SISTEM BERKAS

Penyusun :

Samsoni

Saprudin

Mochammad Bagoes Satria J

UNPAM PRESS
Lembaga Penerbit & Publikasi

Gd. A; R. 212 Universitas Pamulang

Jl. Surya Kencana No. 1 Pamulang | Tangerang Selatan | Banten

SISTEM BERKAS

Penulis:
Samsoni
Saprudin
Mochammad Bagoes Satria J

ISBN: **978-623-7833-61-1**

Editor:
Hadi Zakaria

Desain sampul dan tata letakDesain Sampul:
Ubaid Al Faruq, M.Pd.

Tata Letak:
Aden, S.Si., M.Pd.

Penerbit
UNPAM PRESS

Redaksi:
JL. Surya Kencana No. 1
Pamulang – Tangerang Selatan
Telp. 021 7412566
Fax. 021 74709855
Email: unpampress@unpam.ac.id

Cetakan pertama, 2020

Data Publikasi Unpam Press
| Lembaga Pengembangan Pendidikan dan Pembelajaran

Gedung A. R. 211 Kampus 1 Universitas Pamulang
Jalan Surya Kencana Nomor 1. Pamulang Barat, Tangerang Selatan, Banten.
Website: www.unpam.ac.id | **email**: unpampress@unpam.ac.id

Sistem Berkas/ Samsoni, Saprudin, Mochammad Bagoes Satria J – 1sted.
ISBN 978-623-7833-61-1

I. Sistem Berkas II. Samsoni. III. Saprudin. IV. Mochammad Bagoes Satria J.

Ketua Unpam Press : Pranoto
Koordinator Editorial dan Produksi: Ubaid Al Faruq, Ali Madinsyah
Koordinator Bidang Hak Cipta : Susanto
Koordinator Publikasi dan Dokumentasi : Aden
Desain Cover : Ubaid Al Faruq

Cetakan pertama, 2020

MODUL MATA KULIAH

SISTEM BERKAS

IDENTITAS MATA KULIAH

Program Studi	: S1. Teknik Informatika
Mata Kuliah/Kode	: Sistem Berkas / TPL19
Sks	2
Prasyarat	: --
Deskripsi Mata Kuliah	: Mata kuliah ini merupakan mata kuliah wajib program studi teknik informatika yang membahas tentang media penyimpanan file; dasar-dasar organisasi file : berkas primer, sekunder,langsung dengan banyak kunci & key, relatif dan index sekuensial; majemen kolasi,pengurutan rekaman, sort & merge, perangkat kontrol input dan output konsep umum organisasi dan arsitektur komputer, pengkodean, gerbang logika, memory, dan input/ output
Capaian Pembelajaran	: Setelah menyelesaikan mata kuliah ini mahasiswa mampu membuat struktur berkas pada file dan mengimplementasikan ke dalam program
Penyusun	: 1. Samsoni, S.Kom.,M.Kom 2. Saprudin, S.Kom.,M.Kom 3. Mochammad Bagoes Satria, S.Kom.,M.Kom

Ketua Program Studi	Ketua Tim Teaching
Dr. Ir. Sewaka, M.M	Samsoni, S.Kom., M.Kom
NIDN. 8842760018	NIDN. 0431127505

KATA PENGANTAR

Alhamdulillah puji dan syukur kami panjatkan kehadirat Allah subhanahu wa ta'ala yang telah melimpahkan rahmat dan karunia-Nya, sehingga kami dapat menyelesaikan penulisan modul pembelajaran sistem berkas.

Mata kuliah ini merupakan mata kuliah wajib program studi teknik informatika yang membahas Tentang media penyimpanan file; dasar dasar organisasi file :berkas primer, sekunder,langsung dengan banyak kunci & key,relatif dan index sekuensial; majemen kolasi,pengurutan rekaman, sort & marge,perangkat kontrol input dan output Konsep umum organisasi dan arsitektur komputer, pengkodean, gerbang logika, memory, input/ output.

Kami menyadari, penulisan modul pembelajaran Sistem Berkas ini masih memerlukan penyempurnaan karena keterbatasan pengetahuan dan pengalaman kami sebagai penulis. Oleh karena itu kritik dan saran yang membangun kami harapkan demi kesempurnaan penulisan modul pembelajaran sistem berkas ini.

Tangerang Selatan, 10 Oktober 2020

Penyusun

Samsoni, S.Kom.,M.Kom

NIDN : 0431127505

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

PERTEMUAN 1
KONSEP DASAR SISTEM BERKAS

A. Tujuan Pembelajaran

1. Memahami konsep dasar sistem berkas
2. Mengetahui metode akses pada sistem berkas
3. Memahami struktur direktori
4. Memahami struktur berkas

B. Uraian Materi

Data menginformasikan hal yang diperlukan untuk membangun struktur penyimpanan untuk pengumpulan data. Yang terpenting struktur penyimpanan ini wajib mempermudahkan dalam mengakses suatu data untuk kumpulan informasi yang meliputi pengorganisasian file dan sistem berkas.

1. Pengertian Sistem Berkas dan Akses

Adalah suatu program yang dimana untuk mengetahui proses menyimpan data dari suatu file dan pengorganisasian file itu sendiri. Sisitem akses merupakan proses pengambilan informasi dari dalam file.

Pengarsipan dan Akses yaitu :

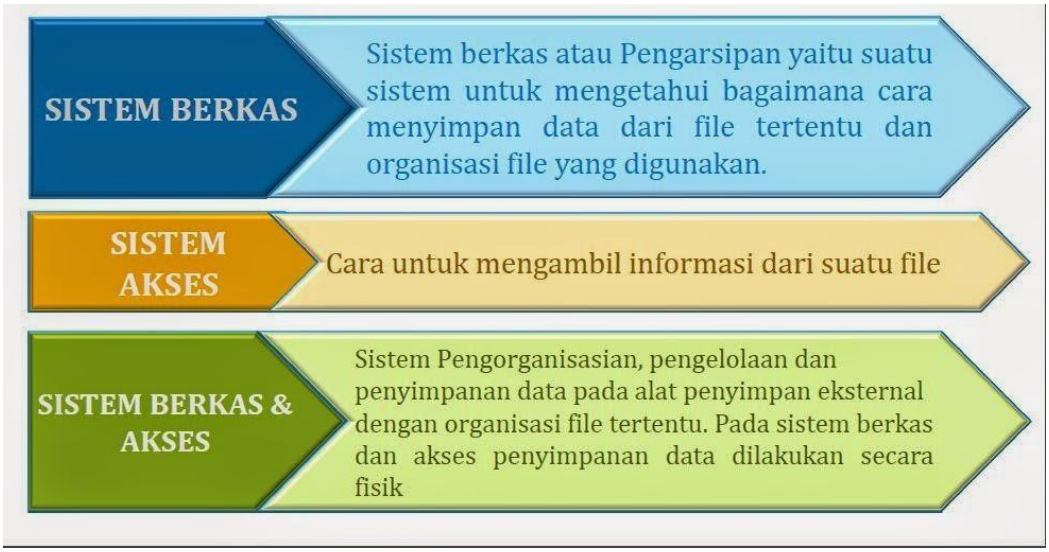
1) Proses pembentukan suatu file dan proses pelacakan record kembali

2) Sistem hubunngan, pemrosessan, dan penyimpanan informasi pada media penyimpanan eksternal menggunakan organisasi file tertentu.

3) Organisasi file adalah cara untuk menyimpan atau menggambarkan record pada suatu file

4) Secara lebih detail pengarsipan dan akses :

   a. Insert adalah menambahkan data baru ke dalam data lama
   b. Up-Date : mengubah data lama menjadi data yang telah di perbaharui.
   c. Organisasi : Membangun kembali record-record dari dalam suatu file.



Gambar 1. 1 Sistem Berkas

Istilah-istilah yang digunakan :

1) Elemen Data

Adalah berupa nilai yang hanya mempunyai satu nama atau karakter seperti tipe nomor, char dan kode atau panjang suatu karakter.

2) Item Data

Sejumlah nama dan himpunan karakter elemen data yang ada dalam suatu atribut.

Sebuah ruang penyimpanan atribut dari suatu entitas contohnya yaitu item data untuk mengetahui identitas mahasiswa dapat di karakteristik menjadi 4 digit dengan nilai 1-999.

3) Entitas

Sekelompok obyek yang teridentifikasikan yang memiliki karakteristik sama atau berbeda dengan yang lain. Contoh suatu obyek yaitu Manusia, lokasi, atau suatu peristiwa. Contoh entisitas motor, siswa, nilai uas, karyawan, dan lain-lain.

4) Attribut

Suatu yang menggambarkan suatu entitas. Semua atribut haruslah mencangkup untuk menyatakan suatu identitas obyek atau atribut wajib untuk dapat menggambarkan suatu keunikan sebuah obyek. Contohnya entisitas sepedah motor terdapat dari atribut No. Registrasi, No. Polisi, tahun pembuatan, bahan bakar, dan lain sebagainya.

5) Field

Tempat menyimpan suatu elemen data. Contohnya : ruang penyimpanan nomor mahasiswa.
Elemen yang memiliki suatu atribut khusus dan suatu data terkecil yang dapat dengan mudah di akses

6) Record (fisik)

Adalah tempat untuk menyimpan suatu rangkaian field yang di dalamnya terdapat elemen-elemen data untuk mengidentifikasi suatu entitas. Contohnya lokasi penyimpanan memerlukan isi dari nama, alamat, tempat tanggal lahir dari salah seorang mahasiswa.

7) File

Yaitu kumpulan record dari tipe tunggal yang didalamnya terdapat elemen-elemen data yang mengidentifikasikan entitas. Contohnya : file mahasiswa yang terdapat satu record untuk satu mahasiswa

8) Access Data

Adalah proses program yang sedang mengakses record-record yang terdapat dari suatu file.

2. Operasi Berkas

Ada 2 hal Untuk memilih organisasi berkas, yaitu :

1) Model operasi berkas

2) Model penggunaannya

Ada 2 proses untuk model pengaplikasiannya, yaitu :

1) Batch, adalah cara yang di lakukan secara dikelompokan

2) Iterative, adalah proses yang di lakukan secara satu persatu record

Model operasi berkas, ada 4 cara yaitu :

1) Creation

2) Update:Insert/ Add
   a. Modification
   b. Delete

3) Retrieval :
   a. Inquiry
   b. Report
   c. Generation

4) Maintenance meliputi:
   a. Restructure
   b. Reorganization

Creation

Ada 2 cara untuk membuat berkas :

1) Pertama merancang struktur berkasnya lalu menyatakan banyaknya record, dan kemudian load record-record tersebt ke dalam suatu berkas.

2) Merancang suatu record dengan cara merekam satu persatu recordnya

Update

Adalah mengubah isi di dalam suatu berkas, fungsinya sendiri yaitu agar berkas selalu diperbaharui.

Ada tiga bagian dalam proses Update yaitu :

1) penambahan record

2) Perbaikan record

3) Penghapusan record

Retrieval

Tujuan untuk mengakses suatu berkas agar kita memperoleh informasi dari berkas itu sendiri. Ada 2 syarat Retrival, yaitu:

1) Comprehensive Retrieval

   Adalah cara untuk memperoleh data dari semua record.

   Contohnya : List Nama, Alamat, dan Display All

2) Selective Retrieval

   Memperoleh informasi dari satu persatu record-record atas dasar syarat tertentu.

   Contohnya :

   List for Upah Karyawan = 80000

   List Nama,NIM for Tahun = 2019/2020

3. Maintenance

Maintenance adalah mengubah suatu berkas yang bertujuan untuk memperbaiki program agar berkas mudah di akses, terdapat 2 cara maintenance:

1) Restructuring

   Mengubah struktur berkas. Misalnya :
   a. Mengubah panjang field
   b. mengubah panjang record
   c. menambahkan field baru

   Perubahan yang di lakukan tidak akan berpengaruh terhadap berkas

2) Reorganisasi

   Mengubah organisasi berkas. Misalnya :
   a. sebelumnya organisasi berkas sequential di ubah ke berkas sequential diindeks.
   b. sebelumnya organisasi berkas langsung diubah ke organisasi berkas sequential (berurutan)

Kriteria teknik di dalam pengarsipan dan akses :

Syarat teknis dalam menyimpan data yang besar adalah :

1) Waktu yang di butuhkan dalam pengaksesan data

2) Kemudahan dalam pembaruan data

3) Dapat dengan mudah di reorganisasi data

4) Menggunakan penyimpanan sekecil mungkin

Adapun kriteria non-teknik :

1) Hemat biaya yang berhubungan dengan perancangan

2) Pemeliharaan atau peremajaan yang mudah

3) Dapat diandalkan

4) Keamanan perusahaan

Representasi Data Logik dan Fisik

Ada 2 aspek data yaitu data logik dan data fisik.

1) Representasi data Logik

   Adalah suatu obyek contonya atribut, integer, nilai, dan operasi yang dikerjakannya. Contoh integer khusu yaitu sekumpulan bilangan yang di dalamnya terdapat tanda, atribut, ukuran dan lain sebagainya.

2) Representasi Data Fisik

Mengevaluasi proses data dan direpresentasikan atau dimuat dalam alat penyimpanan. Perubahan logika ke fisik di gambarkan sebuah penyimpanan tipe data ke media penyimpanan. Contohnya tipe data integer memiliki satu lokasi memori.

0 1 2 3 4.....19

representasi data fisik integer yang berada di lokasi memori 16 bit

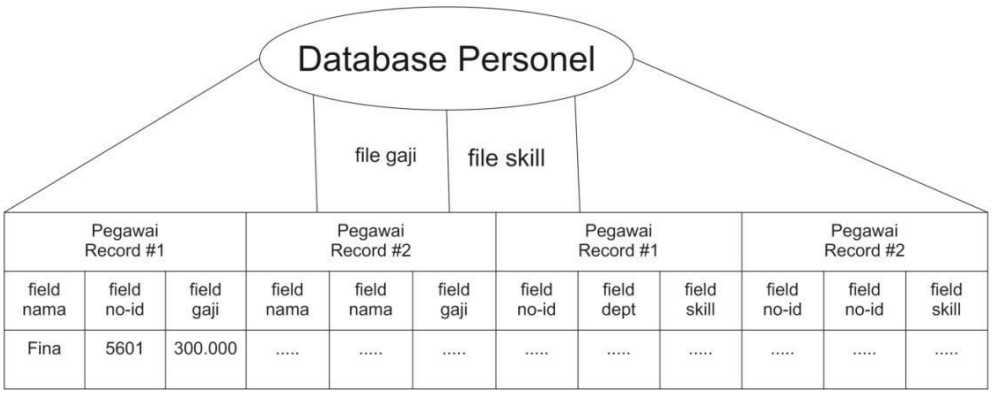
Representasi File

Data yang tersimpan di fungsikan secara berbeda-beda dan untuk tujuan yang berbeda-beda pula. Menggambarkan data secara logik dan fisik di representasikan pada level terendah.

Ruang lingkup file yang digunakan kumpulan yang lebih besar dari rangkaian informasi yang saling berkaitan.

Organisasi Data dan Pengolahan File

Elemen-elemen data umum :



Gambar 1. 2 Database Personel

Karakter

Karakter adalah elemen terkecil di bandingkan dengan elemen-elemen lain yang berada di dalam data komputer. Alphabet, simbol, dan numerik merupakan elemen dasar suatu karakter

Field

Tingkatan data yang lebih tinggi selanjutnya ialah field yang di dalamnya terdapat sekelompok karakter. Field juga bisa dikatakan suatu item. Field yaitu data yang menunjukan atribut dari sekumpulan entitas. Contohnya manusia, tempat, atau peristiwa.

Contoh : nama jalan yaitu field data yang menunjukan suatu atribut berupa tempat atau lokasi. Hasil field nama jalan adalah sekelompok karakter alphabet dari nama tempat. Hasil field jumlah barang juga merupakan kumpulan karakter numerik dari jumlah barang.

Record

Penggabungan field-field data yang menjadi satu record, sebuah record menerangkan kumpulan atribut yang menggambarkan sebuah entitas, Contohnya seperti record daftar upah karyawan untuk satu orang di dalamnya berisi field-field NIP dan rata-rata upah

Ada 2 ukuran sebuah record yaitu, record dengan panjangnya tidak berubah atau disebut fixed length dan record dengan panjang yang berubah-ubah atau variabel length yang di dalamnya berisi beberapa field yang tidak tetap. Record adalah kelompok field yang saling berhubungan.

File

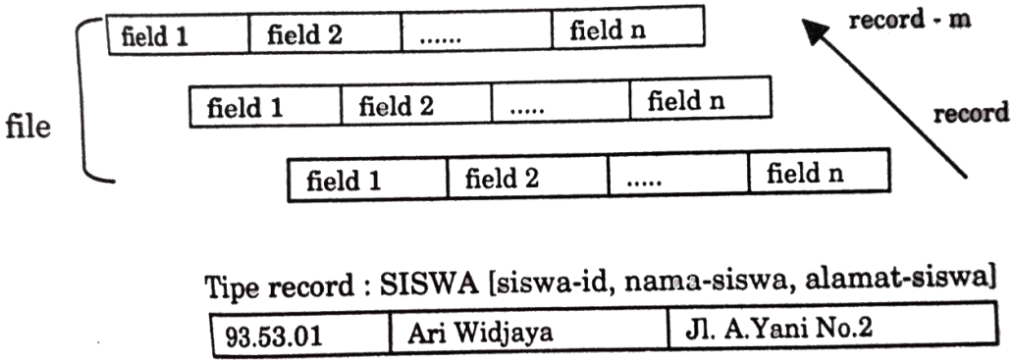
File adalah sekelompok record yang saling berkaitan atau bisa disebut data file (data set). Contohnya seperti file siswa yang di dalamnya terdapat daftar no induk siswa untuk semua siswa di sebuah sekolah. File-file juga sering dianggap sebagai proses pengolahan data contohnya seperti file daftar upah karyawan.

Hubungan antara field, record dan file

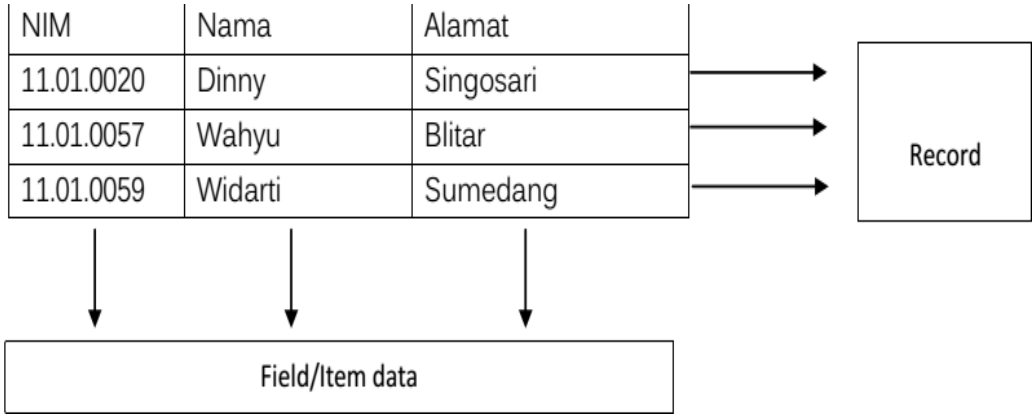
Sebuah record adalah sekelompok elemen yang ber macam-macam dan mungkin direpresentasikan sebgai satu unit. Suatu record dibedakan oleh suatu array yaitu dimana semua elemen mempunyai struktur yang sama. Akan tetapi kumpulan elem yang berada dalam sebuah record mempunyai struktur data yang berbeda-beda.

Elemen-elemen record ialah field. Satu field digambarkan sebagai ruang sebuah record yang di fungsikan sebagai bermacam-macam informasi.

File adalah sekumpulan record yang direlasikan secara logik (dimana record adalah suatu struktur yang secara logika penghubung kan field-field atau elemen-elemen dari informasi).



Gambar 1. 3 Contoh Tabel Siswa



Gambar 1. 4 Contoh File Siswa

4. Sistem Berkas (File Systems)

Akses menulis dan membaca atau read dan write data di dalam file membutuhkan proses yang transparan untuk pembuat aplikasi.

Sistem berkas menyaikan dukungan yang memungkinkan pembuat aplikasi dapat mengakses file tanpa harus menyangkut rincian karakteristik penyimpanan dan peralatan pewaktu. Sistem berkas mengubah kebenaran akses file secara relatif menjadi input atau output level rendah.

Ada banyak tanggung jawab sebuah sistem berkas, yaitu :

1) Permajaan suatu file yang menggambarkan atau melokasikan informasi.
2) Menetapkan alur lintas data antara memori utama dan penyimpanan eksternal
3) Mengatur komunikasi antara CPU dan media penyimpanan dan begitupun sebaliknya

Inti dari bahasan meliputi:

1) Menentukan terlebih dahulu informasi yang diperlukan dalam membuat suatu file.

2) Buatlah informasi menjadi sebuah kode untuk menghemat ruang penyimpanan.

3) Simpan informasi yang akan digunakan sekarang atau dalam waktu yang dekat

4) Penggunaan file berpengaruh terhadap susunan secara berurutan atau secara acak

5) Pengaksessan data di dalam sebuah file harus di batasi, bertujuan agar meningkatkan keamanan suatu file.

5. Jenis-jenis File

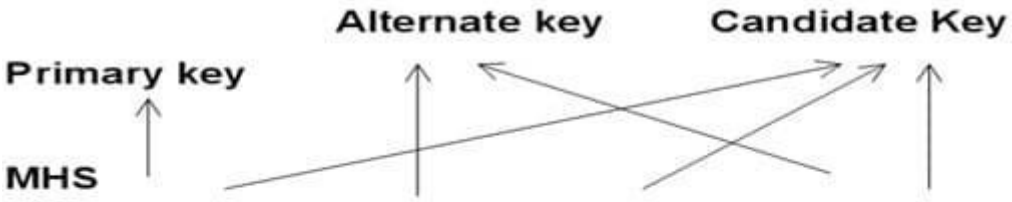
File data dapat digolongkan menurut cara organisasinya se in atau file acak. Dapat juga menurut jenisnya misal file master, file transaksi dll.

File data yang digolongkan menurut jenisnya adalah sebagai berikut :

1) File Induk

Suatu file yang diperlukan untuk memperlancar operasi sistem dan diperbaharui secara teratur. File induk yaitu file yang paling penting dalam sebuah sistem, suatu file induk juga merupakan file yang dipergunakan untuk mengerjakan tugas-tugas utama dalam sebuah sistem dan di perbaharui secara berkala.

Contoh : Sistem sekolah memerlukan file induk tentang daftar siswa, daftar mata pelajaran, dan file lain sebagainya.



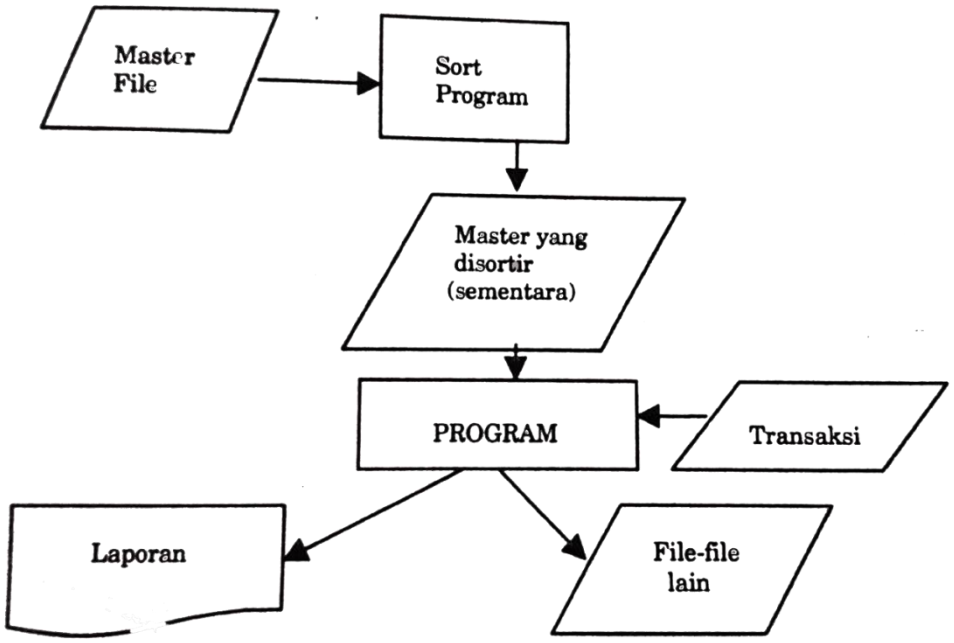
NPM	Nama	Alamat
10296832	Nurhayati	Jakarta
10296126	Astuti	Jakarta
31296500	Budi	Depok
41296525	Prananingrum	Bogor
50096487	Pipit	Bekasi
21196353	Quraish	Bogor

Gambar 1. 5 Contoh File Sistem Sekolah

2) File Transaksi

Fungsi file transaksi adalah agar memperbaharui atau meremajakan file induk dengan data yang diterbarukan.

Contoh : rekaman mengenai pembayaran pajak kendaraan bermotor akan menghasilkan file transaksi, dan file tersebut akan di pergunakan untuk memperbaharui record sebelumnya.



Gambar 1. 6 Contoh File Transaksi

3) File Penunjang

Adalah salinan dari suatu file yang berfungsi untuk menjaga kemungkinan adanya kerusakan dalam file yang asli.

4) File Riwayat hidup

Adalah data yang diperoleh dari waktu-waktu sebelumnya, dan berguna untuk menyusun suatu laporan.

5) File Data Transaksi

Adalah suatu rekaman pada pita atau piringan dari setiap transaksi yang digunakan untuk melengkapi file induk.

6) File Kesalahan

Rekaman mengenai kesalahan yang di rekam pada file. Dalam proses file transaksi untuk melengkapi file induk mungkin akan mengalami kesalahan yang lolos dari proses sebelumnya, maka fungsi dari file kesalahan yaitu sebagai penunjang perbaikan file di kemudian hari.

Menghentikan program setiap kali terjadi kesalahan dan membetulkan kesalahan di nilai tidak menguntungkan sehingga setiap kesalahan di rekam pada file kesalahan.

7) File Laporan (Pencetak)

Merupakan turunan laporan tercetak yang ditahan pada piringan atau pita menunggu printer siap mencetak.

8) File Sementara

File yang disiapkan untuk pemrosesan peralihan.

9) File Pustaka (Library)

Adalah suatu file yang digunakan untuk menyimpan program seperti program utility dan lain-lain.

10) File Kerja (Work)

File kerja terdapat record-record yang di rancang sedemikian rupa hingga menghasilkan sebuah program dan berfungsi sebagai input program lain. Dan file kerja biasa di gunakan dalam proses sortir.

11) File Program

File program terdapat tugas-tugas untuk menjalankan suatu data. Perintah-perintah atau tugas file program biasanya di tulis menggunakan bahasa pemrogaman tingkat tinggi seperti Pascal, BASIC, bahas mesin dan lain-lain.

C. Soal Latihan

1. Apa pengertian dari Sistem Berkas?
2. Sebutkan jenis – jenis file?
3. Apa perbedaan Entitas dan Atribut?

D. Referensi

Handayani, Dewi. 2001. Sistem Berkas. Yogyakarta. J&J Learning

Alan l, Tharp-john Willey &Son. 1988. File Organization and Processing

Hariyanto, Bambang. 2007. Sistem Pengarsipan Dan Metode Akses. Informatika

Bach, Maurice. 2016. The Design of the UNIX Operating System Maurice Bach

Coronel, Carlos. 2010. Database Systems: Design, Implementation, and Management.

PERTEMUAN 2
MEDIA PENYIMPANAN FILE

A. Tujuan Pembelajaran

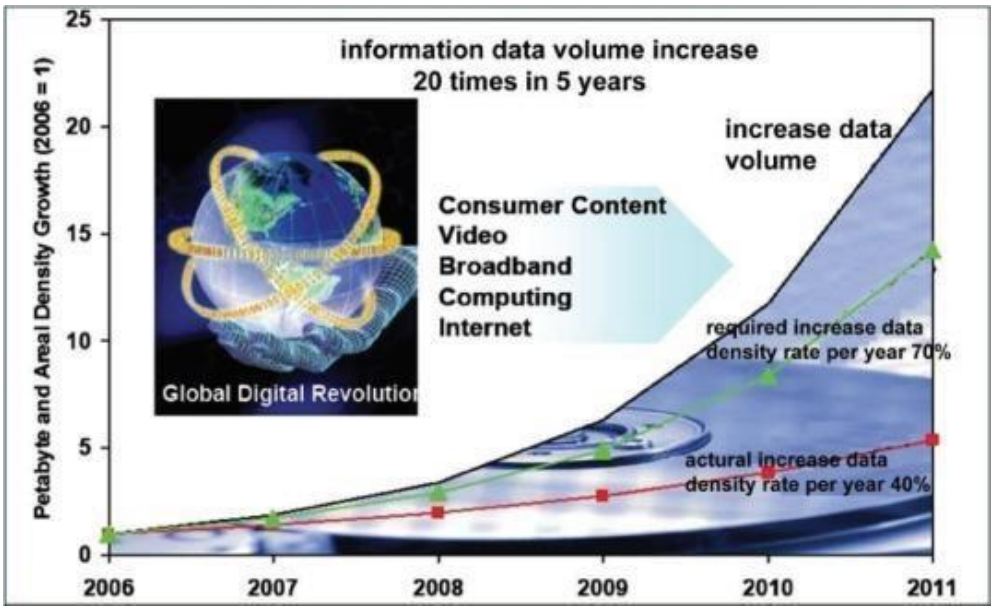
1. Setelah mempelajari materi ini, mahasiswa mampu menjelaskan jenis-jenis media penyimpanan. Mahasiswa mampu melakukan perhitungan untuk organisasi berkas pada magnetic tape.

B. Uraian Materi

1. Media Penyimpanan

Data Storage (penyimpanan data) merupakan perangkat keras (*hardware*) yang dapat digunakan sebagai tempat penyimpanan data pada sebuah *hardware* komputer sehingga bisa dibuka dan dibaca kembali untuk dilakukan proses ulang.

Penyimpanan data adalah salah satu inti dari teknologi informasi (TI) rantai dari produksi data, transfer, penyimpanan, berbagi file, dan akhirnya ke pengolahan data. Untuk mempromosikan dan menemani pengembangan TI, kinerja penyimpanan data informasi harus selalu ditingkatkan. Sebagaimana yang ditunjukkan oleh gambar berikut ini.



Gambar 2. 1 Trend Perkembangan Kapasitas Data

Pada gambar diatas, bisa kita lihat kepadatan data penyimpanan yang diperlukan terus meningkat hingga 70% per tahun.

Data storage (penyimpanan data) atau *memory* (memori) ini bisa dibagi oleh 2 poin berikut :

1) Memori Primer : penyimpanan utama atau penyimpanan internal.

2) Memori Sekunder : penyimpanan sekunder atau penyimpanan eksternal.

2. *Primary Memory* (Memori Primer) atau *Primary Storage* (Penyimpanan Utama)

Primary memory atau memori primer adalah memori yang langsung diakses oleh CPU untuk menyimpan dan mengambil informasi maupun data. Memori primer juga disebut sebagai RAM (*Random Access Memory*). Ini adalah memori yang memiliki sifat *volatile*, yang kehilangan datanya saat power dimatikan. Memori primer dapat diakses langsung oleh CPU melalui alamat dan bus memori dan terus diakses oleh CPU untuk mendapatkan data dan instruksi.

Selanjutnya, komputer berisi ROM (Read Only Memory), yang menampung perintah yang paling banyak dieksekusi seperti program startup (BIOS). Ini adalah memori non volatile yang menyimpan datanya saat power dimatikan. Karena memori utama sering diakses, maka perlu dilakukan lebih cepat. Tapi ukurannya lebih kecil dan harganya mahal.

Diatas disebutkan mengenai sifat memori volatile dan non volatile, dengan sifat tersebut kita bisa mengenal kondisi, dimana hilang atau tidaknya sebuah *file* data (berkas data) maupun *file* program (berkas program) di dalam *storage* (penyimpanan). Volatile, merupakan kondisi dimana file data atau program akan hilang ketika arus listrik terputus atau padam, sedangkan *non volatile* merupakan kondisi dimana file data atau program tidak hilang jika arus listrik terputus atau padam.

Ada 4 bagian pada *primary memory* (memori utama) *atau primary storage* (penyimpanan utama), yaitu :

1) *Input Storage Area* (Area Penyimpanan Input)

   Memiliki fungsi menyimpan data yang akan dibaca.

2) *Program Storage Area* (Area Penyimpanan Program)

   Berfungsi menyimpan perintah proses.

3) *Working Storage Area* (Area Penampungan Kerja)

   Berfungsi sebagai tempat dilakukannya pemrosesan data.

4) *Output Storage Area* (Area Penyimpanan Output)

   Berfungsi sebagai tempat penyimpanan informasi dari hasil pengolahan data untuk sementara, sebelum nantinya dikeluarkan melalui perangkat output.

Primary memory (memori utama) pada komputer terdiri dari 2 bagian, yang pertama adalah RAM (*Random Access Memory*) kemudaian ROM (*Read Only Memory*).

RAM (*Random Access Memory*)

Gambar 2. 2 RAM (*Random Access Memory*)

RAM adalah akronim *random access memory* yang merupakan tempat dimana data disimpan sementara disaat komputer sedang bekerja dan bisa diakses secara acak.

Peran RAM cukup penting karena fungsinya sangat diperlukan untuk mengolah data yang masuk ke dalam *memory* (memori). RAM ini sendiri bersifat volatile, yang artinya ketika listrik padam maka file data atau program akan hilang jika listrik terputus atau padam.

Jenis-Jenis *Random Access Memory* (RAM)

DRAM (*Dynamic Random Access Memory*)

Gambar 2. 3 DRAM (*Dynamic Random Access Memory*)

DRAM adalah sebuah memori semi konduktor yang membutuhkan kapasitor sebagai sarana untuk memproses data. DRAM dalam strukturnya menggunakan 1 transistor dan kapasitor. Hal tersebut membuat DRAM mempunyai kepadatan yang baik. DRAM sendiri mempunyai gelombang kerja di antara 4,7 Megahertz sampai 40 Megahertz.

SDRAM (*Sychronous Dynamic Random Access Memory*)

Gambar 2. 4 SDRAM (Sychronous Dynamic Random Access Memory

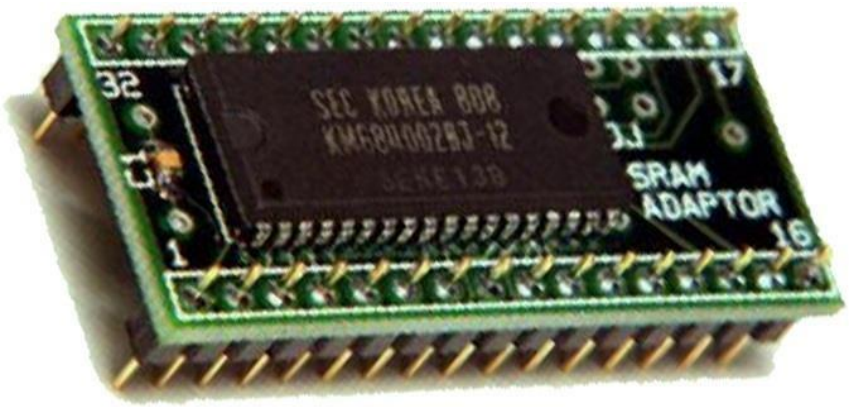
SDRAM adalah sebuah RAM yang mempunyai kecepatan yang lebih tinggi jika dibandingkan dengan RAM yang lain, kecepatannya berkisar antara 100 Megahertz - 133 Megahertz. Dari fisiknya sendiri terlihat memiliki dua slot dibagian bawah. RAM ini nantinya dimasukan ke bagian slot di *mainboard* komputer yang mampu memuat memori dari 16 MB sampai 1GB.

RDRAM (*Rambus Dynamic Random Access Memory*)

Gambar 2. 5 RDRAM (*Rambus Dynamic Random Access Memory*)

Tahun 1995 RDRAM pertama kali diperkenalkan, RDRAM memiliki kecepatan 600 Mbytes/sec. Pada 1996, kecepatannya naik 700 MBps, dan pada 1997 naik 1,6 GBps. RDRAM mulanya dibuat untuk komputer dengan prosesor pentium 4.

SRAM (*Static Random Access Memory*)



Gambar 2. 6 SRAM (*Static Random Access Memory*)

SRAM dibuat dari semi konduktor yang artinya tidak membutuhkan kapasitor dan karena komponen ini hanya menggunakan transistor tanpa kapasitor, kinerjanya menjadi lebih cepat. SRAM memiliki kecepatan yang sanggup menyaingi kecepatan *processor* 500 Megahertz.

EDORAM (*Extended Data Out Random Access Memory*)

Gambar 2. 7 EDORAM (*Extended Data Out Random Access Memory*)

EDORAM diperkenalkan 1995 dan mempunyai kecepatan yang lebih tinggi untuk membaca dan mengirim file data dibading RAM lainnya. EDORAM memiliki slot sebanyak 72 pin, sangat cocok digunakan oleh seluruh komputer pentium.

FPM DRAM (*First Page Mode DRAM*)

Gambar 2. 8 FPM DRAM (*First Page Mode* DRAM)

FPM DRAM merupakan awal dari DRAM. Kecepatan maksimal untuk mengirim cache L2 hampir 176 MBps. FPM berjalan di gelombang16 MHz sampai 66 MHz.

Flash RAM (*Random Access Memory*)

Gambar 2. 9 *Flash* RAM

Flash RAM (*Random Access Memory*) adalah memori yang kapasitasnya rendah, umumnya dipakai untuk alat elektronik misalnya, DVD, Handycam, dan sejenisnya.

VGRAM (*Video Graphic Random Access Memory*) / VGA *Card*

Gambar 2. 10 VGRAM

VGRAM atau yang biasa disebut VGA Card diperuntukan sebagai penyimpanan pixel untuk paparan gambar atau grafik. Penggunaan VGRAM dapat memberikan tampilan gambar maupun video cukup baik dan meringankan beban processor.

DDR SDRAM (*Double Date Rate SDRAM*)

Gambar 2. 11 DDR SDRAM

DDR SDRAM merupakan RAM yang mempunyai kecepatan cukup baik dibandingkan jenis lainnya. DDR SDRAM ini dapat mengeksekusi dua instruksi secara bersamaan. DDR SDRAM ini memiliki slot 184 pin dengan konsumsi listrik yang cukup kecil. Kapasitas memori ini cukup besar sampai 4 GB.

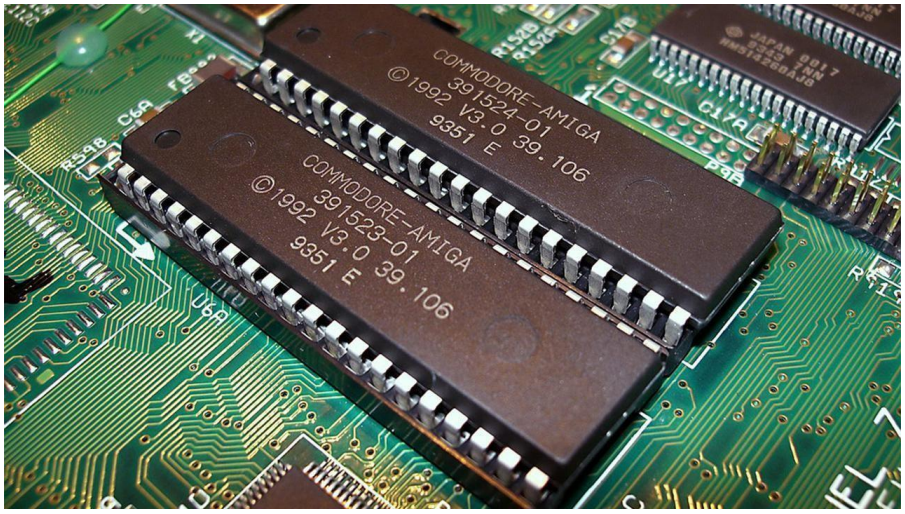
Jenis RAM lainnya merupakan bentuk pengembangan dari DDR SDRAM, misalnya DDR 2, DDR3, dan seterusnya. RAM lainnya ini dipakai pada banyak komputer sekarang, sebab cukup hemat daya, cukup optimal, dan mempunyai kecepatan yang cukup tinggi.

SO-DIMM (*Small Outline Dual in-line Memory Module*)

Gambar 2. 12 SO-DIMM

SO-DIMM adalah jenis memori yang dipakai pada laptop ataupun notebook. Secara fisik bentuknya cukup kecil, yaitu lebih kecil hingga setengahnya ukuran RAM, jadi dapat lebih menghemat ruang. Untuk kapasitasnya sendiri mengikuti RAM yang digunakan pada komputer.

ROM (Read Only Memory)



Gambar 2. 13 ROM (*Read Only Memory*)

ROM merupakan jenis memory yang hanya dapat dibaca. Memori diisi sebuah data ataupun program yang sudah dibuat oleh pabrik produsen. Memori ini umumnya telah diisi sebuah data ataupun program dari produsen dengan kegunaan khusus.

ROM biasanya dipakai menyimpan firmware (program yang berkaitan dengan perangkat keras). Contohnya ROM BIOS yang memiliki isi program dasar komputer.

Jenis-jenis ROM (Read Only Memory)

MASK PROM

Gambar 2. 14 MASK ROM

ROM memiliki data yang diisi melalui mask pada saat perakitan chip. ROM tidak dapat ditulis ulang atau non flashable sehingga ROM jenis ini tidak dapat ditulis ulang.

PROM (Programmable Read Only Memory)

Gambar 2. 15 PROM

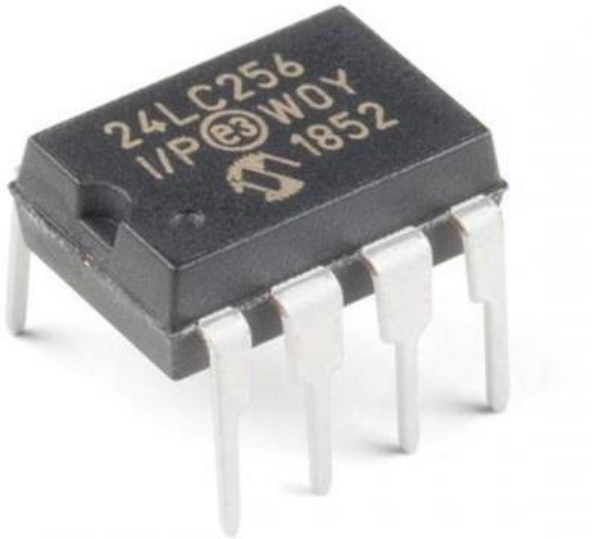
Memori jenis ini hanya bisa ditulis sebuah program, dimana program tersebut dapat diisi oleh pengguna dan akan tersimpan di dalam memori selamanya.

EPROM (*Erasable Programmable Read Only Memory*)

Gambar 2. 16 EPROM

Memori ini bisa diisi sebuah data ataupun program oleh pengguna, dimana data ataupun program tersebut dapat di edit, dan dihapus nantinya.

EEPROM (*Electrically Erasable Progammable Read Only Memory*)



Gambar 2. 17 EEPROM

Memori jenis ini bisa diisi sebuah data ataupun program oleh pengguna, dimana data ataupun program tersebut dapat di edit, dan dihapus nantinya, tanpa perlu melepas atau memindahkan chip EEPROM dari *mainboard*.

3. Secondary Memory atau Secondary Storage

Secondary memory (memori sekunder) adalah perangkat penyimpanan yang tidak dapat diakses langsung oleh CPU dan digunakan sebagai perangkat penyimpanan permanen yang menyimpan data bahkan setelah power dimatikan. CPU mengakses perangkat ini melalui saluran input / output dan data pertama-tama ditransfer ke memori utama dari memori sekunder sebelum mengaksesnya. Umumnya, hard disk drive dan perangkat penyimpanan optik (CD, DVD) digunakan sebagai perangkat penyimpanan sekunder di komputer modern.

Pada perangkat penyimpanan sekunder, data diatur ke file dan direktori sesuai dengan sistem file. Ini juga memungkinkan untuk mengaitkan informasi tambahan dengan data seperti hak akses, pemilik, waktu akses terakhir, dll. Selanjutnya, saat memori utama terisi, memori sekunder digunakan sebagai penyimpanan sementara untuk menyimpan data yang paling jarang digunakan di memori utama. . Perangkat memori sekunder lebih murah dan ukurannya lebih besar.

Terdapat 2 jenis *secondary memory* atau *secondary storage*, yaitu :

1) Yang pertama adala serial atau *Sequential Access Storage Device* akses sekuensial perangkat penyimpanan (SASD)

   Contohnya seperti, pita magnetik, *paper tape*, dan kartu berlubang.

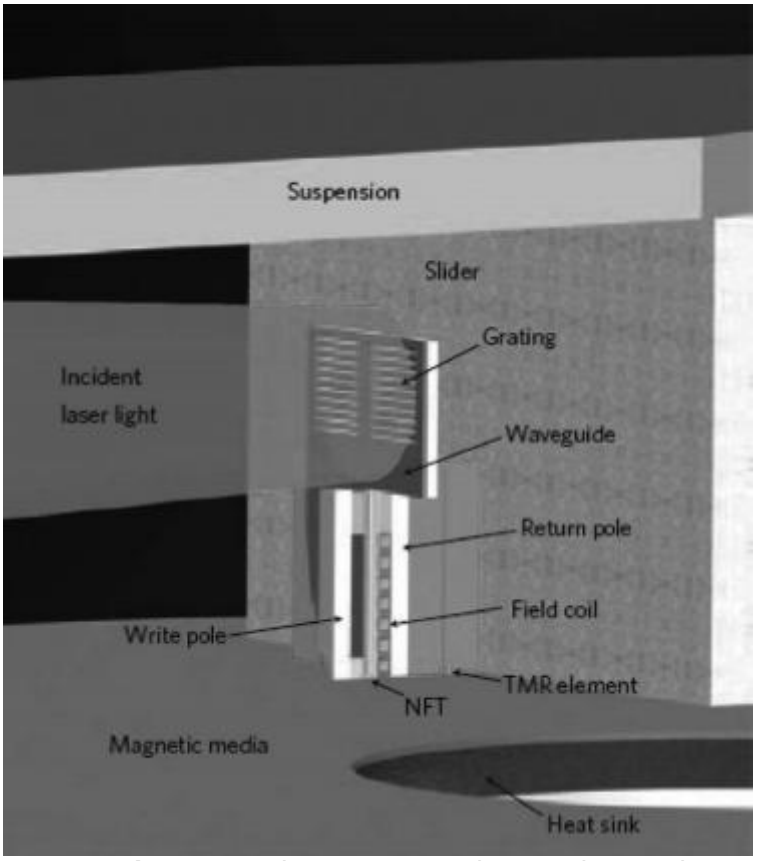
2) Kemudian yang kedua adalah *Direct Access Sotrage Device* akses langsung perangkat penyimpanan (DASD)

   Contohnya seperti, piringan magnetik, disket, dan penyimpanan masal.

Magnetic Data Storage

Magentic data storage adalah alat penyimpanan yang populer. Untuk peningkatan kepadatan (density) penyimpanan ketika ukuran domain dikurangi menjadi kurang dari 100nm, domain magnetik menjadi tidak stabil, dan masalah tersebut yang dikenal sebagai super-paragmanetic harus diselesaikan. Setelah upaya beberapa tahun efek paragmanetic tersebut dapat diatasi dengan pendekatan yang berbeda, solusinya adalah sebagai berikut: *perpendicularity magnetic recording* (PMR), *laser* atau *heat-assisted magnetic recording* (LAMR atau HAMR), dan *patterned media magnetic recording* (PMMR). Ukuran wilayah magnetik, dan ukuran kepala magnetik semua dalam skala nano dan ditentukan oleh nanolithograph.

