Laporan Praktikum

Management Memory

Sistem Operasi

Disusun Oleh:

Nama: Daniel Siahaan
NIM: 11322014
Prodi: Teknologi Informasi

INSTITUT TEKNOLOGI DEL
FAKULTAS VOKASI

Institut Teknologi Del

Minggu/Sesi	:	XII/2 dan 3
Kode Matakuliah	:	1031202/1041202
Nama Matakuliah	:	Sistem Operasi
Panduan Kuliah	:	Panduan ini dibuat untuk mengarahkan mahasiswa memahami mengenai *Management Memory* dalam upaya membantu mereka melaksanakan Pembelajaran Jarak Jauh (PJJ). Panduan ini adalah bagian pertama untuk topik Manajemen Memori.
Setoran	:	Laporan Materi *Management Memory* dikirimkan dalam bentuk PDF dengan aturan penamaan file adalah No_Kelompok_ManagementMemory.
Tujuan	:	1. Mampu menjelaskan tujuan dari manajemen memori. 2. Mampu menjelaskan mengenai relocation, protection, sharing, logical organization dan physical organization dalam melakukan manajemen memori. 3. Mampu menjelaskan jenis-jenis teknik partisi pada memori seperti fixed partitioning, dynamic partitioning, simple paging dan simple segmentation. 4. Mampu menerapkan algoritma penempatan pada dynamic partitioning.

Petunjuk

1. Tugas ini dikerjakan secara individu.
2. Mencontoh pekerjaan dari orang lain akan dianggap plagiarisme dan anda akan ditindak sesuai dengan sanksi akademik yang berlaku di IT Del atau sesuai dengan kebijakan saya dengan memberikan nilai 0.
3. Jawaban diketikkan dalam bentuk laporan mengikuti template yang telah disediakan di- ecourse dan setiap soal harus ditulis secara berurutan.
4. Keterlambatan menyerahkan laporan tidak ditolerir dengan alasan apapun. Oleh karena itu, laporan harus dikumpul tepat waktu.

Management Memory

Teori

1. Jelaskan mengapa manajemen memori diperlukan?

   Jawab:

   Manajemen memori diperlukan karena komputer memiliki keterbatasan dalam jumlah memori fisik yang tersedia untuk menampung program dan data. Manajemen memori diperlukan untuk mengelola dan memantau penggunaan memori oleh sistem operasi dan program yang berjalan di atasnya.
2. Jelaskan fungsi manajemen memori!

   Jawab:

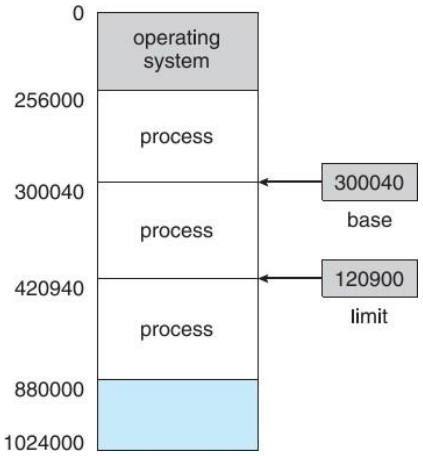
   Fungsi manajemen memori adalah untuk mengelola dan memantau penggunaan memori oleh sistem operasi dan program yang berjalan
3. Jelaskan sifat dari manajemen memori pada sistem operasi yang mendukung *monoprogromming* dan *multiprogramming*!

   Jawab:

   Pada sistem operasi monoprogramming, hanya satu program yang dapat berjalan pada satu waktu. Alokasi memori dapat dilakukan secara statis, dengan program dijalankan dari awal hingga selesai, dan memori akan secara otomatis dibebaskan setelah program selesai dijalankan.

   Sifat manajemen memori pada sistem operasi multiprograming lebih kompleks. Alokasi dapat dilakukan secara dinamis, dengan memori dialokasikan saat program dimulai dan didealokasikan setelah program selesai dijalankan.
4. Jelaskan definisi dari:
   a. *Frame*: unit memori fisik terkecil yang digunakan dalam manajemen memori paging pada sistem operasi. Frame biasanya berukuran 4KB atau 8KB dan merepresentasikan blok memori fisik yang dapat dialoakasikan ke sebuah halaman dalam manajemen memori paging.
   b. *Page*: unit memori virtual terkecil yang digunakan dalam manajemen memori paging pada sistem operasi. Setiap program yang berjaalan pada sistem memiliki sejumlah halaman yang diakses oleh program tersebut.
   c. *Segment*: unit logis terbesar dari suatu program dalam manajemen memori segmen pada sistem operasi. Segmen merepresentasikan bagian program atau data yang memiliki sifat yang sama, seperti kode, data, atau stack.

