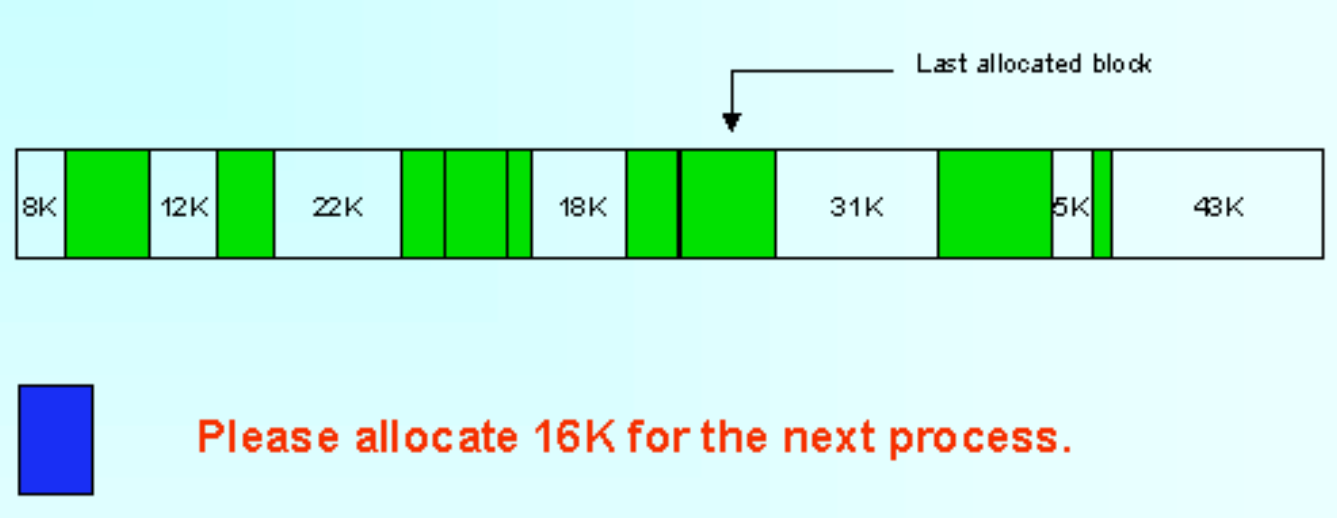
Alokasi Memori (Dynamic Partition)

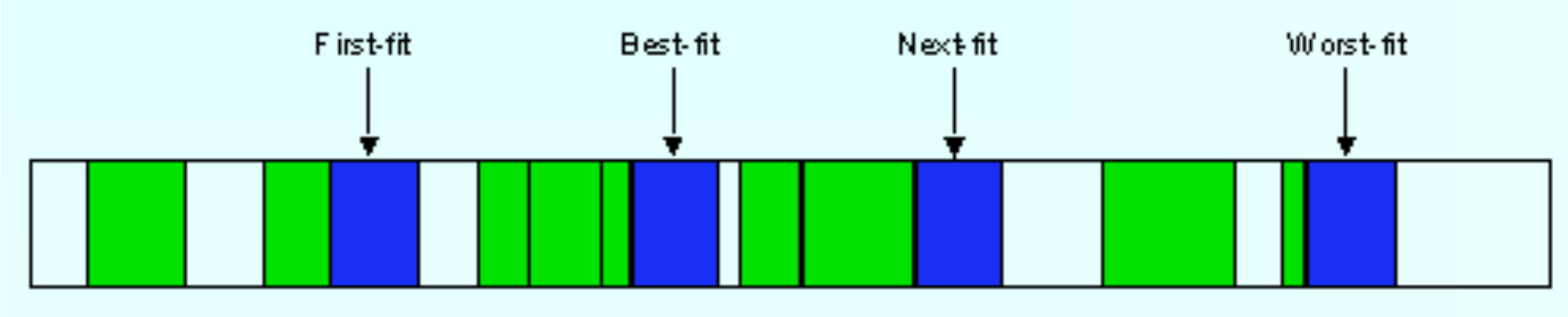
- Variable/dynamic Partition
  - Tergantung besar proses
  - Awalnya terdapat satu ruang kosong (hole) pada ruang proses user

Permasalahan alokasi penyimpanan dinamis

- Metode pemilihan partisi kosong
  - First-fit: scan memori dari awal hingga ditemukan partisi pertama yang kosong dan cukup untuk menampung proses.
  - Best-fit: Tempatkan proses pada partisi kosong yang paling kecil namun cukup dari semua partisi yang ada.
  - Next-Fit: scan memori mulai dari lokasi partisi yang terakhir ditempatkan (*last placement*), hingga ditemukan partisi pertama yang kosong dan cukup untuk menampung proses.
  - Worst-fit: Tempatkan proses pada partisi kosong yang paling besar dari semua partisi kosong yang tersedia.