Dalam perangkat penyimpanan Dalam perangkat penyimpanan data magnetik, seperti drive hard disk (HDD), ukuran bit dari perekaman magnetik dapat turun ke 40 NM, kepadatan areal untuk 630 Gbit/IN2. Langkah selanjutnya hingga 20 NM, kepadatan areal hingga 1 Tbit/IN2 dapat diwujudkan.



Gambar 2. 18 Skema HDD Terintegrasi Termal

Magnetic Tape

Magnetic Tape (pita magnetik) merupakan contoh yang pertama dari memori sekunder. Pita ini digunakan sebagai perangkat masukan dan keluaran (input/output) dimana data dikirim dari pita ke prosesor, dan data diambil dari prosesor kemudian disimpan pada pita yang lainnya.

Pita magnetik biasanya memiliki panjang berukuan 731 meter, lebarnya 2.54 cm, dan memiliki tebal 2 mm. Pada pita magnetik terdapat bercak kecil yang berfungsi untuk menyimpan data, bercak tersebut secara kasat mata tidak terlihat pada bahan plastik ferroksida. Bahan plastik ini disebut sebagai myfar, dan memiliki cara akses tersendiri yaitu *tape drive*.

Banyaknya data yang bisa ditampung bergantung terhadap jenis pita yang dipakai. Pita dengan panjang 731 meter, umumnya bisa menampung data sebanyakan 23.000.000 character (karakter). Untuk penyimpanannya sendiri pada pita jenis ini dengan cara sekuensial.

Gambar 2. 19 Bentuk *Magnetic Tape* (Pita Magnetik)

Representasion (Representasi) *Data* dan *Density* (Kepadatan) pada *Magnetic Tape* (Pita Magnetik)

Pada media pita yang berbentuk bercak magnetik pada lapisan bahan plastik ferroksida, data akan direkam secara digit. 1 bit dinyatakan sebagai magnetisasi positif, dan sebaliknya 0 bit dinyatakan sebagai magnetisasi negatif atau sebaliknya. Pita sendiri terdiri dari sejumlah track, yaitu 9. Untuk kemudia digunakan sesuai fungsinya, 8 track untuk merekam data dan track terakhir dipakai sebagai pengoreksi kesalahan.

Density (kepadatan) merupakan salah satu keistimewaan yang dibutuhkan pada pita, dimana nantinya data akan disimpan. Kepadatan juga merupakan fungsi pita media dan drive dipakai sebagai perekam data ke media Bytes (bpi ekivalen karakter per inci) merupakan satuan yang digunakan pada density (kepadatan).

Paritas dan kontrol kesalahan pada pita magnetik

Cek paritas (parity check) merupakan salah satu cara melakukan pemeriksaan kesalahan pada pita magnetik. Pengecekan paritas sendiri ada 2, yaitu :

1) Paritas Ganjil (Odd Parity)

   Dalam menggunakan teknik paritas ganjil, maka jumlah yang mempresentasikan karakter akan berbentuk ganjil yaitu 1. Jika sudah bernilai 1 maka nilai pada track ke 9 bernilai 0 bit. Namun bila jumlah nilai 1 bit berbentuk genap, maka nilai dari paritas adalah 1.

2) Paritas Genap (Even Parity)

Dalam menggunakan teknik paritas genap, maka jumlah bernilai 1 yang mempresentasikan karakter akan berbentuk genap, dan jika sudah berbentuk genap, maka nilai paritas yang ada di trekk 9 bernilai 0, namun bila nilai 1 berbentuk ganjil, paritasnya 1.

Block system (sistem blok) pada _magnetic tape_ (pita magnetik)

Apa yang disebut sebagai blok adalah sebuah data yang dibaca dari pita, ataupun yang diinput ke dalam pita di suatu grup karakter. Blok merupakan jumlah paling kecil dari sebuah data yang bisa dikirim diantara memori sekunder dan memori primer pada saat diakses. Dari satu atau lebih catatan (*record*) yang terdiri dapat disebut sebagai suatu blok. Catatan fisik (*phsyical record*) merupakan sebuah blok juga.

Terdapat sebuah jarak (*gap*) diantara 2 buah blok, yang memiliki panjang untuk tiap jaraknya 1.524 cm. Banyaknya data maupun catatan yang tersimpan dalam pita dipengaruhi juga oleh ukuran blok.

Mekanisme pengaksesan pada pita magnetik

Sekuensial merupakan bentuk dari cara pengaksesan data pada pita magnetik untuk pembacaan maupun penulisan data. Yang mana berarti perlu menghapus data yang ada di depan lebih dahulu untuk bisa mendapatkan tempat bagi suatu data. Jadi bisa dibilang organisasi berkas dalam pita terbentuk secara skuensial, dan juga mekanisme pengaksesannya secara sekuensial.

Kelebihan menggunakan pita magnetik, antara lain :

1) Kapasitas panjangnya suatu cartatan yang tak terbatas
2) Kepadatan yang cukup tinggi
3) Ruang penyimpanan data yang besar, juga harga yang terjangkau
4) Memiliki kecepatan yang tinggi untuk mengirim data
5) Untuk proses catatan dari sebuah file pita secara keseluruhan sangat efisien.

Kekurangan penggunaan pita magnetik, antara lain :

1) Kecepatan akses yang lambat terhadap catatan
2) *Environment issue* (masalaha terhadap lingkungan)
3) Memerlukan proses yang harus sekuensial.

Piringan Magnetik (Magnetic Disk)

DASD awal yang diciptakan oleh industri komuter adalah piringan magnetik. Kecepatan yang dimiliki piringan magnetik umumnya sangat tinggi, Pada permukaan piringan data akan diambil maupun disimpan, baik untuk akses RAM dengan read mauapun write head yang memiliki posisi diantara piringan magnetik tersebut.

Karakter fisik piringan magnetik

Pada piringan magnetik terdapat jenis alat penyimpanan yang disebut paket piringan (*disk pack*), dimana terdiri dari beberapa lapisan piringan berbahan alumunium. Umumnya untuk setiap pack terdiri dari 11 lapisan piringan, dimana setiap piringan memiliki ukuran keliling sebesar 45.72 cm, dan untuk piringan kecil sebesar 20.32 cm.

Secara fisik piringan magnetik ini memiliki lapisan metal oxide filme yang mengandung magnet seperti pada pita magnetik. Keistimewaan penyimpanan pada lapisan permukaan, ruang kapasitas piringan dan cara pengaksesan ditunjukan dari banyaknya jumlah track pada piringan.

Terdapat 200 hingga 800 track untuk setiap permukaan yang dimiliki oleh piringan. Terdapat 20 permukaan yang digunakan untuk menyimpan data pada paket piringan yang terdiri dari 11 piringan. Pada setiap piringan kedua sisinya digunakan sebaga tempat penyimpanan data, dan untuk piringan paling atas dan juga bawah tidak dipakai karena umumnya pada piringan tersebut lebih rentan terkena kotoran dibandingkan permukaan piringan yang lain.

Terdapat sebuah pengontrol (*controller*), lengan akses (*access arm*), baca (*read*), dan tulis (*write*), juga mekansime untuk rotasi pada paket yang tersusun pada piringan yang digunakan untuk pengaksesan. Kemudian terdapat istilah tidak dapat dilepas (*non removable*) digunakan pada piringan yang pada saat dibuat sudah diisi paket piringan didalamnya sehingga tidak dapat dipindahkan, lalu ada juga bisa istilah dilepas (*removable*) kebalikan dari *non removable*.

Pada pengalamatan catatan, perubahan kode yang terjadi ditangani oleh pengontrol piringan, juga pada pemilihan *drive*. Kontrol aktivitas *read*/*write*, deteksi, dan koreksi kesalahan ditangani oleh penyimpanan penyangga (*buffer storage*), yang diatur oleh pengontrol (*controller*). Kecepatan putaran pada susunan paket piringan yaitu 3600 RPM (*rotation per minute*) yang terus menerus berputar.

Contoh data dan pengalamatan pada piringan magnetik

Layaknya data pada pita magnetik, data yang terdapat pada piringan juga di blok. Banyaknya jumlah data yang diakses pada sebuah perangkat penyimpanan (*storage device*) disebut sebagai pemanggilan blok. Agar data dapat diakses oleh

sebuah program komputer, maka data dari piringan perlu dipindahkan terlebih dahulu ke penyangga (*buffer*) pada penyimpanan utama.

Catatan tidak selalu diakses dengan cara sekuensial, hal ini ditunjukan pada kemampuan piringan untuk mengakses catatan secara langsung (*direct*).

Untuk pengalamatan data yang tersimpan pada piringan, terdapat 2 teknik dasar, antara lain:

1) *Cylinder Method* (Metode Silinder)

   Metode ini menggunakan nomor silinder, permukaan dan catatan sebagai acuan pengalamatannya.

2) *Sector Method* (Metode Sektor)

   Metode ini menggunakan nomor sektor, trek (*track*), dan permukaan sebagai acuan pengalamatannya.

Pergerakan akses kepala piringan (*movable head disk access*)

Untuk setiap catatan yang ada, pergerakan akses kepala piringan memiliki *read*/*write* untuk tiap permukaan piringan penyimpanan. Sekumpulan posisi akses lengan yang digunakan merupak sistem mekanik, jadi untuk kepala *read*/*write* pengalamatan permukaan akan menunjuk pada trek. Secara bersamaan seluruh akses lengan pada perangkat akan dipindahkan, namun hanya kepala yang berstatus hidup akan menampilkan ke atas.

Pengaksesan pada catatan tersimpan dalam paket piringan

Trek pada perangkat dimana terdapat catatan akan ditunjuk oleh pengalamatan catatan yang kodenya sudah dirubah oleh pengontrol piringan. Pada silinder yang tepat akses lengan akan memindahka posisi kepala dari *read*/*write*. Piringan akan berputar sehingga menunjuk catatan pada kepala *read*/write, untuk selanjutnya melalu kanal (*channel*) yang di *request* oleh program pada komputer akan dibaca dan ditransfer.

Access Time = Seek Time (pemindahan arm ke cylinder)
+ Head activition time (pemilihan track)
+ Rotational Delay (pemilihan record)
+ Transfer Time

Gambar 2. 20 Rumus Pengaksesan Catatan

Berikut penjelasannya :

1) Pencarian waktu (*seek time*)

   Pada pergerakan kepala *read*/*write* pada piringan ke silinder yang sesuai membutuhkan waktu, waktu inilah yang dimaksud. Merupakan Waktu yang diperlukan untuk memindahkan kepala baca / tulis pada disk ke posisi silinder yang benar.

2) Waktu pengaktifan kepala (head activation time)

   Pada pergerakan kepala *read*/*write* pada piringan ke posisi trek yang sesuai memerlukan waktu, waktu inilah yang dimaksud. Ini adalah waktu yang diperlukan untuk memindahkan kepala baca / tulis pada disk ke posisi trek yang benar.

3) Penundaan rotasi (rotational delay)

   Saat piringan berputar hingga akhirnya sampai pada posisi catatan yang dimaksud membuthkan waktu, waktu inilah yang dimaksud.

4) Waktu transfer (transfer time)

   Pada banyaknya data yang dikirim terdapat kecepatan, waktu kecepatan inilah yang dimaksud.

Akses tetap kepala piringan

Pengaksesan pada permukaan penyimpanan tidak dapat dipindahkan antara silinder ke silinder, untuk setiap trek yang dimiliki oleh piringan.

Pengaksesan pada piringan magnetik

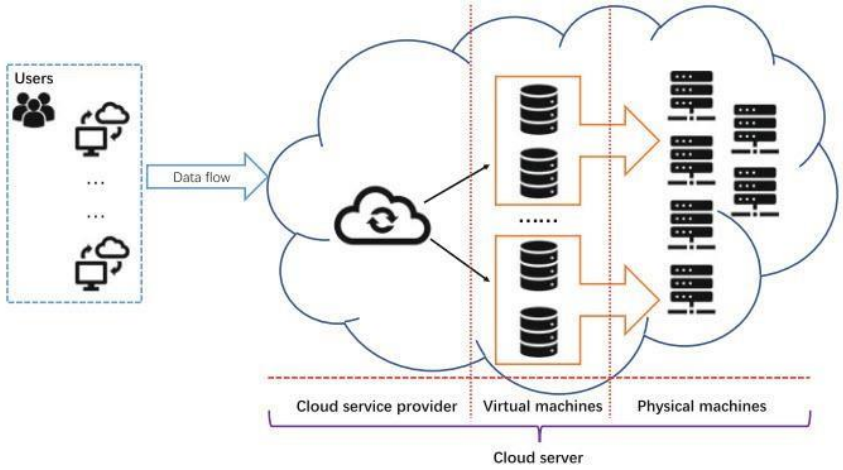
Baik secara sekuensial, indeks sekuensial, maupun langsung dapat dilakukan untuk membentuk suatu data pada piringan magnetik, dan bisa dilakukan metode akses langsung, atau metode akses sekuensial untuk mengambil suatu data dari catatan yang disimpan.

4. Cloud Storage

Cloud storage (penyimpanan awan) berfungsi sebagai komponen fundamental untuk sejumlah besar layanan yang diberikan oleh penyedia jasa *cloud*. Pada bagian ini, kita akan membahas cloud storage dan aplikasinya.

Cloud Storage Architecture

Umumnya, ada dua entitas dalam sistem penyimpanan awan, yaitu pengguna dan penyedia layanan cloud seperti yang ditunjukkan pada gambar berikut :



Gambar 2. 21 *Cloud Storage Architecture*

Pengguna adalah pemilik data. Mereka memiliki sejumlah besar file data yang akan di kirim ke penyedia layanan cloud storage dan juga mengakses data lainnya secara fleksibel dan efisien.

Seperti layanan data yang disediakan oleh penyedia layanan cloud yang memiliki ruang penyimpanan dan sumber komputasi yang signifikan. Ruang Penyimpanan dan kemampuan komputasi disediakan oleh sejumlah besar mesin yang canggih dan perangkat yang digunakan, dan dipelihara oleh penyedia layanan cloud.

Secara gambaran umum, penyedia layanan awan menggunakan kerangka kerja tiga lapisan untuk menyediakan layanan penyimpanan cloud. Di bagian atas kerangka, penyedia layanan cloud berinteraksi dengan semua pengguna untuk menerima permintaan layanan data outsourcing, akses data, dan operasi lain pada data mereka. Setelah menerima permintaan layanan dari pengguna, penyedia layanan cloud menanganinya melalui beberapa algoritma dan mengembalikan hasil yang sesuai ke pengguna sebagai respon.

Aplikasi *Cloud Storage*

Saat ini sudah banyak tersedia layanan aplikasi cloud storage, dari yang gratis hingga yang berbayar, tentunya layanan berbayar memiliki banyak kelebihan misalnya seperti kemanan, kecepatan transfer file, dan lainnya.

Berikut beberapa aplikasi cloud storage yang cukup populer, yaitu : Google Drive, Dropbox, One Drive, Box, Mega, ADrive, Bitcasa.

Aplikasi-aplikasi cloud storage di atas dapat digunakan secara gratis, tentunya dengan limit tertentu yang sudah ditentukan. Jika kita memerlukan

kapasitas penyimpanan yang lebih besar, kita bisa menggunakan layanan premium dari aplikasi-aplikasi tersebut.

Manajemen Penyimpanan

Manajemen Penyimpanan mendefinisikan bagaimana sistem operasi berinteraksi dengan disk dan media penyimpanan. Contohnya seperti OS Windows menyediakan dukungan untuk berbagai jenis media penyimpanan, termasuk hard drive, drive, drive tape dan penyimpanan jaringan seperti SAN (Storage Area Networks) dan iSCSI (Internet Small Computer System Interface).

Berikut ini, kita akan melihat penyimpanan pada hard drive, karena hard drive telah lama menjadi media penyimpanan utama untuk komputer.

Sebuah disk terdiri dari sektor, yang merupakan blok dengan ukuran tertentu yang memiliki alamat. Sebuah hard drive terdiri dari serangkaian blok. Ukuran blok telah dari 2 sampai 8 kbytes, dan blok memiliki angka dari nol ke atas.

Satu tugas untuk sistem operasi adalah mengelola hard drive. Ketika file baru akan disimpan, sistem operasi harus mengalokasikan ruang pada hard disk. Beberapa cara untuk melakukan ini adalah:

1) Alokasi interkoneksi

2) Daftar tertaut

3) Pemetaan berkas

4) Indeks alokasi

Alokasi interkoneksi berarti bahwa file disimpan dalam blok yang terurut, misalnya 1,sampai dengan, 7 dan seterusnya. Sistem operasi hanya perlu melacak mulai blok dan jumlah blok yang menempati file. Masalah pada alokasi interkoneksi adalah sebagai berikut :

1) Alokasi interkoneksi ini tidak selalu bersebelahan blok bebas pada hard disk saat membuat file.

2) Ketika memperluas file, blok bebas mungkin tidak tersedia sehingga ekspansi tidak akan terus menerus atau terhenti.

Cara lain untuk melacak file adalah dengan menggunakan daftar tertaut. Setiap blok memiliki pointer yang menunjuk ke blok berikutnya dalam file. Oleh karena itu, blok tidak perlu urutan yang berturut-turut. Kelemahan dari daftar tertaut adalah bahwa jika pointer tidak diposisikan dengan benar ke blok.

Pemetaan berkas adalah peningkatan dari daftar tertaut. Semua pointer berada di satu tempat, jadi dengan cara ini, kita mendapatkan pemetaan berkas. Jika salah satu pointer salah, itu tidak mempengaruhi yang lain. Alokasi indeks juga merupakan peningkatan dari daftar tertaut, karena semua penunjuk berada

dalam satu tempat. Pointer diakses menggunakan indeks daripada menggunakan pointer ke blok berikutnya dalam cara daftar tertaut.

5. Organisasi Hard Disk

Ada berbagai cara untuk mengatur penyimpanan data dalam sistem komputer. PC biasanya hanya memiliki satu volume pada satu hard drive. Namun, mungkin bermanfaat untuk mengatur disk dengan membaginya menjadi beberapa jilid.

Volume yang digunakan oleh server akan sering diperluas di beberapa hard drive, tujuan meningkatkan kecepatan membaca dan keamanan.

Partisi

Sebuah hard drive terdiri dari satu atau lebih partisi, yang merupakan bagian dari hard drive. Setiap partisi kemudian dapat berfungsi sebagai disk terpisah. Partisi adalah kumpulan dari sektor yang bersebelahan pada disk. Tabel partisi atau pangkalan data lainnya untuk manajemen disk menyimpan sektor awal dan ukuran partisi. Tugas dari Partition Manager adalah untuk membuat, menghapus dan mengelola partisi, sehingga memastikan bahwa semua partisi memiliki ID unik.

Volume

Penyimpanan pada komputer dibagi ke dalam volume yang ditunjuk oleh sebuah huruf seperti C, D atau E. Volume sederhana hanya satu partisi, tetapi Anda dapat mengatur volume sehingga mereka terdiri dari beberapa partisi pada satu atau lebih hard drive. Volume sederhana menggunakan hanya satu hard drive, yang berarti bahwa jika hard disk crash, volume tidak digunakan. Menggunakan beberapa volume dan beberapa disk menghindari hal ini. Oleh karena itu, kita masih dapat mengakses data bahkan jika satu hard drive tidak berfungsi.

Ada 2 jenis volume, yaitu :

1) Volume sederhana mewakili sektor pada satu partisi

2) Volume multi-partisi mewakili sektor dari beberapa partisi

Keuntungan dari volume multi-partisi adalah kinerja, keandalan dan ukuran volume. Beberapa jenis volume multi-partisi adalah spanned volume, striped volume, RAID 1 volume dan RAID 5 volume.

Spanned Volume

Spanned Volume adalah volume yang terdiri dari beberapa partisi. Keuntungan menggunakan spanned volume adalah bahwa seseorang dapat memperluas volume tanpa harus mengganti drive dengan drive baru yang lebih besar, sedangkan kelemahan dari spanned volume adalah risiko yang lebih besar dari kegagalan disk ketika partisi terletak di drive yang berbeda. Spanned

volume tidak memiliki toleransi kesalahan seperti RAID, jadi jika disk gagal, seluruh volume akan hilang.

Striped Volume

Sebuah Striped volume terdiri dari beberapa partisi pada beberapa hard drive. Ketika kita menulis data ke dalam striped volume, data didistribusikan di semua hard drive. Keuntungan dari striped volume adalah bahwa kecepatan membaca meningkat karena komputer dapat membaca data dari disk secara paralel. Kerugian dari striped volume adalah bahwa hal itu dapat mudah menyebabkan kegagalan drive.

RAID 1 Volume

RAID 1 volume terdiri dari dua hard disk. Data yang sama ada di kedua hard disk, yang berarti bahwa satu disk adalah salinan yang lain. Keuntungan dari RAID 1 volume adalah keamanan. Jika salah satu hard drive gagal, data tetap bisa diakses dari hard drive lainnya. Kerugian dari RAID 1 adalah bahwa ia menggunakan dua kali banyak ruang penyimpanan.

RAID 5 dan RAID 6 Volume

Volume RAID 5 telah banyak digunakan untuk menyimpan data di server karena meningkatkan kecepatan membaca dan keamanan. Untuk menggunakan RAID 5, setidaknya tiga disk diperlukan, karena RAID 5 menulis data ke beberapa disk. Keamanan meningkat ketika informasi tambahan tentang data (data paritas) ada di beberapa disk. RAID 5 menghasilkan kecepatan membacacepat karena dapat membaca data dari disk secara paralel.

RAID 5 memerlukan kecepatan tulis sedikit lebih lambat, karena data paritas juga ikut ditulis. RAID 5 dapat mentolerir salah satu disk gagal karena disk lain akan terus bekerja sehingga dapat mengambil data dari sana. Meskipun begitu, jika disk tidak menggunakan kecepatan baca, disk akan lebih lambat, karena harus menggunakan data paritas. RAID 6 adalah peningkatan dari RAID 5, dan sangat mirip dengan RAID 5. RAID 6 adalah alternatif untuk RAID 5 jika sistem penyimpanan berisi empat disk atau lebih karena RAID 6 dapet menoleransi jika dua disk gagal.

C. Soal Latihan

1. Jelaskan waktu yang diperlukan untuk mengakses data dalam sistem disk.
2. Jelaskan yang dimaskud dengan satu silinder dalam disk storage.
3. Jelaskan cara pengaksesan catatan yang disimpan pada paket piringan.

D. Referensi

Einar Krogh. (2020). An Introduction to Windows Operating System.

Gan Fuxi, Wang Yang. (2015). Data Storage at the Nanoscale Advances and Applications.

James O'Reilly. (2017). Network Storage Tools and Technologies for Storing Your Company's Data.

PERTEMUAN 3
ORGANISASI BERKAS PRIMER

A. Tujuan Pembelajaran

1. Memahami konsep dasar organisasi berkas primer
2. Memahami macam – macam metode organisasi berkas
3. Mengetahui tentang pengukuran kuantitatif

B. Uraian Materi

Didalam organisasi berkas kita akan belajar struktur yang baik untuk dapat mengorganisasikan:

1. Data di proses berbeda untuk kebutuhan yang bermacam-macam
2. berpeluang diberlakukannya tugas-tugas utama untuk sebuah data
3. Besar rekaman
4. Berkas dengan proses yang khusus
5. Mengatasi masalah dengan cara yang berbeda dari cara yang sebelumnya dilakukan

1. Organisasi Berkas

Ada 3 jenis organisasi berkas primer ialah sekuensial, langsung, dan sekuensial berindeks. Dari ketiga jenis tersebut mempunyai cara yang berbeda-beda untuk memproses dan mengakses berkas.

Organisasi	Akses
Sekuensial	Sekuensial
Sekuensial berindeks	Sekuensial atau langsung
Langsung	langsung

Gambar 3. 1 Jenis-jenis Organisasi Berkas Primer

Sekuensial

Pada pengorganisasian sekumpulan catatan ke dalam sebuah berkas memerlukan sebuah cara, dan cara paling dasar untuk melakukan hal tersebut adalah dengan sequential file organization (organisasi berkas sekuensial). Waktu pada saat catatan dibuat, kemudia direkam berurut dalam sequential file organization. Pada posisi pertama dalam berkas catatan pertama diletakan, kemudian posisi kedua dalam berkas diletakan catatan kedua, dan begitu

seterusnya. Seluruh catatan tersebut perlu diakses secara berurut pada saat waktu akses dan pada waktu berkas dipakai sebagai masukan (input)

Jika kita akan menambah sebuah catatan pada berkas sekuensial, catatan tersebut akan berada pada bagian berkas akhir. Jadi tidak berarti catatan tersebut akan disimpan berurut secara numerik pada sequential file organization.

Pada setiap jenis pemrosesan data dengan orientasi sekumpulan (batch) sequential file organization sering digunakan. Pemanggilan primitif akses digunakan agar setiap catatan dapat dibaca dan diakses berurut.

Sequential File Organization bisa terdiri dari sejumlah catatan yang jenisnya berbeda

Indeks Sekuensial

Organisasi file indeks sekuensial adalah kombinasi dari organisasi file sekuensial dan organisasi file relatif. Ini adalah metode yang efektif untuk mengatur koleksi rekaman yang perlu diakses secara berurutan atau langsung berdasarkan nilai kunci. Pengaturan file tinta berurutan dirancang untuk kebutuhan pengaksesan secara acak pada file sekuensial. Jadi record dapat diakses secara langsung dengan menggunakan satu atau lebih index.

Struktur file indeks terdiri dari 3 bagian, yaitu:

1) Prime data area

   Daerah data utama yang diurutkan secara sekuensial dan dibagi menjadi blok fisik dengan pointer / nomor record yang tetap

2) Indeks area

   Daerah indeks untuk tiap blok daerah data utama, isi dari indeks area merupakan record pertama dari setiap blok sekuensial.

3) Overflow area

   Daerah ruang bebas untuk penyimpanan record baru selama penyimpanan file Langsung Jika kita akan menambah sebuah catatan pada berkas sekuensial, catatan tersebut akan berada pada bagian berkas akhir.

Jadi tidak berarti catatan tersebut akan disimpan berurut secara numerik pada sequential file organization. Pada setiap jenis pemrosesan data dengan orientasi sekumpulan (batch) sequential file organization sering digunakan. Pemanggilan primitif akses digunakan agar setiap catatan dapat dibaca dan diakses berurut. Sequential File Organization bisa terdiri dari sejumlah catatan yang jenisnya berbeda. File data dapat diakses terlebih dahulu hanya dengan membaca record direktori, dan kemudian mendapatkan pointer ke record data yang sesuai.

Contohnya, ada beberapa informasi yang sering di sebut rekaman yang didalamnya terdapat data seperti entisitas individual. Rekaman dapat dikualifikasikan menjadi beberapa unit yang lebih kecil, yang sering disebut medan-medan yang didalamnya terdapat nilai-nilai khusus pada atribut-atribut yang mewakili individu tersebut.

2. Medan Data

Didalam medan terdapat nilai yang dapat menghasilkan sebuah rekaman. Isi dalam sebuah medan sangat bergantung kepada atribut yang dimiliki pemilik record. Nilai tersebut nantinya akan di jalankan proses komputerisasi nilai-nilai dalam medan haruslah tunduk pada type nilai, domain, kapasitas byte maksimum dan lain sebagainya yang dimiliki medan tersebut.

Record yang ada didalam berkas biasanya mempunyai medan yang berfungsi khusus, ialah untuk identitas record yang mempunyai sifat berbeda baik internal maupun eksternal. Medan ‘tanggal’ yang biasanya mempunyai sebuah record yaitu contoh yang unik.

Medan mempunyai maksimal 8 byte, yaitu terdiri dari 3 medan yang lebih elementer, ialah tanggal mempunyai 2 digit, bulaun mempunyai 2 digit, dan tahun mempunyai 4 digit, Informasi tanggal paling baik diberi tipe berupa bilangan, mengetahui tanggal dapat dibandingkan antara lebih tua atau lebih muda, prosesnya dengan menghitung mundur dan lain-lain.

Merupakan hal yang paling penting untuk pemilihan tipe data di sarankan memenuhi kebutuhan dan mencukupi syarat sistem yang sedang dibangun. Proses menggabungkan satu tipe dan satu representasi ialah merupakan suatu domain yang jelas atau spesifik. Representasi yang tidak jelasakan mengakibatkan sulitnya mencari data kembali.

Rekaman Data

Medan ke-1	Medan ke-2	...	Medan ke-n

Gambar 3. 2 Medan Data

Suatu medan dapat memungkinkan digabungkan dengan medan yang lainnya karena medan tersebut memiliki deskripsi yang jelas, contohnya dengan suatu kejadian atau suatu subyek. Record adalah kumpulan berbagai medan yang mempunyai item data.

Data mahasiswa contohnya, dapat disimpan dengan cara seperti ini :

Record mahasiswa

Nama Mahasiswa	Nomor Mahasiswa	Jenjang	Program Studi	Dosen Wali	SPP	Data lain

Gambar 3. 3 Record Mahasiswa

Dapat dilihat di atas Nama Mahasiswa, Nomor Mahasiswa, Jenjang, Program studi, Dosen Wali, SPP, dan lain-lain di simpan dengan nama-nama medan. Dapat di ambil kunci primernya suatu rekaman dari suatu medan yang berbeda dari medan di rekaman lainya dan semua medan yang tersisa di sebut dengan medan sekunder. Seperti contoh diatas kunci primer yang dapat di ambil yaitu nama mahasiswa atau nomor mahasiswa ssedangkan untuk kunci sekudernya dapat kita amabil yaitu medan, jenjang, program studi, dan lain-lain.

3. Berkas Data

Suatu berkas adalah kumpulan dari record-record atau rekaman yang saling terhubung dan di simpan didalam penympanan komputer.

Contoh alat penyimpanan eksternal yaitu penggerak disk dengan disk magnetiknya. Suatu disk akan memiliki identitas yang di ketahui oleh sistem operasi, dan memiliki struktur yang di tentukan sendiri oleh sistem berkas.

Berikut ini adalah contoh berkas tentang mahasiswa di universitas:

Nama Mahasiswa	Nomor Mahasiswa	Jenjang	Program Studi	Dosen Wali	SPP	Data lain
Dian Kartika	11.50.001	S1	Sistem Informasi	Made	500.000	...
Syda Arlin	11.50.011	S1	Sistem Informasi	Sugeng	500.000	...
Yuniati	11.30.012	DIII	Manajemen Informatika	Made	400.000	...
Sunaryono	11.30.013	DIII	Manajemen Informatika	Made	400.000	...
Zainul	11.55.024	S1	Teknik Informatika	Sugeng	500.000	...
Yenny Noorma	11.50.021	S1	Sistem Informasi	Sugeng	500.000	...
Mustafa	11.50.025	S1	Sistem Informasi	Sugeng	500.000	...
Kusmiyati	11.55.024	S1	Teknik Informatika	Made	500.000	...
Susiana	11.55.024	S1	Teknik Informatika	Made	500.000	...
Dewi Dwi	11.30.014	DIII	Manajemen Informatika	Made	400.000	...

Gambar 3. 4 Berkas Mahasiswa

Untuk Membangun suatu berkas seperti berkas mahasiswa sangat tidak baik bila sebuah tipe rekaman di campur atau di gabungkan dengan tipe rekaman yang lain, contohnya rekaman mahasiswa digabungkan dengan daftar mata kuliah dalam satu berkas yang sama.

Begitupun sebaliknya mempunyai beberapa berkas yang berbeda juga akan mempersulit perancang berkas untuk mecarinya kembali di kemudian hari. Dan para pakar selama ini mencari solusinya, dan akhirnya di temukan solusinya yaitu menggabungkan semua rekaman-rekaman dari semua mahasiswa untuk semua jurusan dalam satu berkas.

File digabungkan secara logika untuk sebagai barisan record. Record-record di letakan disk File diberikan sebagai bentukan dasar di sistem operasi. Meski blok berukuran tetap dan ditentukan sistem operasi, akan tetapi record-record dapat berbeda ukuran.

Direktori File (File directory)

Direktori file yaitu ialah suatu proses di dalam memory atau disk yang didalamnya terdapat data seperti :

1. ruang yang sudah tersedia untuk file tersebut
2. nama file
3. ruang yang sudah digunakan
4. pemilik file
5. pemformatan file dari record-recordnya
6. informasi data lain
7. struktur file

Pengukuran kuantitatif (ukuran performansi dari suatu file)

Perlu adanya pengukuran terhadap jumlah hal-hal yang berhubungan dengan waktu pengaksesan, untuk mengkoreksi kinerja suatu sistem organisasi file, hal-hal tersebut yaitu:

1. R adalah banyaknya penyimpanan yang di butuhkan suatu record
2. T_F adalah waktu yang dibutuhkan untuk mengambil record
3. T_N adalah waktu yang dibutuhkan untuk mendapatkan record berikutnya
4. T_I adalah waktu yang dibutuhkan untuk memperbaharui file dengan cara menyisipkan suatu record
5. T_U adalah waktu yang dibutuhkan untuk mengubah suatu record
6. T_X adalah waktu yang dibutuhkan untuk membaca keseluruh file
7. T_Y adalah waktu yang dibutuhkan untuk menggabungkan kembali file

Record Size (R)

Record size adalah jumlah ruang yang diperlukan untuk satu ukuran nsebuah record biasanya record size sendiri lebih besar dari ruang setiap atribut record itu sendiri.

Contoh : record#2	NIP	: 8 byte
	Nama	: 21 byte
	Alamat	: 31 byte
	Mata Pelajaran	: 11 byte
	Total	: 71 byte

Melihat contoh di atas secara logika totalnya adalah 68 byte, tetapi realistisnya dibutuhkan ruang yang lebih besar dari 68 byte untuk setiap record-recordnya.

Fetch Record (pengambilan record)/Tf

Fetch Record ialah waktu yang diperlukan untuk mengambil sebuah record dari suatu file, dapat dirumuskan dengan Ty. Dan Fetch record bergantung dengan 2 hal seperti berikut:

1. Waktu yang diperlukan untuk menyimpan head atau pembaca disk dilokasi record tersebut tersimpan.
2. pembacaan yang benar

Get Next Record (TN)

Adalah waktu yang diperlukan untuk mencari record berikutnya. Dan bila record berikutnya berada pada posisi yang sama dengan record sebelumnya, maka lebih dapat minimalkan waktu yang dibutuhkan.

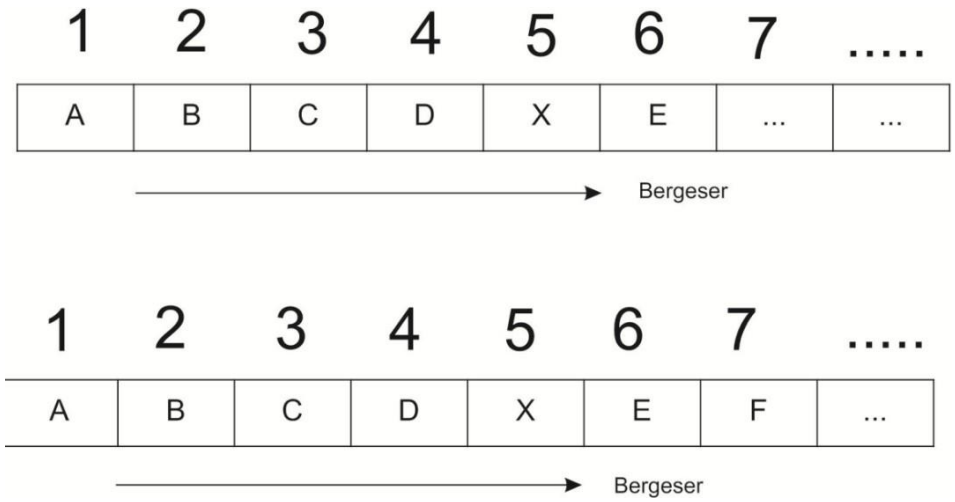
Insert Record (TI)

Adalah waktu yang diperlukan untuk menyisipkan suatu record, dan memiliki nilai waktunya singkat serta juga dapat bernilai panjang.

TI bernilai besar bila insert yang dilakukan pendek (awal track) kecil

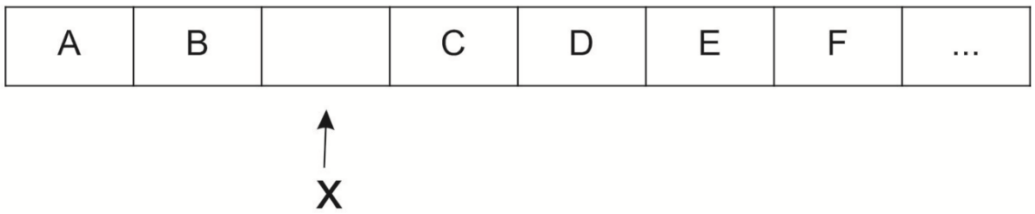
TI bernilai kecil bila insert yang dilakukan di akhir track kecuali bila ada penyediaan tempat khusus pada track.

TI besar : bila penambahan suatu record menyebabkan pergeseran record-record sesudahnya.



Gambar 3. 5 Insert Record

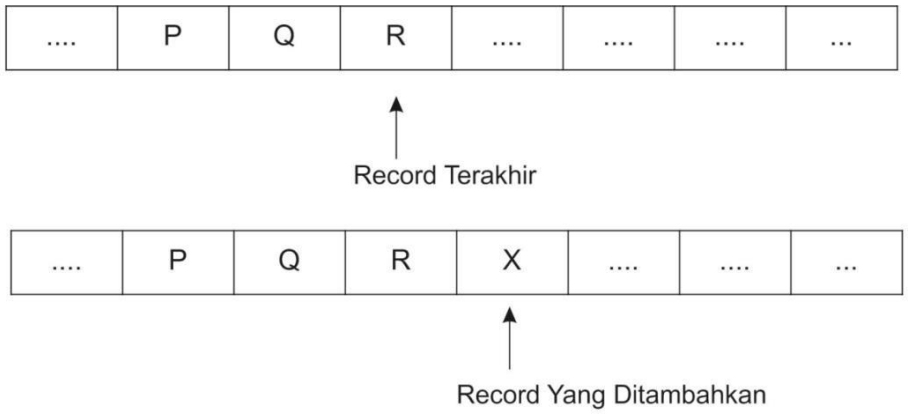
TI Kecil : bila penambahan suatu record tidak menyebabkan pergeseran dari record-record sesudahnya.



Gambar 3. 6 Insert Record 2

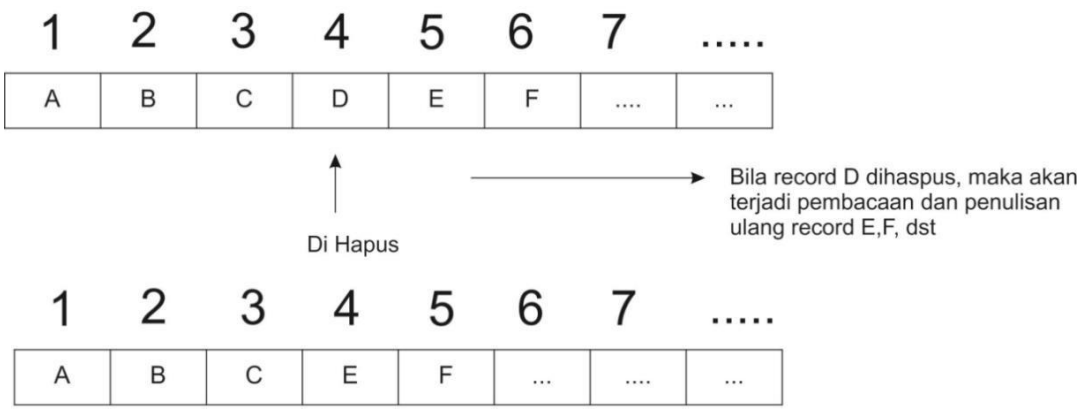
Appending Record

Adalah penambahan record diakhir file (setelah record ter akhir). Alamat blok sebelumnya (terakhir) dari file hampir selalu dijaga dalam direktori untuk membuat akhir file lebih mudah untuk melokasikan.



Gambar 3. 7 *Appending Record*

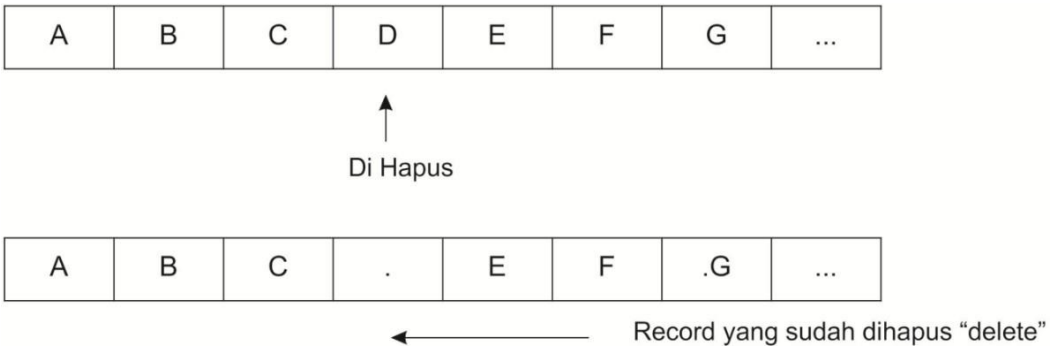
Update Record (Tu)



Gambar 3. 8 *Update Record*

Penghapusan record perubahan dapat menyebabkan record ditulis ulang nanti karena perubahan lokasi.

Cara lain untuk penghapusan suatu record tanpa menulis ulang record sesudahnya yaitu dengan mengisi ruang record lama dengan character NULL atau satu pesan seperti dihapus.



Gambar 3. 9 *Delete Record*

Read Entire File (pembacaan semua record pada file)/ (Tx)

Terkadang suatu file dibutuhkan pembacaan semua dari isi file, notasi Tx ialah waktu pembacaan tersebut file tersebut, dan isi Tx adalah hasil nilai yang tebaca.

Reorganisasi (Ty)

Adalah perancangan kembali rekaman-rekaman sebuah file. Dan terkadang Ty digunakan secara berkala contohnya seperti setiap hari, setiap minggu seterusnya.

Proses organisasi yaitu :

1) Hapus rekaman yang memiliki tanda *
2) Hapus rekaman tidak valid
3) Melakukan pembebsan ruang penyimpanan untuk memasukan record-record yang baru

Organisasi File

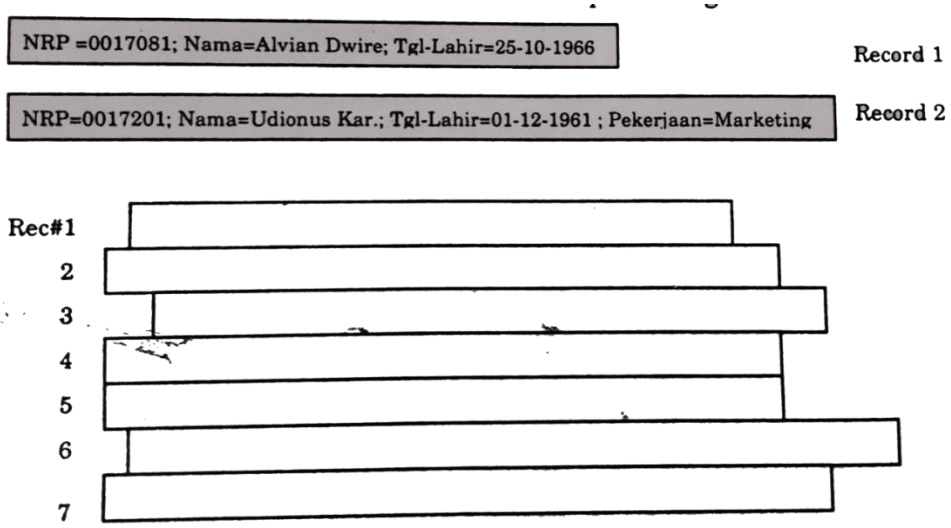
Agar supaya mendapatkan hasil record-record yang dibutuhkan pada sebuah file maka file tersebut haruslah memenuhi syarat dan aturan organisasi file. Ada beberapa contoh organisasi dari suatu file yang biasa digunakan dalam kelompok sistem perorganisasian file dasar yaitu:

1) Pile
2) Sequential file
3) Indexed Sequential file
4) Indexed File
5) Direct File
6) Inverted file dan Multi List

4. Pile (tumpukan)

Rekaman-rekaman dalam pile dikelompokan berdasarkan waktu kedatangan rekaman. Data ini tidak digunakan untuk dikategorikan dianalisa, atau di proses dalam ukuran field.

Tidak harus panjangnya tetap atau variable record length dan tidak juga dibutuhkan atribut-atribut yang sama untuk satu persatu recordnya dalam panjangnya record dari suatu file.



Gambar 3. 10 Pile

Pengunaan Pile

Pile dapat dibuat pada waktu di kelompokan utamanya dalam pemrosesan data yang tidak mudah untuk dikelompokan atau dikumpulkan.

Pile biasa diperlukan untuk mengkelompokan data sebelum terjadinya pemrosesan data.

Analisa dalam file pile menjadi sangatlah rumit karena waktu yang dibutuhkan untuk mendapatkan kembali beberapa record dengan menggunakan statistik.

Kinerja Pile

1) Record Size

Karena nama dan data dalam panjang yang variabel, dua karakter pemisah (= dan, atau dalam contoh diatas) disimpan untuk menandai setiap elemen data. Record size dihitung nilai ukuran "atribut data" dan nilainya berikut koma, tanda sama dengan yang digunakan sebagai karakter pemisah dan tanda-tanda lainnya.

Ukuran record diberikan dengan persamaan :

R=a' (A + V +2) record size rata-rata

dengan

a jumlah field/attribut

a’ = rata-rata jumlah field/attribut

A= ukuran attribute/field (, ; &:= dst)

V = ukuran nilai attribut (isi field)

2) Fetch Record (TF)

Karena data tidak tersusun secara baik, maka nilai Tf relatif tinggi. Waktu yang diperlukan untuk melokasikan satu record dalam suatu file adalah tinggi karena semua record harus dilakukan penelusuran untuk melokasikan satu data item.

Disamping itu pencarian data harus dilakukan secara serial dimana setiap blok dibaca satu persatu sampai record yang dicari ditemukan.

Blok rata-rata = $\sum_{i=1}^{b} i/b$ = ½(1 + b) = ½ Jika b>> 1

dengan :

b = jumlah block total

t'= bulk transfer rate

B = ukuran block

Waktu yang digunakan untuk membaca sejumlah block secara serial.

T = 1/2b$\frac{B}{f'}$

Pemakaian Bulk transfer rate t' karena kita membaca file secara serial dari bagian awal file melalui gap dan batas silinder sampai menemukan blog yang berisi record yang diinginkan.

3) Get Next record (TN)

Selama tidak ada pengurutan record disediakan dalam satu Pile. Pengganti record secara potensial bisa disetiap tempat dalam file. Karena posisinya tidak diketahui maka waktu yang diperlukan untuk menemukan record pengganti yang berubah-ubah :

TN = TF

Dengan asumsi bahwa informasi dari record sebelumnya perlukan untuk penelusuran record pengganti.

4) Insert Time (T!)