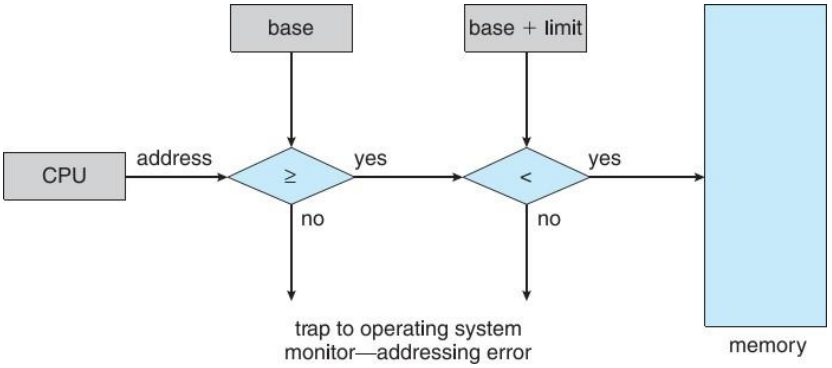
5. Jelaskan definisi dari:
    a. *Register base*: nilai register yang menyimpan alamat awal dari blok memori yang ditugaskan untuk sebuah proses dalam manajemen memori basis register pada sistem operasi.
    b. *Register limit*: nilai register yang menyimpan ukuran maksimal blok memori yang dapat diakses oleh sebuah proses dalam manajemen memori basis register pada sistem operasi.
    c. *Symbolic address*: alamat yang digunakan dalam kode program sebagai simbol atau nama, yang kemudian akan dikonversi menjadi alamat fisik pada saat program dijalankan.
    d. *Relocatable address*: alamat yang dapat berubah-ubah dalam kode program ketika kode program tersebut dipindahkan ke lokasi memori yang berbeda pada saat program dijalankan.
    e. *Absolute address*: alamat fisik yang tetap dari suatu lokasi memori dalam sistem. Alamat absolut didefinisikan pada saat kode program dibuat dan tetap tidak berubah saat program dijalankan.
    f. *Logical address*: alamat yang digunakan oleh program saat melakukan akses ke memori virtual pada sistem. Logical address diproses oleh unit manajemen memori pada CPU dan dikonversi menjadi alamat fisik sebelum diakses oleh proses.
    g. *Physical address*: alamat fisik yang benar-benar mengidentifikasi lokasi memori dalam sistem, digunakan oleh unit manajemen memori pada CPU untuk mengakses memori fisik dan menempatkan data dan instruksi pada lokasi memori yang tepat.
6. Jelaskan bagaimana ruang memori setiap proses harus dipisahkan sebagai bentuk proteksi agar sebuah proses tidak dapat mengakses ruang memori proses yang lain. Berikan penjelasan Anda melalui gambar di bawah.



Jika nilai pada register basis adalah 300040 dan register limit bernilai 120900, maka program hanya dapat mengakses alamat dari 300040 hingga 420939, dan jika program mencoba mengakses alamat di luar rentang tersebut, maka sistem operasi harus melakukan proteksi. Salah satu tindakan yang dapat dilakukan adalah menghentikan eksekusi program atau membuang program tersebut agar tidak dapat mengakses proses lain. Pada gambar yang diberikan, terdapat proses yaitu A, B, dan C, dan proteksi akan dilakukan pada proses B untuk mencegah akses dari proses A dan C.

Oleh karena itu, ada 3 tahap waktu yang terkait dengan manajemen memori, yaitu waktu kompilasi (compile time), waktu pemuatan (load time), dan waktu eksekusi (execution time).

7. Jelaskan bagaimana alur proteksi untuk mengakses memori melalui gambar di bawah.



- ⇨ Cpu harus melakukan pengecekan address setiap proses dan akan melihat base register dan limit register
- ⇨ Jika terdapat kondisi yes dimana base alamat proses lebih besar sama dengan

⇨ Akan dicek kembali limit register jika kondisi tersebut juga sesuai maka akan lanjut ke main memory

⇨ Tetapi jika kondisi tidak sesuai atau bernilai no maka operating system akan melakukan traping atau bisa juga dikeluarkan dari pengeksekusian, atau determinasi, maupun discard.

8. Mengapa *relocation* pada manajemen memori perlu ditangani?

   Jawab: Relocation pada manajemen memori perlu ditangani karena ketika suatu program dijalankan, program tersebut memerlukan alokasi memori yang dapat berbeda-beda pada setiap kali program dijalankan.

9. Mengapa *relocation* penting dalam manajemen memori?

   Jawab: Relocation sangat penting dalam manajemen memori karena memungkinkan program dapat dijalankan pada lokasi memori yang berbeda tanpa perlu melakukan perubahan pada kode program asli.