Ilustrasi Alokasi Penyimpanan Dinamis



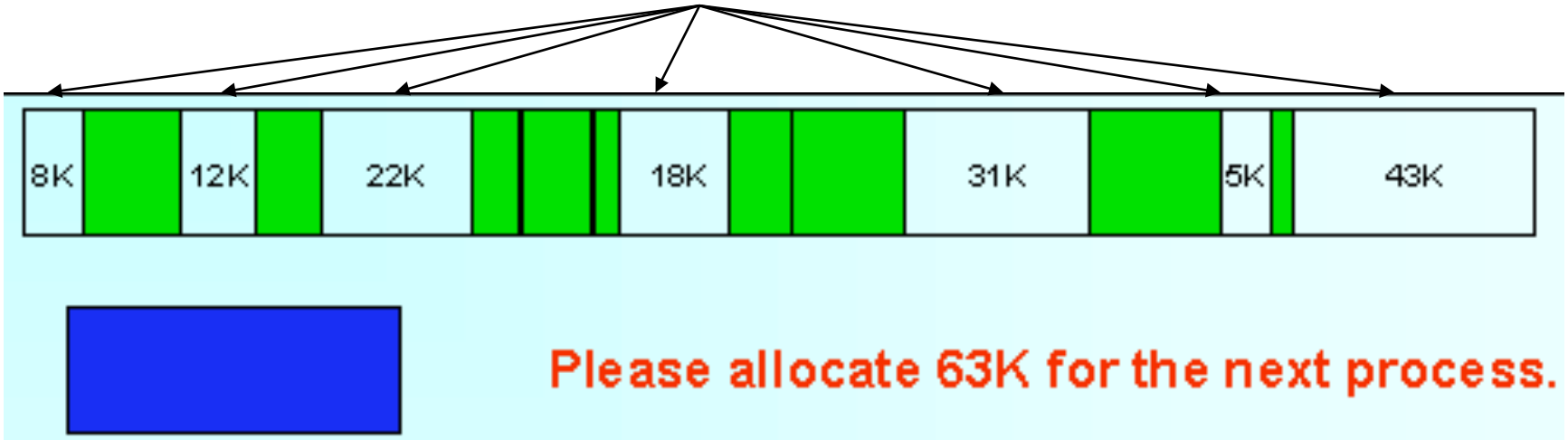


- First-fit and Best-fit lebih baik dibanding Worst-fit dalam hal kecepatan dan utilisasi media penyimpanan.