Adalah waktu yang diperlukan untuk menyisipkan satu record baru ke dalam satu file Pile. Prosesnya berlangsung cepat, karena record yang ditambahkan akan disimpan di akhir file bila:

s = seek time

r = rotasi time

btt = block transfer time = b/t msec

trw = waktu untuk menulis kembali (rewrite), maka:

TI = S+r+btt+ trw

Alamat akhir dari file diketahui, satu record baru yang ditetap kan ditambahkan ke ujung dan akhir dari pointer diperbaharui.

5) Up-Date Record dalam satu File (Tu)

Pembaharuan satu record berisi lokasi dan record lama yang invalid dan menulis satu record baru, kemungkinan lebih besar, record berada di akhir file maka

Tu = Tf TRW : bila ukuran record tetap

TF +TRw +TI : bila ukuran record berubah

6) Read Entire File Pile (pembacaan seluruh file)/ Tx

Satu penelusuran lengkap dalam file organisasi memerlukan membaca file sampai akhir. Hanya 2 kali lebih lama dari penelusuran yang ditentukan, maka :

Tx=27Tf, n dari Y-7-1/28

Record tanpa mencocokkan (attribut) yang cocok aspal dihapus dengan terlebih dahulu diurutkan (sort). File yang di sort di sedia kan secara serial dari nilai-nilai attribut. Menghasilkan file yang tidak terlalu panjang dari Pile sederhana.

Perkiraan waktu yang diperlukan untuk melakukan satu sortir seluruh file adalah :

Tsort(n) = 2b.btt+2b (log 2 btt

= 2n(1+log2(n)JR

C. Soal Latihan

1. Apa pengertian dari Organisasi Berkas Primer?
2. Sebutkan macam cara memproses organisasi file?
3. Apa perbedaan INSERT RECORD (TI) Besar dan Kecil?

D. Referensi

Coronel, Carlos. 2010. Database Systems: Design, Implementation, and Management

Handayani, Dewi. 2001. Sistem Berkas. Yogyakarta. J&J Learning

Alan l, Tharp-john Willey &Son. 1988. File Organization and Process

PERTEMUAN 4
ORGANISASI BERKAS SEKUENSIAL

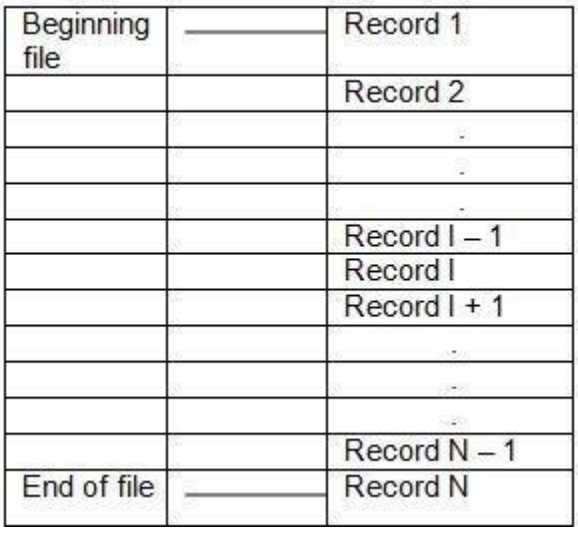
A. Tujuan Pembelajaran

1. Setelah mempelajari materi ini, mahasiswa mampu memahami dan dapat menjelaskan organisasi berkas sekuensial.

B. Uraian Materi

1. Sequential File Organization

Pada pengorganisasian sekumpulan catatan ke dalam sebuah berkas memerlukan sebuah cara, dan cara paling dasar untuk melakukan hal tersebut adalah dengan *sequential file organization* (organisasi berkas sekuensial). Waktu pada saat catatan dibuat, kemudia direkam berurut dalam sequential file organization. Pada posisi pertama dalam berkas catatan pertama diletakan, kemudian posisi kedua dalam berkas diletakan catatan kedua, dan begitu seterusnya. Seluruh catatan tersebut perlu diakses secara berurut pada saat waktu akses dan pada waktu berkas dipakai sebagai masukan (*input*).



Gambar 4. 1 *Sequential File Organization*

Jika kita akan menambah sebuah catatan pada berkas sekuensial, catatam tersebut akan berada pada bagian berkas akhir. Jadi tidak berarti catatan tersebut akan disimpan berurut secara numerik pada sequential *file organization*.

Pada setiap jenis pemrosesan data dengan orientasi sekumpulan (batch) sequential file organization sering digunakan. Pemanggilan primitif akses digunakan agar setiap catatan dapat dibaca dan diakses berurut.

Sequential File Organization bisa terdiri dari sejumlah catatan yang jenisnya berbeda. Contohnya seperti berikut :

Tabel di bawah ini adalah bentuk dalam sistem penggajian pada suatu perusahaan yang telah menggunakan sistem terpadu. Terlihat dari sebuah berkas karyawan yang terdiri dari 2 jenis record.

Tabel 4. 1 Record Data Karyawan

Record Type	No Pegawai	Nama Pegawai	Alamat	Pendidikan	Divisi

Tabel 4. 2 Record Pembayaran Gaji

Record Type	No Pegawai	No Rekening	Jumlah Jam Kerja	Jam Lembur	Tanggal Pembayaran

Catatan pada berkas karyawan di atas tidak membutuhkan format maupun besar ukuran yang sama. Catatan di atas berkas akan disortir menggunakan Record Type dan No Pegawai.

Batch processing lebih sering digunakan pada sequential file organization ketimbang interactive processing hal ini dikarenakan catatan dalam berkas sekuensial perlu dikases berurut.

Pengaksesan catatan berikut secara cepat merupakan kelebihan utama dari teknik seuqential file organization, dan tidak dapat langsung mengakses catatan yang kita mau merupakan suatu keterbatasan dari sequential file organization.

Untuk mendapat waktu akses yang baik, maka berkas sekuensial dapat dipasangkan dengan catatan pada berkas yang telah terurut. Untuk mendapatkan urutan yang tepat dengan pola akses maka, yang pertama perlu ditentukan terlebih dahulu pola aksesnya, kemudian tentukan sequential file organization.

Penulisan catatan pada urutan yang diinginkan dalam media penyimpanan merupakan cara membuat berkas sekuensial. Dan untuk membuat berkas transkasi sekuensial, tugasnya adalah sebagai berikut :

1) *Data collecting*
2) *Data change*
3) *Data editing*
4) *Transaction checking*
5) *Data sorting editing*

Untuk membuat laporan *file sequential*¸terdapat 3 catatan, yaitu :

1) Kepala catatan (*header record*)

   Dimana bagian ini berisi halaman kepala catatan, dan kepala grup, yang dikenal juga sebagai *identifying information* (informasi pengenal)

2) Rincian catatan (*detail record*)

   Dimana bagian ini berisi isi dari *report* (laporan) yang biasanya tersusun pada kolom.

3) Kaki catatan (*footer record*)

   Dimana pada bagian ini dikenal sebagai *summary information* (informasi ringkasan) yang berisi laporan kaki catatan dan kaki grup.

Retrieve (mengambil) secara berurutan pada catatan dalam berkas sekuensial, yang mana urutan catatan akan ditulis pada sebuah berkas sebagai penentu dimana catatan tersebut didapat kembali. Pembuatan laporan (report generation) dan penyelidikan (inquiry) merupakan retrieve dari sebuah berkas. Karena catatan perlu diakses dengan terurut maka lebih efisien jika menggunakan model pembuatan laporan. Sedangkan cara penyelidikan kurang efisien karena perlu mengakses catatan satu persatu.

Disebut sebagai proses update, yaitu sebuah berkas induk yang terdiri dari data yang umumnya tetap, namun terkadang perlu dilakukan perubahan pada berkas. Berikut adalah beberapa faktor dimana frekuensi dari sebuah berkas induk yang perlu diupdate bergantung :

1) Faktor peningkatan data yang berubah

2) Faktor dari keperluan genting dari berkas yang sedang dieksekusi pada berkas induk

3) Faktor kapasitas ukuran dari berkas induk

4) Faktor sejumlah catatan pada berkas induk yang perlu diperbarui, untuk kemudian dibagi sejumlah catatan berkas induk.

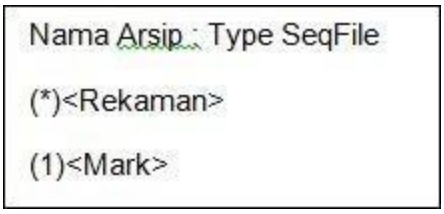
Berkas induk (master file) saat ini akan berubah menjadi berkas induk lama selama proses siklus berikutnya (next cycle) berlangsung, hal inilah yang menybebakan banyaknya berkas induk yang kemudian disebut sebagai pembuatan file (generation file). Dari sekian banyaknya berkas induk ada kemungkinan memiliki nama berkas yang sama, namun tidak akan tertukar karena memiliki nomor pembuatan yang berbeda.

Penggunaan Berkas Sekuensial

Pada pemrosesan data dengan penyesuaian berkas sekuensial sering dipakai. Bidang (field) merupakan penyebutan dari sejumlah elemen catatan.

Setiap catatan umumnya memiliki struktur yang sama. Dengan metode pemanggilan akses primitif, setiap catatan bisa dibaca dan diakses sesuai dengan urutan.

Berikut gambar yang menunjukan cara pendifinisan :



Gambar 4. 2 Pendefinisian

Dari gambar di atas perlu dicatat bahwa lambang bintang (*) mungkin saja bernilai kosong, atau berisi 1 catatan ataupun lebih.

Pengaksesan *Sequential File* Menggunakan Akses Primitif

Perintah open (buka), merupakan sebuah prosedur yang ketika digunakan penulisannya sebagai berikut (input nama berkas, <catatan>) yang berarti berkas sekuensial sudah bisa untuk dibaca, dapat mengakses rekaman pertama yang datanya ada pada rekaman. Pernyataan awalnya adalah acak, dan peranyataan akhirnya adalah catatan pertama bisa diakses, dengan catatan sebagai acuannya.

Perintah read (baca), merupakan sebuah prosedur yang ketika digunakan penulisannya sebagai berikut (input nama berkas, <catatan>) yang berarti dapat membaca catatan setelah catatan yang sedang dieksekusi bisa diakses.

Perintah close (tutup), merupakan sebuah prosedur yang ketika digunakan penulisannya sebagai berikut (input nama berkas).

Perintah rewrite (tulis ulang), merupakan sebuah prosedur yang ketika digunakan penulisannya sebagai berikut (input/output nama berkas) yang berarti berkas sekuensial siap untuk rekam.

Perintah write (tulis), merupakan sebuah prosedur yang ketika digunakan penulisannya sebagai berikut (input/output nama berkas, <catatan>) yang berarti data direkam diwaktu posisi aktual berkas, lalu posisi dipindahkan maju satu. Perintah ini sendiri digunakan untuk menyimpan data variabel ke dalam piringan.

2. File Structure Serial and Sequential

File structure serial merupakan file structure yang cukup sederhana. Ketika berkas dibuat, catatan yang terurut dari berkas dengan urutan catatan dimasukan.

Pada *file structure serial*, catatan fisik yang pertama kali diakses, kemudian dikuti catatan selanjutnya, yaitu catatan ke 2, 3, dan seterusnya. Menerima informasi, mengirim data, berkas urut, dan catatan, merupakan beberapa berkas

yang memiliki urutan kronologis, dan structur serial berguna untuk hal-hal tersebut.

File structure serial bisa dipindahkan ke dalam file structure sequential, caranya yaitu serangkaian catatan dari berkas yang akan dipindahkan diurutkan terlebih dahulu menggunakan pengurutan ascending ataupun descending.

Sekumpulan bidang (*field*) yang digunakan untuk menetapkan urutan catatan membuat sebuah kunci. Persis seperti file structure serial, file structure sequential juga dapat diakses dengan cara yang sama, yaitu catatan pertama diakses kemudian diikuti catatan selanjutnya.

Contoh :

Sequential file yang di *sorting* (pengurutan) menggunakan Nama Karyawan.

No. Rec	NIK	Nama	Alamat	Divisi	Data lainnya..
1	85693214	Andini	Jl. Bahagia	Technical	:.....
2	78542154	Budi	Jl. Diponegoro	Technical	:.....
3	85693325	Cintia	Jl. Kebayoran	Technical	:.....
4	74585569	Doni	Jl. Pasar Minggu	Technical	:.....

Gambar 4. 3 File Sekuensial

Pada file sekuensial di atas struktur serial diberi kunci utama (*primary key*) ‘NIK’ dan struktur sekuensial diberi kunci urut (*sort key*) ‘Nama’

Performansi File Sekuensial

Besaran ukuran pada file sekuensial

Keperluan berkas penyimpanan terhadap berkas sekuensial memakai format catatan yang tetap (*fixed*), artinya semua bergantung pada nilai sejumlah atribut yang bisa jadi = a.

Ukuran tetap = menghasilkan sejumlah field dan ukuran rata-ratanya.

R = a. V

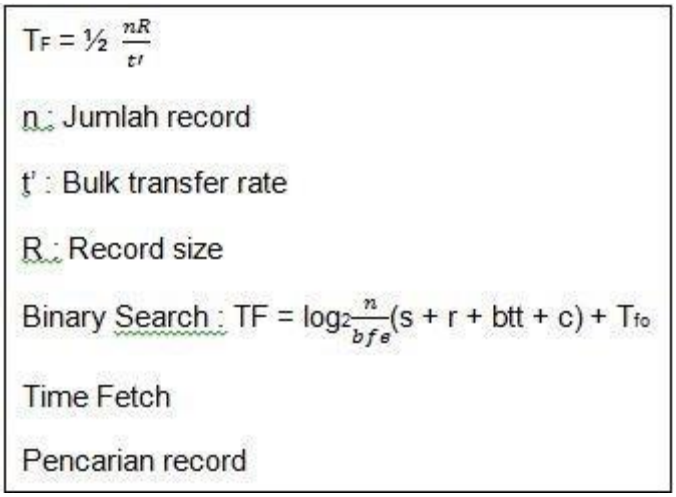
A : Jumlah field/attribut

V : Ukuran nilai attribut (isi field) = Ukuran rata-rata dari attribut/field

Fetch Record dalam File Sekuensial

Gambar 4. 4 Keterangan Pada File Sekuensial

Jika memiliki akses langsung (direct access) bisa terjadi pengurangan soal waktu yang dibutuhkan dalam mengambil sebuah catatan yang akan digunakan. Dalam file sekuensial akses langsung bisa dilakukan hanya untuk attribut yang sesuai atau cocok dimana file telah diletakkan dalam rangkaian.



Gambar 4. 5 Rumus Pada File Sekuensial

Pencarian Catatan (*Record Searching*)

Pencarian catatan pada berkas sekuensial bisa dibuat menggunakan beberapa cara sebagai berikut :

1) Pencarian sekuensial (*seuqential search*)

   Merupakan pencarian atau penelusuran yang bisa dibuat secara serial, dengan dimulai pada catatan pertama, kemudia 2, 3, dan seterusnya sampai ditemukan catatan yang sedang dicari.

   $T_F = T_F$(Pile)

   (Sequensial file/Sequensial search)

2) Pencarian biner (*binary search*)

   Pencarian yang dilakukan bukan dimulai dari catatan per catatan melainkan per blok. Bila blok yang memiliki catatan yang dicari sudah didapatkan, maka kemudian dibuat penelusuran di level catatan.

   Prinsip pencarian biner :

   Contoh :

   NIK : Adalah data yang dicari

   Data : Array yang berisi nilai-nilai data

   Awal : 1

Akhir : N

Tengah L (1 + N) div 2

Langkah-langkah :

a. NIL yang disimpan pada array data harus tersusun berurutan dari nilai terkecil ke besar

b. Jika terdapat N data, maka pencarian dimulai pada data ke (1 + N) div 2 yaitu data yang terletak di tengah (Tengah = (1 + N) div 2).

c. Jika NIL < Data [tengah] maka pencarian diteruskan ke data [awal] sampai ke data [tengah-1] sehingga akhir = tengah – 1, dan dilanjutkan ke langkah pada point f.

d. Jika NIL > data [tengah] maka pencarian diteruskan ke data [tengah + 1] sampai ke data [akhir] sehingga awal = tengah + 1, dan dilanjutkan ke langkah pada point f.

e. Jika nilai = data [tengah] maka pencarian dihentikan dengan hasil, bahwa nilai yang dicari dapat ditemukan.

f. Pencarian akan dihentikan jika awal > akhir dengan hasil, nilai yang dicari tidak dapat ditemukan.

g. Jika kondisi point f tidak dipenuhi, teruskan pencarian ke tengah = (awal + akhir) div 2. Ulangi kembali point c.

***Next Record* Pada File Sekuensial**

Dalam file sekuensial, satu record yang berhasil ditemukan dengan cepat, dapat diakses dengan baik bila berada dalam 1 blok yang sama. Kemungkinan untuk ditemukannya satu record yang berhasil dalam blok yang sama ditentukan oleh sejumlah record per blok (Bfr) yaitu diperlukan dalam 1/Bfr pada kasus berikutnya.

Rumus waktu yang dibutuhkan untuk mendapatkan record :

$TN = \frac{btt}{Bfr} = \frac{R}{t'}$

Time Insert

$T_I = T_F + \frac{1/2n}{Bfr}(btt+T_{RW}) = n\frac{R}{t'} + n\frac{r}{Bfr}$

3. Penelusuran Sekuensial

Teknik penelusuran yang paling sederhana adalah penulusuran sekuensial. Mekanismenya teknik ini akan melakukan pencarian catatan yang dicari dengan cara melakukan pengujian pada setiap urutan catatan hingga pencarian bejalan dengan sukses ataupun gagal di mulai dari bagian kepala.

Contohnya sebagai berikut :

Pada suatu daftar yang berisi record karyawan, kita akan mencari record karyawan yang bernama Aprilia Sulistyowati (field kunci adalah Nam_Karyawan) atau record karyawan dengan nomor pegawai 170493 (field kunci NIK0. Dalam kasus seperti ini berbagai teknik penelusuran diperlukan untuk meningkatkan kemungkinan pengambilan record yang dibutuhkan.

Algoritma Pencarian (*Searching Algorithm*)

Merupakan sebuah algorithm yang melakukan langkah tertentu untuk memperoleh catatan yang dicari, misalnya saja algorithm akan melakukan pencarian dengan nilai A sebagai kuncinya, setelah menjalankan langkah tertentunya maka nantinya algorithm tersebut bisa memperoleh catatan sepenuhnya atau hanya mendapatkan penunjuk (pointer) ke arah catatan yang dicari. Maka nantinya setelah proses selesai akan terdapat 1 antara 2 hasil yaitu berhasil atau gagal.

Algorithm :

1) Seluruh data akan dibandingkan hingga akhirnya berhasil atau gagal dalam mendapatkan data yang dicari.

2) Ketika berhasil, maka seluruh proses penelusuran akan berhenti.

3) Jika gagal, maka penelusuran akan terus berjalan membandingkan seluruh data yang ada.

4) Kekurangannya ialah, jika nilai N adalah 1.000.000 data maka, proses penelusuran akan dilakukan terus hingga sebanyak 1.000.000 data.

5) Dan akhirnya bila data yang diinginkan gagal ditemukan, nanti data yang diinginkan itu akan ditambah ke bagian elemen terakhir pada data.

***Sort Key* (Pengurutan Kunci)**

Untuk meningkatkan efektifitas dan efisiensi dalam sebuah penelusuran dapat dilakukan dengan cara mengurutkan terlebih dahulu suatu daftar yang disesuaikan dengan sebuah key value (nilai kunci). Pada suatu keadaan penggunaan penelusuran sekuensial tidak membutuhkan waktu lama untuk melakukan penelusuran jika catatan yang dicari tidak ada. Hanya akan melakukan penelusuran hingga melewati sebuah tempat dimana logika yang digunakan terdapat dalam daftar, yang mana sampai melewati sebuah catatan yang mempunyai key value lebih besar.

Prosedur SEQFIND2 dalam program berikut menelusuri satu linked list yang diimplementasikan dengan variabel pointer untuk nilai KEYVAL. Daftar berisi satu *node trailer* (simpul ekor).

```
Procedure SEQFIND2 (Daftar: ListPointer ; KeyVal: TipeKunci;Var Lokasi: ListPointer);

Var Found Boolean;

Begin (*SeqFind2*)

    Found :=False;

(* Telusuri sampai elemen ditemukan atau tempat logikanya yang berada dalam list terlewati *)

        While Not Found And (Daftar.Next;

        (* Setting Nilai Lokasi *)

        If Found Then Lokasi := True

                Else Lokasi:= Nil;

End;
```

4. Penelusuran Biner (Binary Search)

Penelusuran Biner (Binary Search) adalah suatu cara yang digunakan dalam penelusuran data dengan mengulangi pembagian setengah dari jumlah data yang ada hingga akan memperkecil area penelusuran menjadi satu data. Dengan cara ini kita akan menghilangkan sementara separuh dari jumlah data. Apabila data yang dicari berhasil ditemukan maka program akan memberikan keluaran, jika gagal ditemukan maka penelusuran akan terus berjalan hingga akhir dari pemagian seluruh data. Algoritma ini umumnya banyak digunakan untuk penelusuran dalam program dengan jumlah data yang banyak, dimana kompleksitas dari algoritma ini adalah O (log n). Pada saat menggunakan metode penelusuran biner, harus diurutkan terlebih dahulu data yang berada dalam array.

Contohnya jika terdapat integer array = {90,10,70,60,30,50,40,20,80}, data para integer array perlu dilakukan pengurutan dahulu dengan cara pengurutan seperti misalnya pengurutan gelembung (bubble sort). Sehingga array kita akan menjadi seperti berikut integer array = {10,20,30,40,50,60,70,80,90}. Apabila angka yang dicari adalah angka 40.

Jika berhasil menemukan data yang dicari, maka program akan berhenti melakukan looping (pengulangan). Dan bila indeks dari data yang dicari tersebut akan ditampilkan, perlu dilakukan penyimpanan indeks dari array dan menampilkannya saja.

Pokok pikiran dalam penelusuran biner pada struktur file sekuensial adalah:

1) Argumen penelusuran meruapakan *attribute key*

2) Penelusuran dilakukan dengan pengaksesan berkas pada bagian tengah, kemudian mebaginya secara terus menerus seperti dengan perbandingan hasil *key value* catatan dengan *key value* yang diinginkan hasilnya.

3) Catatan pertama dan yang terakhir akan dicek agar dapat diketahui dimana catatan berada pada blok tersebut ketikba blok diambil.

4) Jumlah catatan tidak akan berpengaruh ketika total pengambilan, tetapi total pengambilan bergantung pada total blok.

5) Total keseluruhan dalam akses blok diharapkan hasilnya adalah $^{2}\log(b)$.

Pokok pikiran dari penelusuran biner yang paling baik digambarkan secara rekursif. Rekursif sendiri merupakan sebuah mekanisme yang melakukan pemanggilan terhadap dirinya sendiri baik secara langsung ataupun tidak.

Pengujian elemen tengah dalam daftar.

If elemen tengah berisi Key yang diinginkan Then

Stop searching

Else If elemen tengah lebih besar daripada Key yang diinginkan Then

Binary search dari ½ daftar yang pertama

Else binary search dari ½ daftar yang kedua (*elemen tengah lebih kecil dari Key*)

Contoh :

Dengan memperhatikan bagaimana algoritma ini bekerja, maka untuk menemukan nama Fariza berada pada halaman berapa di buku telepon, pertama kita buka halaman tengah pada buku telepon, kemudian lihat nama dengan awalan dari huruf M, dan huruf M lebih besar urutannya ketimbang huruf F untuk nama Fariza, maka dilakukan pencarian secara binary dari A hingga M.

Selanjutnya ganti pencarian di posisi tengah (middle), dan perhatikan nama dengan awalan huruf G, karena huruf G masih lebih besar ketimbang huruf F, maka pencarian binary dijalankan mulai A hingga G. Kemudian beralih lagi ke posisi tengah lalu mulai melihat nama dengan awalan huruf C, karena huruf C urutannya lebih kecil ketimbang huruf F.

Jadi kita lakukan penelusuran biner setengah dari yang kedua, yaitu yang dimulai dari huruf C – G, hingga akhirnya sampai pada halaman yang memuat nama Fariza.

Meskipun arti rekursif secara konsep lebih mudah, tapi cara non rekursif jauh efisien. Cara pencarian binary akan menjalankan satu pencarian binary di array list (hasilnya 0 jika elemen gagal ditemukan).

Perhatikan gambar dibawah ini, yang mengilustrasikan bagaimana prosedur *BINARYFIND* digunakan untuk melokasikan record dengan nilai kunci 27.

Inisialisasi FOUND = False KEYVAL = 27

Tabel 4. 3 Inisialisasi

8	13	17	23	27	31	35	39
[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]
Begin						End	

Langkah 1 *Begin* <= *End* dan berhasil (menemukan data)

Middle = (1 + 8) DIV 2 = 4

list[4] < 27, pindah dari *begin* ke *middle* + 1

Tabel 4. 4 Langkah 1

8	13	17	23	27	31	35	39
[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]
			Middle	*Begin*		*End*	

Langkah 2 *Begin* <= *end* dan gagal

Middle = (5 + 8) DIV 2 = 6

List[6] > 27, dipindahkan *end* ke *middle* – 1

Tabel 4. 5 Langkah 2

8	13	17	23	27	31	35	39
[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]
		Begin		*End*			*Middle*

Langkah 3 *Begin* <= *End* dan gagal

Middle (5 + 5) DIV 2 = 5

List[5] = 27, berhasil = True

Tabel 4. 6 Langkah 3

8	13	17	23	27	31	35	39
[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]
				Middle			

Tiap langkah pengulangan adalah separuh dari jumlah list paling kiri yang dicari. Di kemungkinan terburuk, tidak diperlukan terlalu lama melihat semua elemen list, hanya Log2N elemen.

Ini jauh berguna dibanding pencarian sekuensial, pada list besar. Contohnya, jika list asli memiliki 2048 catatan, pencarian sekuensial perlu kira-kira 1024 pembanding, lalu 2048 pembanding fungsinya mengambil catatan akhir dalam list. Untuk daftar yang sama, hanya diperlukan pencarian biner, dalam kasus terburuk 10 pembanding.

Pencarian biner tidak dapat menjamin pencarian lebih cepat dari daftar yang lebih kecil. Perhatikan bahwa mencari melalui sistem biner biasanya membutuhkan sedikit pembanding, dan setiap pembanding membutuhkan banyak perhitungan. Ketika N sangat kecil, rasionya masih bisa dikontrol.

Contoh program penelusuran biner menggunakan bahasa C

Pada program berikut kita memiliki array dengan index dan value sebagai berikut :

array[0] = 2;

array[1] = 3;

array[2] = 1;

array[3] = 5;

array[4] = 4;

array[5] = 6;

Selanjutnya kita akan melakukan penyusunan value dari array di atas agar tersusun secara ascending (pengurutan dari kecil ke besar), pada contoh program berikut menggunakan algoritma bubble sort untuk penyusunannya, sehingga menghasilkan susunan array sebagai berikut :

array[0] = 1;

array[1] = 2;

array[2] = 3;

array[3] = 4;

array[4] = 5;

array[5] = 6;

Kemudian kita akan memulai algoritma binary search (pencarian biner), untuk itu kita perlu menentukan 4 hal yaitu, left, right, mid, dan key.

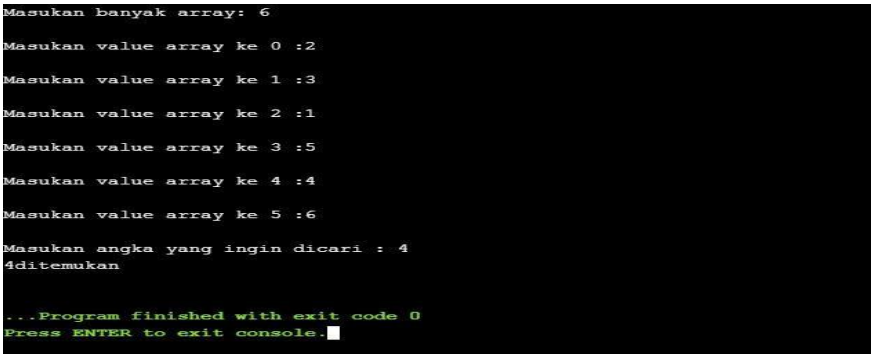
Left digunakan sebagai batas bawah dari pencaran, kemudian right digunakan sebagai batas atas dari pencarian, dan mid akan otomatis ditentukan dari

panjang array yang ada, sedangkan key (kunci) merupakan nilai yang akan kita cari.

```
#include <iostream>

using namespace std;

int binary_search(int a[], int l, int r, int key) {
    while (l <= r) {
        int m = l + (r - l) / 2;
        if (key == a[m])
        return m;
        else if (key < a[m])
                r = m - 1;
                else
                l = m + 1;
                }
                return -1;
                }
        int *bubble_sort(int a[],int n){
                int param;
                int y = n-2;
                while(y >= 0){
                        int index = 0;
                        while(index <= y){
                        if(a[index] > a[index+1]){
                                param = a[index];
                                a[index] = a[index+1];
                                a[index+1] = param;
                        }
                        index++;
                }
                y--;
```

```
        }
        return a;
}
    int main(int argc, char const* argv[]) {
    int n, key;
    cout << "Masukan banyak array: ";
    cin >> n;
    int* a = new int[n];
    for (int i = 0; i < n; i++) {
        cout<<endl<<"Masukan value array ke "<<i<<" :";
        cin >> a[i];
        }
        cout <<endl<< "Masukan angka yang ingin dicari : ";
        cin >> key;
        a = bubble_sort(a,n);
         int res = binary_search(a, 0, n - 1, key);
        if (res != -1)
                std::cout << key << "ditemukan " << endl;
        else
                std::cout << key << "tidak ditemukan" << endl;
        return 0;
}
```

Hasil tampilan program :



Gambar 4. 6 Tampilan Program

C. Soal Latihan

1. Faktor apa saja yang perlu ditimbang dalam memilih berapa banyak generasi suatu file yang harus tetap dipertahankan?
2. Sebutkan, dan jelaskan jenis-jenis record dalam pembuatan berkas laporan sekuensial
3. Apa yang anda ketahui tentang penelusuran sekuensial, jelaskan dan tuliskan algoritma penelusuran sekuensial.

D. Referensi

Arnab Bhattacharya (2015). Fundamentals of Database Indexing and Searching.
Narasimha Karumanchi (2017). Data Structures and Algorithms Made Easy
Paul Deitel, Harvey Deitel. (2016). C how to program : with an introduction to C++

PERTEMUAN 5
ORGANISASI BERKAS LANGSUNG

A. Tujuan Pembelajaran

1. Memahami konsep dasar organisasi berkas langsung
2. Memahami macam – macam metode hashing
3. Mengerti tentang kunci sebagai alamat rekaman yang unik

B. Uraian Materi

Metode pencarian interpolasi atau biner masih tidak dianggap memenuhi kebutuhan manusia terhadap pemberi informasi yang baik. Pemakaian teknologi oleh manusia pada biasanya dan lebih tepatnya teknologi informasi dirasa bahwa waktu yang diperlukan untuk mendapatkan sebuah informasi masih sangat lambat.

Peningkatan teknik pencarian kembali data yang tersimpan harus seimbang dengan peningkatan alat record data untuk kemampuan atau menampung jumlah data yang sangat besar.

1. Kunci Sebagai Alamat Rekaman Yang Unik

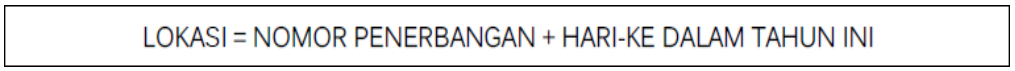
Amat diharapkan supaya proses langsung menuju ke alamat tempat record menggunakan kunci tertentu disimpan. Untuk memperoleh record yang digabungkan dengan suatu kunci primer.

Hal tersebut dimungkinksn terjadi bila salah satu kunci record juga ialah suatu alamat lokasi rekaman. Untuk aplikasi record yang di dalamnya terdapat NIP atau nomor induk siswa, ada 10 digit (kunci rekaman yang merupakan pengkumpulan 6 digit tahun angkatan + 2 digit kode jurusan + 2 digit nomor urut), untuk memperoleh waktu pencarian akan lebih baik, yaitu 1 probe untuk setiap rekaman yang dicari. maka diperlukan lokasi sebanyak 99.999.999.

Akan tetapi, teknik tersebut mempunyai kekurangan karena di butuhkan ruang yang sangat besar untuk dapat menambpung semua record. Untuk menampung data dari NIP siswa di perlukan ruang yang besar. Meskipun jumlah dari siswa mungkin hanya 600, pasti ada kurang lebih nomor yang tidak terisi karena ada beberapa alasan, contohnya sudah lulus, mengundurkan diri, cuti, dan lain-lain. dengan demikian tidak semua ruang digunakan.

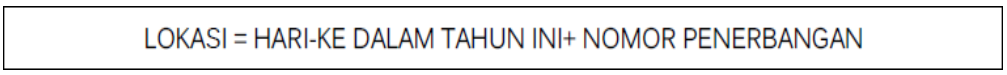
Konversi Kunci Rekaman Menjadi Satu Alamat Yang Unik

Contoh yang sering digunakan untuk menggambarkan konversi catatan kunci adalah sistem pemesanan tiket. Jika nomor penerbangan maskapai adalah dari 1 hingga 1000, dan Anda ingin memantau reservasi penerbangan tahunan dengan 1 hingga 366 hari (1 tahun), Anda dapat menautkan nomor penerbangan dan tanggal kedatangan untuk mendapatkan catatan lokasi yang berisi data reservasi penerbangan untuk tanggal tertentu.



Gambar 5. 1 Nomor Penerbangan

Tanda + merupakan hubungan, maka untuk menyediakan semua kemungkinan penerbangan dibutuhkan ruang alamat sebesar 8888666 unit. Hasilnya tersebut dapat diperkecil sampai 30% bila memakai kombinasi:



Gambar 5. 2 Lokasi Hari

Karena kecil kemungkinan dalam satu maskapai terdapat lebih dari 150 penerbangan, maka ruang alamat maksimum yang diperlukan adalah 25588.

Menentukan Alamat Dengan Konversi Kunci

Dibutuhkan satu fungsi untuk meletaklan cakupan nilai kunci yang lebih besar ke dalam cakupan yang lebih kecil nilai dari alamat. Fungsi yang disebut tersebut yaitu fungsi hash.

Hash atau kunci memungkinkan alamat output hasil proses hashing tidak lagi alamat yang unik, tetapi kemungkinan alamat bagi kunci yang di hash.

Alamat untuk melokasikan alamat yang didapat dari fungsi hash ialah home-address bagi record tersebut. Tidak terbatas akan bentuk fungsi yang akan melokasikan kunci ke cakupan alamat, tetapi agar supaya fungsi tersebut membentuk kemungkinan alamat yang:

1. Dapat untuk mengurangi terbentuknya kelompok. pengkelompokan terjadi bila hasil hashing 2 kunci record yang berbeda menuju ke alamat yang sama. Dan juga dapat untuk membagikan kunci secara merata ke dalam cakupan alamat.

2. Dapat dilakukan dengan maksimal. Hal tersebut bertujuan agar waktu pembacaan dapat ditekan sebaik mungkin.

Maka fungsi hashing terdapat 2 aspek, ialah:

1. Fungsi hash itu sendiri

2. Metode untuk menyatakan kolisi

Resolusi kolusi dibutuhkan agar rekaman yang dirubah ke suatu lokasi yang memiliki kapasitas lebih besar. Di ketahui tulisan di atas menerangkan bahwa satu lokasi memiliki kapasitas satu rekaman.

2. Metode Hashing

Sistem Berkas langsung memiki kelebihan yang maksimal, akan tetapi juga memiliki kekurangan.

Untuk menghilangkan kekurangannya itu maka digunakanlah metode lain yang di sebut metode hashing.

Dipergunakannya metode Hashing intinya ialah agar menurangi banyaknya ruang alamat yang dipakai dan untuk mengubah kunci yang mempunyai cangkupan yang besar ke nilai alamat yang mempunyai cangkupan yang kecil.

Skema Hashing

Ide utama di balik hashing adalah memasukkan setiap kunci yang dimuat ke dalam tabel ke dalam pohonnya sendiri dengan cara yang cerdas setiap pohon menerima sekitar jumlah kunci yang sama terlepas dari nilai-nilai mereka dan tanpa pengetahuan tentang distribusi aktual nilai-nilai ini.

Jika itu mungkin, kita akan memiliki kunci N / H di setiap pohon. Misalkan kita memiliki 2 ** 20 (sekitar 1 juta) kunci untuk memuat. Jika kita dimuat mereka dalam satu pohon (HASHEXP:0, H = 1), menggunakan pencarian pohon biner untuk menemukan atau menolak kunci pencarian akan memerlukan 21 perbandingan antara kunci pencarian dan beberapa kunci di pohon. Namun, jika kunci seragam dimuat ke pohon H = 1024 (HASHEXP:10), masing-masing akan berisi hanya sekitar 1024 kunci. Diberi kunci pencarian, kita akan tahu jumlah ember di mana ia berada. Mencari di antara 1024 kunci di pohon itu melalui pencarian biner kemudian akan memerlukan hanya 11 perbandingan kunci untuk menemukan atau menolak kunci pencarian; yaitu, kecepatan pencarian akan praktis dua kali lipat.

Fungsi Hashing Strategi membagi dan menaklukkan ini tidak sulit untuk dibayangkan. Tapi bagaimana kita memastikan bahwa setiap ember menerima kira-kira jumlah kunci input yang sama jika kita tidak tahu apa-apa tentang nilai-nilai mereka priori? Jika kunci input kami adalah, katakanlah, angka alami dari KEY = 1 ke KEY = 1024 dan kami memiliki 8 pohon untuk diisi, kami akan cukup bagi kunci dalam 8 rentang yang sama, dengan tepat 4 kunci per pohon. Untungnya, ada transformasi tertentu yang disebut fungsi hash. Fungsi hash yang baik memiliki tiga sifat dasar, yaitu :

1) Hal ini dapat menerima kunci sewenang-wenang N dengan nilai-nilai sewenang-wenang sebagai argumen dan mengembalikan nomor alami TN (nomor pohon) dari 1 hingga H: TN = hash_function (KUNCI).

2) Setiap nomor TN dari 1 hingga H akan diberikan ke sekitar tombol N/H, dengan sangat sedikit atau tidak ada variasi antara nomor TN yang berbeda.

3) Untuk setiap nilai kunci yang diberikan, dapat mengembalikan satu dan hanya satu nilai TN. Dengan kata lain, akan ada tidak ada situasi ketika kunci yang sama ditetapkan ke dua pohon yang berbeda.

4) Hal ini cukup cepat untuk menghitung. Memang, jika sangat lambat untuk menghitung dan karenanya mengambil waktu yang sangat lama untuk menemukan pohon mana kunci milik, itu akan mengalahkan tujuan pencarian cepat dan efisien di tempat pertama.

Untuk melihat bagaimana prestasi seperti itu dapat ditarik, mari kita membentuk kunci komposit dari sepasang variabel (Team_SK, Player_ID) dari data set Bizarro.Player_candidates. Sekarang mari kita pasang semua 10.000 nilai yang berbeda ke dalam koktail dari fungsi bersarang yang digunakan sebagai fungsi hash di bawah ini untuk mendistribusikan nilai TN yang dihasilkan ke dalam angka dari 1 sampai 8. Key-indexed array Freq di bawah ini digunakan untuk mendapatkan frekuensi pada jumlah TN nilai ditempatkan ke dalam setiap ember.

Alasan kunci didistribusikan begitu merata adalah bahwa fungsi MD5 yang disediakan oleh SAS adalah hash fungsi itu sendiri. Di atas, mengkonsumsi kunci komposit (Team_SK, Player_ID) diubah menjadi karakter string dengan fungsi CATS dan menghasilkan string karakter 16 byte (yang disebut tanda tangan). PERINGKAT fungsi mengembalikan posisi byte pertama tanda tangan dalam urutan pemeriksaan. Akhirnya, MOD menggunakan pembagi 8 untuk mendistribusikan nomor posisi (mulai dari 0 hingga 255) antara nilai tn dari 1 sampai 8. Meskipun ada sekitar 4 persen variabilitas antara bucket yang paling banyak dan paling sedikit diisi, untuk semua maksud praktis dan tujuan menggunakan pencarian pohon biner dalam salah satu ember ini akan sama cepatnya. Berbeda dengan biner-mencari semua 10.000 kunci, itu akan menghemat sekitar 3 perbandingan kunci per pencarian biner sepenuhnya Ember.

Karena membandingkan kunci biasanya merupakan bagian yang paling komputasi pajak dari algoritma pencarian, mendistribusikan kunci di antara pohon-pohon mungkin membenarkan overhead komputasi TN dengan menggunakan hash fungsi di atas. Ekspresi yang diberikan di atas hanyalah salah satu contoh dari fungsi hash yang layak digunakan hanya untuk menggambarkan ide. Ia bekerja dengan baik karena MD5 itu sendiri adalah fungsi hash.

Meskipun fungsi hash internal bekerja untuk hash objek di balik layar berbeda, secara konseptual melayani tujuan yang sama untuk mendistribusikan kunci secara merata di seluruh jumlah pohon AVL yang dialokasikan.

Fungsi-fungsi Hash yang Sering Digunakan Dalam Hashing

Hashing dengan Kunci Modulus N

Fungsi hash yang paling sering digunakan adalah fungsi modulus N.

f (kunci) = kunci mod N

Gambar 5. 3 Fungsi Modulus N

Simbol N adalah sebuah ukuran suatu tabel. sisa pembagian kunci N adalah Hasil fungsi modulus. Salah satu contohnya, untuk N=12 maka: 30 mod N = 6 40 mod N = 4 Keuntungan fungsi ini hanya menghasilkan nilai dalam rentang ruang alamat (0) sampai dengan (N-1)

Hashing dengan Kunci Modulus P

Fungsi selanjutnya ialah fungsi hashing modulus P adalah fungsi variasi dari fungsi modulus N, rumusnya yaitu :

f (kunci) = kunci mod P

Gambar 5. 4 Fungsi Modulus P

P sebagai bilangan prima terkecil yang lebih besar atau sama dengan N. dan N adalah ukuran tabel. P ini kemudian menjadi ukuran tabel baru yang menggantikan N. Contoh: untuk N=12 maka P=13 30 mod P = 4 40 mod P = 1

Hashing dengan Pemotongan

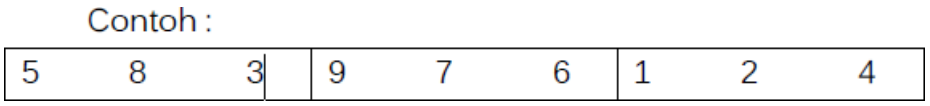
Fungsi lain dari metode hashing ialah pemotongan. Sebagai contoh yaitu Mahasiswa di suatu Universitas. Para Mahasiswa mempunyai NIM (Nomor Induk Mahasiswa) mempunyai 10 digit, terdiri dari 6 digit tahun angkatan dan 4 digit nomor urutnya.

Jika ingin dijadikan 8 digit saja, maka dapat diberlakukan pemotongan jumlah dari digit. Pemotongan dapat dilakukan pada bagian mana saja, tentunya dengan konsekuensi masing-masing.

Pada NIM, jika dilakukan pemotongan pada bagian belakang NIM, akan didapat 6 digit yang sama (tahun angkata) sehingga kemudian kolisinya akan lebih besar jika pemotongannya berada di bagian depan NIM.

Hashing dengan Lipatan

Hashing dengan lipatan ialah fungsi hashing yang dapat ditentukan dengan melihat kondisi digit bagian awal dan digit yang di hasilkan. Contohnya 9 digit NIM akan di perkecil menjadi 3 digit, maka digit bagian awal dibagi dengan 3, lalu dilipat pada batas antar bagian.



Gambar 5. 5 *Hashing* dengan Lipatan

Garis diatas adalah batasan yang dimana akan dilakukannya lipatan.

Jumlah dari susunan tersebut ialah::

385

976

421

.....+

Jika penjumlahan dilakukan dengan mengabaikan carry, hasil yang di dapat yaitu 672

Lalu jika tidak mengabaikan carry maka hasil yang didapat yaitu 782.

Hashing dengan Penggeseran

Hashing penggeseran mempunyai proses yang hampir sama dengan fungsi hashing lipatan, hanya saja perbedaannya yaitu sesudah ditentukan batasannya. Digit asli dipotong kemudian digeser dan di jumlahkan.

583

976

124

.....+

Yang didapat yaitu 573 (tanpa ada carry). Lalu jika seluruh hasil penjumlahan tersebut dijumlahkan dengan menggunakan carry dan hanya 3 digit di bagian akhir saja yang dipakai, maka hasil yang didapat 683.

Hashing dengan Pengkuadratan

Fungsi hashing dengan penguadratan yaitu fungsi hashing dilakukan dengan menguadratkan kunci, untuk mendapatkan alamat yang diperbolehkan dari hasil penguadratan ini kemudian dapat dilakukan penggabungan dengan pemotongan dan lipatan. Contohnya kunci 782 akan menghasilkan alamat 117. F(782) =117.

Hashing dengan Konversi Radix

Dalam konversi radix, kunci dianggap dalam base selain 10 yang kemudian dikonversi ke dalam basis 10, misalkan kunci 5 6 7 8 didalam base 13 akan di

dapat 12098, diperoleh dari: 5 6 7 8, Posisi: 3 2 1 0 Hasil tersebut masih dapat digabungkan dengan fungsi hash lain seperti fungsi pemotongan atau lipatan untuk mendapatkan digit alamat yang diinginkan.

Struktur file akses langsung dibuat dengan tujuan untuk kebutuhan akses data secara langsung (Direct). Direct Access kadang kadang disebut dengan Random Access untuk menekankan bahwa dalam faktanya permintaan record itu sendiri tidak dalam urutan khusus.

Dalam struktur file akses langsung timp record disimpan pada suatu lokasi yang dihitung dari nilai kunci utamanya. Semua pro gram dengan penunjuk file harus mengetahui "Algoritma transform- masi yang nantinya digunakan untuk menghitung alamat awal penyimpanan dan semut penunjuk alamat untuk record.

Algoritma Transformasi menempatkan tiap kunci utama dan mengkonversinya menjadi alamat record relatif Relative Record Address (RRA). RRA adalah nomor antara 1-M, dimana M adalah nomor lokasi penyimpanan dalam file.

File akses langsung biasanya memerlukan ruang penyimpanan lebih daripada nomor data record itu sendiri karena:

1) Banyak sekelompok kunci utama bukan merupakan nomor se quensial dari 1-N, dimana N = Nomor data record 2.

2) Tidak ada Algoritma Transformasi yang bisa membuat mereka menjadi suatu himpunan rangkaian.

Hal ini menunjukkan bahwa file akses langsung memerlukan lebih daerah penyimpanan daripada file serial atau sequential untuk menyimpan data yang sama.

Masalah lain dari proses Algoritma Transformasi adalah bisa menghasilkannya RRA yang sama dari 2 nilai kunci utama. Record record dengan satu RRA yang sama disebut Synonyms, dan pe nyimpanannya memerlukan daerah Overflow, daerah yang kosong (free) untuk menyimpan record-record yang baru dan/atau yang synonyms.

Untuk mengakses file langsung, pemakai program harus mem berikan nilai kunci utama untuk record yang diinginkan bagi algo ritma transformasi. Dengan catatan suatu file akses langsung juga diakses melalui suatu rutin serial-access. Rutin ini harus menguji untuk record-record yang kosong dimana record dengan data yang tidak valid.