10. Mengapa *protection* perlu ditangani?

    Jawab:

    - Mencegah akses yang tidak sah: Proteksi memastikan bahwa proses hanya dapat mengakses memori yang diizinkan untuk proses tersebut.
    - Mencegah perubahan data yang tidak sah: Proteksi juga melindungi data dari modifikasi yang tidak diizinkan.
    - Mencegah serangan dari luar: Proteksi juga melindungi sistem dari serangan dan malware yang mencoba mengambil alih kontrol sistem atau mencuri data yang sensitif.
    - Mencegah konflik antar proses: Proteksi memastikan bahwa setiap proses hanya dapat mengakses data atau kode program yang diizinkan untuk proses tersebut, sehingga menghindari konflik dan kerusakan pada sistem.

11. Mengapa *sharing* perlu ditangani?

    Jawab:

    Penggunaan memori yang efisien: Sharing memungkinkan beberapa proses untuk menggunakan memori yang sama secara bersamaan.

    Meningkatkan kecepatan: Sharing memungkinkan beberapa proses untuk menggunakan data yang sama secara bersamaa, yang dapat meningkatkan kecepatan sistem.

    Komunikasi antar proses: Sharing memungkinkan proses untuk berkomunikasi dengan mudah. Jika suatu proses memerlukan akses ke data yang dimiliki oleh proses lain,

sharing memungkinkan untuk berbagai data tanpa perlu mengalihkan data atau menggunakan teknik komunikasi yang lebih kompleks.

12. Mengapa *local organization* dan *physical organization* perlu ditangani?

    Jawab:

    ⇨ Local organization dikarenakan main memory dikelompokkan secara linier atau berdimensi satu yang berupa ruang alamat yang terdiri dari deretan byte tau word.

    ⇨ Physical organization dikarenakan diperlukan metode yang mengatur aliran data dari memori utama ke memori sekunder dan sebaliknya

13. Berikan perbandingan antara *fixed partitioning*, *dynamic partitioning*, *simple paging* dan *simple segmentation*! Buatkan dalam bentuk tabel.

    Jawab:

Nama	Pemabagian memory	Kelebihan	Kekurangan
Fixed partitioning	⇨ Saat startup sistem ⇨ Satu partisi (berdekatan) per job	Mendukung multiprogramming	⇨ Mengharuskan loading program secara contiguous/berdekatan ⇨ Metode alokasi job Partisi pertama yang tersedia dan ukurannya lebih besar dari ukuran program ⇨ Untuk dapat bekerja dengan baik: Semua job harus berukuran sama dan ukuran memori diketahui terlebih dahulu ⇨ Ukuran partisi yang dibuat asalasalan berakibat buruk. Misalnya: Partisi yang terlalu kecil Job-job yang terlalu besar turnaround nya menjadi semakin besar.

Dynamic partitioning	⇨ Memori untuk job dialokasikan ketika di load ⇨ Satu partisi contiguous per jo	Keterbuangan memori relatif kecil	⇨ Ulitisasi memori secara penuh hanya ketika job pertama yang di load. – Subsequent allocation: memori terbuang ⇨ External fragmentation: fragments diantara blok

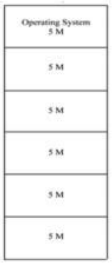
Simple paging	⇨ Memori virtual dibagi menjadi blok-blok yang ukurannya tetap yang dinamakan page (ukurannya adalah pangkat 2, diantara 512 bytes dan 8192 bytes, tergantung arsitektur) ⇨ Memori fisik dibagi juga menjadi blokblok yang ukurannya tetap yang dinamakan frame. ⇨ Lalu kita membuat suatu page table yang akan menterjemahkan memori virtual menjadi memori fisik.	⇨ Jika kita membuat ukuran dari masing masing pages menjadi besar akses memori akan relatif lebih cepat. ⇨ Jika kita membuat ukuran dari masing-masing pages menjadi kecil kemungkinan terjadinya fragmentasi internal akan menjadi lebih kecil.	⇨ Jika kita meml dari ukuran masingmasingnjadi pages besar kemungkinan terjadinya fragmentasi yang internal sangat besar. kita ⇨ Jika membuat ukuran - dari masing ges masing cil menjadi ak memori a relatif le lambat.

Simple Segmentat Ion	⇨ Alamat logika terdiri dari dua bagian yaitu nomor segmen (s) dan offset (d) yang dituliskan dengan <nomor segmen,offset> ⇨ Pemetaan alamat logika ke alamat fisik menggunakan tabel segmen (segment table),	Bila ada proses yang besar maka segmentasi: ⇨ Saling berbagi. ⇨ Protek si.	⇨ Segmen dapat membesar. ⇨ Muncul fragmentasi luar
	terdiri dari: Segmen basis (base) berisi alamat fisik awal		

14. Pada *fixed partitioning* terdapat dua teknik yaitu *equal-size* dan *unequal-size*. Berikan penjelasan Anda terhadap kedua teknik tersebut dan lengkapi dengan contoh.