Fragmentasi

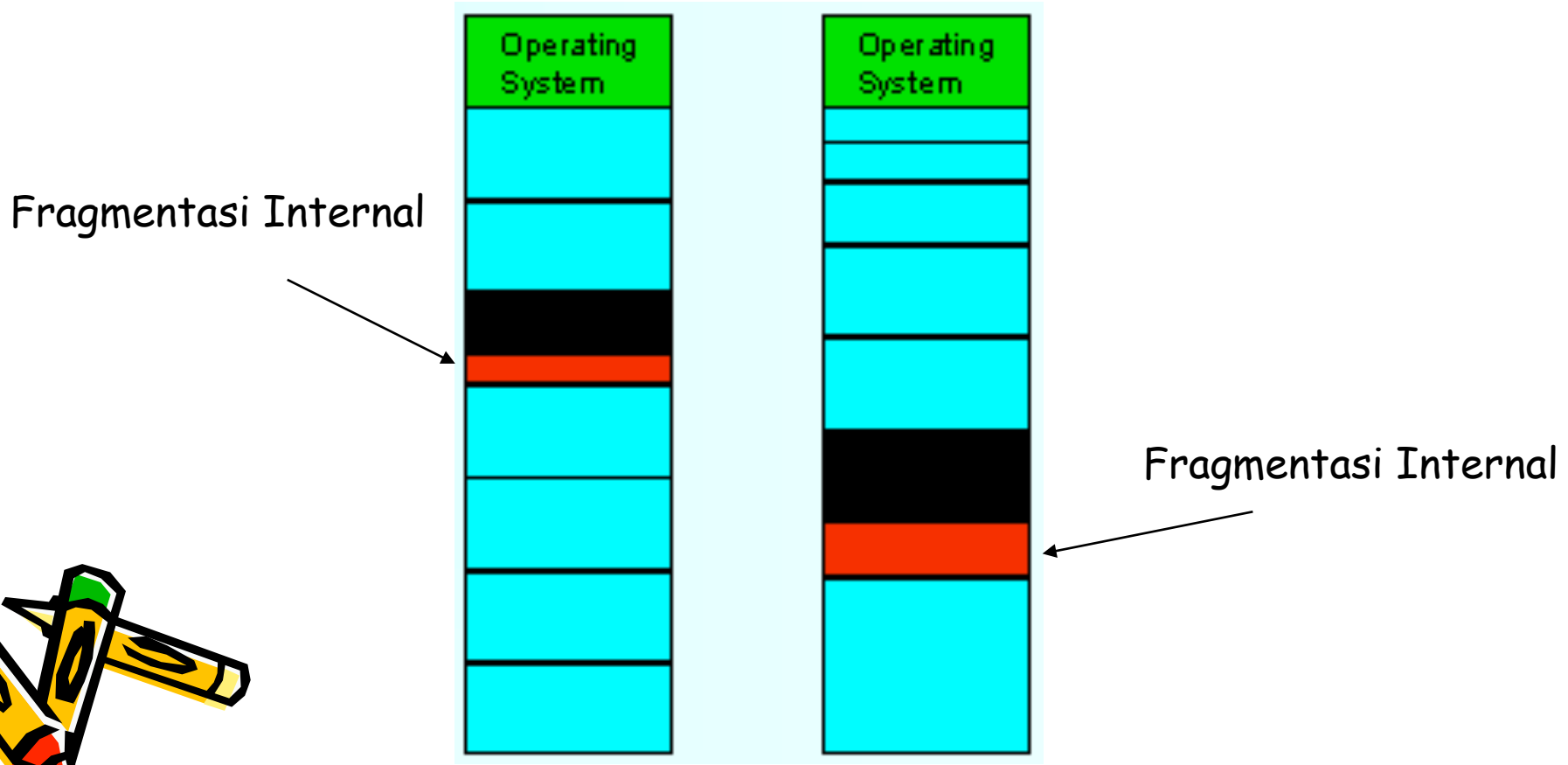
- **Fragmentasi Eksternal** ⇒ terdapat kumpulan partisi kosong berukuran kecil yang tersebar tidak terurut dan tidak dapat digunakan.
- Kumpulan partisi kosong yang tidak cukup besar untuk menampung permintaan proses yang datang.

Fragmentasi Eksternal

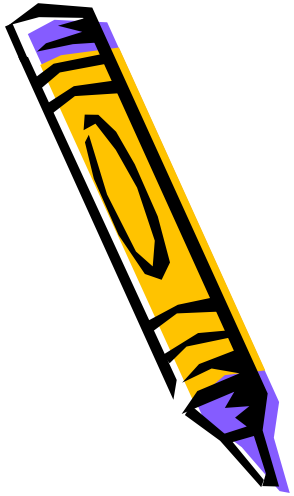


Fragmentasi

- Fragmentasi Internal ⇒ proses terlalu kecil untuk ditempatkan di blok memori yang mengakibatkan pemborosan ruang.



Compaction



- Cara Mengatasi Fragmentasi Eksternal
  - *Compaction*
    - *Compaction* $\Rightarrow$ menempatkan semua isi memori pada satu lokasi
    - Compaction hanya dapat dilakukan bila relokasi bersifat dinamis pada saat run-time
  - Pengalamatan menggunakan *paging*

Pertanyaan

- Apa yang dimaksud dengan fragmentasi eksternal ?
- Apa yang dimaksud dengan fragmentasi internal ?
- Apa yang dimaksud compaction?
- Bagaimana metode first-fit bekerja?
- Bagaimana metode Best-Fit bekerja?
- Bagaimana metode Worst-fit bekerja?