File-file akses langsung berguna hanya jika ketika salah satu record pada saat digunakan untuk pemrosesan dan ketika nilai kunci diketahui. Sistem Pengolahan secara On-Line seperti Sistem Pelayan an Rekening Bank, biasanya memakai tipe struktur ini. Untuk me menulis laporan dari file akses langsung harus diproses dengan cara yang sama seperti file Serial.

3. Metode Akses pada File Akses Langsung (Direct Access File)

Alternatif penelusuran secara sekuensial suatu file untuk satu record pada mekanisme perolehan kembali record dikenal sebagai akses secara langsung.

Kita bisa mengakses record secara langsung satu record pada saat pencarian secara langsung di awal record dan membacanya. Dimana bila penelusuran secara sekuensial memerlu kan operasi sebanyak O(n), sedangkan untuk akses secara langsung hanya memerlukan operasi sebesar 0(1). Tidak menjadi soal berapa besar filenya, kita masih bisa mendapatkan record yang diinginkan dengan pencarian tunggal.

Permasalahan utama akses langsung diketahui di permulaan kebutuhan record adalah kadang-kadang informasi tentang lokasi record berada di file indeks yang terpisah, tetapi sesaat kita meng. asumsikan bahwa kita tidak memiliki suatu indeks. Dengan meng asumsikan bahwa kita mengetahui nomor relatif record (RRN) dari record yang kita inginkan.

RRN merupakan konsep penting yang timbul dari pandangan file sebagai sekumpulan record daripada se bagai sekumpulan byte. Jika suatu file merupakan serangkaian record, RRN dari record memberikan posisi relatifnya ke awal file. Record pertama memiliki RRN 0, berikutnya RRN 1, dan seterusnya.

Di nama dan alamat file, kita bisa menghubungkan satu record dengan ke RRN-nya dengan menandai nomor keanggotaan yang di relasikan ke urutan dimana kita memasukkan record dalam file. Seseorang dengan record pertama memiliki nomor keanggotaan nomor 1001, nomor kedua 1002, dan seterusnya. Dengan memberikan nomor anggota, kita bisa mengurangi dengan 1001 untuk mendapat kan RRN suatu record.

RRN memberitahukan kepada kita posisi relatif record. Untuk mendukung akses langsung, kita harus bekerja dengan pemakaian record dengan panjang tetap. Jika semua record panjangnya sama, maka bisa menggunakan RRN record untuk menghitung byte offset di awal record relatif ke awal file. Contoh, jika menginginkan record dengan RRN 546 dan file yang kita memiliki ukuran panjang record tetap 128 byte per recordnya, maka byte offset bisa dihitung :

Byte Offset = 546 x 128 = 69 888

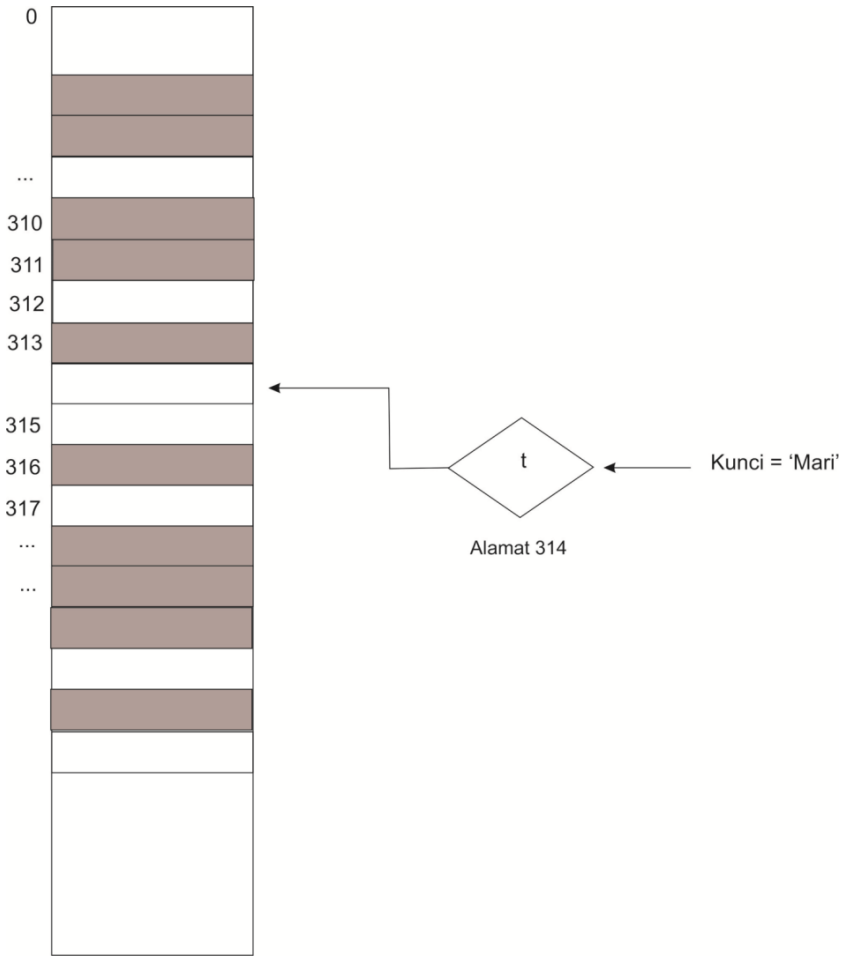
Bentuk umum record file dengan panjang tetap, ukuran record r, byte offset satu record dengan RRN n adalah :

Byte offset = nxr

Metode file dengan akses langsung mentransformasikan kunci dengan menggunakan algoritma perhitungan sebelum digunakan sebagai suatu Address. Metode ini cepat selama bisa menghindari operasi file lanjutan, tetapi kekuatan dari metode ini adalah data di alokasikan sesuai dengan satu kunci attribut tunggal.

Bagaimanapun juga record dalam file langsung tidak di link ke record sebelumnya ataupun ke sesudahnya.

Direct File menggunakan ruang ekstra di file utama (m>n), untuk menjaga memberikan beberapa record ruang kosong di urutan ketika akan menyisipkan record baru ke file.



Gambar 5. 6 Metode Akses pada File Akses Langsung

4. Transformasi Kunci ke Address

Solusi permasalahan dengan menggunakan Key sebagai Alamat file yang meliputi pemakaian prosedur perhitungan yang menterjemahkan Key sebagai Alamat file yang meliputi pemakaian prosedur perhitungan yang menterjemahkan nilai Key-Attribute ke dalam "Relative Addres" dalam ruang file.

Setiap calon Relative Address dikenali sebagai 1 slot dimana Batu record bisa ditempatkan.

Prosedur yang berbeda digunakan untuk menghitung alamat relatif dengan mengenali kunci seseorang dalam file karyawan dan dari kunci yang sama dalam file Sales-Personel. Prosedur ini ter gantung pada tipe Kunci. Perhitungan ini disebut Transformasi Key- to-Address

Dua kategori perhitungan ini bisa berbeda. Ketetapan prosedur yang menterjemahkan identifikasi field ke bentuk alamat unik dan teknik pengacakan (Randomizing) yang menterjemahkan Kunci ke Alamat.

Prosedur Deterministik

Didapat dari himpunan semua nilai kunci dan menghitungnya menjadi sekumpulan alamat relatif yang khusus.

Transformasi Randomizing

Menterjemahkan Nilai-Kini menjadi Alamat Relatif mengguna kan prosedur "Algoritma Randomizing"

Contoh : Transformasi Key-To-Address

Satu transformasi randomizing untuk satu file personel meng gunakan nomor sosial_sekurity sebagai Key. Asumsi nilai dari digit orde rendah nomor ini didistribusikan dengan rata dan karenanya memungkinkan untuk memperoleh 3 digit nomor unik untuk setiap karyawan. Jika salah satu menginginkan ruang yang mungkin untuk 500 karyawan, nilai Kunci bisa dibagi dengan 500, menghasilkan sisa dengan nilai antara 0 sd 499.

Alamat yang sama bisa terjadi :

Ali	Atau	322-45-6178	memberikan address	178
Joe		123-45-6284		284
Marry		036-23-0373		373
Pete		901-23-4784		284

Gambar 5. 7 Transformasi *Key-to-Address*

Disini Joe dan Pete keduanya ditandai untuk satu record dengan nomor sama = 284 Record untuk Joe dan Pete akan bertubrukan jika keduanya di tempatkan secara langsung dalam satu file.

5. Hashing

Penentuan alamat suatu record pada file data dengan me lakukan perhitungan pada file data dengan melakukan perhitungan pada Key-Value dari record tersebut. Yang ditulis sebagai :

Gambar 5. 8 Perhitungan Fungsi Transformasi

Dalam transformasi Key Value ke Address record, bisa timbul kemungkinan bahwa 2 record dengan Key-value yang berbeda di peta kan pada satu alamat record yang sama. Hal ini disebut dengan collision (Kolisi), yaitu :

H(K1)=HK(2) dimana K1<>k2

Untuk mengatasi Kolisi adalah:

1) Linear Search

   Dengan cara mencari tempat kosong terdekat dari alamat record yang mengalami kolisi.

2) Rehashing

   Tersedia lebih dari satu fungsi Hash. Jika terjadi Kolisi, per hitungan dengan memakai fungsi Hash yang lain sampai tidak terjadi Kolisi lagi.

3) Area Overflow

   Record yang mengalami kolisi ditempatkan pada aren Overflow Untuk menghubungkan file data dengan area Overflow diguna kan pointer.

Kriteria pemilihan fungsi transformasi :

1) Membutuhkan ruang seminimal mungkin

2) Menghindari tabrakan (Kolisi)

Jadi pada Direct File memungkinkan untuk mengakses sebuah record secara langsung dengan menggunakan Key Attribute tanpa mem pertimbangkan posisi Key tersebut dalam urutannya di keseluruhan file.

6. Ringkasan

1) Pengaksesan paling cepat pada file adalah dengan mengetahui Alamat dari record yang akan diaksses atau diambil.

2) Untuk mencapai pengalamatan secara langsung kunci dari record digunakan untuk melokasikan record dalam file.Direct Access diagram bisa dilihat digambar 13-1

3) Metode Direct File menggunakan perhitungan t untuk menyedia kan alamat record dengan satu Kunci.

4) Pemrosesan file secara langsung, atau akses langsung, me mungkinkan komputer secara langsung menempatkan record record yang diinginkan dengan menggunakan satu kunci record.

5) Pemrosesan secara langsung memerlukan penyimpanan disk, di mana devise disknya disebut devise penyimpanan akses langsung (direct-access storage device/DASD) karena komputer bisa secara langsung menempatkan record pada disk.

6) Keuntungan tambahan dari akses langsung adalah kemampuan untuk membaca, mengubah, dan menghasilkan satu record pada tempat yang sama di disk, dimana hal ini disebut sebagai pe remajaan dalam tempat.

C. Soal Latihan

1. Apa pengertian dari organisasi berkas langsung?
2. Apa itu Fungsi Hash pada organisasi berkas langsung?
3. Apa yang di maksud kunci sebagai alat rekaman yang unik?

D. Referensi

Handayani, Dewi. 2001. Sistem Berkas. Yogyakarta. J&J Learning Alan l, Tharp-john Willey &Son. 1988. File Organization and Processing.

PERTEMUAN 6
ORGANISASI BERKAS BANYAK KUNCI

A. Tujuan Pembelajaran

1. Setelah mempelajari materi ini, mahasiswa memahami dan dapat menjelaskan organisasi berkas banyak kunci.

B. Uraian Materi

1. Organisasi Berkas Banyak Kunci

Organisasi file multi-kunci merupakan organisasi file yang memungkinkan beberapa bidang kunci untuk mengakses catatan. Inilah yang disebut organisasi file multi-kunci.

Sekarang banyak terdapat organisasi yang memiliki file, dimana file tersebut dapat diakses dengan banyak cara, tiap-tiap organisasi memiliki kunci yang berbeda. Teknik organisasi ini adalah bagian utama dalam implementasi database.

Hal tersebut terkait dengan sifat dari perangkat media penyimpan yang sedemikian rupa, sehingga catatan data yang disimpan dalam file tertentu perlu diberikan pengenal. pengidentifikasi, atau kunci yang terdiri dari beberapa kombinasi karakteristik record.

Contohnya jika terdapat sebuah isi file record dari bidang tertentu, misalnya record “Hazmi”. Dengan demikian, dalam file personil, record “Hazmi” dapat diberikan kunci "19 "(jika itu adalah record kesembilan belas dalam file), atau "01230402 " (jika ada di perangkat 01, silinder 23, jalur 04, record kedua), atau "15216 " (jika nomor karyawan Hazmi adalah 15216).

Terlepas dari metode yang digunakan untuk menetapkan kunci, Namun, Semua kunci untuk record di file siapa pun harus unik, yaitu tidak ada dua record yang memiliki kunci bernilai sama. Alasan terkait hal ini karena record tidak dapat berbagi alamat fisik yang sama karena dua record tidak dapat berbagi ruang fisik yang sama, dan dua record yang berbagi kunci yang sama akan menyebabkan ambiguitas, yaitu, perintah yang berisi kunci non-unik tidak akan cukup untuk membuat perangkat keras menentukan yang mana dari beberapa record yang berisi kunci itu harus diakses.

Ada cukup jumlah cara yang bisa digunakan dalam organisasi berkas multi-kunci. Hampir seluruh teknik yang digunakan tergantung di pembuatanindeks yang dapat langsung mengakses nilai kunci.

Ada dua teknologi dasar yang menyediakan hubungan antara indeks dan catatan data di dalam file, sebagai berikut :

1) Terbalik

2) Daftar multi

Banyak program kompilator bahasa tidak mendukung organisasi file multi-kunci. Untuk tujuan ini, dibutuhkan paket tambahan yang dapat mendukung sistem pemrosesan manajemen data yang hampir sepenuhnya diatur menggunakan beberapa file kunci..

Banyak sistem informasi interaktif membutuhkan dukungan dari beberapa organisasi file kunci. Contohnya : Sistem dengan banyak pengguna di bank, seperti pelanggan, teller, petugas kredit, karyawan bank, manajer cabang, dan lainnya. Semua pengguna ini perlu menggunakan format catatan yang sama seperti bawah ini untuk akses data.

ID	Nama		Kode Kelompok		No. Social Society	Saldo	Overdraw Limit
	Depan	Akhir	Cabang	Type			

Gambar 6. 1 Format Catatan

Adanya pengguna yang berbeda membutuhkan sistem untuk mengakses catatan dengan cara yang berbeda. Misalnya, teller akan menggunakan ID untuk mengidentifikasi akun catatan, dan kemudian pemberi pinjaman perlu mengakses semua catatan berdasarkan nilai batas cerukan, atau mengakses semua akun catatan berdasarkan nilai No. Kemudian, manajer cabang dari asosiasi sosial akan membuat laporan rutin untuk semua akun catatan yang diklasifikasikan menggunakan data ID, dan pelanggan perlu mengakses catatan mereka sendiri dengan memberikan data ID yang mereka miliki (atau kombinasi dari nama, nomor dan jenis sosial).

Tabel berikut menunjukan perbedaan kebutuhan akses record tiap pengguna dalam cara yang berbeda.

Tabel 6. 1 Tabel Kebutuhan Akses

Nasabah	Memerlukan akses terhadap record account-nya sendiri berdasarkan data ID, ataupun kombinasi Nama, No.SS, dan Type
Teller	Memerlukan akses untuk mengidentifikasi record dari account yang ada, berdasarkan nilai ID
Petugas Kredit	Perlu mengakses semua catatan yang ada berdasarkan nilai "batas penarikan"
Pegawai Bank	Memerlukan akses untuk membuat laporan rutin yang disortir bedasarkan nilai ID
Manajer Bank	Memerlukan akses seluruh record berdasarkan nilai Kode Group

Dengan memiliki banyak file yang berbeda dapat mendukung semua jenis teknologi akses. Setiap file diatur untuk suatu tujuan, Sistem bank pada paragraf sebelumnya adalah contoh yang harus dimiliki:

1) Atur akun yang diarsipkan dengan nilai kunci ID untuk melayani pengguna, yaitu pelanggan, teller, karyawan bank, dan pelanggan.

2) File account dengan organisasi sekuensial yang mempunyai pengurutan record berdasarkan batas penarikan, agar pengguna mendapat pelayanan, yaitu petugas kredit.

3) File account dengan organisasi relatif yang mempunyai kunci No.Soc, untuk melayani pengguna yaitu, pegawai bank.

4) Berkas akun dengan organisasi sekuensial mempunyai record dengan pengurutan berdasarkan kode group, untuk melayani pengguna yaitu, manajer cabang.

5) File account dengan organisasi relatif yang memiliki *key value* Nama, No.SS, dan tipe. Digunakan agar pengguna (nasabah) mendapatkan pelayanan.

Di atas terdapat 5 berkas, seluruhnya memilik kesamaani catatan. Semua berkas tersebut hanya beda pada organisasi dan pengaksesan.

Bukan sebuah cara yang baik menggunakan pengulangan data dari beberapa berkas pengaksesan catatan dengan bergam teknik, dan juga teknik tersebut perlu kapasitas yang cukup banyak pada alat penyimpanan dan lamanya durasi yang diperlukan untuk melakukan pemutakhiran record secara bersamaan.

Pengaturan file dengan banyak kunci adalah cara terbaik untuk menyelesaikan masalah di atas. File tersebut menggunakan indeks, bukan data

perulangan seperti yang dijelaskan di atas. Organisasi berkas dengan banyak kunci merupakan konsep yang cukup banyak Ini dicapai dengan membuat beberapa indeks. Fungsi dari indeks ini adalah untuk memberikan akses yang berbeda ke catatan. Cara ini menggunakan daftar tertaut (*linked-list*).

Pada paragraf di atas terdapat kata linked-list, berikut akan dijelaskan mengenai linked-list.

Linked-List

Linked-list (daftar tertaut) adalah struktur data yang digunakan untuk menyimpan koleksi data. Linked-list memiliki sifat-sifat berikut :

1) Elemen berturut-turut yang dihubungkan oleh pointer.
2) Elemen terakhir menunjuk ke NULL
3) Ukurannya apat semakin besar maupun mengecil selama pelaksanaan program.
4) Dapat dibuat hanya ketika sedang diperlukan

Ada banyak struktur data lain yang melakukan hal yang sama seperti linked-list. Sebelum membahas linked-list penting untuk memahami perbedaan antara linked-list dan array. Kedua linked-list dan array digunakan untuk menyimpan koleksi data, dan karena keduanya digunakan untuk tujuan, kita perlu membedakan penggunaannya. Itu berarti dalam kasus mana array yang cocok dan dalam kasus mana linked-list cocok untuk digunakan.

Keuntungan dari linked-list yaitu, bahwa mereka dapat diperluas dalam waktu yang konstan. Untuk membuat array, kita harus mengalokasikan memori untuk jumlah elemen tertentu. Untuk menambahkan lebih banyak elemen ke array saat penuh, kita perlu membuat array baru dan menyalin array lama ke yang baru. Ini dapat memakan cukup banyak waktu. Kita dapat mencegah hal ini dengan mengalokasikan banyak ruang pada awalnya tapi kemudian kita bisa mengalokasikan lebih dari yang kita butuhkan. Dengan linked-list, kita dapat mulai dengan ruang untuk hanya satu elemen yang dialokasikan dan menambahkan elemen baru dengan mudah tanpa perlu melakukan penyalinan dan mengalokasikannya.

Ada sejumlah kekurangan yang dimiliki linked-list. Kelemahan utama dari linked-list adalah akses waktu untuk elemen individu. Sedangkan Array adalah Random-Access, yang berarti dibutuhkan O (1) untuk mengakses elemen dalam array. Daftar tertaut mengambil O (n) untuk akses ke elemen dalam daftar dalam kasus terburuk.

2. Inverted File

Inverted file atau file terbalik merupakan sebuah kunci pada Indeks terbalik dengan semua nilai kunci, setiap nilai kunci memiliki penunjuk ke rekaman terkait. Inversi sendiri metode dasarnya adalah menyediakan koneksi antara indeks dan catatan data dalam file.

Contoh inverted file (file terbalik) “Account” terhadap No.SS (Nomor Social Society) yang menghasilkan indeks inversi.

Tabel 6. 2 Inverted File

No.SS	Address
741258963	8
789654123	12
589632145	1
554478963	10
563252236	13
445666987	15
322566963	2
353265567	20
545563212	17
332654789	11
585665478	14
522365412	7
256545856	19
544587412	18
554121239	16
252365698	9
125225687	4
526336549	5
554887963	3
522569874	6

Untuk file relatif dengan nilai kunci "ID", indeks terbalik dengan "No.SS" sebagai kunci akan menghasilkan file, yang dapat langsung diakses berdasarkan nilai "ID" atau "No.SS".

Tabel 6. 3 Inverted File 2

No.SS	ID
741258963	55489
789654123	12896
589632145	14785
554478963	10124
563252236	13785
445666987	15563
322566963	21429
353265567	20584
545563212	17445
332654789	11786
585665478	14564
522365412	74563
256545856	19889
544587412	18778
554121239	16566
252365698	97841
125225687	44523
526336549	55642
554887963	33215
522569874	65698

Pengertian Kunci Utama

Kunci Utama (primary key) Adalah nilai yang digunakan dalam database untuk mengidentifikasi baris dalam tabel. Kunci utama juga digunakan dalam struktur penyimpanan data di file.

Struktur Inverted File

Struktur Inverted File memajukan direktori record pada sebuah berkas terangkai, yang mana record memiliki penunjuk terhadap semua subset. Struktur inverted file dapat dilihat pada tabel berikut :

Bagian File Personal		
Alamat Record	Nomor Karyawan	Umur
101	10001	19
102	10002	27
103	10003	29
104	10004	40
105	10005	32

Inverted File dengan Indeks Umur	
Umur	Alamat Record
18-25	101,....
26-30	102, 103,....
31-35	104, 105...

Gambar 6. 2 Struktur Inverted File

Terdapat 2 berkas dalam struktur inverted file ini, direktori dan berkas data. 2 berkas tersebut diterapkan menggunakan struktur terpisah. Berkas data diterpakan sebagai *direct access* (akses langsung), dan direktori diterapkan sebagai *serial access*. Ini mengurangi pemeliharaan direktori saat catatan baru ditambahkan ke berkas. Jika nilai kunci dicari, file direktori dapat berupa file berurutan, atau dapat diindeks dalam urutan nilai besar.

Direktori File

Direktori file adalah sebuah penamaan terhadap sejumlah file, ini adalah cara untuk mengelompokan file sehingga kita dapat mengatur sejumlah file tersebut.

File direktori memiliki organisasi berurutan diindeks, yaitu record dapat diakses secara acak atau secara berurutan. Ada satu file direktori per file data. Ada satu record direktori pada tabel berikut :

Tabel 6. 4 Direktori File

Data Key	Key Number	Pointer to Data Record

Ada satu record direktori pada gambar di bawah untuk setiap Key yang muncul di setiap record data. Dengan demikian, file data yang terdiri dari 1.000 record, masing-masing dengan lima kunci, memerlukan sebuah urutan langsung oleh 5.000 Record data. File direktori tersimpan dalam file kunci.

Panjang field kunci data telah ditetapkan di dalam file direktori tertentu, namun file direktori yang berbeda mungkin memiliki field kunci data yang berbeda panjang. Panjang field kunci data sama dengan panjang kunci terbesar dalam file data yang sesuai. Kunci data yang lebih pendek dari maksimum yang dibiarkan di dalam file direktori dan diisi ke kanan dengan nilai null. Ini akan menjamin urutan yang benar dalam file direktori.

Nomor kunci menunjukkan field kunci dalam rekaman data yang berisi nilai dalam field kunci data terkait. Programmer harus menetapkan nomor kunci untuk setiap field kunci dalam record data. Tombol bernomor dari satu sampai sepuluh disebut secondary key (kunci sekunder), dan setiap nomor tombol nol disebut primary key (kunci utama). Hal ini tidak diperlukan untuk menetapkan primary key, tetapi ada keuntungan yang signifikan yang diperoleh dari penggunaannya.

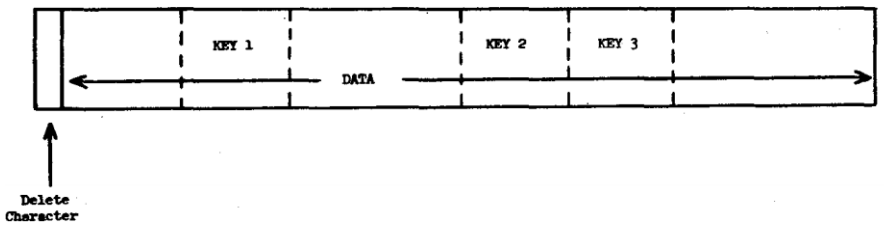
Bidang ketiga di setiap record direktori adalah lokasi fisik relatif dari sebuah data record (dalam file data) yang berisi kunci data tersebut.

Kunci dimana catatan direktori tertentu diketahui perangkat keras atau sistem operasi adalah seluruh direktori record. Bahkan jika dua record data berisi nilai kunci, dan nomor kunci yang sama, alamat fisik mereka yang berbeda menjamin catatan direktori yang unik. Demikina juga jika salah satu data record berisi nilai

yang sama di dua field kunci yang berbeda, keunikan dalam file direktori terjamin dengan nomor kunci yang berbeda yang ditetapkan ke dua field kunci.

Data File (Berkas Data)

File data memiliki organisasi langsung, dan berisi semua data pengguna. Kunci setiap rekaman data adalah lokasi fisik rekaman relatif terhadap awal file, yang memastikan kunci unik untuk setiap record. Kunci yang ditunjukkan pada gambar berikut adalah kunci yang diketahui pengguna dan yang muncul di file direktori, kunci ini bukanlah kunci dimana record data diketahui hardware atau sistem operasi.



Gambar 6. 3 Data Record

File data dapat diakses terlebih dahulu hanya dengan membaca record direktori, dan kemudian mendapatkan pointer ke record data yang sesuai.

Cara Pengaksesan Berkas Terbalik (*Inverted File*)

Metode akses pada inverted file menggunakan indeks yang disebut inverted index atau inverted directory (file yang dibalik). Setiap daftar inverted index, alamat maupun key untuk semua record mempunyai atribut yang sama. Misalnya : Inverted index dapat ponting ke *address* catatan seluruh pegawai yang berusia 18 hingga 25, 26 hingga 30, 31 hingga 35, dan seterusnya.

Untuk melokasikan record untuk berbagai alternatif seperti, usia, jenis kelamin, dan status pernikahan, beberapa file diperlukan. Berkas terbalik ini membuat efisiensi pada saat akses catatan yang mempunyai relasi logika cukup tinggi.

Metode ini memberikan kemudahan yang lebih baik untuk penggunaan satu atau beberapa rekaman atribut dan file pelacakan. Data juga dibutuhkan untuk mendeskripsikan relasi dan alamat catatan yang diidentifikasi oleh berkas terbalik, yang diidentifikasi sebagai data keluaran tambahan. Contoh file inverted bisa dilihat dari struktur pada tabel diatas.

Penggunaan Inverted File

Penggunaan inverted file dapat kita lihat pada contoh berikut :

No.Rec<S-Id, Nama, Divisi...><Pointeer>

1	1234:	Andini:	GA;...	2
2	5768:	Bagas:	TEC;...	3
3	9101	Candra:	DRV;...	0
4	1011	Doni:	TEC;...	0
5	1213	Farah:	GA;...	0
6	.....	.....	.....	.....

DIRECTORY

Search Key	Address List
GA	1,5
TEC	2,4
DRV	3

DATAFILE

No.Rec<S-Id, Nama, Divisi...>

1	1234:	Andini:	GA	:...
2	5768:	Bagas:	TEC	:...
3	9101:	Candra:	DRV	:...
4	1011:	Doni:	TEC	:...
5	1213:	Farah:	GA	:...

Gambar 6. 4 Sekumpulan Tabel Penggunaan Inverted File

Indeks pengalamatan tidak langsung

1) Pembalikan dapat memakai *key* yang tidak memiliki nilai unik, misalnya indeks menurut *key* kode grup.

Kode-Group Tipe-Cabang	ID
DT 0001	111112, 111113, 211014
DT 0002	156253
NW 0001	112321, 208456
NW 0002	302156
NE 0001	306125

Gambar 6. 5 Record dengan Inverted Index berdasarkan nilai Kode Group

Pada gambar di atas menjelaskan penggunaan pengindeksan terbalik berdasarkan kunci Kode-Group.

2) Struktur indeks terbalik memakai *addressing* nilai No.Soc. secara tidak langsung.

No. Society	ID
00589654	111112
00256984	111113
11256365	211014
11456879	156253
22456315	112321
32145689	208456
58963147	302156
55693582	306125
44569856	585696

Gambar 6. 6 Record dengan Inverted Index berdasarkan nilai No.SOC

Pada gambar di atas menjelaskan penggunaan index inversion dengan menggunakan nilai kunci No.Soc berdasarkan pengalamatan tidak langsung.

Merupakan salah satu ciri dari inverted file adalah beberapa pertanyaang yang dapat dijawab tanpa perlu mengakses file data. Misalnya, permintaan berikut ini dapat diproses menggunakan inverted file seperti pada gambara diatas dan diatas.

1) Apakah ada nomor rekening untuk No. Society “22456315”.
2) Berapa banyak rekening berdasarkan Kode-Group “DT0001”.
3) Berapa banyak rekening berdasarkan cabang “NE”.
4) Apakah cabang “NE” memiliki rekening Tipe “0002”.
5) Apakah No.Soc “00256984” memiliki rekening tipe “0002”.

Pertanyaan di atas merupakan pertanyaan-pertanyaan yang dapat dijawab tanpa perlu mengakses file data.

3. File Multi-List

Merupakan sebuah pendekatan yang memiliki indeks untuk tiap key sekunder. File multi-daftar memiliki perbedaan dengan berkas terbalik yang mana pada indeks terbalik, nilai kunci memiliki sebuah pointer data catatan pertama nilai kunci, Dalam indeks multi-daftar, nilai kunci hanya memiliki penunjuk ke rekaman data pertama dengan nilai kunci.

Catatan data memiliki sebuah pointer bagi catatan data berikutnya dengan nilai kunci, begitu juga selanjutnya. Karena ada daftar tertaut dari catatan data bagi tiap nilai kunci sekunder.

Contoh File Data dengan Struktur Multi-List

Alamat Record	ID	Nama	Kode-Group Tipe Cabang	Next	No.Soc	Saldo	Batas Penarikan	Next
1	112113	Aprilia	DT 001	112115	55857896	100.2	0	112147
2	115446	Andini	NW 001	112147	85214569	21.30	100	112115
3	554655	Adelia	DT 002	558456	25896325	585.1	200	0
4	112115	Nadia	DT 001	554213	74125896	563.1	100	445698
5	112147	Fariza	NW 001	554521	74589632	25.23	0	778569
6	778569	Agung	NW 002	0	25639874	25.22	0	554213
7	445698	Bagas	NE 002	963258	58963214	21.1	100	585696
8	554213	Doni	DT 001	478963	41257856	52.32	0	745856

Gambar 6. 7 File dengan Banyak Kunci

Pada gambar berikut menunjukan sebuah multi-list kunci tambahan grup kode. Setiap record data memiliki pointer untuk akses ke catatan selanjutnya.

Kode-Group Tipe Cabang	ID
DT 001	112113
NW 001	115446
DT 002	554655
DT 001	112115
NW 001	112147
NW 002	778569
NE 002	445698
DT 001	554213

Gambar 6. 8 Multi-List dengan Kunci Sekunder Kode Group

Nilai kunci perlu diurutkan, dan struktur indeks adalah tabel dengan pengalamatan tidak langsung, dan memiliki koneksi rekaman data yang diatur oleh ID. Hasil dari struktur multi-daftar adalah kunci sekunder dengan nilai tunggal atau unik.

Dari hal tersebut didapati bahwa terpada N catatan data, ada N nilai kunci tambahan dalam indeks yang menunjukkan catatan pertama.

Teknik multi-list memberikan kemampuan jenis akses sama dengan teknologi file terbalik, tetapi dapat menangani 2 jenis file berbeda. Contohnya sebagai berikut :

1) Berapa banyak jumlah account dengan Kode Group “NE002”.

2) Tampilkan nilai ID untuk account ID dengan Kode Group “NW002”.

3) Tampilkan nilai ID untuk account dengan Type “001”.

C. Soal Latihan

1. Jelaskan pengertian organisasi dengan banyak kunci?
2. Susunlah sebuah sistem pemberian kode dengan ruang seminimal mungkin untuk informasi di bawah ini :
   a. Jumlah karyawan di PT. UNPAM yang memberikan informasi tentang pendidikannya yaitu, tahun masuk perguruan tinggi, fakultas, dan program studi. Berapakah jumlah maksimal file yang harus tersedia?
   b. Jumlah spare part komputer di sebuah tempat servis komputer yang mencatumkan jenis motherboard, tipe processor, dan tipe RAM yang dapat disupport.
   c. Catatan kependudukan yang mencantumkan, tempat tanggal lahir, nama, dan pekerjaan.
3. Apa yang dimaksud dengan inverted list, dan multi list?
4. Metode apa yang memberikan fasilitas yang lebih besar dalam penelusuran suatu file untuk record yang menggunakan 1 attribut maupun lebih?
   Terdapat sebuah arsip sebagai berikut :

No. Rec	NIK	No. Society	Nama	Divisi
1	87.556.454	4567895	Andini	Sales
2	78.221.321	1236528	Aprilia	Technical
3	54.663.215	5896325	Bagas	Admin
4	55.321.632	3569654	Fariza	Technical
5	21.253.326	2541236	Nita	Sales

   Jelaskan bagaimana agar arsip di atas dapat dibentuk menjadi file langsung.

D. Referensi

Dewi Handayani U.N. (2001). Sistem Berkas.
Wladston Ferreira Filho (2017). Computer Science Distilled.
Narasimha Karumanchi (2017). Data Structures and Algorithms Made Easy.

PERTEMUAN 7
ORGANISASI BERKAS DENGAN BANYAK KUNCI

A. Tujuan Pembelajaran

1. Mahasiswa dapat menjelaskan tentang Organisasi berkas dengan banyak kunci
2. Mahasiswa dapat menjelaskan tentang berkas terbalik (*inverted file)*
3. Mahasiswa dapat menjelaskan tentang multilist file

B. Uraian Materi

1. Organisasi Berkas Dengan Banyak Kunci

Organisasi berkas banyak kunci merupakan organisasi berkas yang dapat diakses dengan banyak cara, yang masing-masing mempunyai kunci yang berbeda. Teknik organisasi ini merupakan jantung dari implementasi basis data.

Beberapa teknik digunakan dalam organisasi file dengan banyak kunci. Ini adalah dua teknik dasar yang digunakan untuk mengalokasikan hubungan antara indeks dan catatan data dalam file:

1) Terbalik

2) Daftar Ganda

Saat menggunakan organisasi file multi-kunci, paket perangkat lunak lain diperlukan untuk mendukung sistem pemrosesan data manajemen, sehingga banyak program penyusun tidak menyediakan fungsi untuk mendukung organisasi file multi-kunci.

Pada sistem informasi interaktif memerlukan dukungan organisasi berkas banyak kunci untuk memudahkan dalam mengakses data.

Contoh, sistem Online Shop mempunya beberapa pemakai seperti penjual, pembeli dan suplier , Dimana semauya perlu mengakses data dalam format yang sama dengan catatan :

Tabel 7. 1 Contoh tabel barang

Kd_barang	Nama_barang	Kd_kategori	Kd_suplier	stok	Harga

Memerlukan cara mengakses file yang berbeda karena adanya pemakai yang berbeda. Seorang penjual memerlukan akses record tersebut menurut kd_barang, ke_kategori, atau kd_suplier . Pembeli memerlukan akses ke record

berdasarkan kombinasi nama_barang atau kategori. Suplier memerlukan akses record menurut kd_suplier.

Iterasi data dalam banyak file bukanlah cara yang baik untuk mengakses catatan dengan cara berikut banyak kunci karena memerlukan ruang penyimpanan yang besar dpan sulit untuk memutakhirkan record secara serentak.

Untuk itu organisasi file dengan banyak kata kunci adalah cara terbaik untuk menyelesaikan masalah di atas, yaitu menggunakan indeks, bukan loop data. Konsep multi-key access diwujudkan dengan membuat beberapa indeks untuk memberikan data yang berbeda pada record. Metode ini menggunakan linked list.

Linked-list (daftar tertaut) adalah struktur data yang digunakan untuk menyimpan koleksi data. Linked-list memiliki sifat-sifat berikut :

1) Elemen berturut-turut yang dihubungkan oleh pointer.

2) Elemen terakhir menunjuk ke NULL

3) Ukurannya apat semakin besar maupun mengecil selama pelaksanaan program.

4) Dapat dibuat hanya ketika sedang diperlukan

Ada banyak struktur data lain yang melakukan hal yang sama seperti linked-list. Sebelum membahas linked-list penting untuk memahami perbedaan antara linked-list dan array. Kedua linked-list dan array digunakan untuk menyimpan koleksi data, dan karena keduanya digunakan untuk tujuan, kita perlu membedakan penggunaannya. Itu berarti dalam kasus mana array yang cocok dan dalam kasus mana linked-list cocok untuk digunakan. ‘

Keuntungan dari linked-list yaitu, bahwa mereka dapat diperluas dalam waktu yang konstan. Untuk membuat array, kita harus mengalokasikan memori untuk jumlah elemen tertentu. Untuk menambahkan lebih banyak elemen ke array saat penuh, kita harus Buat array baru dan salin array lama ke array baru. Ini dapat memakan cukup banyak waktu. Kita dapat mencegah hal ini dengan mengalokasikan banyak ruang pada awalnya tapi kemudian kita bisa mengalokasikan lebih dari yang kita butuhkan. Dengan linked-list, kita dapat mulai dengan ruang untuk hanya satu elemen yang dialokasikan dan menambahkan elemen baru dengan mudah tanpa perlu melakukan penyalinan dan mengalokasikannya.

Ada sejumlah kekurangan yang dimiliki linked-list. Kelemahan utama dari linked-list adalah akses waktu untuk elemen individu. Sedangkan Array adalah Random-Access, yang berarti dibutuhkan O (1) untuk mengakses elemen dalam array. Daftar tertaut mengambil O (n) untuk akses ke elemen dalam daftar dalam kasus terburuk.

2. Berkas Terbalik (Inverted file)

Struktur berkas Terbalik

Struktur file terbalik memperluas direktori rekaman file. File pada tipe struktur ini ada 2 yaitu file data dan direktori. File data direktori diimplementasikan dengan struktur file yang berbeda.file data diakses secara langsung (direct access), sedangkan file direktori di akses secara sekuensial atau sekuensial berindek jika recordnya banyak dengan nilai kunci untuk menelusurinya.

Tabel 7. 2 File Data Barang

File data Barang		
Record address	Kd_barang	Kd_kategori
1	B001	K002
2	B002	K003
3	B003	K001
4	B004	K002
5	B005	K001

Tabel 7. 3 File Direktori Dari File Data Barang

File Inverted dengan Indeks kd_kategori	
kd_kategori	Record address
K001	3,5
K002	1,4
K003	2

Ada beberapa istilah dalam berkas inversi:

1) Primary Key

   Sebuah Key yang digunakan untuk menentukan struktur penyimpanan dari file.

2) Secondary Key

   Sebuah Key yang tidak digunakan untuk menentukan struktur penyimpanan file.

3) Completely inverted

   Setiap bidang data memiliki file indeks terbalik.

4) Partialy inverted file

   File yang tidak sepenuhnya terbalik tetapi memiliki setidaknya satu indeks terbalik.

Ada dua file dalam struktur file terbalik, yaitu direktori dan file data. Masing-masing diimplementasikan dalam struktur terpisah. File data diimplementasikan sebagai akses langsung (akses langsung), dan direktori diimplementasikan sebagai akses serial. Ini mengurangi pemeliharaan direktori saat catatan baru ditambahkan ke file. Jika Anda mencari berdasarkan nilai kunci, file Katalog dapat berupa file berurutan, atau dapat diindeks sebagai file berurutan dengan nilai yang besar.

Direktori File

Direktori file adalah sebuah penamaan terhadap sejumlah file, ini adalah cara untuk mengelompokan file sehingga kita dapat mengatur sejumlah file tersebut.

File direktori memiliki organisasi berurutan diindeks, yaitu record dapat diakses secara acak atau secara berurutan. Ada satu file direktori per file data. Ada satu record direktori pada gambar berikut :

Ada satu record direktori pada gambar di bawah untuk setiap Key yang muncul di setiap record data. Dengan demikian, file data yang terdiri dari 1.000 record, masing-masing dengan lima kunci, memerlukan sebuah urutan langsung oleh 5.000 Record data. File direktori tersimpan dalam file kunci.

Panjang field kunci data telah ditetapkan di dalam file direktori tertentu, namun file direktori yang berbeda mungkin memiliki field kunci data yang berbeda panjang. Panjang field kunci data sama dengan panjang kunci terbesar dalam file data yang sesuai. Kunci data yang lebih pendek dari maksimum yang dibiarkan di dalam file direktori dan diisi ke kanan dengan nilai null. Ini akan menjamin urutan yang benar dalam file direktori.

Nomor kunci menunjukkan field kunci dalam rekaman data yang berisi nilai dalam field kunci data terkait. Programmer harus menetapkan nomor kunci untuk setiap field kunci dalam record data. Tombol bernomor dari satu sampai sepuluh disebut secondary key (kunci sekunder), dan setiap nomor tombol nol disebut primary key (kunci utama). Hal ini tidak diperlukan untuk menetapkan primary key, tetapi ada keuntungan yang signifikan yang diperoleh dari penggunaannya.

Bidang ketiga di setiap record direktori adalah lokasi fisik relatif dari sebuah data record (dalam file data) yang berisi kunci data tersebut. Kunci dimana catatan direktori tertentu diketahui perangkat keras atau sistem operasi adalah seluruh direktori record. Bahkan jika dua record data berisi nilai kunci, dan nomor kunci yang sama, alamat fisik mereka yang berbeda menjamin catatan direktori yang unik. Demikina juga jika salah satu data record berisi nilai yang sama di dua field kunci yang berbeda, keunikan dalam file direktori terjamin dengan nomor kunci yang berbeda yang ditetapkan ke dua field kunci.

Data File (Berkas Data)

File data memiliki organisasi langsung, dan berisi semua data pengguna. Kunci setiap rekaman data adalah lokasi fisik rekaman relatif terhadap awal file,

yang memastikan kunci unik untuk setiap record. Kunci yang ditunjukkan pada gambar berikut adalah kunci yang diketahui pengguna dan yang muncul di file direktori, kunci ini bukanlah kunci dimana record data diketahui hardware atau sistem operasi.

File data dapat diakses terlebih dahulu hanya dengan membaca record direktori, dan kemudian mendapatkan pointer ke record data yang sesuai.

Metode Akses File Terbalik

Metode ini menggunakan indeks yang dibalik (inverted index atau inverted directory). Record file inverted merupakan alamat tau kuncin untuk semua record dalam file primer. Indeks file inverted bisa menunjukan alamat record dari semua file primer. Contohnya bisa dilihat pada tabel 2, yang bisa menunjukan alamat record dari tabel 1.

Beberapa file diperlkan jika mengalokasikan record untuk berbagai macam alternatif (seperti umur, jenis kelamin, status, dll). Dengan file inverted pengaksesan record yang memiliki banyak hubungan logis lebih efisien.

Metode akses file terbalik Metode akses pada inverted file menggunakan indeks yang disebut inverted index atau inverted directory (file yang dibalik). Setiap daftar inverted index, alamat maupun key untuk semua record mempunyai atribut yang sama.

Misalnya : Indeks terbalik dapat mengarah ke catatan alamat semua karyawan yang berusia 18-25, 26-30, 31-35. Untuk mencari berbagai catatan alternatif (seperti umur, jenis kelamin, dan status perkawinan), diperlukan beberapa file.

File terbalik ini dapat meningkatkan efisiensi pengaksesan rekaman dengan banyak hubungan logis. Metode ini memberikan kemudahan yang lebih baik untuk penggunaan satu atau beberapa rekaman atribut dan file pelacakan. Data juga diperlukan untuk mendeskripsikan hubungan dan alamat record yang diidentifikasi oleh file terbalik, yang diidentifikasi sebagai data keluaran tambahan. Contoh file inverted bisa dilihat dari struktur pada tabel diatas.

Tabel 7. 4 File Data Mahasiswa

File Data Mahasiswa			
Address	NIM	Nama	Kd_prodi
1	2013001	Anton	TI
2	2013002	Celin	AK
3	2013003	Deni	TI
4	2013004	Abdul	MIPA
5	2013005	Erlin	AK
6	2013006	Feri	MIPA
7	2013007	Ridwan	TI
8	2013008	Beni	AK

Tabel 7. 5 File Direktori Data Mahasiswa

Search key	address
TI	1,3,7
AK	2,5,8
MIPA	4,6

Pada tabel diatas menunjukan pemakaian pengindekan terbalik berdasarkan nilai kd_prodi.

3. Multilist File

Struktur file Multilist

File multi-list memiliki indeks untuk setiap kunci sekunder. Dalam beberapa daftar, nilai kunci memiliki penunjuk ke catatan data pertama dengan nilai kunci, catatan data memiliki penunjuk ke catatan data berikutnya dengan nilai kunci, dan seterusnya. Kemudian ada daftar catatan data yang ditautkan untuk setiap nilai kunci kedua.

Contoh file data dengan struktur daftar berganda:

Tabel 7. 6 Contoh Struktur Multilist

Kd_kategori	Kd_barang
K001	B0123
K002	B0132
K003	B0101

Tabel diatas menunjukan indeks Multi list untuk secondary Key kd_barang, sedangkan tabel dibawah menujukan data file.

Tabel 7. 7 Tabel Data dan Barang

Record address	Kd_barang	Nama_barang	Kd_kategori	Kd_suplier	stok	Harga
1	B0123	Buku tulis	K001	S001	2	5000
2	B0250	Pensil	K001	S002	3	2500
3	B0345	Kopi	K002	S003	5	1500
4	B0101	Sabun mandi	K003	S004	6	5000
5	B0212	Roti tawar	K002	S005	2	10000
6	B0012	Susu murni	K002	S003	7	16000
7	B0254	Penghapus	K002	S001	5	2000
8	B0243	Teh celup	K002	S003	8	3000
9	B0112	Pasta gigi	K003	S004	10	7000
10	B0132	Gula pasir	K002	S005	12	16000

Tabel 7. 8 Contoh Struktur Multilist

Kd_kategori	Kd_barang	Panjang
K001	B0123	2
K002	B0132	6
K003	B0101	2

Mengenai jumlah catatan pada daftar tertaut berfungsi untuk memperoleh metode akses data terbaik.

Dari tabel diatas terdapat 3 cara potensial untuk pengaksesan :

1) Cara pengaksesan sekuensial memerlukan akses sapai 10 data record

2) Menggunakan indeks kd_suplier memerlukan 5 akses data record

3) Menggunakan indeks kd_kategori memerlukan 3 akses data record

jadi bisa disimpulkan dari ketiga cara tersebut, pengaksesan terbaik adalah menggunakan indeks kd_kategori.