Equal size partition

Ukuran setiap blok di partisi tetap akan sama. Setiap proses yang kurang dari ukuran partisi dapat dimuat di partisi tetap dengan ukuran yang sama. Contoh:

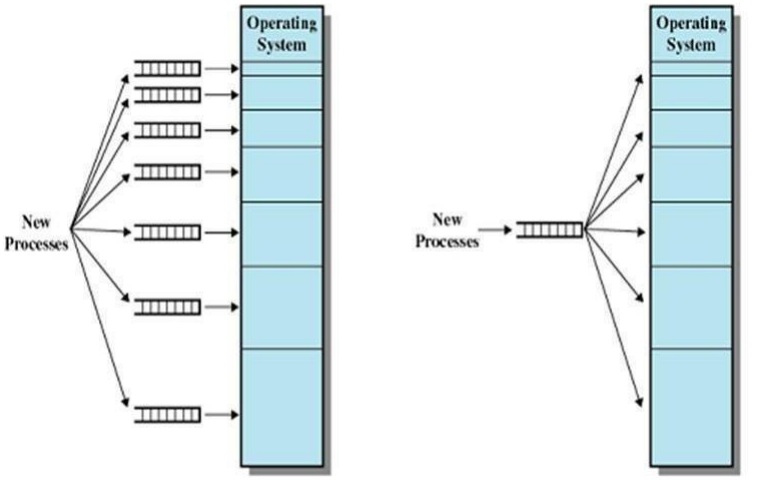


⇨ **Unequal size partition**

Ukuran setiap blok di partisi tetap bervariasi di mana proses ditugaskan ke blok di mana ia cocok: dengan kata lain, proses mungkin antri untuk menggunakan partisi terbaik yang tersedia. Dalam partisi dengan ukuran yang tidak sama dibandingkan dengan partisi dengan ukuran yang sama, pemborosan memori diminimalkan, dan mungkin tidak memberikan throughput terbaik karena beberapa partisi mungkin tidak digunakan. Partisi

ukuran yang tidak sama menggunakan dua jenis antrian di mana proses ditugaskan ke blok memori. Mereka adalah antrian ganda dan antrian tunggal.

15. Pada *fixed partitioning* dengan model *unequal-size* terdapat dua jenis antrian seperti pada gambar di bawah. Jelaskan masing-masing model tersebut.



(a) One process queue per partition (b) Single queue

a. One process quee per partition

   a. Keuntungan:

   Fragmentasi internal berkurang

   b. Kekurangan:

   Tidak optimal dari sudut pandang sistem

b. Single quee

   c. Keuntungan:

      Tingkat fleksibilitas, sederhana, membutuhkan OS S/w minimal dan overhead pemrosesan

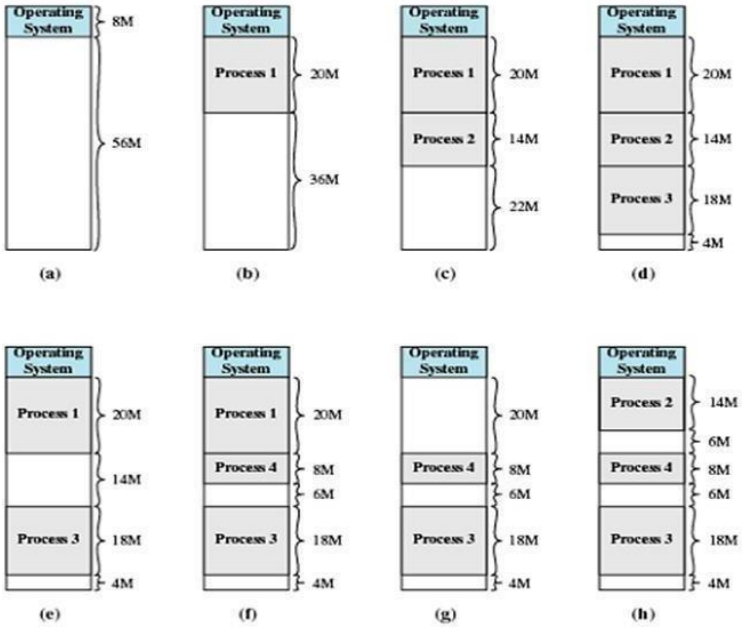
   d. Kekurangan:

1. Membatasi jumlah proses aktif dalam sistem
2. Pekerjaan kecil tidak akan memanfaatkan ruang partisi secara efisien

16. Jelaskan kelebihan dan kekurangan dari *fixed partitioning*.

Jawab: Pada saat pembuatan sistem, memori utama dibagi menjadi beberapa partisi statis. Setiap partisi memiliki ukuran yang sama atau lebih besar sehingga sebuah proses dapat dimuat ke dalam partisi tersebut. Metode ini memiliki kelebihan yaitu sederhana untuk diterapkan dan memiliki sedikit overhead pada sistem operasi. Namun, kelemahannya adalah penggunaan memori yang tidak efisien karena adanya fragmentasi internal. Selain itu, jumlah maksimum proses aktif telah ditetapkan secara kaku.