4. Contoh Program

Struktur Basis Data Barang

	#	Nama	Jenis
☐	1	kd_barang	varchar(10)
☐	2	nama_barang	varchar(25)
☐	3	stok	int(11)
☐	4	harga	int(11)
☐	5	kd_kategori	varchar(10)
☐	6	kd_suplier	varchar(10)

Gambar 7. 1 Basis Data

File koneksi.php

```
<?php

mysql_pconnect("localhost","root","");

mysql_select_db("multikey");

?>
```

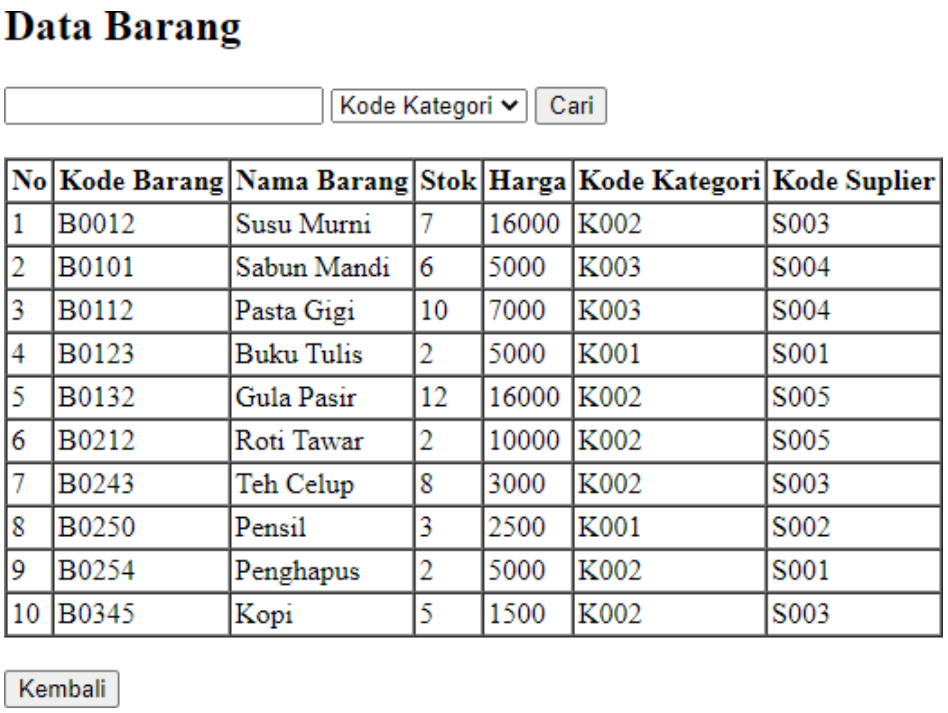
File index.php

```
<?php
```

```
require_once("koneksi.php");
extract($_GET);
if(empty($_GET['cari'])){
    $query=mysql_query("select * from barang");
}
else{
    $query=mysql_query("select * from barang where $kat_cari like '%".$cari."%'");
}
$jumlahdata=mysql_num_rows($query);
$row=mysql_fetch_assoc($query);
?>
<!DOCTYPE html>
<html>
<head>
<title></title>
</head>
<body>
<div>
<div>
<div><h2>Data Barang</h2></div>
<div>
<form action="index.php" method="get">
<input type="search" name="cari">
<select name="kat_cari">
<optionvalue="kd_kategori">Kode Kategori</option>
<optionvalue="kd_suplier">Kode Suplier</option>
```

```
                    </select>
                    <input type="submit" value="Cari">
                </form>
            </div><br>
            <div>
                <?php
                    if(isset($_GET['cari'])){
                    echo "Pencarian berdasarkan ".$kat_cari." = ".$cari;
                    }
                ?>
            </div>
        </div>
        <div>
            <table border="1" cellspacing="0" cellpadding="2">
                <tr>
                    <th>No</th>
                    <th>Kode Barang</th>
                    <th>Nama Barang</th>
                    <th>Stok</th>
                    <th>Harga</th>
                    <th>Kode Kategori</th>
                    <th>Kode Suplier</th>
                </tr>
                <?php
                    $x=1;
                    if($jumlahdata>0){
                        do{
```

```
                ?>
                <tr>
                        <td><?php echo $x ?></td>
                        <td><?php echo $row['kd_barang']; ?></td>
                        <td><?php echo $row['nama_barang']; ?></td>
                        <td><?php echo $row['stok']; ?></td>
                        <td><?php echo $row['harga']; ?></td>
                        <td><?php echo $row['kd_kategori']; ?></td>
                        <td><?php echo $row['kd_suplier']; ?></td>
                </tr>
                <?php
                                $x++;
                                }while($row=mysql_fetch_assoc($query));
                        }
                ?>
            </table>
        </div>
    <br>
        <div>
            <a href="index.php">
                    <input type="Button" value="Kembali">
            </a>
        </div>
    </div>
    </body>
    </html>
```



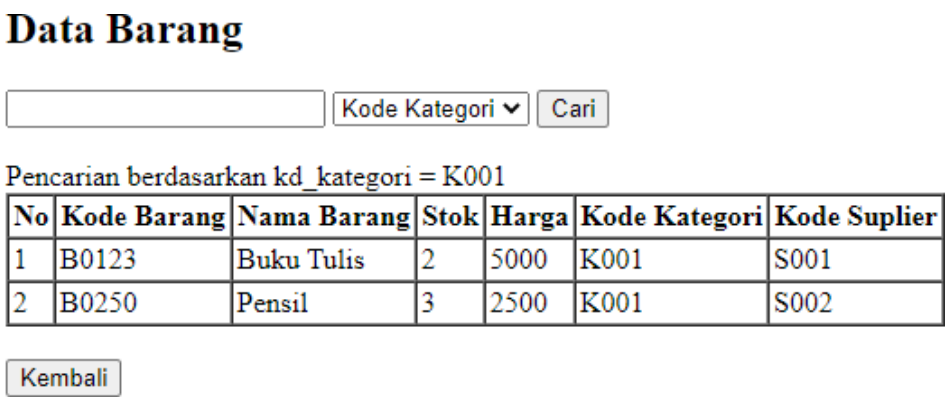
Data Barang

Kode Kategori Cari

No	Kode Barang	Nama Barang	Stok	Harga	Kode Kategori	Kode Suplier
1	B0012	Susu Murni	7	16000	K002	S003
2	B0101	Sabun Mandi	6	5000	K003	S004
3	B0112	Pasta Gigi	10	7000	K003	S004
4	B0123	Buku Tulis	2	5000	K001	S001
5	B0132	Gula Pasir	12	16000	K002	S005
6	B0212	Roti Tawar	2	10000	K002	S005
7	B0243	Teh Celup	8	3000	K002	S003
8	B0250	Pensil	3	2500	K001	S002
9	B0254	Penghapus	2	5000	K002	S001
10	B0345	Kopi	5	1500	K002	S003

Kembali

Gambar 7. 3 Tampilan Data Barang



Data Barang

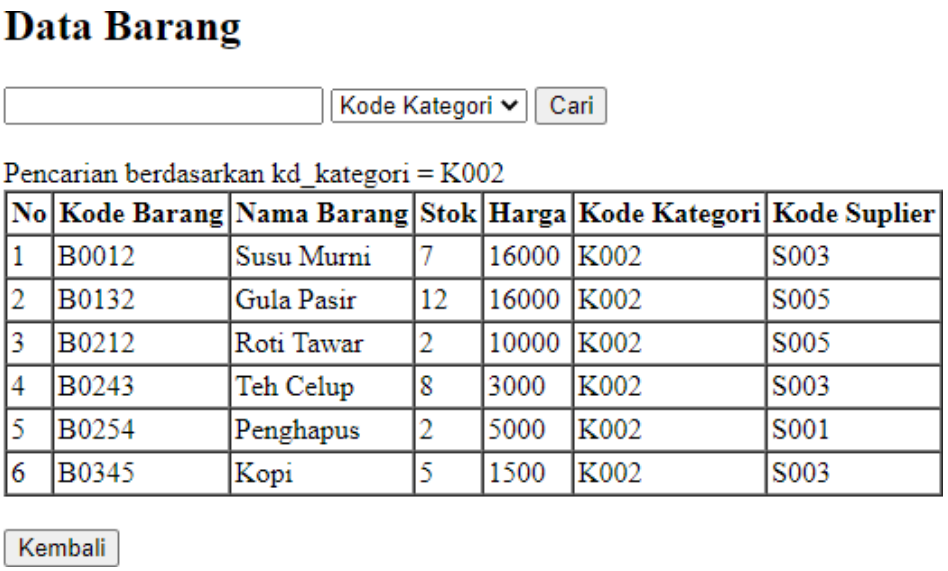
Kode Kategori Cari

Pencarian berdasarkan kd_kategori = K001

No	Kode Barang	Nama Barang	Stok	Harga	Kode Kategori	Kode Suplier
1	B0123	Buku Tulis	2	5000	K001	S001
2	B0250	Pensil	3	2500	K001	S002

Kembali

Gambar 7. 2 Pencarian Data Barang Berdasarkan Kode Kategori K001



Data Barang

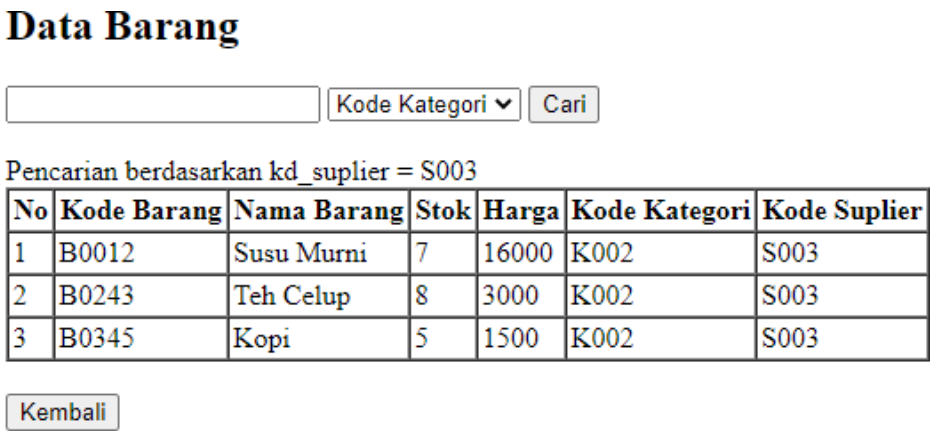
Kode Kategori Cari

Pencarian berdasarkan kd_kategori = K002

No	Kode Barang	Nama Barang	Stok	Harga	Kode Kategori	Kode Suplier
1	B0012	Susu Murni	7	16000	K002	S003
2	B0132	Gula Pasir	12	16000	K002	S005
3	B0212	Roti Tawar	2	10000	K002	S005
4	B0243	Teh Celup	8	3000	K002	S003
5	B0254	Penghapus	2	5000	K002	S001
6	B0345	Kopi	5	1500	K002	S003

Kembali

Gambar 7. 4 Pencarian Data Barang Berdasarkan Kode Kategori K002



Data Barang

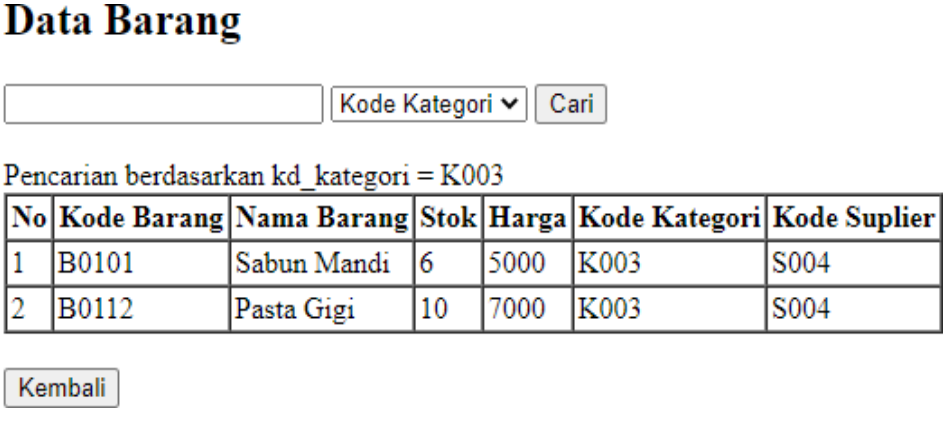
Kode Kategori Cari

Pencarian berdasarkan kd_suplier = S003

No	Kode Barang	Nama Barang	Stok	Harga	Kode Kategori	Kode Suplier
1	B0012	Susu Murni	7	16000	K002	S003
2	B0243	Teh Celup	8	3000	K002	S003
3	B0345	Kopi	5	1500	K002	S003

Kembali

Gambar 7. 5 Pencarian Data Barang Berdasarkan Kode Kategori K003



Data Barang

Kode Kategori Cari

Pencarian berdasarkan kd_kategori = K003

No	Kode Barang	Nama Barang	Stok	Harga	Kode Kategori	Kode Suplier
1	B0101	Sabun Mandi	6	5000	K003	S004
2	B0112	Pasta Gigi	10	7000	K003	S004

Kembali

Gambar 7. 6 Pencarian data barang berdasarkan kode kategori S003

C. Soal Latihan

1. Apa yang dimaksud dengan organisasi file banyak kunci?
2. Sebutkan dan jelaskan yang anda ketahui teknik dalam organisasi berkas banyak kunci!
3. Apa perbedaan antara inverted file dan multilist file?

D. Referensi

Handayani, Dewi. 2001. *Sistem Berkas*. Yogyakarta : J&J Learning.

PERTEMUAN 8
ORGANISASI BERKAS RELATIF

A. Tujuan Pembelajaran

1. Mahasiswa dapat menjelaskan tentang organisasi berkas relatif.
2. Mahasiswa dapat menjelaskan pengertian Teknik Pemetaan Langsung, kalkulasi.
3. Mahasiswa dapat menjelaskan tentang teknik teknik yang dapat dikalkulasi.

B. Uraian Materi

1. Organisasi Berkas Relatif

Definisi File Relatif

1) Metode efektif untuk mengatur sekumpulan rekaman yang memerlukan akses cepat ke rekaman disebut organisasi file relatif. Dalam file relatif, ada hubungan antara kunci yang digunakan untuk mengidentifikasi record dan lokasi record di penyimpanan tambahan.

2) Gunakan kunci yang diperlukan untuk mengidentifikasi file yang direkam.

3) Rekaman tidak memerlukan nilai kunci untuk diurutkan secara fisik.

4) Hubungan ini direpresentasikan sebagai R, yang merupakan fungsi pemetaan.

R(NILAI KEY) ⟶ ADDRESS

Dari nilai kunci ke alamat dalam penyimpanan tambahan.

Pemrosesan file relatif

1) Saat menulis record ke file relatif, fungsi pemetaan R digunakan untuk mengubah nilai kunci dari record ke alamat tempat record disimpan.

2) File yang sesuai harus disimpan pada media DASD (seperti disk atau drum).

Catatan:

1) Kita tidak perlu mengakses semua record file master, kita hanya perlu mengakses record yang dibutuhkan secara langsung.

2) Catatan dalam file terkait dapat diperbarui secara langsung tanpa merekam ulang semua catatan.

3) Keuntungan dari file relatif ini adalah bahwa catatan dapat diakses secara langsung. Rekaman dapat diambil, disisipkan, dimodifikasi atau dihapus; catatan lain dalam file yang sama tidak terpengaruh.

Teknik dasar terbagi menjadi 3 yang digunakan untuk menyatakan fungsi pemetaan R, Dimana R(NILAI KEY) ⟶ ADDRESS, yaitu :

1) Teknik Pemetaan Langsung (Direct Mapping)
2) Teknik Pencarian Tabel (Directory Look Up)
3) Teknik Kalkulasi (Calculating)

2. Teknik Pemetaan Langsung (Direct Mapping)

Teknologi ini adalah teknologi sederhana yang mengubah nilai kunci record menjadi alamat.

Pemetaan langsung ada dua yaitu:

1) Pengalamatan mutlak
2) Pengalamatan relative

Pengalamatan Mutlak

R (nilai kunci) ⟶ alamat

Nilai kunci = alamat mutlak

Fungsi pemetaan ini benar-benar dapat dialamatkan. Nilai kunci yang diberikan oleh pengguna program sama dengan alamat sebenarnya yang tercatat di penyimpanan tambahan.

Saat menyimpan record, pengguna harus menentukan di mana menyimpan record (nomor silinder, nomor permukaan, nomor record) (jika pengalamatan silinder digunakan), atau (nomor sektor, nomor record) (jika digunakan untuk pengalamatan sektor). Demikian pula, saat mengambil record, pengguna harus mengetahui dan menentukan posisi absolut. Keuntungan pengalamatan dibagi menjadi 2

Keuntungan :

1) Fungsi pemetaan R sangat sederhana.
2) Tidak butuh waktu lama untuk menentukan lokasi record di penyimpanan sekunder (pengambilan lebih cepat).

Kelemahan :

1) Pengguna harus tahu persis catatan mana yang disimpan secara fisik.
2) Tergantung pada perangkat, memperbaiki atau mengubah perangkat tempat file tersebut berada akan mengubah nilai kunci.
3) Tergantung pada ruang alamat. Pengaturan ulang file relatif akan menyebabkan nilai kunci berubah.

Pengalamatan Relatif (Relative Addressing)

R(NILAI KEY) ⟶ ADDRESS

NILAI KEY = ALAMAT RELATIF

Pengalamatan relatif dari sebuah record dalam sebuah berkas disebut urutan record tersebut dalam berkas. Sebuah berkas dengan N record mempunyai record alamat relative dari himpunan (1,2,3, …., N-2,N,N-1) record yang ke 1 mempunyai alamat relative 1 dan 1-1.

Keuntungan :

1) Fungsi pemetaan R sangat sederhana.
2) Nilai key dari sebuah record dapat ditentukan lokasi recordnya dalam sebuah penyimpanan sekunder tanpa memerlukan waktu proses yang berarti.
3) Retrieve lebih cepat.

Kelemahan :

1) bukan device dependent
2) Merupakan address space dependent
3) Terjadinya pemborosan ruangan

3. Teknik Pencarian Tabel (Directory Look Up)

Teknologi pencarian tabel lebih dari sekedar teknologi pemetaan langsung. Itu hanya membutuhkan biaya perawatan baru.

Perlu diperhatikan bahwa metode ini hampir sama dengan teknik yang digunakan untuk menghasilkan indeks sekuensial. Prinsip dasar dari metode tabel pencarian adalah tabel atau direktori nilai kunci atau alamat. Untuk menemukan record dalam file relatif, pertama-tama lihat di direktori nilai kunci record dan tunjukkan alamat penyimpanan record tersebut.

Keuntungan dari Pencarian Tabel :

1) Fungsi Pemetaan R sangat sederhana.
2) Penetuan nilai key tidak perlu waktu proses yang lama.

Kelemahan dari Pencarian Tabel :

1) Alamat relatif adalah address space dependent.
2) Terjadinya pemborosan ruangan.

4. Teknik Kalkulasi (Calculating)

Pendekatan lain yang umum mengimplementasikan

R (NILAI KEY) ⟶ ADDRESS

Adalah dengan melakukan kalkulasi terhadap nilai key, hasilnya adalah alamat relatif.

1) Salah satu kelemahan dari teknik pengalamatan relatif adalah ruang harus disediakan sebanyak jangkauan nilai key, terlepas dari berapa banyak nilai key.

2) Salah satu masalah dari teknik ini adalah ditemukannya alamat relatif yang sama untuk nilai key yang berbeda.

3) Keadaan dimana :

   R(K1) = R(K2) disebut benturan

   K1 ≠ K atau collision

4) Sedangkan nilai key K1 dan K2 disebut synomin.

5) Synonim adalah dua atau lebih nilai key yang berbeda pada hash ke home address yang sama.

Teknik-teknik yang terdapat pada kalkulasi :

1) Scatter storage techniques

2) Randomizing techniques

3) Key-to-address transformation methods

4) Direct addressing techniques

5) Hash table methods

6) Hashing

Scatter storage techniques

Sebuah metode dan perangkat untuk melakukan penyimpanan pengungkapan dan pengembalian dalam sistem untuk menyimpan informasi yang diungkapkan menggunakan teknik hashing. Untuk mencegah kontaminasi dan penyimpanan media dengan catatan kedaluwarsa otomatis, teknik pengumpulan sampah digunakan yang menghapus semua catatan lengkap di lingkungan dan peneliti ke dalam sistem penyimpanan data.

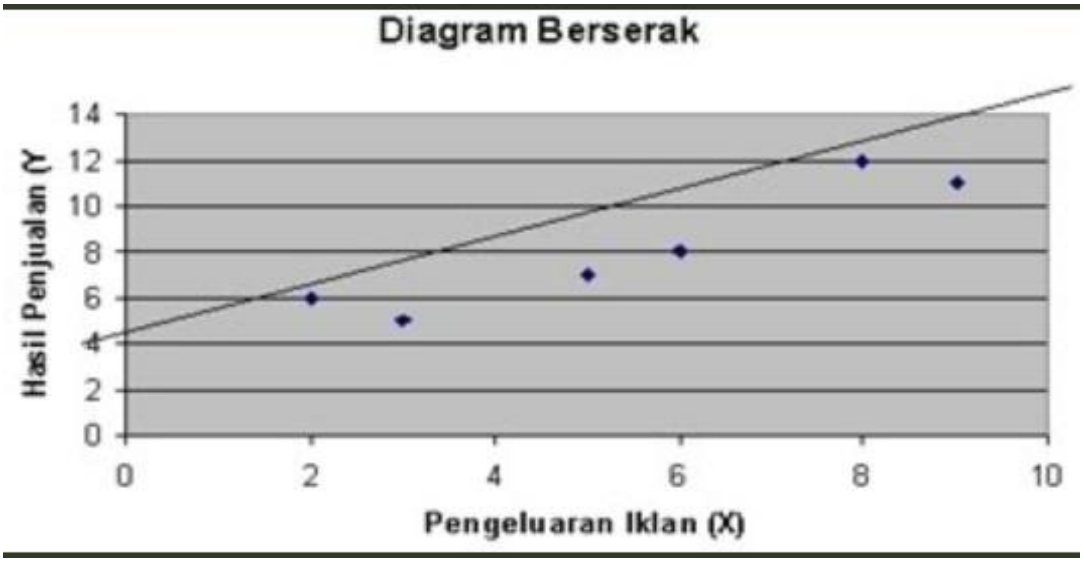
Lebih khusus lagi, masalah apa pun dengan memasukkan, mengambil, atau menghapus record adalah kemampuan untuk mencari seluruh rangkaian record yang ditemukan untuk record terakhir, lalu menghapusnya dan menutup rantai

pengumpulan. Sampah ini secara otomatis menghapus record sampah yang berakhir di sekitar probe, sehingga secara otomatis membersihkan ruang penyimpanan. Karena tidak ada kontaminasi jangka panjang yang dapat terakumulasi dalam sistem ini, ini sangat berguna untuk database besar yang banyak digunakan dan yang memerlukan akses cepat yang disediakan oleh hashing.

Contoh :

Variabel X	Variabel Y
2	6
3	5
5	7
6	8
8	12
9	11

Gambar 8. 1 Variabel



Gambar 8. 2 Diagram Berserak

Randomizing techniques

Teknik acak sederhana terinspirasi dari metode eksplorasi probabilistik, yang berguna untuk transformasi, menciptakan area bebas benturan, dan mendeskripsikan metode transformasi iteratif yang memudahkan dalam mencari solusi dari masalah.

Contoh :

Pencarian Nomor acak pada pseudo-code

```
prefixGenerator :: (Ord a, Arbitrary a) => Gen ([a],[a])
prefixGenerator = frequency [
    (1, return ([],[])),
    (2, do {
            xs1 <- orderedListEj13 ;
            xs2 <- orderedListEj13 ;
            return (xs1,xs2)
       }),
    (2, do {
            xs2 <- orderedListEj13 ;
            return ((take RANDOMNUMBERHERE xs2),xs2)
       })
    ]
```

Gambar 8. 3 Pencarian Nomor Acak

Key-to-address transformation methods

Teknik yang digunakan dalam teori kode koreksi kesalahan digunakan untuk memecahkan masalah penanganan file besar. Pendekatan baru untuk file masalah ini menjelaskan desain khusus untuk menunjukkan kelayakan.

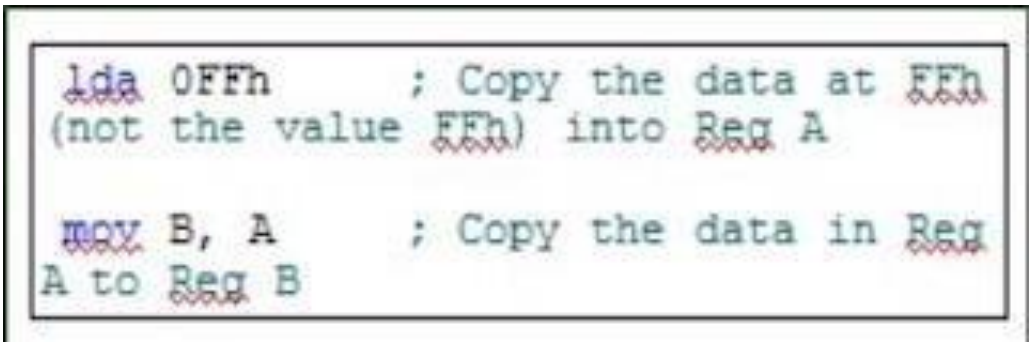
Efektivitasnya diilustrasikan lebih lanjut dengan membandingkan hasil pengujian yang diperoleh dari simulasi komputasi, yang menggunakan data tipikal, dengan nilai yang dihitung dari model ideal.

Direct addressing techniques

Teknik sederhana yang bekerja dengan baik jika U adalah alam semesta (kemungkinan nilai ruang dilambangkan dengan K). U = {0,1, ..., m-1}, nilai m tidak terlalu besar dengan asumsi tidak ada dua elemen yang memiliki kunci yang sama dalam teknik pengalamatan langsung, instruksi lain ditampilkan menggunakan pengalaman langsung, yang berarti bahwa Data yang kami rujuk sebenarnya disimpan dalam struktur lain, baik itu di registri atau di lokasi memori.

Contoh :

Instruksi ditampilkan menggunakan pengalamatan langsung. artinya, data sebenarnya disimpan dalam struktur lain, sebagai register atau lokasi memori.



Gambar 8. 4 Instruksi Ditampilkan

Hash table methods

Ketika kunci berada dalam tabel hash, metode ini harus menghasilkan hasil yang sama ketika dipanggil dengan parameter yang sama.

Contoh :

Deteksi linier, misalnya, dapat menampung 128 data. Fungsi hash dapat mengubah nama file dari 0 menjadi 127. Jika itu membuat file sebagai 129, ukuran tabel harus diperluas ke batas waktu tertentu.

Hashing

Hash. Hashing dapat diartikan sebagai kegiatan mengubah suatu objek menjadi rangkaian angka / karakter / simbol serupa. Tujuan utamanya adalah bahwa dua kunci berbeda tidak memiliki nilai alamat relatif yang sama.

Ada beberapa fungsi Hash yang umum digunakan

Division reminder

Pengingat partisi gunakan pembagian, untuk distribusi yang tidak diketahui

Contoh :

Nilai NPM yang tadinya 10 digit dipotong menjadi hanya 2 digit dengan mengambil 2 nomor terakhir. misalnya 2016110067 akan memiliki home address 67.

Mid Square

menggunakan metode perpangkatan, untuk file dengan faktor cukup rendah.

Contoh :

NPM 2016110067 memiliki home address =

22+02+12+62+12+12+02+02+62+72=4+0+1+36+1+1+0+0+36+49= 128

Folding

Menggunakan metode penjumlahan, mudah dalam perhitungan baik bila panjang key = panjang address.

Contoh :

Nilai NPM 2016110067 yang berisi 10 digit dibagi kedalam 5 bagian masing 2 digit, terus dilipat pada bagian-bagian tersebut. NPM 2016110067 akan menjadi 5 bagian yaitu 20,16,11,00 dan 67 yang dijumlahkan dan menghasilkan 114 (dengan carrier) atau 14 (tanpa carrier)

Keuntungan Menggunakan hashing :

1) Nilai kunci sebenarnya dapat digunakan karena telah diubah menjadi alamat.

2) Nilai kunci telah mengubah ruang alamat, tetapi nilai kunci akan tetap ada.

Kelemahan :

1) Perlu waktu untuk mengimplementasikan fungsi hash

2) Mengatasi konflik membutuhkan waktu pemrosesan dan akses I / O.

Penampilan fungsi hash bergantung pada :

1) Distribusi nilai kunci yang digunakan
2) Jumlah kunci yang digunakan terkait dengan ukuran ruang alamat.
3) Jumlah record yang dapat disimpan di alamat tertentu tanpa menimbulkan konflik.
4) Teknologi untuk menangani tabrakan

Fungsi hash yang umum digunakan :

1) Hashing dengan fungsi Kunci Modulus N
2) Hashing dengan fungsi Kunci Modulus P
3) Hashing dengan fungsi Pemotongan
4) Hashing dengan fungsi Lipatan
5) Hashing dengan fungsi Pergeseran
6) Hashing dengan fungsi Pengkuadratan
7) Hashing dengan fungsi Konversi Radix

Hashing dengan Modulus N

1) Merupakan fungsi hashing yang paling popular
2) Rumus : f(kunci) = kunci mod N

   dengan : N : ukuran tabel atau berkas

   Mod : sisa pembagian

Contoh :

misal N = 12

30 mod N = 6, diperoleh dari 30 dibagi 12 menghasilkan 2 dgn sisa 6

40 mod N = 4, diperoleh dari 40 dibagi 12 menghasilkan 3 dgn sisa 4

Hashing dengan Modulus P

1) Fungsi hashing Kunci mod P merupakan variasi Kunci modulus N.
2) Rumus : f (kunci) = kunci mod P
3) Dimana : P : merupakan bilangan prima terkecil yang lebih besar atau sama dengan P

Contoh :

Jika N = 12 maka P = 13

30 mod P = 4, diperoleh dari 30 mod 13 hasilnya 2 dgn sisa 4

40 mod P = 1, diperoleh dari 40 mod 13 hasilnya 2 dgn sisa 1

Hashing dengan Pemotongan

1) Melakukan pemenggalan sejumlah digit yang pertama atau yang terakhir dari sejumlah digit.

2) Keuntungan, cepat dan mudah dalam implementasinya

3) Kerugian, terbatasnya ukuran ruang alamat

Contoh 1:

Jika fungsi F menghapus 6 digit akhir dari digit 123456789

Maka f(123456789) = 123 hashing memetakan 123456789 ke alamat 123

Hashing dengan Lipatan

1. Nilai kunci dibagi menjadi beberapa bagian, masing-masing memiliki jumlah digit yang sama (kecuali bagian awal atau akhir). Bagian-bagian ini kemudian dilipat antara satu bagian dengan bagian lain. Hasil penjumlahan setelah dilipat dan digit dengan orde paling tinggi dipenggal menjadi alamat relative.

   Contoh :

   Nilai kunci, target alamat relative menggunakan 4 digit artinya Nilai Kunci dibagi menjadi 4 digit

   Jawab : 2.5345.6718

   dilipat dengan urutan terbalik, hasilnya :

   2

   5345

   8176

   15521

   Digit dengan order tertinggi dihilangkan menjadi 5521

Hashing dengan Pengkuadratan

1) Hashing dengan pengkuadratan adalah fungsi hashing dengan cara mengkuadratkan kunci.

2) Hasil pengkuadratan ini kemudian dapat dikombinasi dengan pemotongan atau lipatan untuk mendapatkan alamat yang diijinkan.

Contoh :

Pengkuadratan kunci 782 akan menghasilkan kemungkinan alamat 117 diperoleh dari : $7^{2+}8^{2+}2^{2+}$= 49 + 64 + 4 = 117

Hashing dengan Konversi Radix

1) Dalam konversi radix, kunci dianggap dalam base selain 10 yang kemudian dikoversi ke dalam basis 10, misal 5678 dalam base 13 akan menghasilkan 12098, diperoleh dari :

Posisi : $\underset{5}{3}\ \underset{6}{2}\ \underset{7}{1}\ \underset{8}{0}$

$(5x13^3) + (6x13^2) + (7x13^1) + (8x13^0)$ =10985 + 1013 + 91 + 8 = 12098

C. Soal Latihan

1. Bisa anda jelaskan yang dimaksud dengan Organisasi Berkas Relatif ?
2. Apa perbedaan dari berkas relatif dan sekuensial
3. Menurut anda lebih efektif berkas relative atau berkas sekuensial? Coba jelaskan

D. Referensi

Handayani, Dewi. 2001. Sistem Berkas. Yogyakarta:J&J Learning.

PERTEMUAN 9
ORGANISASI BERKAS INDEKS SEKUENSIAL

A. Tujuan Pembelajaran

1. Mahasiswa dapat memahami dan dapat menjelaskan organisasi berkas indeks sekuensial
2. Mahasiswa dapat mejelaskan cara pembuatan berkas sekuensial
3. Mahasiswa menjelaskan pengertian prime dan overflow data area, dan dapat menyebutkan contohnya.

B. Uraian Materi

1. Pengertian Organisasi Berkas Indeks Sekuensial

Organisasi file indeks sekuensial adalah kombinasi dari organisasi file sekuensial dan organisasi file relatif. Ini adalah metode yang efektif untuk mengatur koleksi rekaman yang perlu diakses secara berurutan atau langsung berdasarkan nilai kunci. Pengaturan file tinta berurutan dirancang untuk kebutuhan pengaksesan secara acak pada file sekuensial. Jadi record dapat diakses secara langsung dengan menggunakan satu atau lebih index.

Struktur file indeks terdiri dari 3 bagian, yaitu:

1. Prime data area

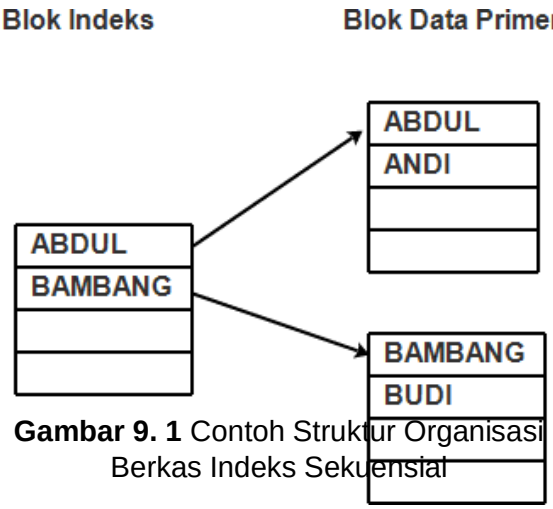
   Daerah data utama yang diurutkan secara sekuensial dan dibagi menjadi blok fisik dengan pointer / nomor record yang tetap

2. Indeks area

   Daerah indeks untuk tiap blok daerah data utama, isi dari indeks area merupakan record pertama dari setiap blok sekuensial.

3. Overflow area

   Daerah ruang bebas untuk penyimpanan record baru selama penyimpanan file



Gambar 9. 1 Contoh Struktur Organisasi Berkas Indeks Sekuensial

Gambar 9. 2 Daftar Isi

Contoh lain dari organisasi berkas indeks sekuensial adalah daftar isi buku. Pada gambar 2, poin “A. PENYUSUNAN RENCANA PEMBELAJARAN SEMESTER(RPS) “ merupakan blok indeks, dan “1.Definisi RPS” merupakan blok data.

2. Indeks

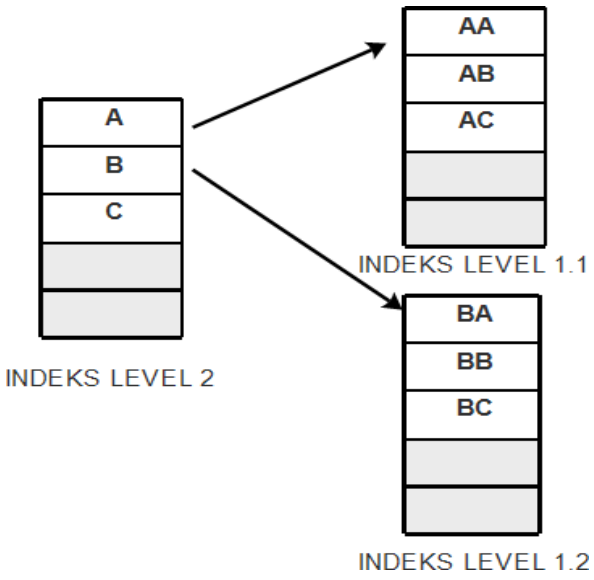
Indeks merupakan sekumpulan entri yang Berisi nilai kunci dari catatan data utama, yang memungkinkan akses cepat ke catatan yang diperlukan. Indeks selalu dalam keadaan terurut berdasarkan kunci recordnya sehingga dapat ditelusuri dengan cepat.

Indeks level-1

Indeks level-1 berisikan nilai kunci record pertama dari setiap blok data primer. Seperti gambar 1, indeks level 1 jumlah maksimum record 4 dan berisikan 2 record, yaitu Abdul yang merupakan nilai kunci record pertama dan Bambang nilai kunci record kedua.

Indeks level-2

Indeks level 2 berisikan nilai kunci record pertama dari setiap blok indeks level 1



Gambar 9. 3 Contoh indeks level 2

3. Insert Record

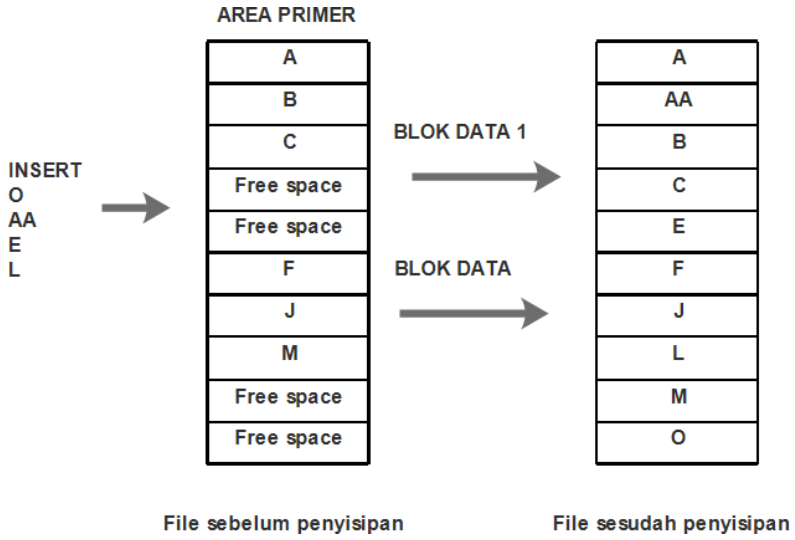
Untuk menyisipkan file baru pada organisasi berkas indeks sekuensial, harus ada ruang kosong (free space) yang telah disiapkan. Ada 3 cara untuk menyisipkan file baru pada organisasi berkas indeks sekuensial, yaitu:

1) Berkas terpisah

Catatan yang disisipkan disimpan dalam file terpisah. Cara ini kurang efisien karena jumlah file atau blok akan menjadi banyak sedangkan isi dari file tidak seragam, dan file indek jugak akan menjadi lebih banyak.

2) Free Space in Every Blok

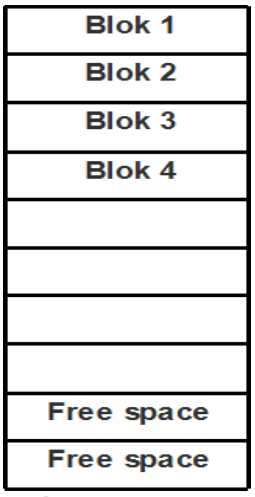
Dalam setiap blok disediakan ruang kosong dan akan terisi jika ada penyisipan record baru.



Gambar 9. 4 Contoh Free Space In Every Blok

3) Free Space in Every Cilinder

Menyediakan beberapa blok pada akhir setiap silinder sebagai persediaan (overflow area). Jika ada penambahan record baru akan ditampung di overflow area.



Gambar 9. 5 Free Space In Every Silinder

4. Blok Indeks dan Data

Dalam metode indeks dan blok data, file indeks diatur dalam blok dan memiliki struktur pohon, sedangkan file data diatur dalam blok dan memiliki struktur berurutan. Blok indeks dan blok data dibuat dengan jumlah pengisian atau ruang kosong tertentu untuk memasukkan atau menghapus file.

Misal :

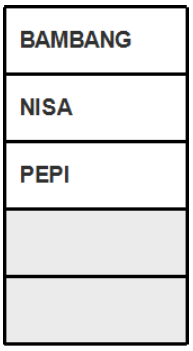
Setiap blok data mempunyai kapasitas menampung 5 record dan blok data mempunyai kapasitas menampung 5 pasang (nilai kunci, pointer).

Permintaan :

INSERT BAMBANG

INSERT PEPI

INSERT NISA



Gambar 9. 6 Hasil Insert BAMBANG, PEPI, NISA

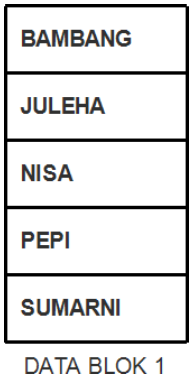
Hasil dari penyisipan BAMBANG, PEPI dan NISA ditunjukan pada gambar di atas, dan pada gambar tersebut bagian yang berwarna abu-abu merupakan ruang bebas atau padding. Penyisipan data pada blok harus disusun dengan urutan sekuensial menaik.

Kasus 1

Permintaan :

INSERT JULEHA

INSERT SUMARNI



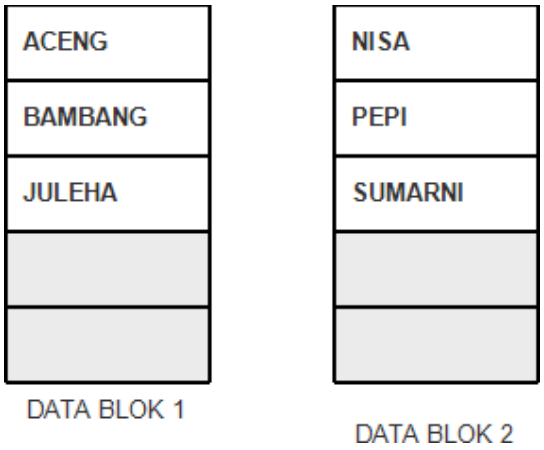
Gambar 9. 7 Hasil Insert JULEHA dan SUMARNI

Untuk penyisipan data JULEHA dan SUMARNI, hanyak DATA BLOK 1 yang digunakan.

Kasus 2

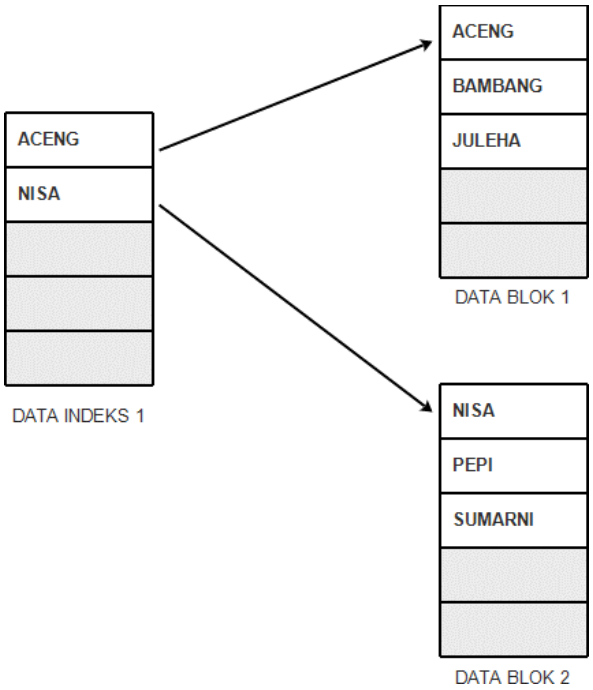
Permintaan :

INSERT ACENG



Gambar 9. 8 Blok Data Dipecah Menjadi 2

Seharusnya ACENG memasuki blok data 1, tetapi blok data sudah penuh. Kemudian solusi untuk situasi ini adalah dengan membagi blok data 1 menjadi 2, separuh dari konten blok data disimpan di blok data 1, dan separuh lainnya dipindahkan ke blok data baru, yaitu blok data 2. kemudian memodifikasi blok indeks 1. Hasilnya ditunjjukan pada gambar 6.



Gambar 9. 9 Hasil Pemecahan Blok Data dan Modifikasi Blok Indeks

Perlu dingingat data harus diurutka terlebih dahulu sebelum blok data dipecah menjadi 2 bagian. Begitu juga dengan blok indeks.

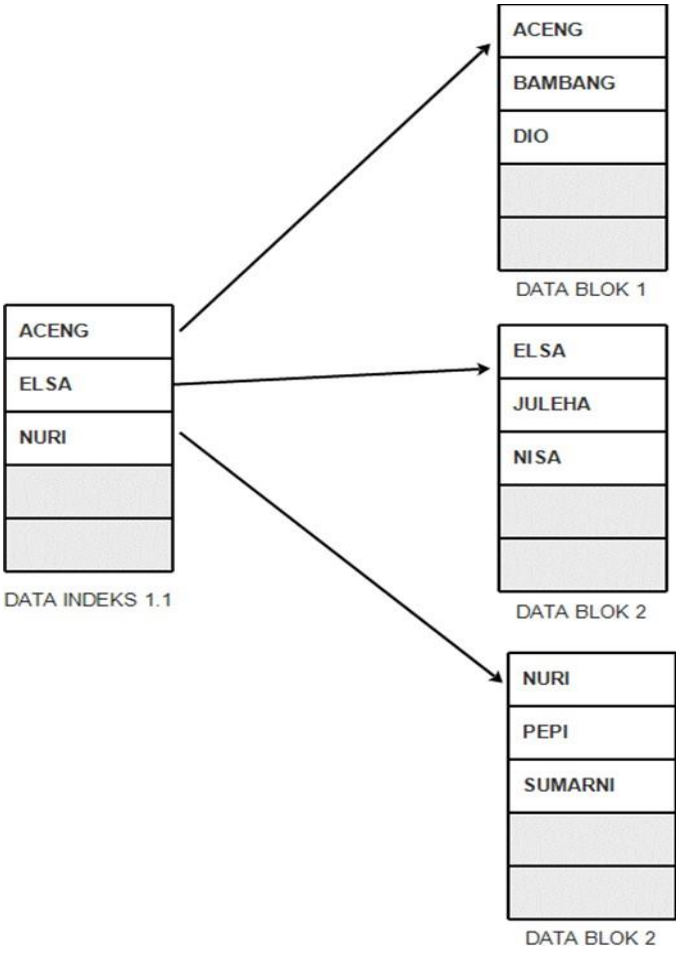
Kasus 3

Permintaan:

INSERT NURI

INSERT ELSA

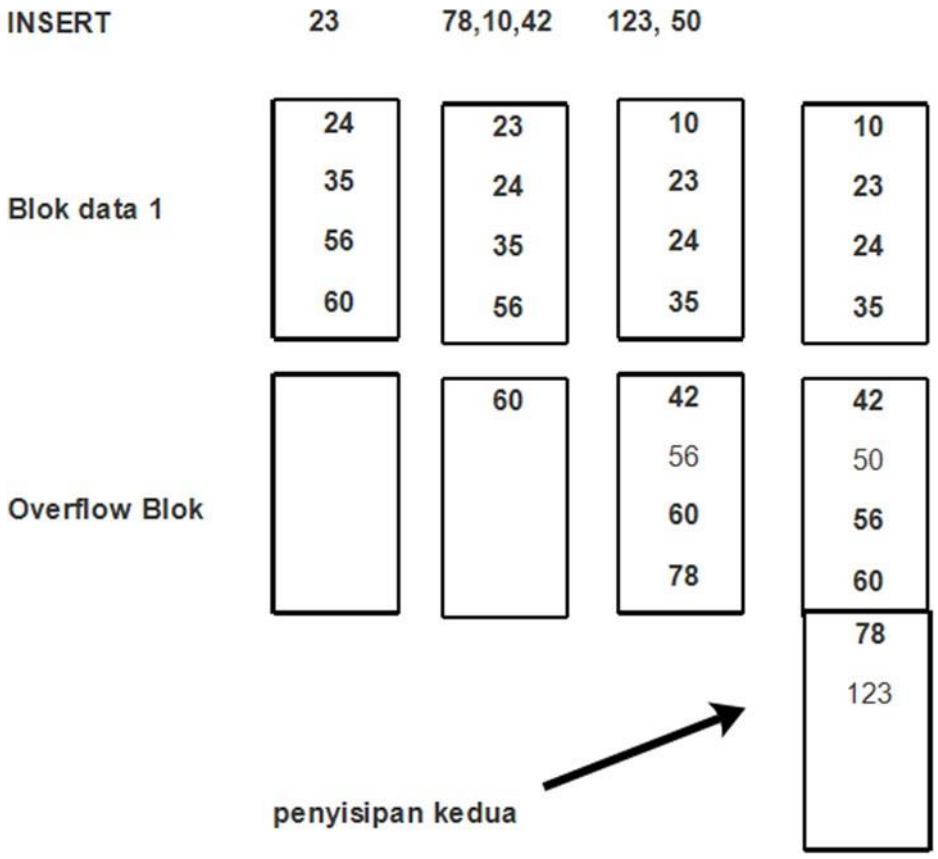
INSERT DIO



Gambar 9. 10 Hasil Penyisipan NURI, ELSA dan DIO

5. Prime dan Overflow Data Area

Prime data area adalah area primer atau area utama penyimpanan record, yang diurutkan secara sekuensial menaik. Urutan data bisa berubah akan tetapi address atau nomor record tetap. Sedangkan overflow merupakan area atau daerah berisi ruang bebas yang digunakan untuk menyimpan record yang baru selama pembuatan file. Jika area primer penuh makan record akan disimpan di area overflow.



Gambar 9. 11 Overflow

6. Contoh program Organisasi berkas indeks sekuensial

Kode program bahasa C++

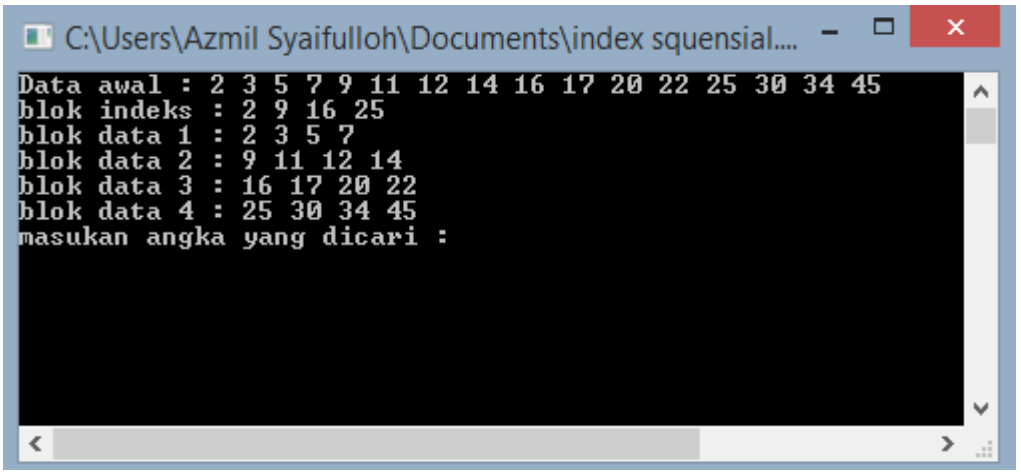
pencarianIndekssekuensial.cpp

```
#include <iostream>
using namespace std;
    int main(){
        const int data[4][4]={{2,3,5,7},{9,11,12,14},{16,17,20,22},{25,30,34,45}};
        int x,z[2],i,j,jumlahindex=0,ketemu;
        cout<<"Data awal : ";
        for(int i=0;i<4;i++){
                for(int j=0;j<4;j++){
                        cout<<data[i][j]<<" ";
```

```
            }
        }
        cout<<endl;
        cout<<"blok indeks : ";
        for(int i=0;i<4;i++){
        for(int j=0;j<1;j++){
                    z[i]=data[i][j];
                    cout<<z[i]<<" ";
                    jumlahindex++;
            }
        }
        cout<<endl;
        for(int i=0;i<4;i++){
            cout<<"blok data "<<i+1<<" : ";
            for(int j=0;j<4;j++){
                    cout<<data[i][j]<<" ";
            }
            cout<<endl;
        }
        cout<<"masukan angka yang dicari : ";cin>>x;
        for(int i=0;i<jumlahindex;i++){
            if(x<z[0]){
                    cout<<"Data tidak ditemukan !!!!";
                    i=jumlahindex;
            }
            if(x >= z[i]){
                    for(j=0;j<4;j++){
```

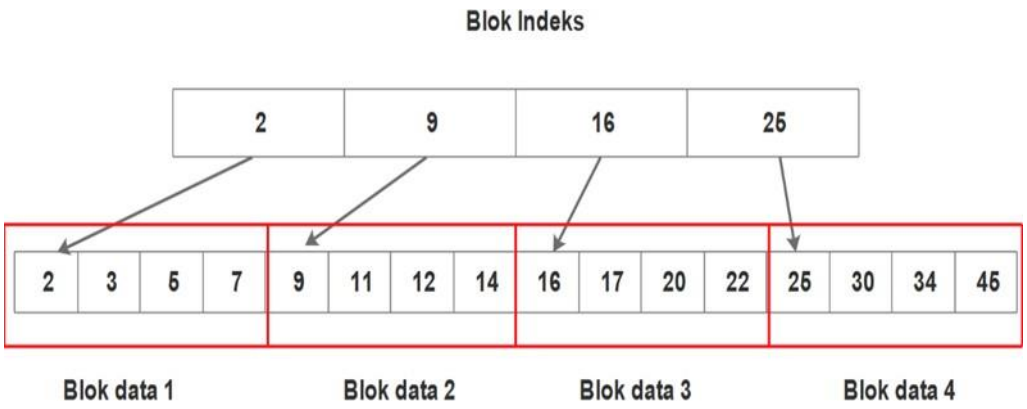
```
                                    if(x==data[i][j]){
                                        cout<<"Data ditemukan pada indeks
"<<z[i]<<" data ke "<<j+1;
                                        ketemu=1;
                                    }
                                }
                            }
                        }
                        if(ketemu==0){
                            cout<<"Data tidak ditemukan!!";
                        }
                    }
```

Dari kode program diatas jika dijalankan akan tampil sebagai berikut :



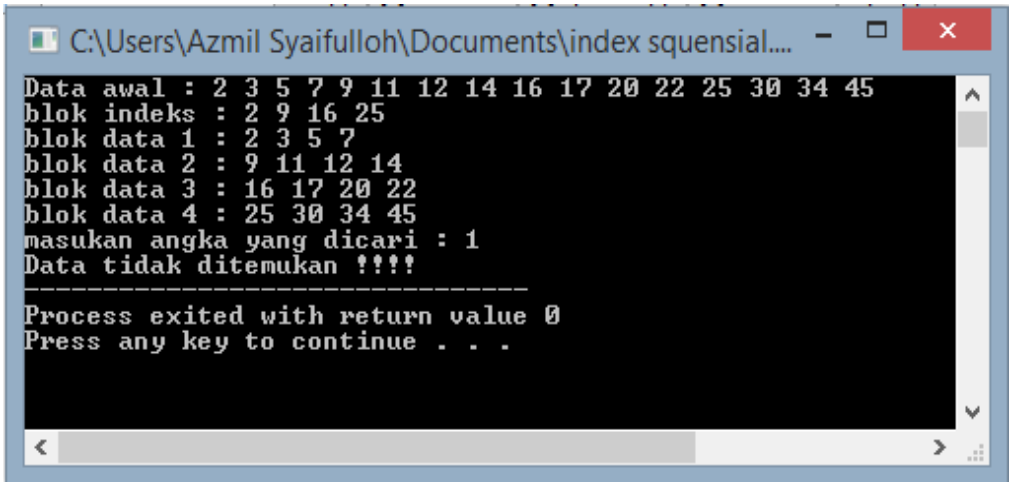
Gambar 9. 12 Tampilan Program Organisasi Berkas Indeks Sekuensial

Program diatas ditentukan blok indeks dapat menampung 4 record dan setiap blok data primer dapat menampung 4 record. Gambar diata dapat dituliskan dalam tabel sebagai berikut.



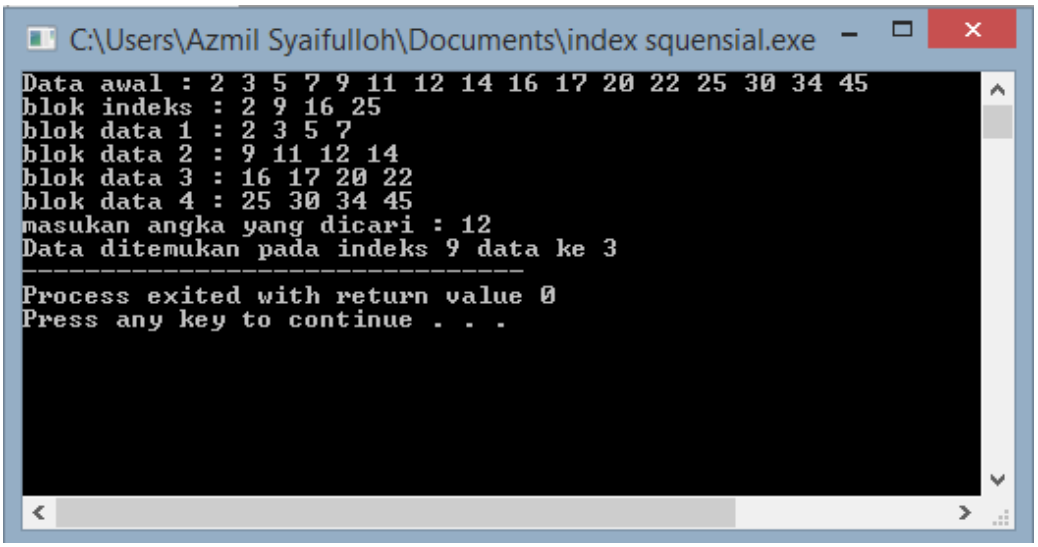
Gambar 9. 13 Pembagian Record dan Indeks

Setiap blok menampung 4 record data dan record pertama dari setiap blok data menjadi indeks. Dan record pada blok data maupun blok indeks harus diurutkan secara sekuensial ascending/ menaik.



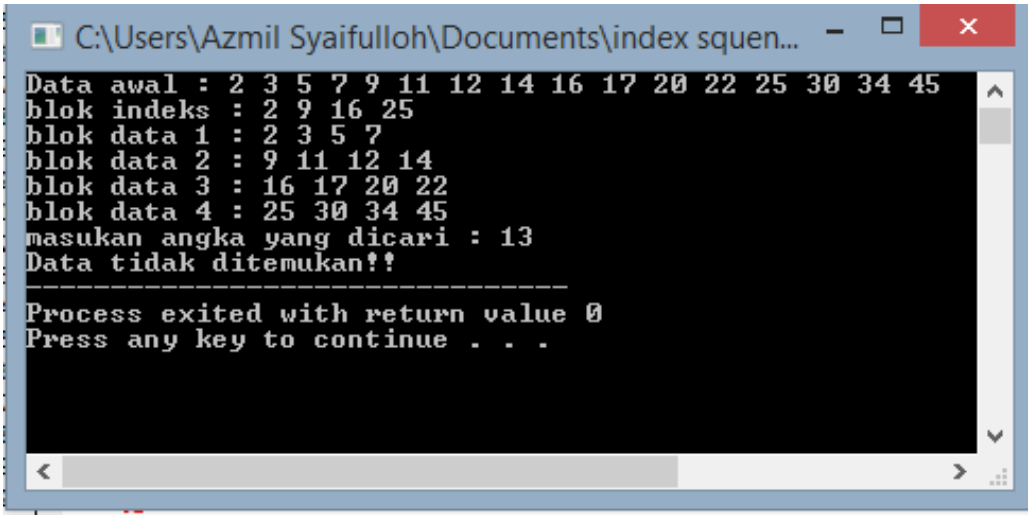
Gambar 9. 14 Program Mencari Nilai Key 1

Jika data yang dicari adalah 1, maka data yang dicari tidak ditemukan. Karena nilai key data yang dicarilebih kecil dari nilai key (kunci) indeks terkecil, jadi dari awal proses pencarian record langsung dihetikan. Jika nilai kunci yang Anda cari lebih besar dari nilai kunci indeks minimum, pencarian record akan diteruskan sampai ketemu.



Gambar 9. 15 Program Mencari Nilai Key 3

Misalkan data yang dicari nilai key nya adalah 12, maka akan dibandingkan terlebih dahulu dengan nilai key indeks. 12 >= 2? jika benar akan dibandingkan dengan nilai key selanjutnya. 12 >= 9 ? benar maka dibandingkan lagi dengan nilai key indeks selanjutnya. 12 >= 16 ? salah, jika salah maka indeks yang diakai adalah indeks sebelumnya yaitu nilai key indek 9. Jadi data yang dicari hanya data yang ber indeks 9 saja. Selanjutnya data dibanding kan satu persatu denga nilai key pencarian, jika ketemu nilai key yang sama maka data yang dicari ditemukan. Seperti gambar diatas, data ditemukan pada indeks 9 urutan data ke 3. Bagaimana jika data yang dicari tidak ditemukan di indeks 9? Jika data tidak ditemukan maka proses pencarian selesai dengan hasil tidak ditemukan, tidak perlu mencari dengan nilai key indeks yang lain. Seperti gambar dibawah ini.



Gambar 9. 16 Program Mencari Nilai Key 13

C. Soal Latihan

1. Apa itu organisasi file indeks sekuensial?
2. Apa perbedaan organisasi berkas indeks sekuensial dan organisasi berkas sekuensial?
3. Sebutkan kelebihan organisasi berkas indeks sekuensial dibanding organisasi berkas yang lain!
4. Sebutkan dan jelaskan struktur organisasi berkas indeks sekuenial!

D. Referensi

Handayani, Dewi. 2001. *Sistem Berkas*. Yogyakarta : J&J Learning.

PERTEMUAN 10
MOUNTING, SHARING DAN PROTEKSI

A. Tujuan Pembelajaran

1. Mahasiswa dapat memahami proses mounting pada sistem file
2. Mahasiswa dapat memahami file sharing pada sistem file
3. Mahasiswa dapat memahami proteksi pada sistem file

B. Uraian Materi

1. Mounting

Proses menautkan sistem file yang baru ditemukan (dalam perangkat) ke direktori home. Perangkat yang akan diinstal dapat berupa CD-ROM, floppy disk atau zip drive. Setiap sistem file yang akan dipasang akan diberi titik pemasangan atau direktori di pohon direktori sistem yang sedang diakses. Sistem file dijelaskan di / etc / fstab (fstab adalah singkatan dari tabel sistem file). Anda dapat menggunakan perintah mount untuk melihat daftar sistem file yang dipasang kapan saja. Karena izin disetel ke hanya baca di file fstab, tidak perlu khawatir pengguna lain akan mencoba mengubah dan menulis ke titik pemasangan yang baru.

Red Hat Linux dan sistem operasi serupa UNIX mengakses file dengan cara yang berbeda dari MS-DOS, Windows dan Macintosh. Di Linux, semua konten disimpan di lokasi yang dapat ditentukan dalam struktur data. Linux bahkan menyimpan perintah sebagai file.

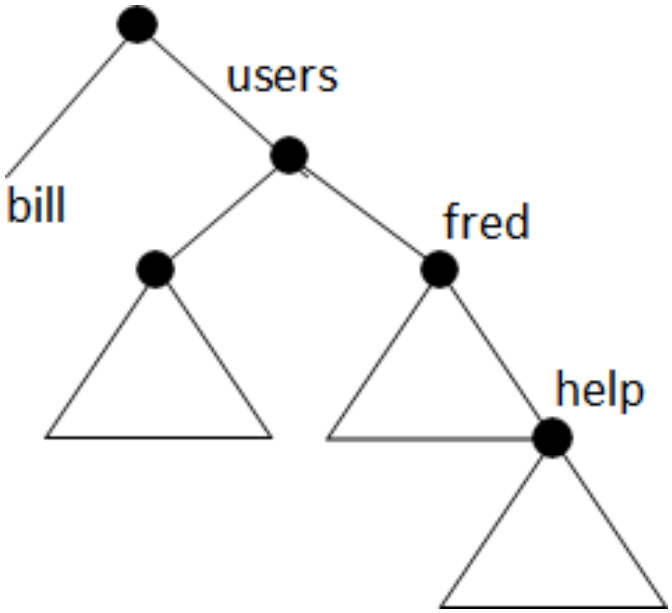
Seperti sistem operasi modern lainnya, Linux memiliki struktur pohon, struktur hierarki, dan organisasi direktori yang disebut sistem file. Semua ruang yang tersedia di disk diatur dalam pohon direktori. Basis dari sistem ini adalah direktori root yang dilambangkan dengan garis miring ("/"). Isi dari sistem berkas dapat digunakan dengan memasukkan sistem berkas ke dalam sistem direktori melalui proses yang disebut pemasangan.

Mounting Overview

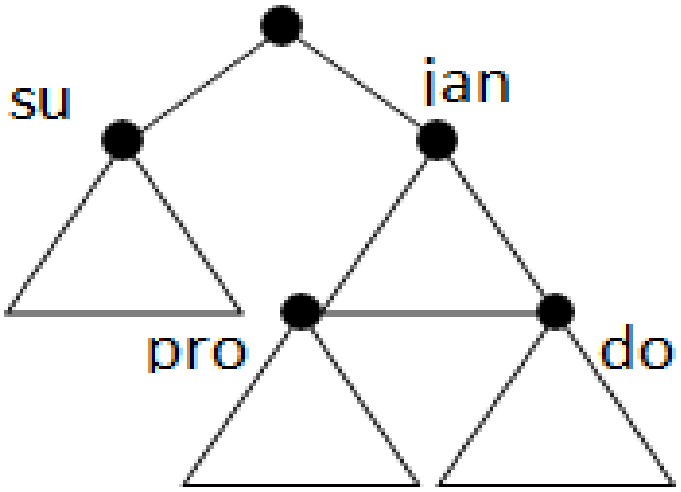
Dengan melakukan mount, file system, direktori, alat dan file lainnya dapat digunakan di lokasi tertentu, sehingga direktori tersebut dapat diakses.

Mount Point

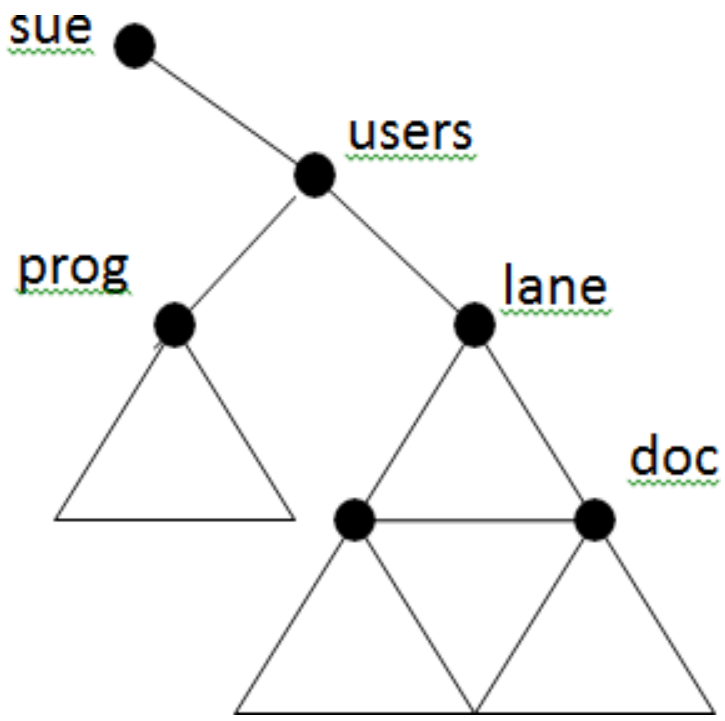
Mount point adalah direktori dimana file baru dapat diakses. Untuk me-mount file system atau direktori, mount point harus berupa direktori, dan untuk me-mount file, mount point juga harus berupa file. Jika file atau direktori yang akan digunakan sebagai mount point berisi data, selama direktori / file tersebut adalah file lain Atau direktori ditetapkan sebagai titik pemasangan, itu tidak dapat diakses.



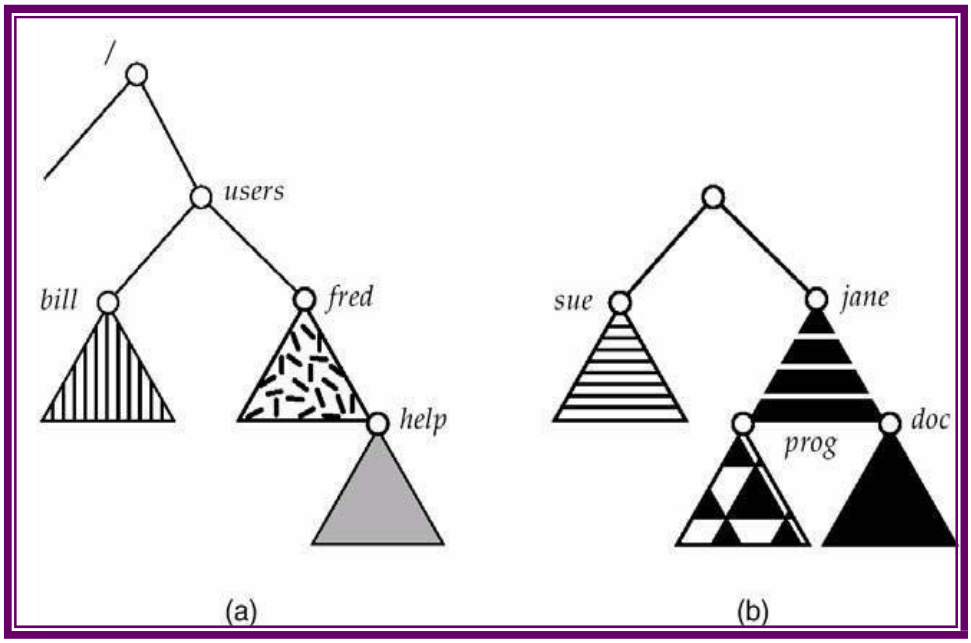
Gambar 10. 1 Mount Point 1



Gambar 10. 2 Mount Point 2

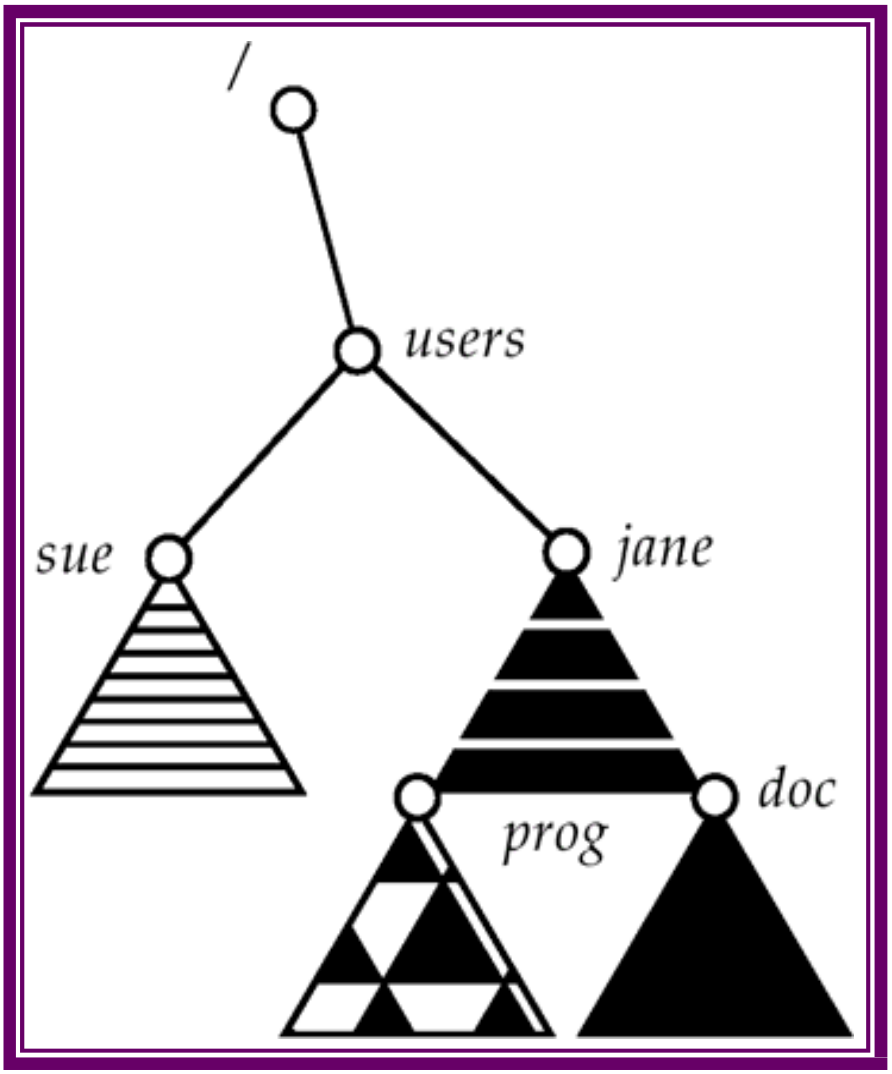


Gambar 10. 3 Mount Point 3



Gambar 10. 4 Mount Point 4

(a) Sistem eksis (b) partisi yang tidak di-mount



Gambar 10. 5 Mount Point 5

Mounting Sistem Berkas, Direktori, dan Berkas

Ada dua jenis penginstalan yaitu,penginstalan jarak jauh dan penginstalan lokal.

1) Instalasi jarak jauh diselesaikan oleh sistem jarak jauh, di mana datanya dikirim melalui jalur telekomunikasi.

2) Selesaikan penginstalan lokal di sistem lokal Setiap sistem file dikaitkan dengan perangkat yang berbeda. Sebelum menggunakan sistem file, itu harus ditautkan ke struktur direktori yang ada (bisa root atau file tambahan lainnya).

Sebagai contoh, kita dapat me-*mount* dari /home/server/database ke *mount point* yang dispesifikasikan sebagai /home/user1, /home/user2, and /home/user3:

/home/server/database /home/user1

/home/server/database /home/user2

/home/server/database /home/user3

Contoh Mounting Lainnya :

C:\mount C:\cygwin\usr\X11R6\li \X11\fonts on/usr/X11R6/lib/X11/fonts/type

system (binmode)C:\cygwin\bin on /usr/bin type system(binmode)

C:\cygwin\lib on /usr/lib type system(binmode)

C:\cygwin on / type system (binmode)

d: on /tmp tpye system (binmode)

c: on /cygdrive/c typer user (binmode,noumount)

f: on /cygdrive/f typer user(binmode,noumount)

g: on /cygdrive/g typer user(binmode,noumount)

h: on /cygdrive/h typer user(binmode,noumount)

x: on /cygdrive/x typer user(binmode,noumount)

2. Sharing

Dapat berbagi berkas dengan pengguna lainnya yang teregistrasi. Hal pertama yang harus kita lakukan adalah menentukan dengan siapa berkas tersebut akan dibagi dan akses seperti apa yang akan diberikan kepada mereka. Berbagi berkas berguna bag pengguna yang ingin bergabung dengan pengguna lain dan mengurangi usaha untuk mencapai sebuah hasil akhir.

Banyak Pengguna

Sistem operasi yang dibuat untuk banyak pengguna, masalah berbagi file, Penamaan file dan perlindungan file sangat penting. Oleh karena itu, sistem operasi harus dapat menampung / mengelola file sharing dengan menyediakan struktur direktori yang memungkinkan pengguna untuk berbagi.

Karena masalah akses file, kami dapat mengizinkan pengguna lain untuk melihat, mengedit, atau menghapus file. Untuk mengedit file menggunakan sistem file web, pertama-tama kita harus menyalin file dari sistem file web ke komputer lokal, mengeditnya di komputer lokal, dan kemudian mengirim file kembali ke sistem dengan nama file yang sama. Misalnya, kami dapat

mengizinkan semua pengguna terdaftar untuk melihat file di direktori (tetapi mereka tidak dapat mengedit atau menghapusnya).

Untuk memberikan contoh lain, kami hanya mengizinkan satu pengguna untuk melakukan operasi apa pun pada direktori dan semua isinya (lihat semua file, edit, Tambahkan file atau bahkan hapus konten file). Sistem file web memungkinkan kita untuk menentukan izin akses di tingkat file. Oleh karena itu, kami dapat mengizinkan semua orang untuk melihat isi direktori, atau hanya mengizinkan beberapa pengguna untuk mengakses direktori. Di kebanyakan sistem, banyak pengguna mengadopsi konsep pemilik file / direktori pengguna dan grup.

1) Pemilik / Pengguna: Pengguna yang dapat mengubah atribut, memberikan hak akses, dan memiliki sebagian besar kontrol dalam sebuah file atau direktori.

2) Grup: beberapa pengguna yang berbagi file.

Remote File Sistem

Dalam metode implementasi Remote File Sistem, terdapat 3 Metode implementasi :

1) FTP (*File Transfer Protocol)* digunakan untuk akses anonim (mentransfer file tanpa memiliki account di sistem remote) dan akses autentik (membutuhkan ijin).

2) DFS (*Disributed File System*) digunakan untuk remote direktori terlihat dari mesin lokal, dengan menggunakan akses autentik

3) WWW (*World Wide Web*) biasanya menggunakan akses anonim untuk remote file sistem.

Client-Server Model

Server : mesin yang berisi berkas

Klien : mesin yang mengakses berkas

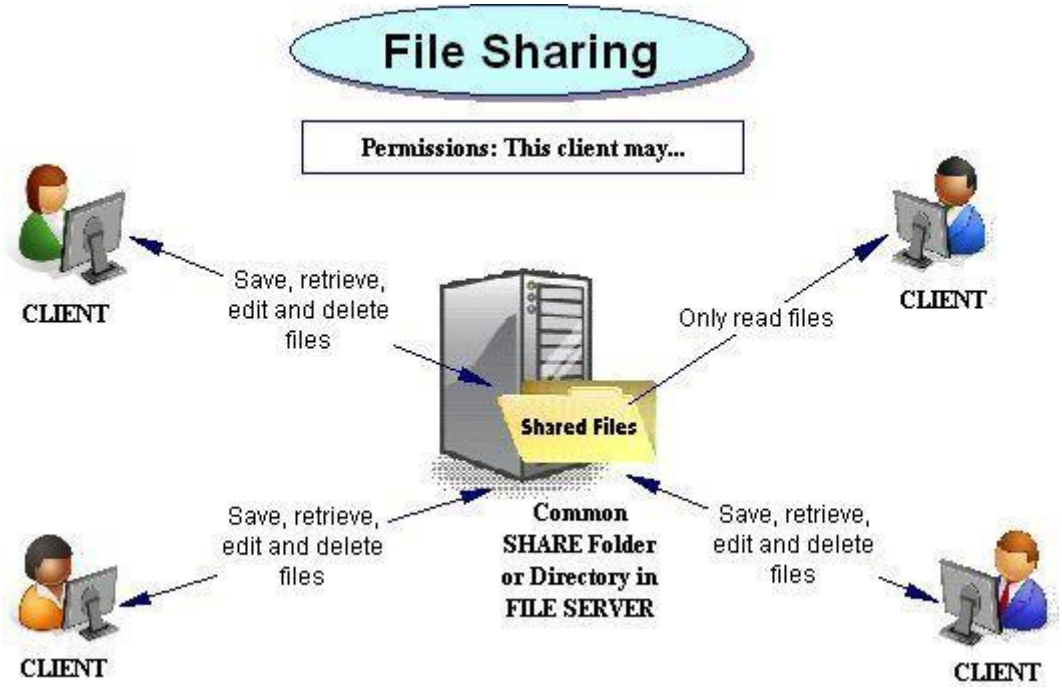
Server dapat menyediakan layanan untuk banyak pengguna, dan klien dapat menggunakan banyak server. Proses identifikasi klien biasanya sangat sulit, metode yang biasa dilakukan adalah melacak alamat IP, tetapi karena alamat IP dapat dipalsukan, metode ini kurang efektif. Ada juga orang yang menggunakan proses kunci enkripsi, tetapi ini lebih rumit karena server klien harus menggunakan algoritma enkripsi yang sama dan pertukaran kunci aman

Unix Semantics

1) Penulisan terhadap suatu file yang terbuka akan langsung terlihat oleh pengguna lain yang sedang membuka file tersebut pada waktu yang

bersamaan.

2) Memperbolehkan pengguna untk men-share pointer dari posisi terakhir.



Session Semantics

Gambar 10. 6 File Sharing

1) Penulisan terhadap suatu file yang terbuka tidak langsung terlihat oleh pengguna lain yang sedang menggunakan file yang sama.

2) Ketika file di tutup, perubahan yang terjadi pada file tersebut hanya dapat dilihat pada sesi start berikutnya.

3. Proteksi

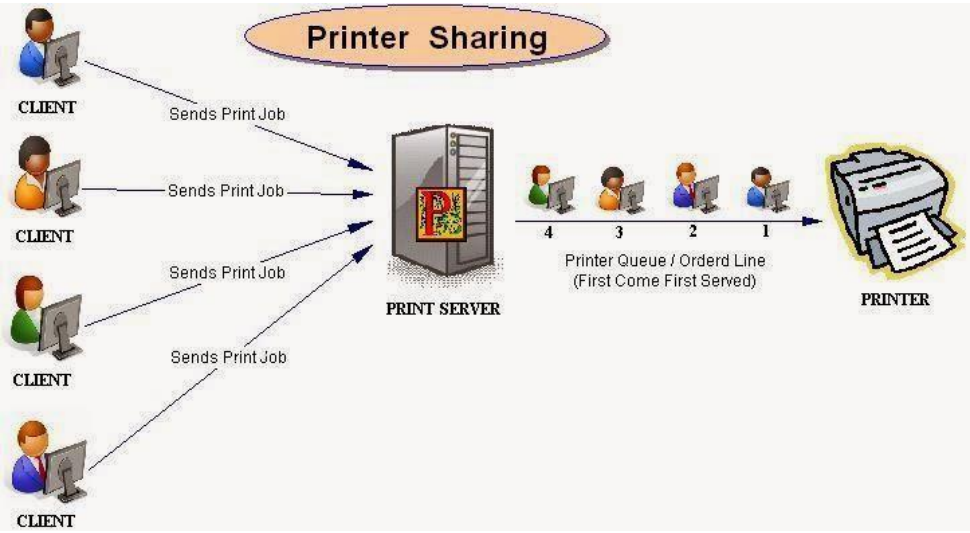
Suatu mekanisme yang digunakan untuk mengamankan berkas(informasi) anda dari akses orang lain. Dalam pembahasan tentang perlindungan file, kita akan lebih banyak membahas tentang aspek keamanan dan mekanisme integritas yang digunakan untuk mencegah file dari gangguan akses eksternal. Dengan memberikan batasan akses kepada setiap pengguna ke file tertentu, mekanisme keamanan file mutlak diperlukan.

Tujuan perlindungan : untuk menghindari *physical damag*e dan *improper access*

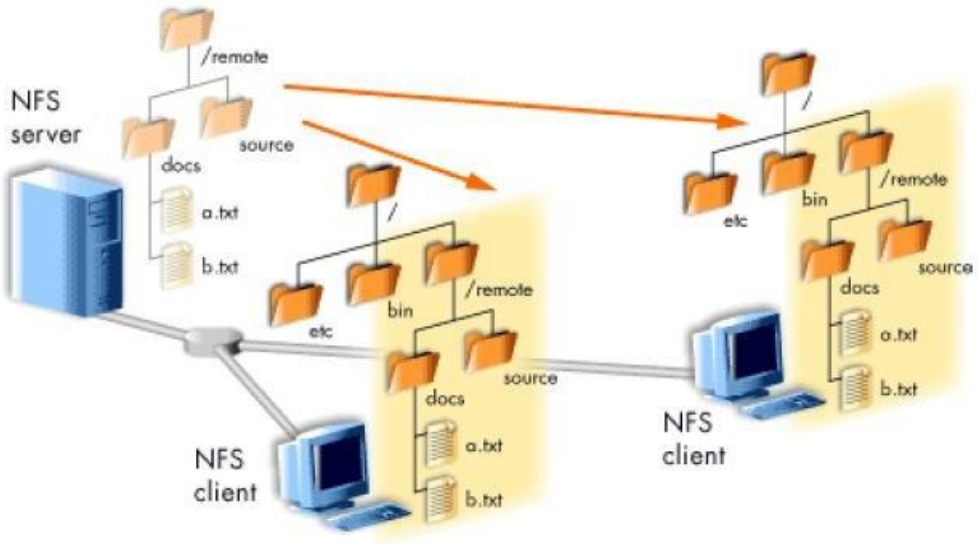
1) Physical damage : masalah hardware

2) Improper access : pengaksesan ke suatu berkas yangtidak sesuai dengan keinginankita

Contoh Sharing



Gambar 10. 7 Printer Sharing



Gambar 10. 8 Network Sharing

Tipe Akses

Perlindungan terkait dengan kemampuan untuk mengakses file tertentu secara langsung. Dari segi panjang, jika sistem menentukan bahwa akses file selalu tertutup atau selalu dirilis ke semua pengguna lain, maka sistem tidak memerlukan mekanisme proteksi. Jenis akses ini dibagi menjadi beberapa jenis (akses terkontrol) yang dapat kita atur dan tentukan. Beberapa jenis akses meliputi:Read/Baca: membaca berkas

1) Write/Tulis: menulis berkas

2) Execute/Eksekusi: memasukkan berkas ke memori dan dieksekusi

3) Append/Sisip: menulis informasi baru pada baris akhir berkas

4) Delete/Hapus: menghapus berkas

5) List/Daftar: mendaftar nama dan atribut berkas

Operasi lain yang mungkin ada pada sistem tertentu (seperti mengganti nama, menyalin, atau mengedit) adalah kombinasi dari jenis kontrol akses di atas. Misalnya, salin file sebagai rangkaian permintaan baca dari pengguna. Dengan cara ini, dalam hal ini, pengguna dengan hak kontrol akses baca juga dapat menyalin, mencetak, dll

Kontrol Akses

Cara paling umum untuk memecahkan masalah perlindungan file adalah dengan membiarkan pengguna menentukan akses ke file secara langsung. Ini dapat dilakukan dengan mengaitkan setiap file atau direktori dengan daftar kontrol akses (ACL) yang berisi nama pengguna dan jenis akses yang diberikan pengguna. Misalnya, dalam sistem VMS, untuk melihat daftar direktori dan daftar kontrol akses, ketik perintah "DIR / SECURITY" atau "DIR / SEC". Salah satu output perintah adalah daftar yang mirip dengan berikut ini.

WWW-HOME.DIR;1 [HMC2000,WWART] (RW,RWED„E)

(IDENTIFIER=WWW_SERVER_ACCESS,OPTIONS=DEFAULT,ACCESS=READ) (IDENTIFIER=WWW_SERVER_ACCESS,ACCESS=READ)

Baris pertama menampilkan nama file WWW-HOME.DIR, lalu menampilkan nama grup pemilik HMC2000 dan nama pengguna WWART di sebelahnya, diikuti dengan grup tipe.akses RW, RWED„E

Keterangan :

R = Baca

W = Tulis

E = Eksekusi

D = Hapus

Dua baris berikutnya disebut daftar kontrol akses. Garis satu-ke-satu disebut entri kontrol akses (ACE) dan terdiri dari tiga bagian, yaitu :

1) Bagian pertama, disebut IDENTIFIER / Identification, menunjukkan nama grup atau nama pengguna (seperti [HMC2000, WWART]) atau otoritas akses khusus (seperti WWW_SERVER_ACCESS).

2) Bagian kedua adalah daftar "opsi / pilihan".

3) Bagian terakhir adalah daftar izin / akses ACCESS, seperti membaca atau mengeksekusi Izin / izin akses ini diberikan kepada siapa saja yang merujuk ke bagian "identifikasi".

Cara kerjanya :

Ketika pengguna meminta akses ke file / direktori, sistem operasi memeriksa daftar kontrol akses untuk melihat apakah nama pengguna ada dalam daftar. Jika registrasi sudah benar, maka permintaan akses akan dikabulkan, begitu pula sebaliknya, permintaan akses akan ditolak. Keuntungan dari metode ini adalah penggunaan metode akses yang kompleks, yang sulit untuk disusupi secara sembarangan. Masalah utamanya adalah ukuran daftar akses. Bayangkan jika kita mengizinkan semua orang untuk membaca file tersebut, maka kita harus mendaftarkan semua nama pengguna dan hak akses baca mereka. Selain itu, teknik ini memiliki dua konsekuensi yang tidak diinginkan :

1) Membuat daftar seperti itu adalah pekerjaan yang membosankan dan tidak efektif.

2) Entri direktori ukuran tetap berukuran variabel sebelumnya, mengarah ke manajemen ruang kosong yang lebih rumit. Masalah ini dapat diatasi dengan menggunakan daftar akses yang disederhanakan. Untuk menyederhanakan ukuran daftar kontrol akses, banyak sistem menggunakan tiga kategori pengguna berikut:

3) Pemilik: pengguna yang membuat file.

4) Grup: Sekelompok pengguna yang berbagi file dan membutuhkan izin akses yang sama.

5) Semesta: semua pengguna

6) Membuat daftar seperti itu bisa menjadi pekerjaan yang membosankan dan tidak efektif.

7) Entri direktori berukuran tetap yang sebelumnya menjadi ukuran variabel, mengarah ke manajemen ruang kosong yang lebih kompleks.

Contoh 1 :

Sistem UNIX di mana kontrol akses dibagi menjadi 3 bagian. Setiap bagian mewakili kategori pengguna (yaitu pemilik, grup, dan semesta). Setiap bagian dibagi lagi menjadi 3 bit tipe akses -rwx, di mana r mengontrol akses baca, w mengontrol akses tulis, dan x mengontrol eksekusi. Dengan cara ini, 9 bit diperlukan untuk merekam semua informasi perlindungan file.

Berikut adalah keluaran dari perintah "ls -al" di sistem UNIX :

-rwxr-x--- 1 david karyawan 12210 Nov 14 20:12 laporan.txt

Baris di atas menyatakan bahwa berkas laporan.txt memiliki akses penuhterhadap pemilik berkas (yi.david), grupnya hanya dapat membaca dan mengeksekusi, sedang lainnya tidak memiliki akses sama sekali.

Contoh 2 :

Pada sistem UNIX, perlindungan direktori ditangani dengan cara yang sama seperti perlindungan file, misalnya, mengaitkan perlindungan direktori dengan setiap subdirektori dan menggunakan pemilik, grup, dan semesta (lainnya) sebagai 3 bit RWX. Informasi dari kiri ke kanan dalam file mencakup perlindungan file atau direktori, jumlah link file, nama pemilik, nama grup, ukuran file dalam byte, tanggal pembuatan, dan nama file.

Tabel 10. 1 Proteksi file dan direktori pada UNIX

-rw-rw-r--	1	pbg	staff	31200	Sep	3	08:30	intro.ps
drwx------	5	pbg	staff	512	Jul	8	09:33	private/
drwxrwxr-x	2	pbg	staff	512	Jul	8	09:35	doc/
drwxrwx---	2	pbg	student	512	Aug	3	14:13	student-proj/
-rw-r—-r--	1	pbg	staff	9423	Feb	24	1993	program.c
-rwxr-xr-x	1	pbg	staff	20471	Feb	24	1993	program
drwx—-x--x	4	pbg	faculty	512	Jul	31	10:31	lib/
drwx------	3	pbg	staff	1024	Aug	29	06:52	mail/
drwxrwxrwx	3	pbg	staff	512	Jul	8	09:35	test/

Pendekatan Pengamanan Lainnya

Cara lain untuk mengatasi masalah perlindungan adalah dengan menetapkan kata sandi untuk setiap file. Jika kata kunci dipilih secara acak dan sering diubah, metode ini sangat efektif karena membatasi akses ke file hanya untuk pengguna yang mengetahui kata kunci tersebut. Cara ini memiliki beberapa kelemahan, diantaranya :

1. Semakin banyak kata kunci yang perlu diingat pengguna, yang tidak praktis.
2. Jika hanya satu kata sandi yang digunakan dalam semua file, begitu orang lain mengetahui kata sandi tersebut, orang tersebut dapat dengan mudah mengakses semua file lainnya. Beberapa sistem (seperti TOPS-20) memungkinkan pengguna memasukkan kata kunci dengan subdirektori untuk menangani masalah ini alih-alih file tertentu.
3. Umumnya, hanya satu kata kunci yang dikaitkan dengan semua file lainnya. Oleh karena itu, keamanan menjadi segalanya atau tidak sama sekali. Untuk

mendukung keamanan lebih detail, kita harus menggunakan banyak kata kunci.

C. Soal Latihan

1. Jelaskan apa yang dimaksud dengan mounting,sharing, dan proteksi ?
2. Dalam mounting, ada berapa prose dan jenis mounting dalam sistem berkas ?
   a. sebutkan prose dan jenis-jenisnya
   b. gambarkan mounting pointnya
3. Ada berapa metode yang ada pada sharing di sistem berkas ?
4. Jelaskan pengertian dari owner dan group pada sharing di sistem berkas ?
5. Buatlah contoh perintah untuk kontrol akses pada proteksi di sistem berkas?

D. Referensi

Masyarakat Digital Gotong Royong (MDGR), Pengantar Sistem Operasi Komputer: Plus Studi Kasus Kernel Linux,

Indonusa Esa Unggul, 2003. Materi Kuliah Sistem Operasi. Universitas Indonusa Esa Unggul.

Silberschatz, Galvin, Gagne. 2002. Operating System Concepts, 6th ed. John Wiley & Sons.

Tananbaum, Andrew S. 1992. Modern Operating System 2nd ed. Engrewood cliffs, New Jersey: Prentice Hall Inc.

Stallings, Williem. 2000. Operating System 4th ed. Prentice Hall.

Operation System Concepts, Sixth Edition; Abraham Silberschatz, dkk, hal 394- 395

PERTEMUAN 11
MANAJEMEN KOLISI

A. Tujuan Pembelajaran

1. Mahasiswa dapat Menjelaskan tentang manajemen kolisi
2. Mahasiswa dapat Menjelaskan tentang linear probing
3. Mahasiswa dapat Menjelaskan tentang linear quentient
4. Mahasiswa dapat Menjelaskan tentang double hashing

B. Uraian Materi

1. Pengertian Manajemen Kolisi

Hash berfungsi untuk mendidtribusikan data secara menyeluruh ke dalam berkas file. Tujuan dari hashing adalah Temukan fungsi untuk mempermudah penghitungan untuk memetakan nilai kunci ke nilai posisi kosong. Salah satu masalah dari fungsi hash adalah menghasilkan banyak kolisi atau sinonim. Jadi timbul kemungkinan bahwa 2 record dengan nilai kunci yang berbeda dipetakan dalam alamat yang sama atau yang sering disebut kolisi.