17. Jelaskan pengertian Anda mengenai *dynamic partitioning* melalui gambar di bawah.



⇨ Partisi Dinamis

1. Partisi dibuat secara dinamis, sehingga setiap proses dimuat ke dalam partisi dengan ukuran yang sama persis dengan proses tersebut.
2. Tidak ada fragmentasi internal; penggunaan memori utama lebih efisien.
3. Penggunaan prosesor yang tidak efisien karena perlunya pemadatan untuk melawan fragmentasi eksternal.

18. Jelaskan mengapa fragmentasi eksternal dapat terjadi pada *dynamic partitioning* dan jelaskan solusinya.

Jawab:

Dynamic partitioning adalah teknik pembagian memori dimana memori utama dibagi menjadi beberapa partisi yang memiliki ukuran yang bervariasi sesuai dengan kebutuhan proses. Pada dynamic partitioning, proses ditempatkan pada partisi yang cukup besar untuk menampung keseluruhan proses.

Fragmentasi eksternal dapat terjadi pada dynamic partitioning karena partisi-partisi yang digunakan oleh proses tidak selalu memiliki ukuran yang sama. Hal ini dapat menyebabkan adanya ruang kosong di antara partisi-partisi tersebut yang tidak dapat digunakan oleh proses yang membutuhkan memori yang lebih besar dari ukuran ruang kosong tersebut. Dengan kata lain, terdapat fragmen-fragmen memori yang tersisa antara partisi-partisi yang tidak dapat digunakan, meskipun sebenarnya total memori yang kosong tersedia di dalam sistem masih cukup.

Solusi untuk mengatasi fragmentasi eksternal pada dynamic partitioning adalah dengan melakukan kompaksi (compaction). Kompaksi adalah proses penggabungan fragmen-fragmen kecil memori yang kosong tersebut menjadi satu ruang kosong yang lebih besar dan dapat digunakan oleh proses-proses selanjutnya. Namun, proses kompaksi memerlukan waktu yang relatif lama dan overhead yang tinggi, karena memerlukan pemindahan data dari satu lokasi ke lokasi lainnya di dalam memori. Oleh karena itu, kompaksi biasanya dilakukan hanya ketika tidak ada lagi ruang kosong yang cukup besar untuk menampung proses-proses baru.

19. Jelaskan tiga algoritma penempatan (*placement*) yang digunakan pada *dynamic partitioning* dan urutkan ketiga algoritma tersebut dimulai dari kinerja yang paling baik hingga paling buruk.

Terdapat 3 algoritma penempatan (placement) antara lain:

- Best-fit

  Memilih blok yang ukurannya paling dekat dengan permintaan

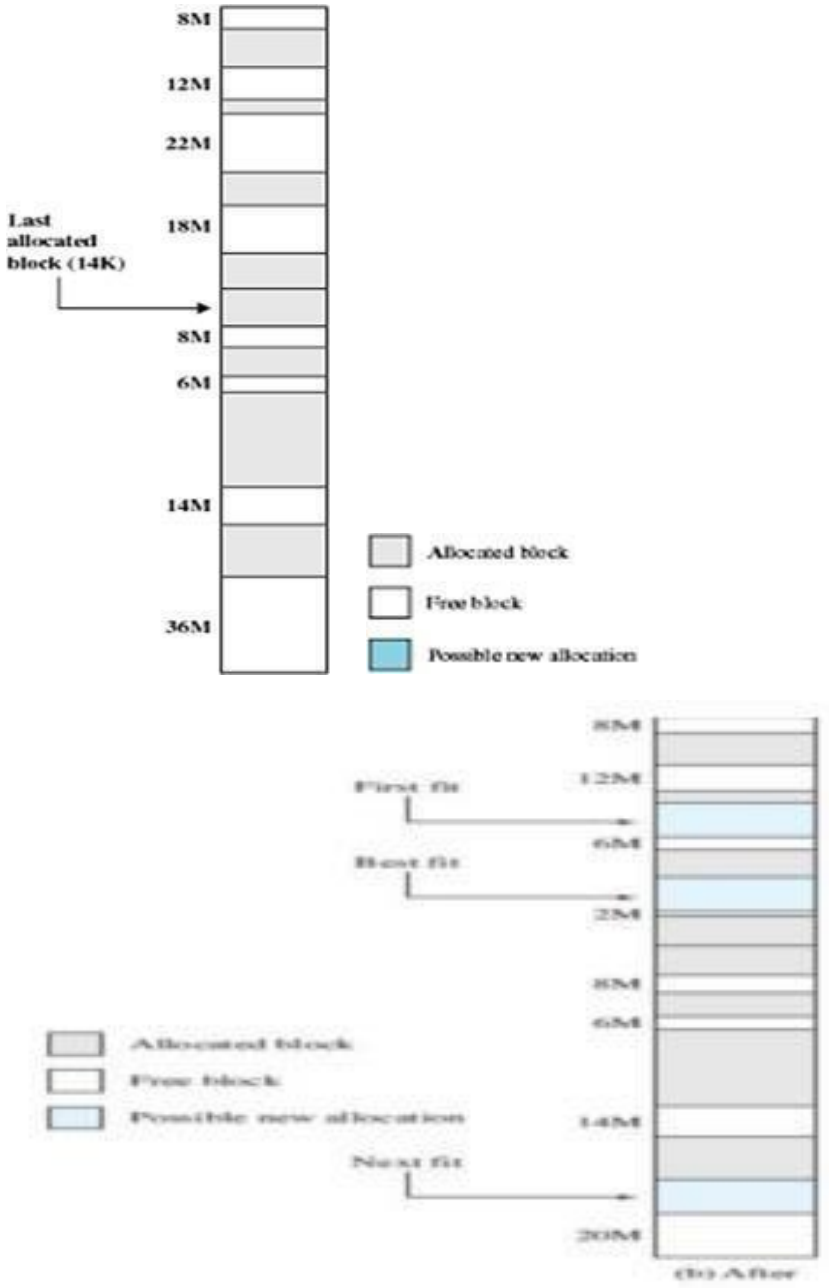
- First-fit

Memindai memori dari awal dan memilih blok pertama yang tersedia yang cukup besar

- Next-fit

  Pindai memori dari lokasi penempatan terakhir dan pilih blok berikutnya yang tersedia yang cukup besar

20. Diketahui sebuah proses baru dengan ukuran 14 MB akan diletakkan ke memori dengan gambar blok memori seperti gambar di bawah. Dengan mengikuti ketiga algoritma penempatan pada No. 17, gambarkan blok memori yang baru, beri tanda blok memori dari setiap algoritma dan sisa blok memori dari setiap algoritma.



21. Jelaskan kelebihan dan kekurangan dari *dynamic partitioning*.

 Pada saat sistem dibuat, memori utama dibagi menjadi beberapa partisi statis dengan ukuran yang sama atau lebih besar. Sebuah proses dapat dimuat ke dalam partisi yang memiliki ukuran sama atau lebih besar dari proses tersebut.

Keuntungannya adalah pengaplikasiannya yang sederhana dan overhead sistem operasi yang sedikit. Namun, kelemahannya adalah penggunaan memori yang tidak efisien karena adanya fragmentasi internal, dan jumlah maksimum proses aktif telah ditentukan sebelumnya.

22.

Praktikum

A. Manajemen Memori pada LINUX

Perintah free digunakan untuk menampilkan jumlah memori fisik dan memori swap (VM), total yang belum dan yang telah digunakan oleh komputer, termasuk juga jumlah memori yang digunakan secara bersama-sama dan buffer yang digunakan oleh kernel.
Terdapat beberapa parameter yang dapat digunakan oleh perintah free. Berikut adalah parameter yang dapat digunakan antara lain:

-b menampilkan jumlah memori dalam byte (B)
-k menampilkan jumlah memori dalam kilobyte (KB), ini digunakan sebagai default.
-m menampilkan jumlah memori dalam megabyte (MB).
-t menampilkan sebuah baris yang berisi total.
-V menampilkan informasi mengenai versi dari perintah free yang kita gunakan.

Ketikkan perintah free dan output eksekusi perintah tersebut. Lakukan langkah-langkah berikut dan amati hasilnya:

1. Ketikkan perintah **free**

```
sanhenra@ubuntu: ~$ free  total  used
      free shared buffers cached
Mem: 1016832   320204 696628   0          13716
148336
-/+ buffers/cache: 158152 858680
Swap: 1646620   0 1646620
```

2. Ketikkan perintah **free** dengan menambahkan opsi **-t**

```
sanhenra@ubuntu: ~$ free -t total used
        free shared buffers cached
Mem: 1016832    338780 678052   0           14008
163228
-/+ buffers/cache: 161544 855288
Swap: 1646620   0 1646620
Total: 2663452  338780 2324672
```