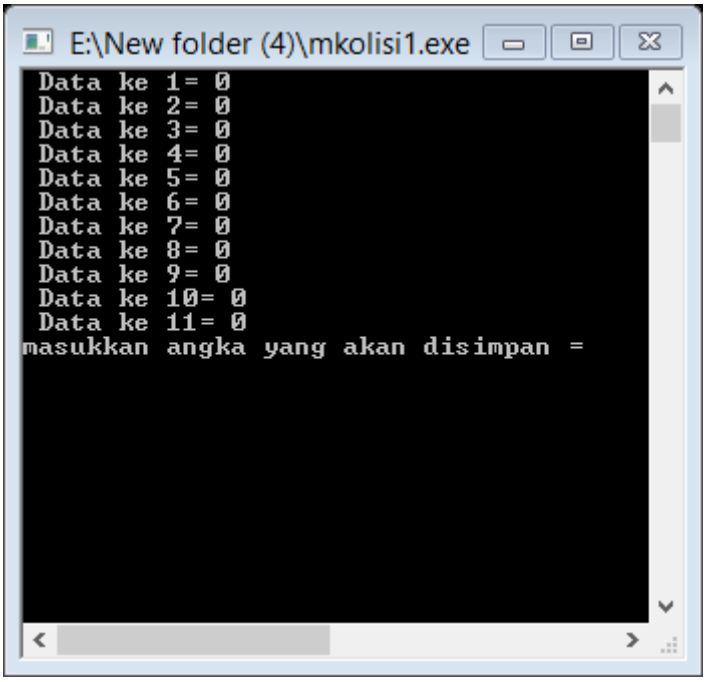
Salah satu cara untuk mengatasi kolisi adalah menyimpan record yang bertubrukan ke dalam alamat selanjutnya. Misalkan kita akan menambahkan record dengan nilai kunci 35 dan dan kapasitas berkasnya adaah 11. Nilai kunci ini akan dilakukan fungsi hash dan didapat alamat / indeks 2. Akan tetapi pada alamat 2 sudah terisi dengan nilai kunci 24.

Maka yang akan kita lakukan adalah mencari lokasi kosong selain 2 dengan cara menambahkan indeks nya dan didapatkan indeks 3. Kemudian dilakukan penelusuran kembali, jika indeks masih kosong amakan record akan disimpan, jika indeks 3 sudah terisi makan akan dilakukan penelusuran kembali dengan menambahkan indeks nya.

Contoh program C++

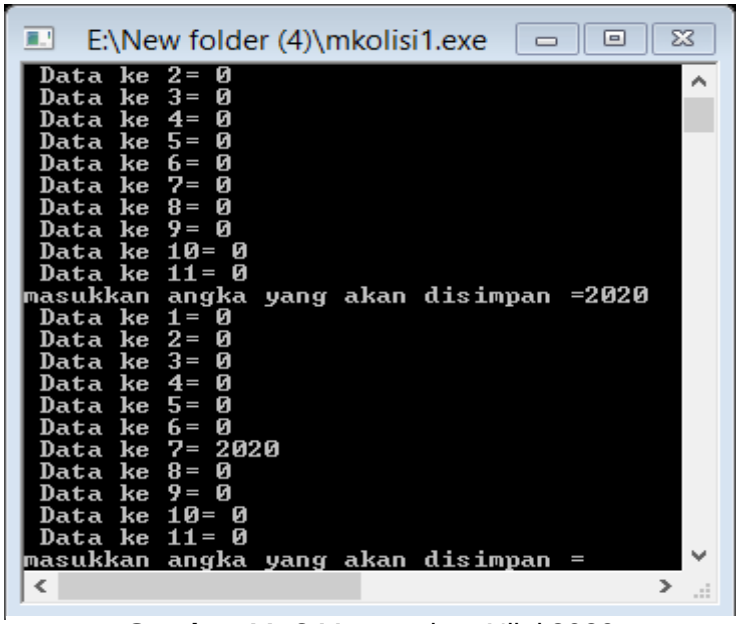
```
#include <iostream>

using namespace std;

int main(){

int data[11]={},a,hash;

awal:

for(int i=0;i<11;i++){

        cout<<" Data ke "<<i+1<<"= "<<data[i]<<endl;

}
```

```
cout<<"masukkan angka yang akan disimpan =";
cin>>a;
hash=a % 11;
for(int i=0;i<11;i++){
        if(i==hash-1){
                if(data[hash-1]==0){ data[hash-1]=a;
                        goto awal;
                }
                else{
                        hash+=1;
                }
        }
}
}
```



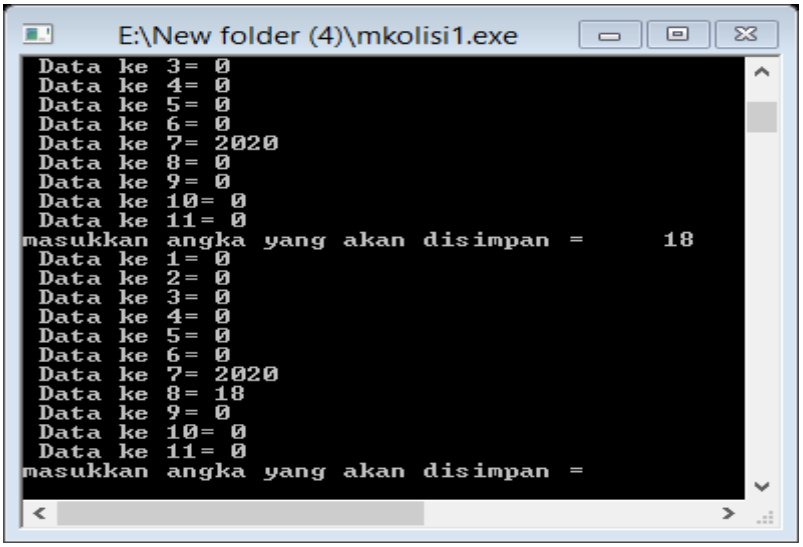
Gambar 11. 1 Contoh Program Mengatasi Kolisi

Pada program diatas lokasi yang kosong diinisialisasikan dengan nilai 0 atau isa dengan “null” untuk memudahkan dalam menunjukkan bawhwa lokasi tersebut kosong. Jadi pada program diatas semua lokasi masih kosong. Kita akan mencoba memasukan sebuah nilai, misalkan 2020.



Gambar 11. 2 Memasukan Nilai 2020

Nilai 2020 akan masuk ke dalam lokasi data ke 7. Kenapa demikian? Kenapa tidak masuk ke lokasi data ke 1 saja? Karena sebelum disimpan akan dilakukan fungsi has terlebih dahulu. Hash = nulai_kunci mod kapasitas berkas. Diketahui kapasitas berkasnya adalah 11 dan nilai_kuncinya adalah 2020, 2020 mod 11 = 7 , makan data akan dimasukan ke lokasi data ke 7.



Gambar 11. 3 Memasukan Nilai Kunci 18

Nilai 18 seharusnya disimpan di lokasi data ke 7, berhubung lokasi 7 sudah ada isinya maka akan dilakukan penelusuran secara sekuensial dimulai dari lokasi berikutnya. Lokasi 8 masih kosong maka data disimpan di lokasi 8. Dan jika lokasi selanjutnya juga sudah ada isinya maka dilakukan penelusuran lagai sampai lokasi kosong ditemukan. Jika nilai kosong tidak ditemukan maka program akan berhenti.

Dalam pengaplikasiannya fungsi hash memetakan beberapa nilai kunci yang berbeda dalam alamat relatif yang sama. Saat tabrakan terjadi, lokasi kosong harus ditemukan untuk nilai kunci tabrakan.

Metode penanganan tabrakan:

1) pengalamatan terbuka (*open aqddressing*)
   a. Linear Probing
   b. Linear quotient
   c. Double hashing/rehasing

2) Coalesced hashing

Pengalamatan Terbuka (Open Addressing)

Dengan mengasunsikan satu nilai kunci yang dicampur(dengan banyak fungsi) ke satu alamat yang telah siap ditempati dengan nilai kunci yang berbeda. Proses open addressing menelusuri tabel untuk sau lokasi kosong dan menyimpan nilai kunci di lokasi kosong yang ditemukan.

Diasumsikan dimakan 1 kunci **k** dipetakan dengan fungsi hash **h** ke satu alamat **A** dalam ukuran table **t. A** dalam rang (value) = 1,2,3,4. ,t adalat nilai kunci berbeda yang berdiri sendiri.

2. Linear Probing

Deteksi linier adalah teknik paling sederhana dalam pengalamatan terbuka. Pilih bilangan bulat positif d (perpindahan). Jika E. coli muncul di posisi A, kembali ke A + d, A + 2d, A + 3d, A + 4d, , A + td, sampai Anda menemukan posisi kosong.

Setiap alamat dihitung dengan modulus (t), asalkan nilainya tidak melebihi kapasitas tabel maksimum. Asumsikan bahwa d adalah bilangan bulat 1.

Urutan pencarian adalah :

A, A + 1, A + 2, A + 3, ..., A + j (= t), 1, 2, 3, ..., A-1

Penelusuran dimulai dari lokasi A sampai nilai A+d = t, kemudian penelusuran dilanjutkan dari lokasi awal tabel sampai lokasi A-1. Dengan cara ini menjamin semua lokasi akan diuji jika diperlukan dan dengan mudah menentukan apakah tabel penuh ketika kembali ke lokasi A.

Contoh :

Diketahui sebuah tabel dengan ukuran tabel 11 dan akan diidi dengan 8 record dengan nilai kunci sebagai berikut: 11,23,54,34,43,73.

Alamat lokasi akan dihitung sebagai berikut :

1) (11 mod 11) +1 = 1,uji di lokasi 1; kosong, simpan 11 di lokasi 1
2) (23 mod 11) + 1 = 2, uji lokasi 2; kosong simpan 23 di lokasi 2
3) (54 mod 11) + 1= 11, uji lokasi 11; kosong, simpan 54 di lokasi 11
4) (34 mod 11) +1 = 2, uji lokasi 2; terjadi kolisi.maka 2 +1 =3, uji lokasi 3; kosong. Simpan 34 di lokasi 3.
5) (43 mod 11) + 1 = 11, uji lokasi 11;terjadi kolisi. 11+1= 12, 12 > t maka pengujian kembali ke lokasi 1,terjadi kolisi. 1+1 2, terjadi kolisi. 2+1 = 3 , tejadi kolisi,. 3+1=4; kosong. Simpan 34 di lokasi 4
6) (73 mod 11) +1 =8, uji lokasi 8; kosong. Simpan 73 di lokasi

Tabel 11. 1 Hashing Dengan Linear Probing

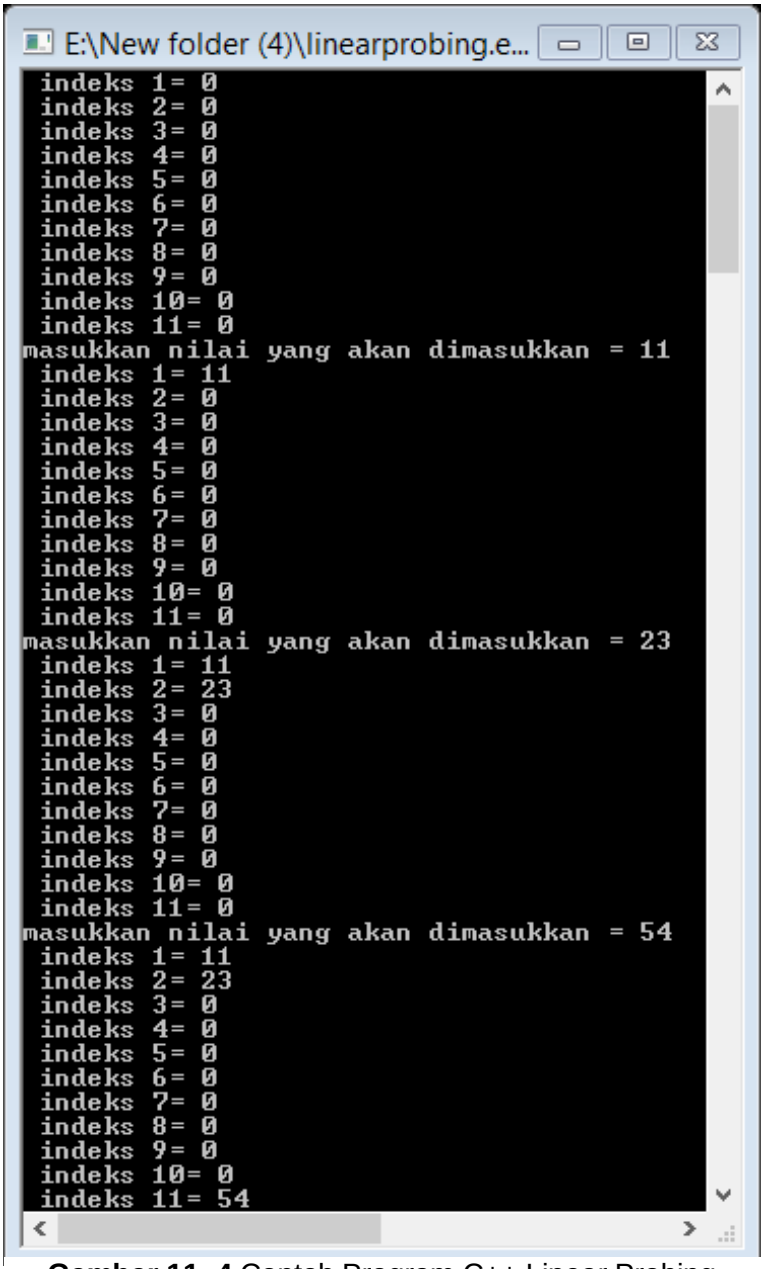
Indeks	1	2	3	4	5	6	7	8	9	10	11
Nilai kunci	11	23	54	34	-	-	-	73	-	-	54

Contoh program C++ linear probing

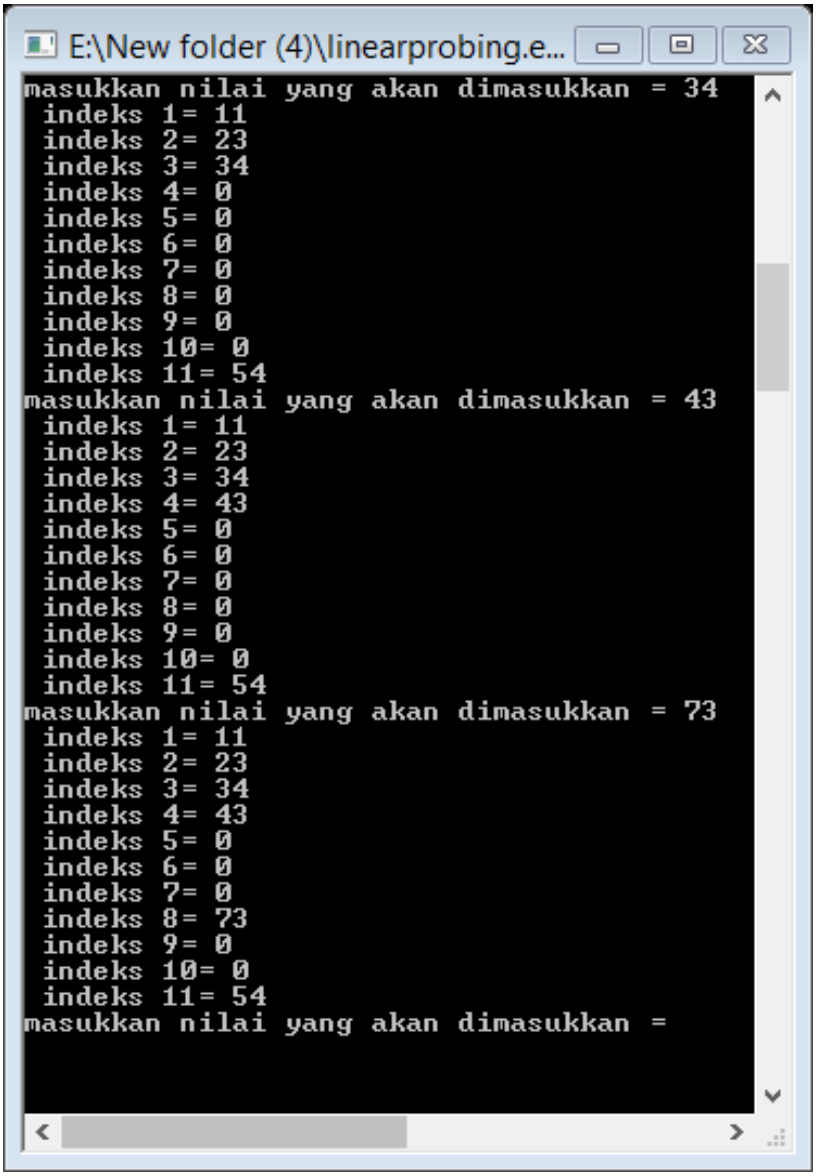
```
#include <iostream>

using namespace std;

int main(){

    int d=1;

    int t=11;

    int data[11]={};

    int a, hash;

    awal:

    cout<<d<<t;

    for(int i=0;i<11;i++){

        cout<<" indeks "<<i+1<<"= "<<data[i]<<endl;

    }

    cout<<"masukkan nilai yang akan dimasukkan = ";
```

```
cin>>a;
hash= (a % t) + 1;
cout<<hash;
if(data[hash-1]==0){
    data[hash-1]=a;
    goto awal;
}
else if(data[hash-1]!=0){
    if(hash!=t){
            for(int i=hash;i<t;i++){
            if(data[i]==0)
            {
                data[i]=a;
                goto awal;
            }
    }
}
    else if(hash==t){
            for(int i=0;i<hash-1;i++){
                if(data[i]==0){
                    data[i]=a;
                    goto awal;
                }
            }
    }
    }
```

}



Gambar 11. 4 Contoh Program C++ Linear Probing



Gambar 11. 5 Contoh Program C++ Linear Probing 2

3. Linear Quotient

Linear Quotient merupakan teknik pengalamatan berdasarkan pembagian (metode hasil bagi linear). Digunakan sebagai contact person untuk fungsi pemisahan hash. Q adalah hasil bagi saat kunci k dibagi dengan ukuran tabel t untuk menentukan alamat A.

Jika q =0 makan akan diatur menjadi 1.

q = k div t

A = k mod t

A_1=(A + q) mod t

Jika diketahui julah kapasitas berkas adalah 11, dan nilai kunci yang akan dimasukan dengan dalam berkas adalah sebagai beriku:

Maka penempatan nilai kunci akan dihitung menggunakan metode linear quentient sebagai berikut :

1) 20 mod 11 = 9 ; uji lokasi 9;kosong. Simpan 20 di lokasi 9;
2) 13 mod 11= 2 ; uji lokasi 2; kosong. Simpan 13 di lokasi 2.
3) 30 mod 11 = 7; uji lokasi 7; kosong. Simpan 30 di lokasi 8.
4) 24 mod 11= 2. Uji lokasi 2; terjadi kolisi (24+(24 div 11)) mod 11=26 mod 11= 4 Uji loasi 4 ; kosong. Simpan 24 di lokasi 4
5) 75 mod 11 = 9; uji lokasi 9. Terjadi kolisi (75+(75 div 11)) mod 11 = 81 mod 11 =4 Uji lokasi 4.terjadi kolisi (75+(81 div 11)) mod 11 =82 mod 11 = 5 Uji lokasi 5; kosong. Simpan 75 di lokasi 5

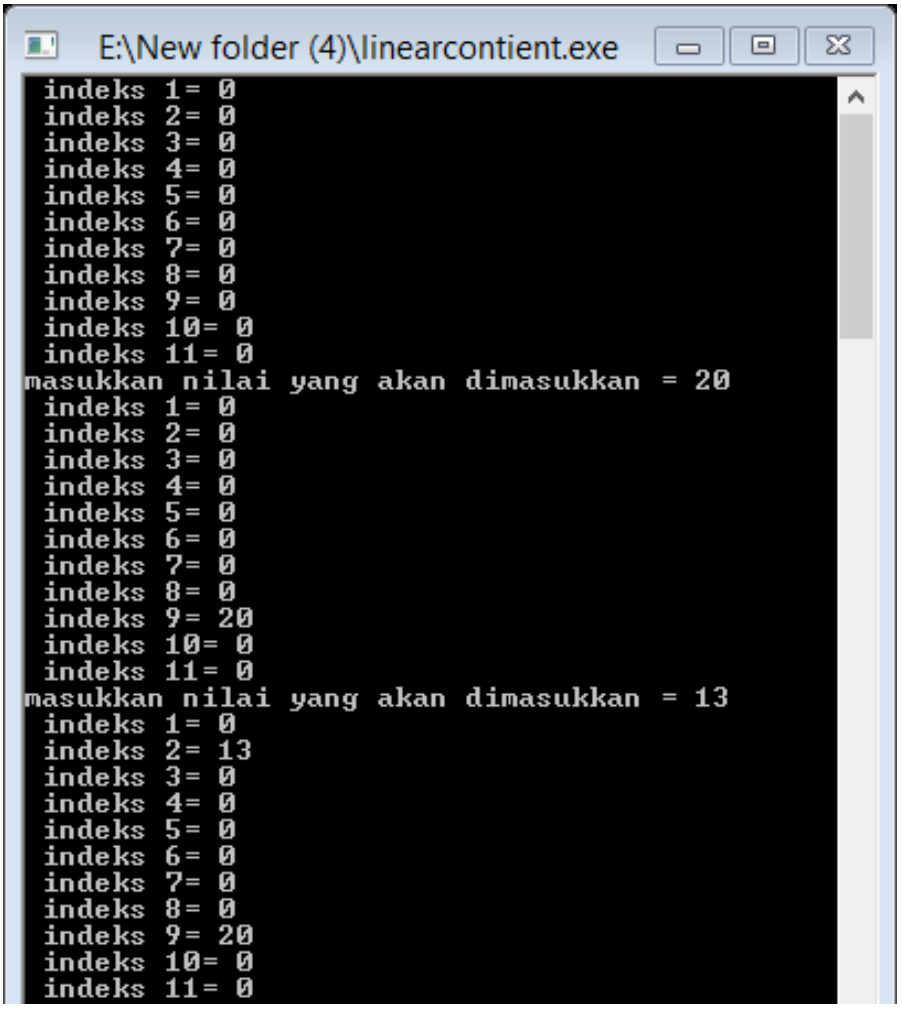
Tabel 11. 2 Linear Quentient

Indeks	1		2	3	4	5	6	7	8	9	10	11
Nilai kunci	-		13	-	24	75	-	-	30	20	-	-

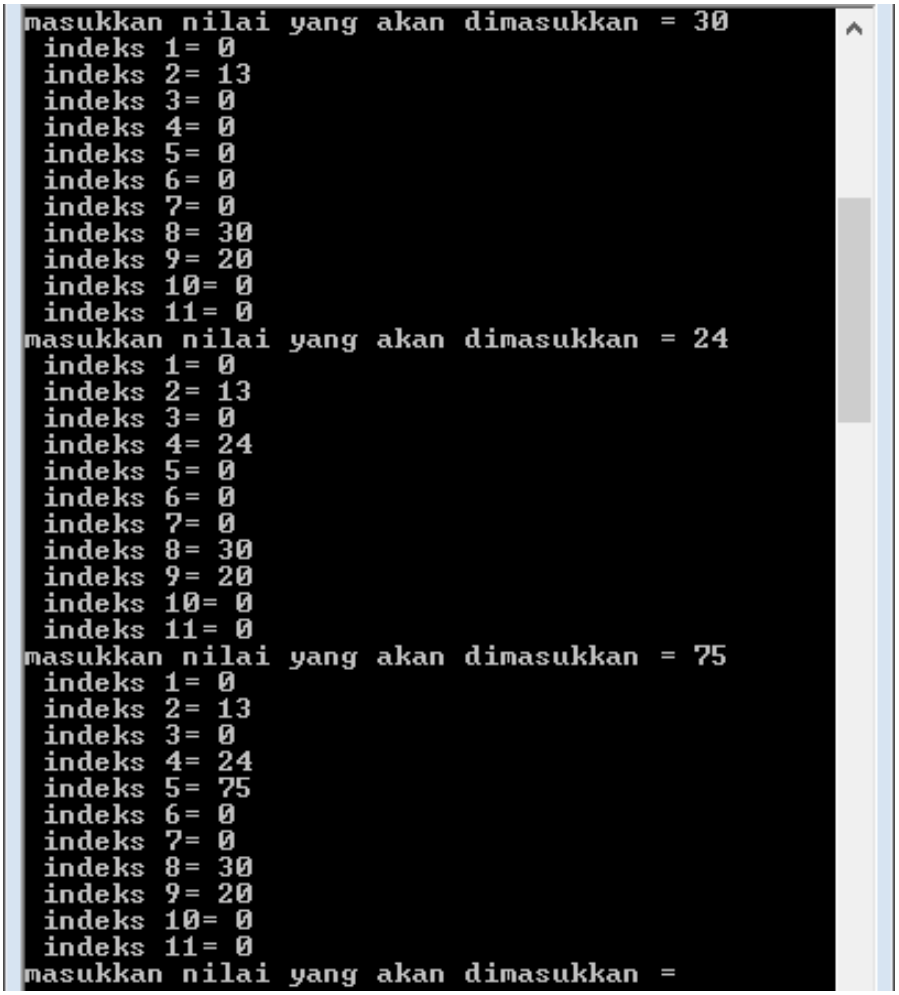
Contohprogram C++ linear quentient

```
#include <iostream>

using namespace std;

int main(){

    int q;

    int t=11;

    int data[11]={};
```

```
        int a, hash,b;
        awal:
        for(int i=0;i<11;i++){
                cout<<" indeks "<<i+1<<"= "<<data[i]<<endl;
        }
        cout<<"masukkan nilai yang akan dimasukkan = ";
        cin>>a;
        hash= (a % t);
        if(data[hash-1]==0){
                data[hash-1]=a;
                goto awal;
        }
        else if(data[hash-1]!=0){
                cariulang:
                q=a/t;
                hash=(a + q) % t;
                        if(data[hash-1]==0){
                                data[hash-1]=a;
                                goto awal;
                        }
                        else if(data[hash-1]!=0)
                        {
                                q=(a+q)/t;
                                goto cariulang;
                        }
        }
}
```



Gambar 11. 6 Contoh Program Linear Contient



Gambar 11. 7 Contoh Program Linear Contient 2

4. Double Hashing

Salah satu implementasi pendekatan ini untuk menghasilkan fingsi hash teknik ini berbeda dengan linear probing karena nilai konstantanya bergantung pada alamat yang dihasilkan oleh fungsi hash yang pertama.

Double hashing mengurangi kemungkinan pengelompokan nilai yang sama. Teknik ii menghasilkan alamat sebagai berikut:

$A_0, A_1, A_2, \ldots\ldots\ldots\ldots A_t$

A_0 = h(k), alamat awal yang dihasilkan oleh hashing
Aj=(A_0 + j^2) od t, untuk j=1,2,3,4....,t

Jika 1 kunci k di hash ke suatu alamat A yang siap ditempatkan, maka secara berturut turut pengujian dibuat di A+1,A+4,A+9,A+16 dan seterusnya

5. Coalesced Hashing

Gabungkan hash adalah metode menggunakan pointer untuk menautkan rekaman dari rantai sinonim.

Tabel 11. 3 Home Address

address	Record	Medan penghubung
1	8	7
2	16	^
3	10	6
4	25	5
5	32	^
6	17	^
7	15	^

Pada tabel diatas menunjukan bahwa semua record memiliki home address, akan tetapi tidak semua memiliki medan penghubung. 8 disipkan perama dan memiliki home address. 16 disiipkan ke address 2. 15 seharusnya disisipkan pada adress 1, akan tetapi sudah terisi dengan record lain, maka 15 disisikan pada lokasi lain dengan menunjukan lokasi yang sekarang pada medan enghubung lokasi aslinya.

Berikut adalah algoritmua untuk coalesced hashing :

1) Hash semua catatan kunci yang akan disisipkan untuk mendapatkan alamat rumah atau alamat kandidat yang mungkin ditempati catatan ini.

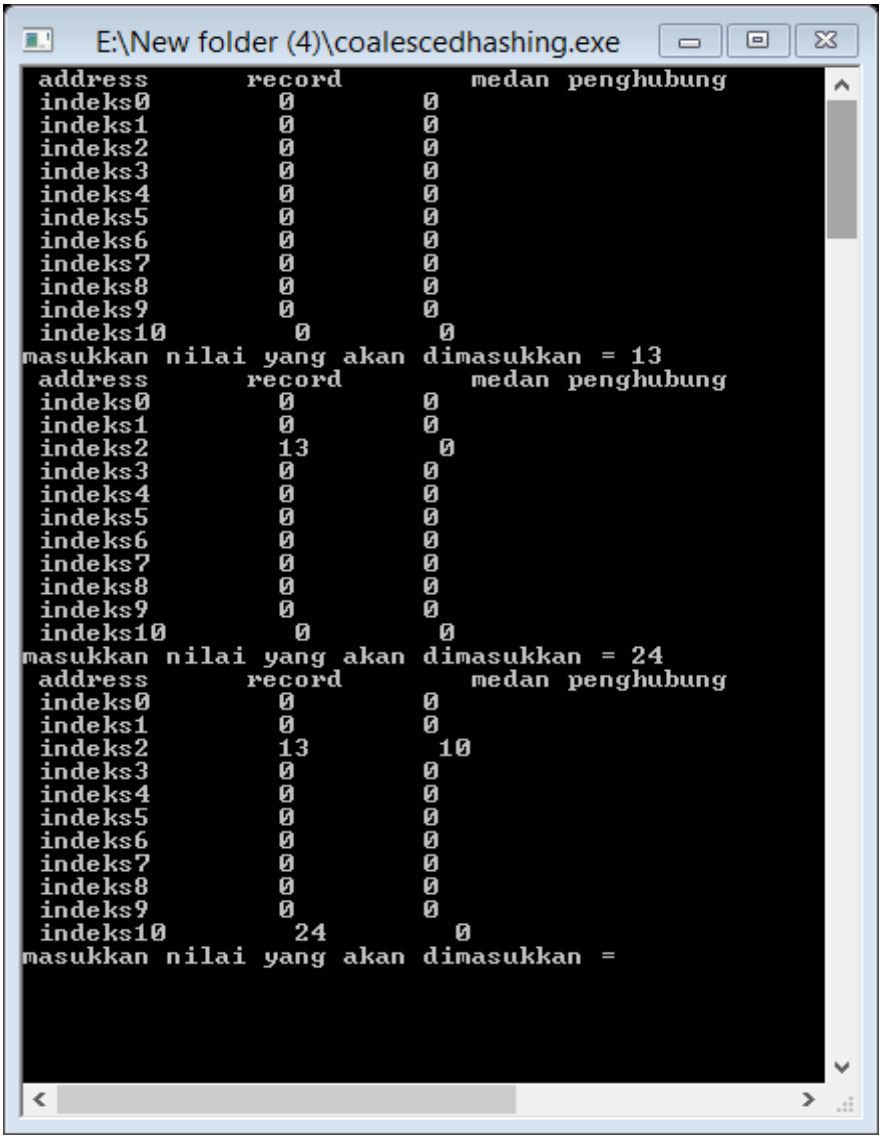
2) Jika posisinya kosong, masukkan record pada posisi itu. Jika posisi sudah diisi, harap akhiri prosedur perekaman ganda, jika tidak:
   a. Temukan rantai terakhir dari rantai sinonim dengan mengikuti penunjuk di bidang koneksi.
   b. Cari lokasi paling bawah berkas, jika tidak ditemukan akhiri program.
   c. Sisipkan record pada lokasi kosong dan atur medan penghubung lokasi aslinya agar menunjukan loksdi ysng bsru.

Contoh program C++ coalesced hashing

```
#include <iostream>

using namespace std;

int main(){

   int d=1;

   int t=11;
```

```
    int data[11][2]={};
    int a, hash;
        for(int i=0;i<11;i++){
        for (int j=0;j<2;j++){
        data[i][j]=0;
        }
        }
awal:
cout<<" address     record      medan penghubung"<<endl;
for(int i=0;i<11;i++){
        cout<<" indeks"<<i;
        for (int j=0;j<2;j++){
                cout<<"      "<<data[i][j];
        }
        cout<<endl;
        }
    cout<<"masukkan nilai yang akan dimasukkan = ";
    cin>>a;
    hash= (a % t) + 1;
    if(data[hash-1][0]==0){
    data[hash-1][0]=a;
    goto awal;
    }
    else if(data[hash-1][0]!=0){
    for(int i=t-1;i>hash-1;i--){
        for(int j=0;j<2;j++){
        if(data[i][0]==0){
```

```
                    data[i][0]=a;
                    data[hash-1][1]=i;
                        goto awal;
                        }
                        else{
                        cout<<"berks penuh";
                        goto awal;
                    }
                }
                }
                goto awal;
            }
        }
```



Gambar 11. 8 Tampilan Program C++ Coalesced Hashing

C. Soal Latihan

1. Jelaskan yang anda ketahui tentang manajemen kolisi?
2. Mengapa kolisi bisa terjadi? Berikan contohnya
3. Jika diketahui kapasitas berkas adalah 15, dan aka disipkan nilai kunci, 13,24,52,64,78,33,42,54,65,73,77,85
   a. hitung menggunakan metode linear probing
   b. hitung dengan menggunakan linear quentient
4. apa perbedaan pen addressing dengan coalesced hashing?
5. Buat sebuah program untuk menyisikkan record dengan mengguakan linear probing!

D. Referensi

Handayani, Dewi. 2001. *Sistem Berkas*. Yogyakarta : J&J Learning.

PERTEMUAN 12
PENGURUTAN REKAMAN

A. Tujuan Pembelajaran

1. Mahasiswa Memahami mengenai algoritma pengurutan .
2. Mahasiswa Mampu mengimplementasikan algoritma pengurutan

B. Uraian Materi

1. Pengurutan Rekaman

Pengurutan rekaman adalah kesanggupan kamputer dalam menyotir data dan itu salah adalah salah satu kebutuhan dalam sistem informasi.record demi record muali dari file transaksi harus disusun dalam urutan, untuk membantu memproses pencarian file utama secara gampang (efisen). Dan juga dalam membuat laporan umumnya pada record yang disusun sesuai urutan tertentu untuk menghasilkan laporan yang mudah dibaca.

Dua macam Pengurutan :

1) Ascending (urutan naik) adalah urutan bilangan yang diurutkan, semakin kecil nilainya maka semakin besar nilainya.

2) Descending (Urutan menurun) adalah kebalikannya, yaitu pengurutan dimulai dari nilai yang lebih besar kemudian ke nilai yang lebih kecil.

Sebagai contoh misalkan diberikan data berupa bilangan berikut ini :

2 9 3 5 6 1 7

Hasil sorting :

Ascending adalah 1 2 3 5 6 7 9

Descending adalah 9 7 6 5 3 2 1

Ada 6 metode pengurutan yang sering seing digunakan adalah :

1) Pengurutan Seleksi (Selection Sort)

2) Pengurutan Gelembung (Bubble Sort)

3) Pengurutan Cepat (Quick Sort)

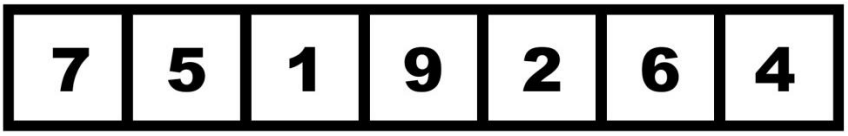
4) Pengurutan Heap (Heap Sort)

5) Pengurutan Shell (Sheel Sort)

2. Pengurutan Seleksi (Selection Sort)

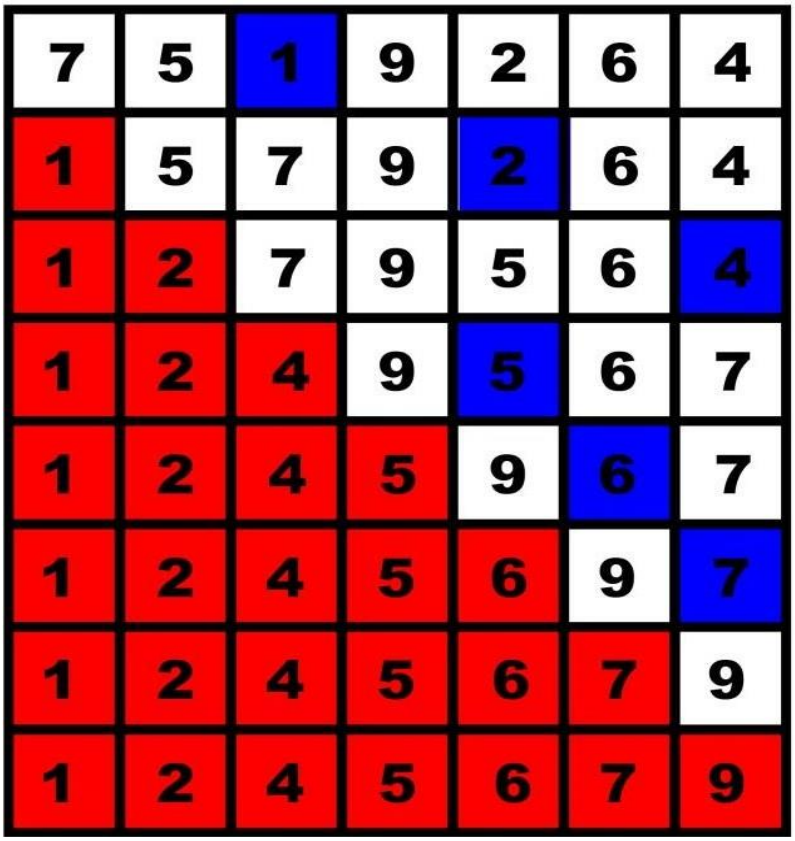
Algoritme pengurutan sederhana lainnya adalah pengurutan selektif. Lakukan beberapa langkah untuk melakukan pemilihan elemen struktur data. Untuk jenis ascending, ini adalah ide dasarnya.

(Ascending order), elemen terkecil di antara elemen yang tidak diurutkan akan menyimpan indeks, dan kemudian nilai elemen dengan indeks yang disimpan akan ditukar dengan elemen pertama yang tidak disortir. Di sisi lain, untuk mengurutkan dalam urutan turun, maka tukarkan elemen indeks terbesar. Perhatikan array berikut



Gambar 12. 1 Selection Sort

Menentukan elemen terkecil dalam daftar tak berurutan dalam satu tarikan napas. Elemen terkecil ini diwarnai biru. Untuk bagian larik yang diurutkan, warnanya merah.

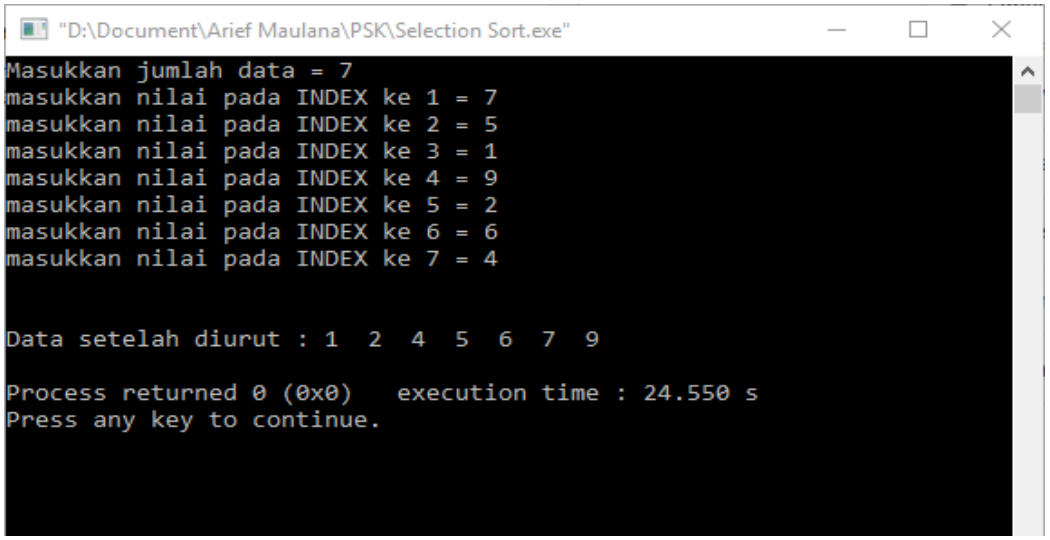


Gambar 12. 2 Elemen Terkecil

Contoh program menggunakan selection sort :

```
#include <stdio.h>

int main()
{
   int a,b,temp,total;
   int data[20];
   printf("Masukkan jumlah data = ");scanf("%d",&total);
   for(a=0;a<total;a++)
   {
      printf("masukkan nilai pada INDEX ke %d = ",a+1);scanf("%d",&data[a]);
   }

   for(a=0;a<total-1;a++)
   {
      for(b=a+1;b<total;b++)
      {
         if(data[a]>data[b])
         {
            temp=data[b];
            data[b]=data[a];
            data[a]=temp;
         }
      }
   }

   printf("\n\nData setelah diurut : ");
   for(a=0;a<total;a++)
   {
      printf("%d  ",data[a]);
   }
   printf("\n");

   return 0;
}
```

Gambar 12. 3 Program Selection Sort



Gambar 12. 4 Program Selection Sort 2

3. Pengurutan Gelembung (Bubble Sort)

Salah satu proses penyortiran yang paling sederhana. Ini disebut pengurutan gelembung karena setiap akor akan menggelembung perlahan ke posisi yang benar.

Satu karakter semacam ini sangat mudah dimengerti. Namun, di antara semua metode pengurutan yang dibahas, metode pengurutan gelembung yang paling tidak mudah (efektif) disebut pengurutan gelembung, karena setiap tombol perlahan-lahan akan menggelembung ke posisi yang benar.

Proses pengurutan Bubble Sort

Berikut merupakan proses pengurutan Bubble Sort dengan array "7 9 4 1 5".

Proses pertama :

(7 9 4 1 5) menjadi (7 9 4 1 5)

(7 9 4 1 5) menjadi (7 4 9 1 5)

(7 4 9 1 5) menjadi (7 4 1 9 5)

(7 4 1 9 5) menjadi (7 4 1 5 9)

Proses kedua :

(7 4 1 5 9) menjadi (4 7 1 5 9)

(4 7 1 5 9) menjadi (4 1 7 5 9)

(4 1 7 5 9) menjadi (4 1 5 7 9)

(4 1 5 7 9) menjadi (4 1 5 7 9)

Proses ketiga :

(4 1 5 7 9) menjadi (1 4 5 7 9)

(1 4 5 7 9) menjadi (1 4 5 7 9)

(1 4 5 7 9) menjadi (1 4 5 7 9)

(1 4 5 7 9) menjadi (1 4 5 7 9)

Jika diperhatikan, ketiga proses tersebut masih berjalan, jadi tidak ada single swap dalam satu proses. Lakukan proses ini untuk verifikasi data. Kelebihan dari metode ini adalah merupakan metode yang paling sederhana, namun kekurangannya adalah kurang efisien karena jika memilah data yang besar maka prosesnya akan memakan waktu yang lama.

Contoh program sorting menggunakan Bubble Sort :

```c
#include <stdio.h>

int main()
{
   int i,j,n,t, A[100];
   printf("Masukkan banyak data : "); scanf("%d", &n);

   for(i=1; i<=n; i++)
   {
      printf("Data %d = ", i); scanf("%d", &A[i]);
   }

   for(i=1; i<=(n-1); i++)
   {
      for(j=n; j>=(i+1); j--)
      {
         if(A[j-1]>A[j])
         {
            t=A[j-1];
            A[j-1] = A[j];
            A[j] = t;
         }
      }
   }

   printf("\nUrutannya adalah : ");
   for(i=1; i<=n; i++)
   {
      printf("%d ", A[i]);
   }
   printf("\n");
   return 0;
}
```

Gambar 12. 5 Program Sorting Menggunakan Bubble Sort

4. Pengurutan Cepat (Quick Sort)

Bandingkan elemen (disebut pivot) dengan elemen lain dan susun sehingga elemen lain yang lebih kecil dari poros berada di sebelah kiri, dan elemen lain yang lebih besar dari poros ada di sebelah kanan.

Oleh karena itu, terbentuk dua sublist, yang terletak di kiri dan kanan poros. Kemudian di sublist kiri dan kanan, kami mempertimbangkan daftar baru dan melakukan proses yang sama seperti sebelumnya. Begitu seterusnya, hingga tidak ada lagi sublist. Jadi ada proses rekursif.algoritma metode quick dapat dituliskan sebagai berikut :

Algoritma metode quick dapat dituliskan sebagai berikut :

1. algoritma quicksort
2. deskripsi
3. x data[(L+R) div 2]
4. I L
5. J R
6. while (I < = J) do
7. while (data[I] < x) do inc(I) endwhile
8. while (data[J] > x) do dec(J) endwhile
9. If (I < = J) then
10. tukar(data[I],data[j])
11. inc(I)
12. Dec(J)
13. Endif
14. Endwhile
15. If (L < J) then quicksort(larikpixel, L, J)
16. If (I < R) then quicksort(larikpixel, I, R)

Contoh program sorting menggunakan Bubble Sort :

```
#include<stdio.h>
void quicksort (int a[], int lo, int hi){

int x=a[(lo+hi)/2];
    while (i<=j){
      while (a[i]<x) i++;
        while (a[j]>x) j--;

        if (i<=j)
        {
            h=a[i]; a[i]=a[j]; a[j]=h;
            i++; j--;
        printf("besar: %d | ",a[i]);
     printf("pivot: %d | ",x);
     printf("kecil: %d | \n",a[j]);
          for(s=0; s<10;s++){
                        printf("%d - ",a[s]);
                    }
                        printf("\n");
                        printf("\n");

        }

    } ;

    if (lo<j) quicksort(a, lo, j);
    if (i<hi) quicksort(a, i, hi);
}
void printlist(int list[],int n){
   int i;
   for(i=0;i<n;i++)
     printf("%d - ",list[i]);
}

main(){
 int MAX_ELEMENTS = 10;

    int list[10] ={12,35,9,11,78,17,23,15,31,20};

   printf("\nThe list before sorting is: \n");
   printlist(list,MAX_ELEMENTS);
    printf("\n=====================================================\n"
        );
   // sort the list using quicksort
   quicksort(list,0,MAX_ELEMENTS-1);
 printf("\n");
   // print the result
   printf("The list after sorting using quicksort algorithm:\n");
   printlist(list,MAX_ELEMENTS);
    printf("\n");;
}
```

Gambar 12. 6 Program Sorting Menggunakan Bubble Sort 2

```
C:\>g++ quick3.cpp -o quick3

C:\>quick3

The list before sorting is:
12 - 35 - 9 - 11 - 78 - 17 - 23 - 15 - 31 - 20 -
================================================
besar: 17 ¦ pivot: 78 ¦ kecil: 31 ¦
12 - 35 - 9 - 11 - 20 - 17 - 23 - 15 - 31 - 78 -

besar: 9 ¦ pivot: 20 ¦ kecil: 23 ¦
12 - 15 - 9 - 11 - 20 - 17 - 23 - 35 - 31 - 78 -

besar: 20 ¦ pivot: 20 ¦ kecil: 17 ¦
12 - 15 - 9 - 11 - 17 - 20 - 23 - 35 - 31 - 78 -

besar: 15 ¦ pivot: 9 ¦ kecil: 15 ¦
9 - 15 - 12 - 11 - 17 - 20 - 23 - 35 - 31 - 78 -

besar: 12 ¦ pivot: 12 ¦ kecil: 12 ¦
9 - 11 - 12 - 15 - 17 - 20 - 23 - 35 - 31 - 78 -

besar: 15 ¦ pivot: 12 ¦ kecil: 11 ¦
9 - 11 - 12 - 15 - 17 - 20 - 23 - 35 - 31 - 78 -

besar: 17 ¦ pivot: 15 ¦ kecil: 12 ¦
9 - 11 - 12 - 15 - 17 - 20 - 23 - 35 - 31 - 78 -

besar: 35 ¦ pivot: 23 ¦ kecil: 20 ¦
9 - 11 - 12 - 15 - 17 - 20 - 23 - 35 - 31 - 78 -

besar: 35 ¦ pivot: 35 ¦ kecil: 31 ¦
9 - 11 - 12 - 15 - 17 - 20 - 23 - 31 - 35 - 78 -


The list after sorting using quicksort algorithm:
9 - 11 - 12 - 15 - 17 - 20 - 23 - 31 - 35 - 78 -

C:\>_
```

Gambar 12. 7 Program Sorting Menggunakan Bubble Sort 3

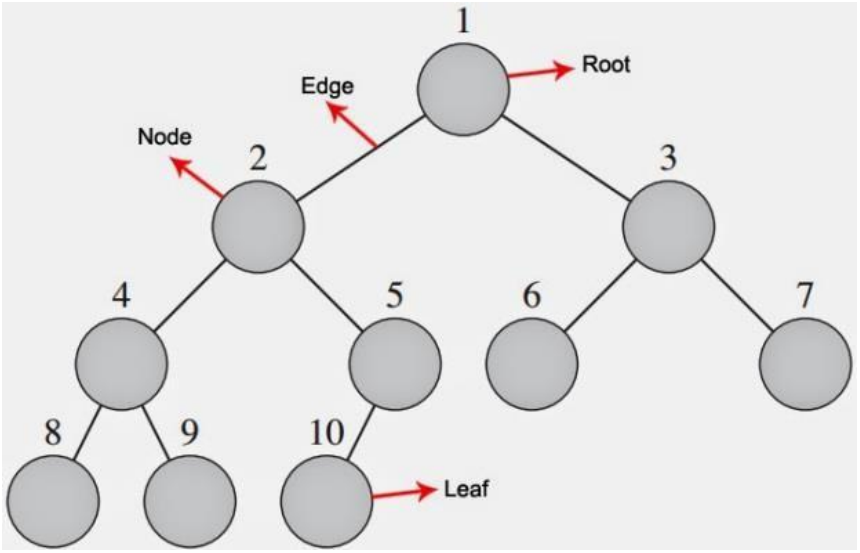
5. Pengurutan Heap (Heap Sort)

Heap Sort adalah algoritma pengurutan yang paling lambat dari algoritma dengan kompleksitas O (n log n). Tetapi tidak seperti algoritma Merge Sort dan Quick Sort, algoritma Heap Sort tidak membutuhkan banyak rekursif atau menggunakan banyak tabel (array).