3. Ketikkan perintah **free** dengan menambahkan opsi **-b**

```
sanhenra@ubuntu:  ~$  free  -b  total
        used free shared buffers cached
Mem:   1041235968 346853376
694382592                                0 14352384 167145472
-/+ buffers/cache: 165355520
875880448
Swap: 1686138880   0 1686138880
```

4. Ketikkan perintah **free** dengan menambahkan opsi **-k**

```
sanhenra@ubuntu:  ~$  free  -k  total
        used free shared buffers cached
Mem: 1016832 338724 678108      0           14016
163228
-/+ buffers/cache: 161480 855352
Swap: 1646620   0 1646620
```

5. Ketikkan perintah **free** dengan menambahkan opsi **-m**

```
sanhenra@ubuntu: ~$ free -m
      total used free shared buffers cached Mem:
       993330 662 0 13          159
-/+ buffers/cache: 157   835
Swap:   1608      0    1608
```

6. Ketikkan perintah **free** dengan menambahkan opsi **-V**

```
sanhenra@ubuntu: ~$ free -V
```

```
procps version 3.2.7
```

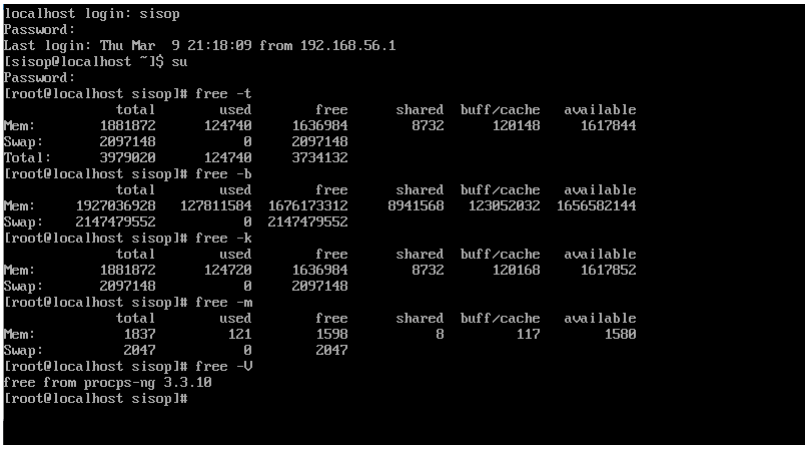
Sekarang anda bandingkan hasil yang anda peroleh tadi dengan isi file /proc/meminfo. File /proc/meminfo ini berisi tentang informasi memori yang terdapat pada komputer kita.

7. Amati isi file /proc/meminfo, apa kesimpulan Anda?

```
[root@si ~]# cat
/proc/meminfo
MemTotal: 2059440 kB
MemFree: 293156 kB
Buffers: 178696 kB
Cached: 1042952 kB
SwapCached: 0 kB
Active: 1045888 kB Inactive:
506152 kB
HighTotal:  0 kB
HighFree:   0 kB
LowTotal: 2059440 kB
LowFree: 293156 kB
SwapTotal: 2048248 kB
SwapFree: 2048144 kB
Dirty:     196 kB
Writeback:   0 kB
AnonPages: 330372 kB
Mapped:  59128 kB
Slab:   163644 kB
PageTables: 28256 kB
NFS_Unstable: 0 kB
Bounce:     0 kB
CommitLimit: 3077968 kB
Committed_AS: 1390340 kB
VmallocTotal: 34359738367
kB
```

VmallocUsed: 263888 kB
VmallocChunk: 34359473775 kB
HugePages_Total: 0
HugePages_Free: 0
HugePages_Rsvd: 0
Hugepagesize: 2048 kB

Output Program:



B. Manajemen memori pada DOS

Untuk memeriksa memori yang terdapat pada komputer yang menggunakan sistem operasi DOS (Disk Operating Systems) dapat menggunakan perintah MEM. Perintah MEM akan menampilkan informasi tentang daerah memori yang telah dialokasikan, daerah memori yang belum digunakan, dan program-progam yang telah dimuat. Selain itu, informasi lain dapati diperoleh dengan cara menyertakan parameter tambahan pada saat memberikan perintah.

PERHATIAN:

Parameter-parameter yang disajikan di sini dapat berjalan di DOS 98 atau sebelumnya, seperti DOS 6.0. Apabila anda menjalankan DOS emulator pada sistem operasi Windows XP maka tidak semua parameter di bawah ini berjalan dengan baik. Anda dapat mengetahui parameter apa saja yang tersedia beserta keterangan fungsinya dengan menjalankan perintah berikut pada shell prompt anda:

8. Ketikkan perintah **mem** dengan menambahkan opsi **/?**

```
C:\Users\good>mem /?
Displays the amount of used and free memory in your system.

MEM [/PROGRAM | /DEBUG | /CLASSIFY]

/PROGRAM or /P Displays status of programs currently loaded in memory.
/DEBUG or /D Displays status of programs, internal drivers, and other information.
/CLASSIFY or /C Classifies programs by memory usage. Lists the size of programs, provides a summary of memory in use, and lists largest memory block available.
```

9. Ketikkan perintah **mem** dengan menambahkan opsi **/CLASSIFY atau /C.** Menginformasikan program-program yang telah dimuat ke dalam memori dan jumlah memori yang digunakan masing-masing program tersebut. Selain itu juga ditampilkan informasi seperti yang ditampilkan pada perintah MEM tanpa parameter.

```
C:\Users\good>mem /C

Conventional Memory:

 Name    Size in Decimal Size in Hex

----------------------------------------
MSDOS    12288 (12.0K) 3000
 KBD      3360  (3.3K)  D20
HIMEM     1248  (1.2K)  4E0
COMMAND3968 (3.9K)    F80
 DOSX    34704 (33.9K) 8790
 FREE      112  (0.1K)   70
 FREE    599488 (585.4K) 925C0

Total FREE: 599600 (585.5K)
```

```
Upper Memory:

 Name    Size in Decimal Size in Hex

----------
SYSTEM  200688 (196.0K) 30FF0
 DOSX      128  (0.1K)    80
 MOUSE    12528 (12.2K) 30F0
MSCDEXN    352  (0.3K)   160
T REDIR   2176  (2.1K)   880
 FREE      1168  (1.1K)   490
 FREE     44976 (43.9K) AFB0

Total FREE: 46144 (45.1K)
Total bytes available to programs (Conventional+Upper):    645744 (630.6K)
Largest executable program size: 598288 (584.3K)
```

```
Largest available upper memory        44976 (43.9K)
block:

  1048576 bytes total contiguous
    extended memory 0 bytes
    available contiguous extended
    memory
  941056 bytes available XMS
  memory
 MS-DOS resident in High Memory
                  Area
```

10. Ketikkan perintah **mem** dengan menambahkan opsi **/DEBUG atau /D**. Menampilkan daftar program dan driver yang telah dimuat ke dalam memori. Juga ditampilkan ukuran setiap modul, alamat segmen, jenis modul, ringkasan Ketikkan memori secara keseluruhan, dan informasi lain yang sangat berguna untuk pemrograman.

```
C:\Users\good>mem /D

Address Name Size Type

 ----------
000000  000400 Interrupt Vector
000400                  000100 ROM
Communication Area
000500                   000200 DOS
Communication Area

000700 IO   000370 System Data
CON
                 System Device Driver
       AUX   System Device Driver
       PRN    System Device Driver
      CLOCK$ System Device Driver
       COM1  System Device Driver
       LPT1   System Device Driver
       LPT2   System Device Driver
       LPT3   System Device Driver
       COM2  System Device Driver
       COM3  System Device Driver
       COM4  System Device Driver

000A70 MSDOS 001690 System
Data


002100IO  002150 System Data
KBD
          000D20 System Program
          HIMEM
            0004E0  DEVICE=
     XMSXXXX0
Installed Device Driver
```

```
          000490 FILES=
          000090 FCBS=
          000200 LASTDRIVE=
          0007D0 STACKS=
004260 COMMAND 000A20
Program
 004C90 MSDOS 000070 -- Free
 -- 004D10 COMMAND 000560
 Environment
005280 DOSX 008790 Program
00DA20 MEM      0004A0
Environment
00DED0  MEM 0174E0 Program
 0253C0          MSDOS
          07AC20 -- Free --
 09FFF0          SYSTEM  031000
 System Program

 0D1000 IO 003100 System Data
       MOUSE     0030F0  System
       Program
 0D4110 MSDOS 000490 -- Free
 -- 0D45B0 MSCDEXNT 000160
 Program
0D4720 REDIR 000880 Program
0D4FB0 DOSX 000080 Data
0D5040 MSDOS 00AFB0 -- Free --


  655360 bytes total conventional
  memory 655360 bytes available
  to MS-DOS 598288 largest
  executable program size
```

```
1048576 bytes total contiguous
   extended memory 0 bytes
   available contiguous extended
   memory
941056 bytes available XMS
memory
    MS-DOS resident in High
    Memory Area
```

11. Ketikkan perintah **mem** dengan menambahkan opsi **/FREE atau /F.**

Menampilkan daerah yang masih belum digunakan pada memori konvensional dan upper memory area. Juga ditampilkan alamat segment dan ukuran masing-masing daerah yang masih kosong tersebut.

```
C:\Users\good>mem /F
```

Cobalah opsi lainnya dan jelaskan kegunaan masing-masing opsi tersebut!