Oleh karena itu, Heap Sort adalah pilihan yang baik untuk kumpulan data yang besar. Algoritme dimulai dengan membuat tabel heap, membuat heap dari kumpulan data, lalu memindahkan jumlah record terbesar ke akhir kumpulan hasil.

Array heap kemudian dibangun kembali dan kemudian menempatkan item terbesar di sebelah item yang sebelumnya dipindahkan. Ini diulangi sampai tabel heap habis. Secara umum, algoritma ini membutuhkan dua tabel, satu tabel untuk menyimpan heap, dan satu tabel untuk menyimpan hasil. Meskipun lebih lambat dari merge atau quicksort, algoritme ini cocok untuk digunakan dengan ukuran data yang besar.

Dalam heap sorting terdapat 3 bagian yaitu node, edge dan leaf, dimana node adalah masing-masing indeks dalam array, edge adalah garis yang menghubungkan setiap node, dan leaf adalah setiap node tanpa anak (node turunan). Selain itu, ada root, yang merupakan node awal di heap, seperti yang ditunjukkan pada gambar.



Gambar 12. 8 Pohon Heap

Pohon heap adalah pohon biner lengkap (CBT Complete Binary Tree), di mana nilai kunci pada simpul membuat nilai kunci pada simpul anak tidak lebih besar dari nilai kunci pada simpul induk.

Pohon heap terbagi menjadi dua jenis yaitu Max-Heap dan Min-Heap, dimana max-heap adalah kondisi pohon heap yang memiliki nilai tertinggi pada simpul akar, dan nilai setiap simpul anak lebih kecil dari nilai simpul induknya. Min-heap dan max-heap adalah kondisi yang berlawanan. Dalam min-heap, nilai minimum terletak di node root, dan nilai setiap node turunan lebih besar dari nilai node induknya. Dalam metode penyortiran heap, tipe pohon heap yang digunakan adalah Max-Heap.

Dan memvisualisasikan array sebagai pohon heap adalah menemukan simpul akar terlebih dahulu, yaitu, simpul pertama dari pohon heap adalah indeks pertama dalam larik, dan indeksnya adalah 0, tetapi simpul awal pada pohon heap berada di posisi 1, dan larik dengan indeks Nilai awal yang berbeda yaitu indeks 0. Setelah menemukan node root, sekarang hanya perlu mencari node anak dari node root, kemudian membagi node anak menjadi dua, yaitu node anak kiri dan node anak kanan, dan mencari node anak kiri, node anak kanan dan node induk, gunakan rumus berikut :

Left Child : 2i (Contoh : Left child dari 1 adalah 2 x 1 = 2)

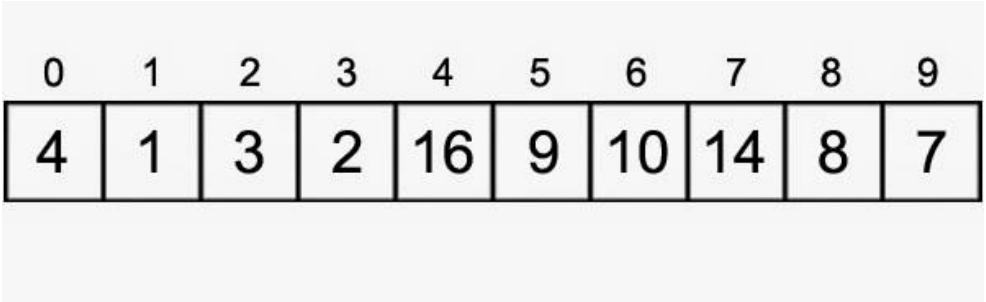
Right Child : 2i + 1 (Contoh : Right Child dari 1 adalah (2 x 1) + 1 = 3)

Parent : $\lfloor i/2 \rfloor$ (Contoh : Parent dari 3 adalah 3 / 2 = 1)

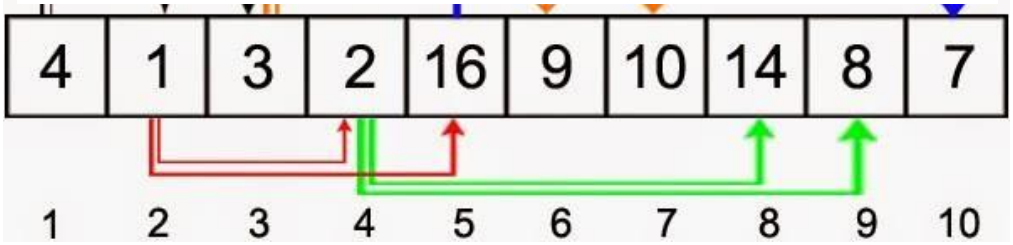
NB : Untuk i adalah posisi node yang ingin dicari left/right childnya atau parent nodenya dan untuk lambing ($\lfloor \rfloor$) adalah floor yaitu pembulatan kebawah missal

3 / 2 = 1,5 dibulatkan kebawah menjadi 1. Berikut adalah contoh cara memvisualisasikan sebuh array menjadi sebuah heap tree :

Contoh : Kita memiliki sebuah aray A = 4, 1, 3, 2, 16, 9, 10, 14, 8, 7. Dan untuk memvisualisasikan array tersebut gunakan rumus yang sudah disediakan dan prosesnya akan terlihat seperti ini :



Gambar 12. 9 Visualisasi Array



Gambar 12. 10 Visualisasi Array 2

Pengurutan Shell (Sheel Sort)

Metode ini juga disebut metode pertumbuhan yang semakin berkurang. Metode ini dikembangkan oleh Donald L. Shell pada tahun 1959 dan oleh karena itu sering disebut sebagai metode pemilahan shell. Metode ini mengurutkan data dengan cara membandingkan satu data dengan data lain yang memiliki jarak tertentu. Secara singkat metode ini dijelaskan sebagai berikut. Pada langkah pertama, kita mengambil elemen pertama dan membandingkannya dan membandingkannya dengan elemen yang agak jauh dari elemen pertama.

Kemudian kami membandingkan elemen kedua dengan elemen lainnya pada jarak yang sama seperti di atas. Begitu seterusnya sampai semua item dibandingkan. Pada tahap kedua, proses tersebut diulangi secara bertahap.

yang lebih kecil, pada langkah ketiga jarak dikurangi lagi, seluruh proses dihentikan jika jaraknya sama dengan satu. Contoh proses penyortiran menggunakan metode Shell Sort.

Jarak	A[1]	A[2]	A[3]	A[4]	A[5]	A[6]
Awal	22	10	15	3	2	8
Jarak = 3	22	10	15	3	2	8
	3	10	15	22	2	8
	3	2	15	22	10	8
	3	2	8	22	10	15
Jarak = 1	3	2	8	22	10	15
	2	3	8	22	10	15
	2	3	8	22	10	15
	2	3	8	22	10	15
	2	3	8	10	22	15
	2	3	8	10	15	22
Akhir	2	3	8	10	15	22

Gambar 12. 11 Proses Penyortiran Metode Shell Short

C. Soal Latihan

1. Tuliskan algoritma pengurutan bubble sort secara ascending.
2. Tuliskan algoritma pengurutan shell sort secara ascending.

D. Referensi

Handayani, Dewi. 2001. Sistem Berkas. Yogyakarta:J&J Learning
https://rizkinawawi.wordpress.com/2020/07/08/pertemuan-12-pengurutan-rekaman/
https://matheusrumetna.com/materi/sistem_berkas/BAB_VII_PENGURUTAN_REKAMAN.pdf

PERTEMUAN 13
BERKAS SORT DAN MERGE

A. Tujuan Pembelajaran

1. Mahasiswa dapat menjelaskan tentang berkas sort dan merge
2. Mahasiswa dapat menjelaskan tentang teknik sort dan merge

B. Uraian Materi

1. Organisasi Berkas Sort dan Merge

Pengertian Berkas Sort dan Merge

1) Suatu kebutuhan dalam sistem informasi adalah kemampuan computer untuk menyortir data.

2) Record dari berkas transaksi harus disusun dalam urutan tertentu dalam mengupdatean sekuensial master file secara efisien.

3) Membantu membuat laporan agar mudah dibaca

Dalam sistem penyortiran dikenal dua metode, yaitu :

1) Metode Sort Internal

2) Metode Sort Eksternal

Metode Sort internal

Semua record yang akan diproses dimuat kedalam memori komputer lalu diproses sort (sortir)

Metode Sort Eksternal

Record-record diproses tidak semuanya dapat dimuat ke dalam memori komputer, karena keterbatasan memori computer.

Perbedaannya :

1) Pada metode sort internal, semua record yang akan diproses dimuat ke dalam memori komputer lalu diproses sort (sortir)

2) Pada metode sort eksternal, record-record yang diproses tidak semuanya dapat dimuat ke dalam memori komputer, karena keterbatasan memori komputer.

3) Metode sort eksternal di dalam penerapannya nanti, menggunakan pula metode sort internal.

Contohnya :

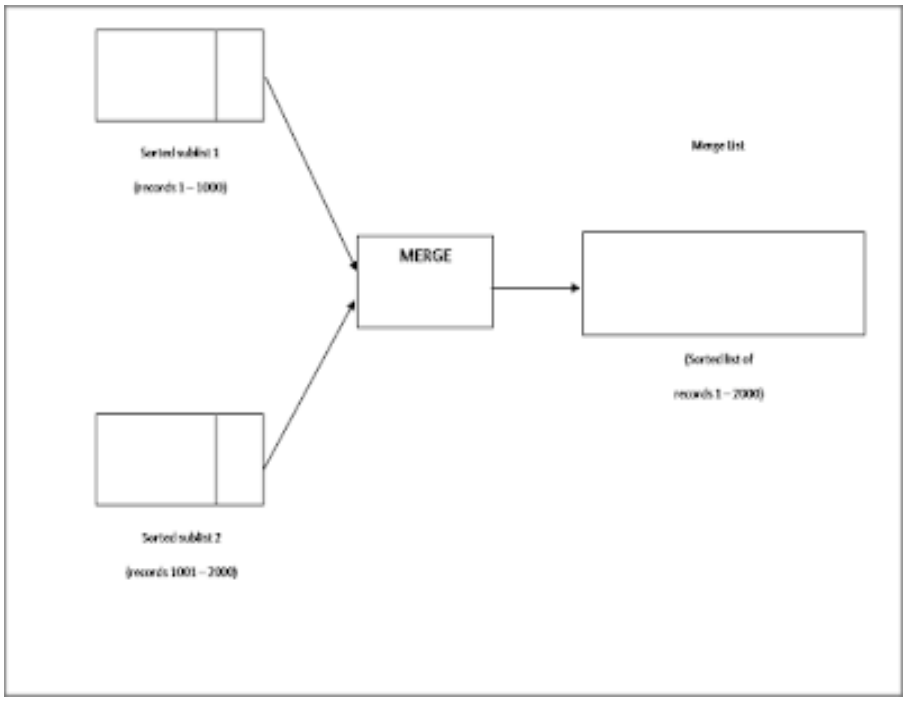
Sebuah file berisi 2000 record harus disortir ke dalam memori yang hanya dapat menampung 1000 record sekaligus. Untuk itu digunakan metode sort eksternal.

Langkah-langkah penyortiran ini adalah:

1) Record-record dibagi ke dalam beberapa file agar dapat ditampung sekaligus di memori komputer, lalu masing-masing bagian di sortir internal. Bagian-bagian file yang telah tersortir ini disebut sorted sublist.

   Maka didapat : Sorted sublist 1 (record 1-1000) dan Sorted sublist 2 (record 001-2000)

2) Setelah itu kedua sorted sublist ini (RUN) digabung (merge), sehingga didapat berkas gabungan (merge file) yang record-recordnya telah disortir.



Gambar 13. 1 Langkah-langkah Penyortiran

Bisa kita simpulkan langkah – langkah untuk metode sort ekstrenal ini adalah :

1) Internal sort, dimana file dibagi menjadi beberapa bagian file kemudian di sortir.

2) Merge, dimana bagian-bagian file ini (sorted sublist) digabung menjadi satu atau lebih file gabungan. File-file gabungan ini kemudian digabung lagi sampai akhirnya didapatkan sebuah file gabungan yang berisi semua record-record yang telah tersortir.

3) Output, dimana menyalin file gabungan yang telah disortir kemudian storage terakhir.

Faktor yang mempengaruhi metode pemeringkatan eksternal :

1) Jumlah record yang akan disortir.

2) Ukuran rekaman (panjang rekaman).

3) Jumlah penyimpanan yang digunakan.

4) Kapasitas memori.

5) Tetapkan nilai kunci dalam file input

Teknologi pengurutan / penggabungan file berbeda satu sama lain dalam beberapa cara berikut :

1) Metode penyortiran internal digunakan

2) Jumlah memori utama yang dicadangkan untuk klasifikasi internal

3) Mendistribusikan sublist yang diurutkan di penyimpanan sekunder ke satu atau beberapa file yang digabungkan melalui langkah penggabungan (proses penggabungan)

Ada 4 teknik dalam sort/merge file, yaitu :

1) Natural Merge

2) Balanced Merge

3) Polyphase Merge

4) Cascade Merge

Natural Merge

Merge yang menangani dua input file sekaligus disebut two way

Natural Merge

Merge yang menangani M input file sekaligus disebut M way natural merge. M menunjukkan derajat merge.

Pada natural merge terbagi lagi menjadi :

2 way natural merge

3 way natural merge

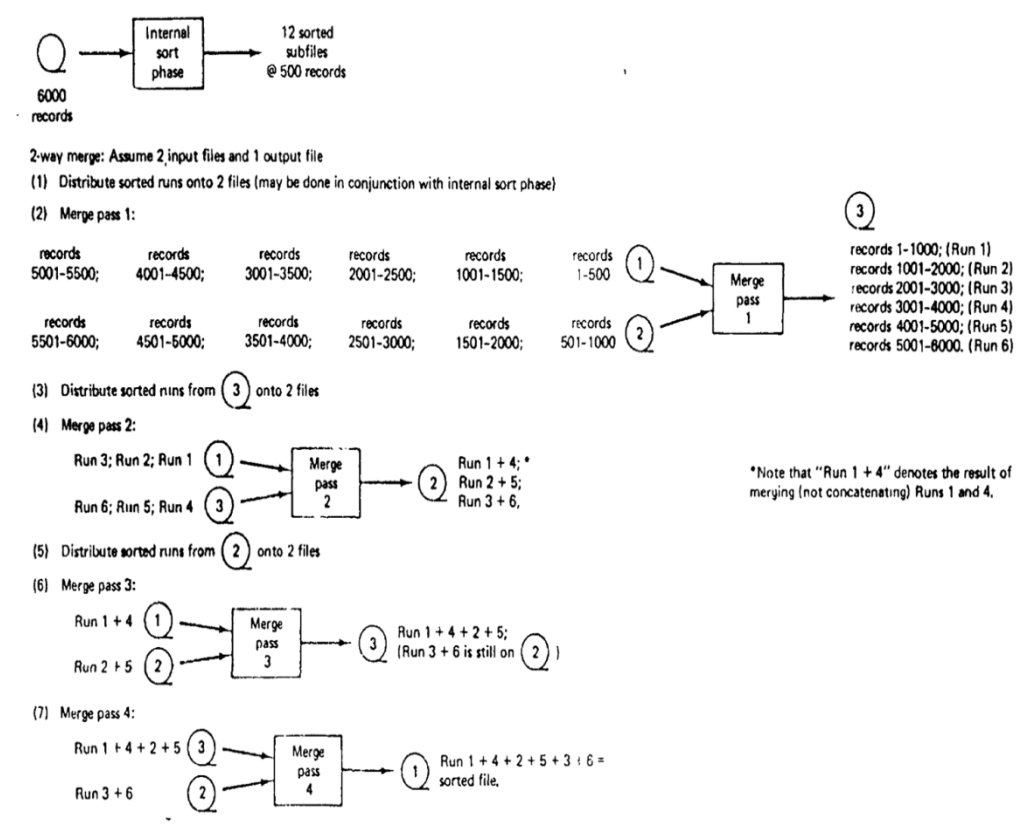
2. M Way Natural Merge

Pada M way natural merge, dapat diartikan sebagai merge dengan M input file dan hanya satu output file.

Contoh :

Sebuah file yang terdiri dari 6000 record hendak disortir ke dalam memori komputer yang kapasitasnya 1000 record.

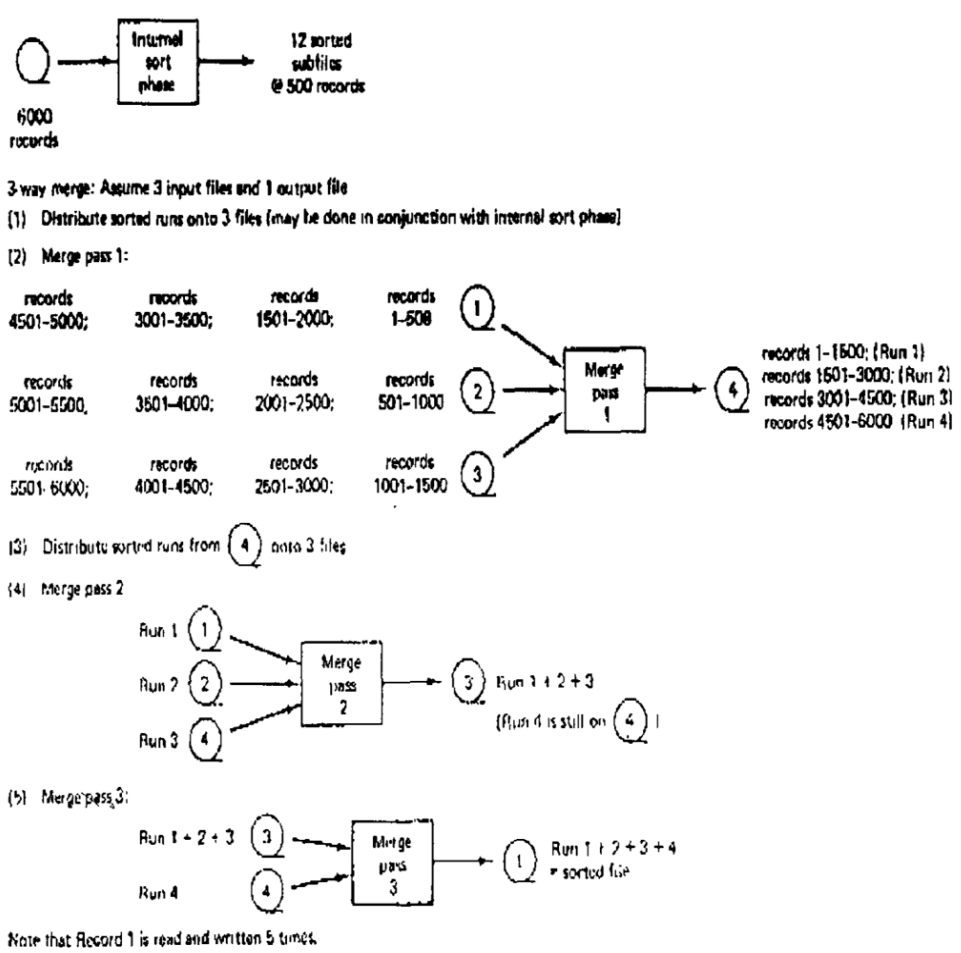
Buatlah dengan menggunakan 3 way natural merge.



Gambar 13. 2 Natural Merge 2 - WAY

Penggabungan 2 arah. Dengan asumsi 2 file masukan dan 1 file keluaran

1) Penyortiran terdistribusi dari 2 file (dapat digabungkan dengan langkah-langkah penyortiran internal sampai selesai.
2) Gabungkan Tahap 1
3) Salah satu dari 3 hasil penggabungan sementara di file 3 dipindahkan ke file 1.
4) Fase kedua ditetapkan.



Gambar 6 Contoh Natural Merge 3–WAY

Gambar 13. 3 Natural Merge 3 - Way

3. Balance Merge

Hal ini terlihat dari metode natural merge bahwa jika kita menggunakan file input M, hanya file M + 1 yang digunakan. Ketika saldo dikonsolidasikan, jika file input M digunakan, seluruh file yang digunakan adalah file 2 M.

Seperti halnya natural merge, balance merge juga ada beberapa cara yaitu :

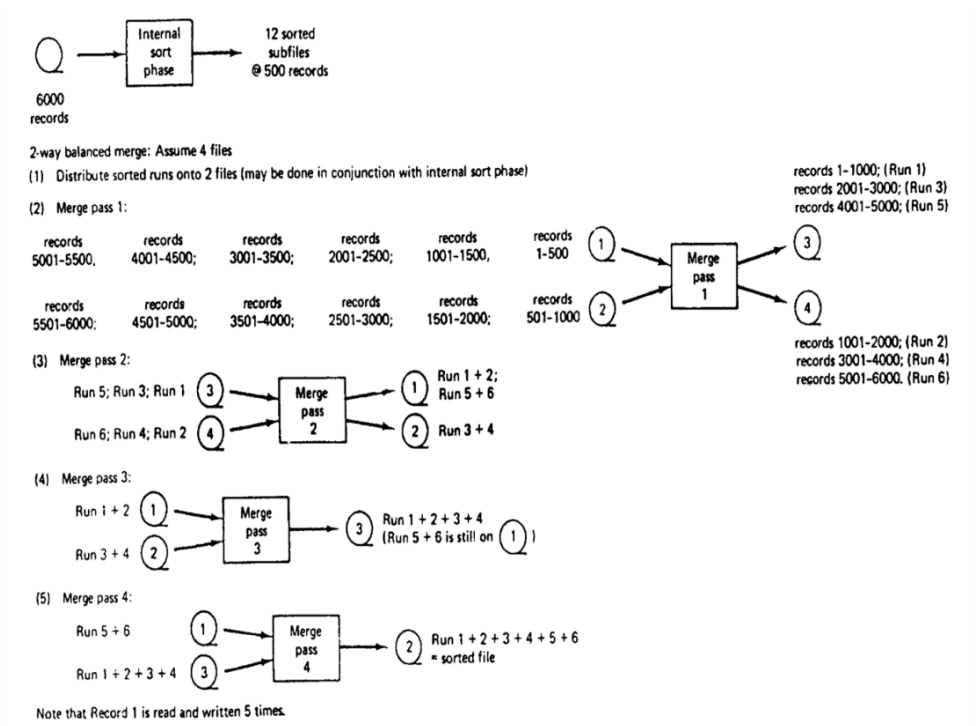
1) 2 way balance merge

2) 3 way balance merge,

3) M way balance merge.

2 way balance merge berarti merge dengan 2 input file sekaligus dan hasilnya 2 output file.

M way balance merge berarti M input file dengan M output file



Gambar 7 Contoh Balanced Merge 2 – WAY

Gambar 13. 4 Balanced Merge 2 - Way

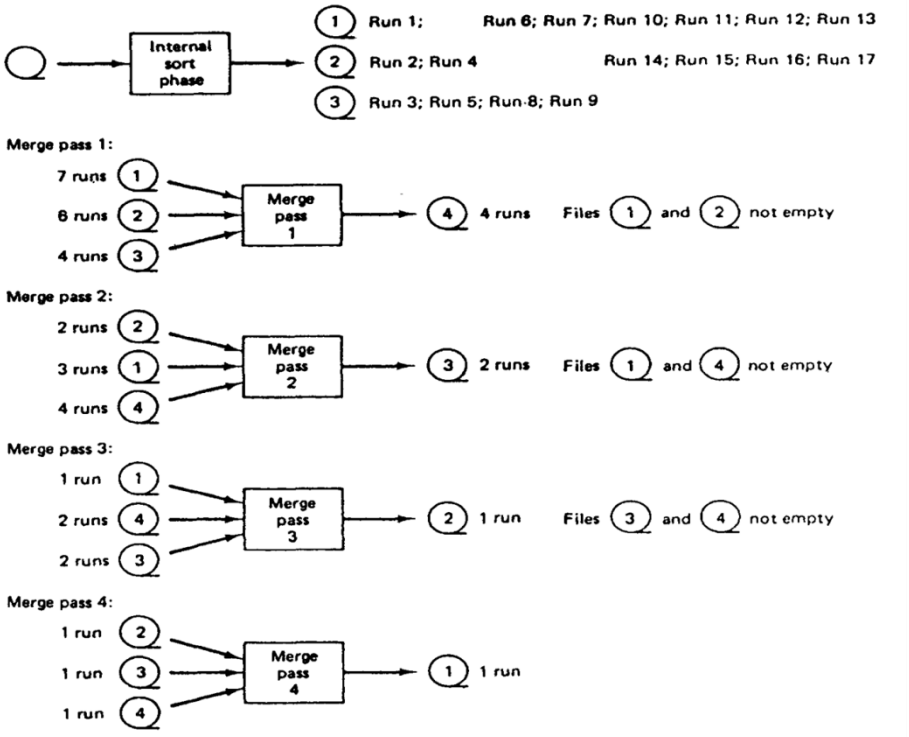
4. Polyphase Merge

Kita dapat melihat bahwa keseimbangan mode M menggunakan file 2 M (file input M dan file output M). Karena hanya file M yang dimasukkan dan direkam ke dalam satu file setiap kali, ada file M-1 dalam status idle. Inilah mengapa penggabungan poplyphase menyelesaikan kerusakan ini. Pada penggabungan multi fase mode M, digunakan file input 2M-1 dan 1 file output sekaligus, sehingga jumlah file input yang digunakan adalah 3 file input.

Contoh :

Setelah phase sort internal, misalkan kita mempnyai 17 subfile atau 17 run yang akan didistribusikan ke input file. Jika kita menggunakan 2 way polyphase merge, berarti 17 run tersebut didistribusikan ke dalam 3 input file.

Dari pendistribusian tersebut, maka diperoleh :

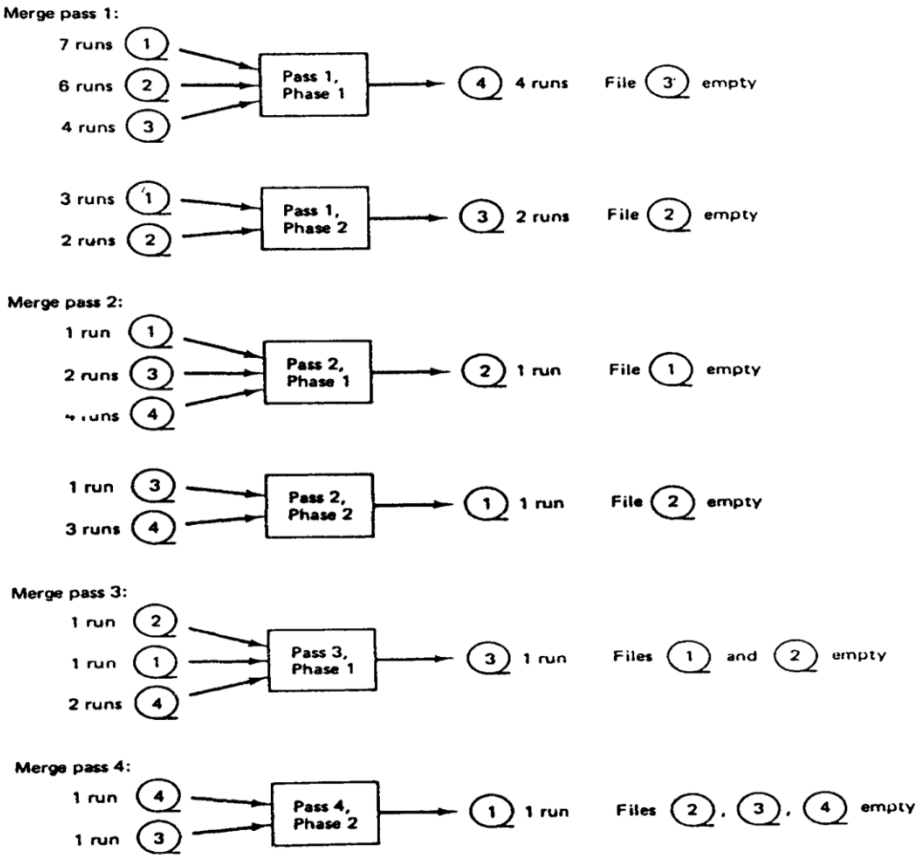


Gambar 13. 5 Polyphase Merge

5. Cascade Merge

Jenis lain dari unbalanced merge yang berusaha mengurangi penyalinan dan pembacaan record-record disebut cascade merge. Cascade merge dengan derajat M menggunakan input file 2M-1, 2M-2 dan 2M-3, …, kemudian 2 input file selama tiap tahap merge.

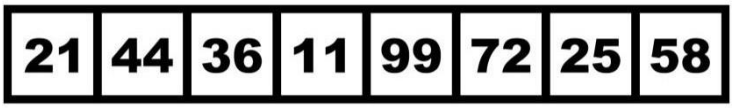
Setiap merge pass dimulali dengan merge dari : 2 M-1 input file ke 1 ouput file. Pada cascade merge, pendistribusian runnya saa dengan pendistribusian run oada polyphase merge, hanya berbeda pada phase mergenya.



Gambar 13. 6 Cascade Merge

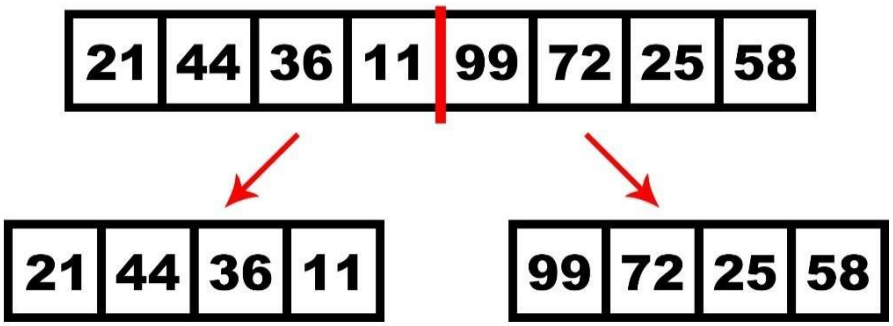
6. Algoritma Merge Sort

kita memiliki list dengan urutan acak, seperti berikut :



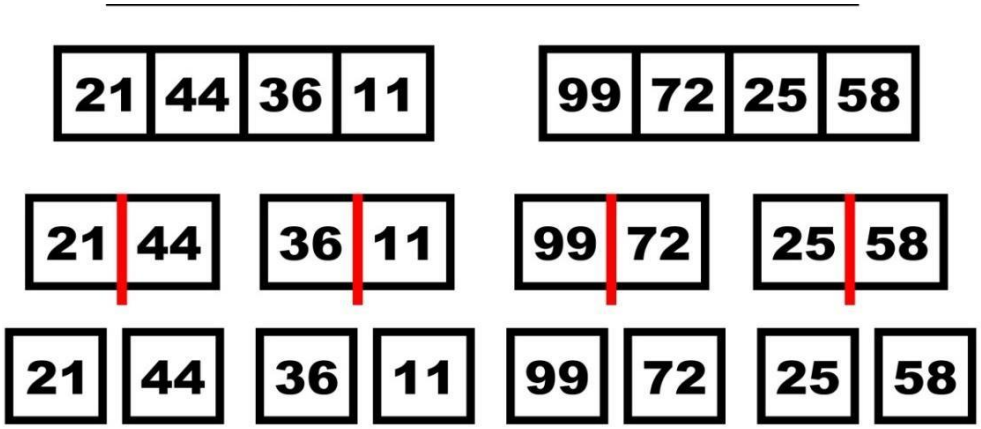
Gambar 13. 7 List Merge Sort

Algoritma ini menggunakan metode devide and conquer, jadi akan memecah list ini menjadi 2 bagian sama rata.



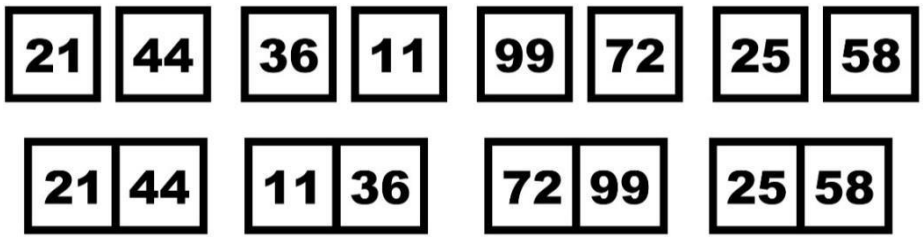
Gambar 13. 8 List Merge Sort 2

Apakah elemen tersebut sudah menjadi elemen tunggal? tentu saja belum, maka kita akan membagi kembali sampai kita mendapat elemen tunggal dari setiap index List.



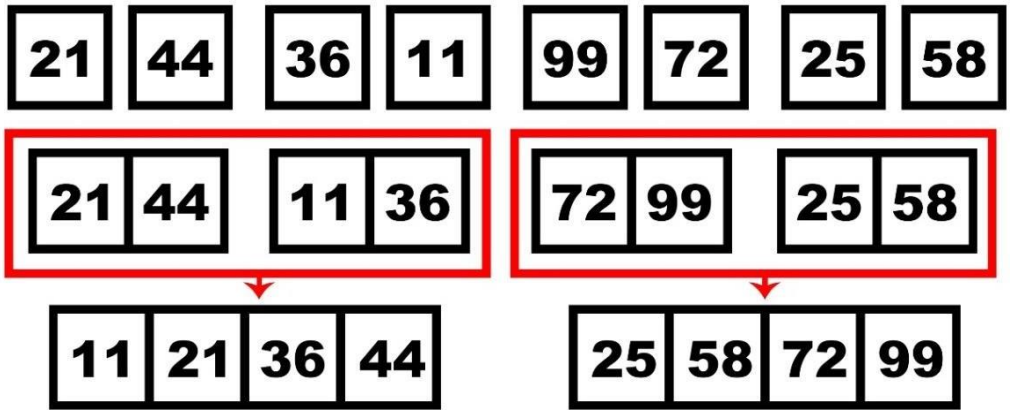
Gambar 13. 9 List Merge Sort 3

Sekarang, semua elemen sudah menjadi elemen tunggal, selanjutnya kita akan melakukan sorting dengan menggunakan kebalikan dari metode devide and conquer, dimana kita akan melakukan perbandingan antara 2 elemen dan menggabungkannya menjadi elemen baru



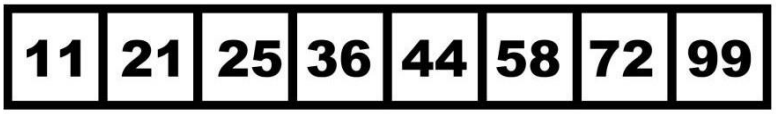
Gambar 13. 10 List Merge Sort 4

Selanjutnya kita akan membandingkan 2 elemen kembali hingga semua item dalam List tersusun Ascending.



Gambar 13. 11 List Merge Sort 5

Lakukan lagi hingga semua data berhasil tersusun.



Gambar 13. 12 List Merge Sort 6

Kelebihan Algoritma Merge Sort :

1) Performa sangat bagus untuk List yang memiliki banyak index
2) memiliki waktu pengerjaan yang konsisten (worst case,average case,best case)

Kekurangan Algoritma Merge Sort :

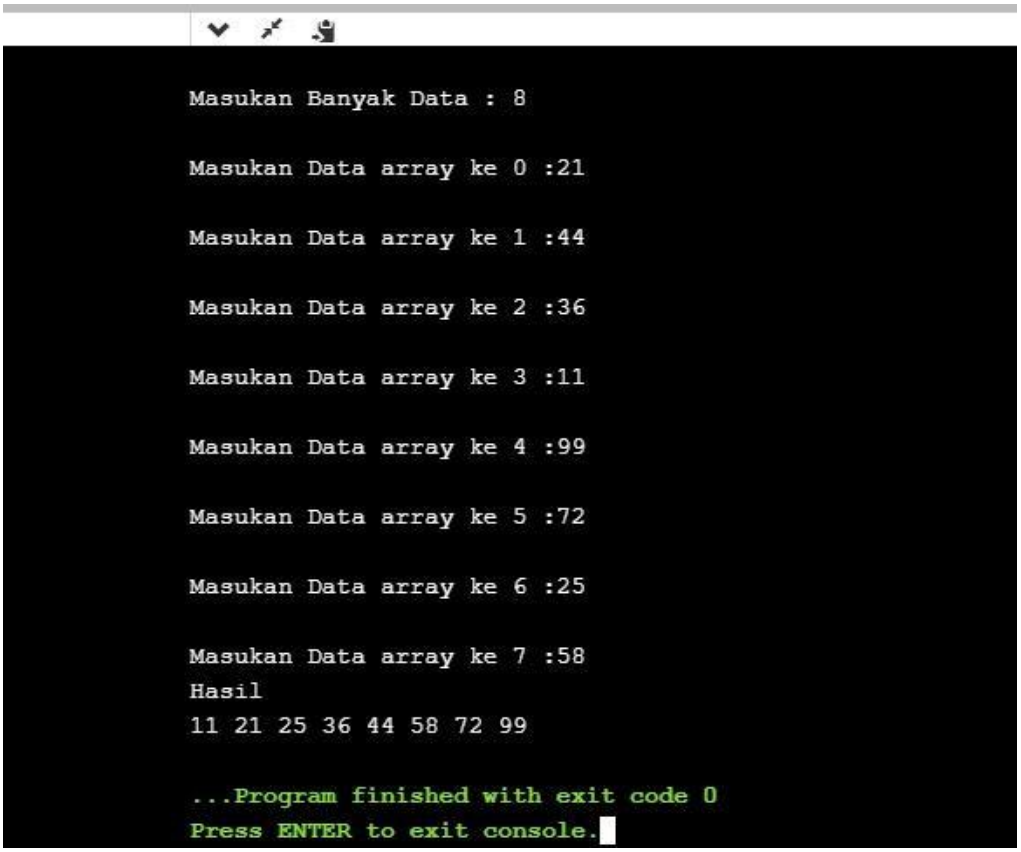
1) Performa buruk untuk list dengan index sendikit dibanding algoritma sorting lainnya seperti bubble sort dan insertion sort
2) Jika data sudah tersorting sejak awal, maka ia akan tetap melakukan sorting dari awal.
3) Menggunakan memory lebih untuk melakukan split data.

Contoh program Sort merge C++

```cpp
#include <iostream>
    using namespace std;

    void merge(int arr[], int l, int m, int r)
    {
        int i, j, k;
        int n1 = m - l + 1;
        int n2 = r - m;

        int L[n1], R[n2];

        for (i = 0; i < n1; i++)
            L[i] = arr[l + i];
        for (j = 0; j < n2; j++)
            R[j] = arr[m + 1 + j];

        i = 0;
        j = 0;
        k = l;
        while (i < n1 && j < n2)
        {
            if (L[i] <= R[j])
            {
                arr[k] = L[i];
                i++;
            }
            else
            {
                arr[k] = R[j];
                j++;
            }
            k++;
        }

        while (i < n1)
        {
            arr[k] = L[i];
            i++;
            k++;
        }

        while (j < n2)
        {
            arr[k] = R[j];
            j++;
            k++;
        }
    }

    void mergeSort(int arr[], int l, int r)
    {
        if (l < r)
        {

            int m = l + (r - l) / 2;

            mergeSort(arr, l, m);
            mergeSort(arr, m + 1, r);

            merge(arr, l, m, r);
        }
    }

    void show(int A[], int size)
    {
        int i;
        for (i = 0; i < size; i++)
            cout << A[i] << " ";
    }

    int main()
    {
        int size;
        cout << "\nMasukan Banyak Data : ";

        cin >> size;

        int arr[size];

        for (int i = 0; i < size; ++i)
        {
            cout << "\nMasukan Data array ke "<<i<<" :";
            cin >> arr[i];
        }

        mergeSort(arr, 0, size);

        cout << "Hasil\n";
        show(arr, size);
        return 0;
    }
```

Gambar 13. 13 Program Sort Merge C++

Output :



Gambar 13. 14 Tampilan Output Program Sort Merge C++

C. Soal Latihan

1. Gunakan metode merge sort untuk pengurutan data berikut dan tunjukan proses pengurutan langkah demi langkah 40 75 12 68 72 9 56 18 60 5 20 65 2
2. Gunakan metode bubble sort untuk pengurutan data berikut dan tunjukan proses pengurutan langkah demi langkah 40 75 12 68 72 9 56 18 60 5 20 65 2

D. Referensi

Handayani, Dewi. 2001. Sistem Berkas. Yogyakarta:J&J Learninghttps://rizkirahadiansyah.wordpress.com/2018/03/28/macam-macam-metode-sorting/

PERTEMUAN 14
PENGENALAN KONTROL INPUT/OUTPUT

A. Tujuan Pembelajaran

1. Mahasiswa dapat menjelaskan definisi dan persyaratan control I/O
2. Mahasiswa dapat menjelaskan pengertian direktori berkas dan control informasi
3. Mahasiswa dapat menjelaskan pengertian kontrol peralatan

B. Uraian Materi

1. Input dan Output

Ini adalah mekanisme pengiriman data secara bertahap dan terus menerus dari proses ke perangkat melalui aliran data. Fungsi input / output pada dasarnya mengimplementasikan algoritma input / output di tingkat aplikasi. Ini karena kode aplikasi sangat fleksibel, dan kesalahan aplikasi tidak akan dengan mudah menyebabkan sistem macet.

Unit Input/Output adalah bagian dari sistem mikroprosesor yang di gunakan mikrosesor untuk berhubungan dengan perangkat lain.

1) Unit input adalah unit yang di gunakan untuk memasukan data dari luar ke dalam mikroprosesor. Contoh : data input yang berasal dari keyboard dan mouse.

2) Unit output biasanya di gunakan untuk menampilkan data atau untuk memunculkan data yang dikirimkan oleh mikroprosesor.

Contoh : suatu data yang di tampilkan monitor dan printer.

Bagian Input dan Ouput ini memerlukan sinyal kontrol, antara lain untuk baca I/O (input/output Read [IOR]) dan untuk tulis I/O (input/output Write [IOW]).

Dalam Komputer terdapat 3 bagian penting :

1) Perangkat Keras (Hardware)

2) Alat computer yang berbentuk fisik yang dapat menjalankan sistem operasi computer. Dan terdiri dari beberapa bagian :
   a. Input Device (Alat masukan)
   b. Process Device (Alat Proses)
   c. Output (Alat Keluaran)

3) Software (perangkat lunak)

4) Bagian dari sistem komputer yang tidak memiliki bentuk fisik dan tidak terlihat merupakan kumpulan data elektronik yang disimpan dan dikelola oleh komputer dalam bentuk program yang dapat menjalankan perintah..

5) Perlengkapan berpikir

6) Orang yang menggunakan komputer

Sistem komputer terdiri dari 5 bagian struktural dasar :

1) Unit masukan (unit masukan)

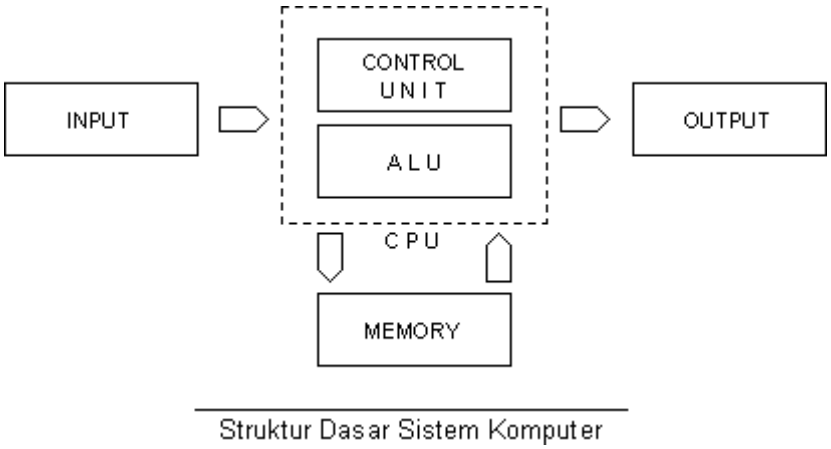
2) Unit kontrol (unit kontrol)

3) Satuan Logika dan Aritmatika (Satuan Aritmatika dan Logika / ALU)

4) Unit Memori / Penyimpanan

5) Unit keluaran (unit keluaran)

Unit kontrol dan ALU membentuk satu unit yang disebut central processing unit (CPU). Gambar berikut menunjukkan hubungan antara setiap unit sistem komputer.



Gambar 14. 1 Struktur Dasar Komputer

Macam-macam Input/Output

1) Struktur Input Output

   Bagian ini meliputi dari Struktur input output, Interupsi I/O, dan Direct Memory Access (DMA).

2) Interuspi Input Output

   Operasi interupsi input output dijalankan, terdapat 2 kemungkinan : yaitu Synchronous input output dan Asynchronous input output. Kontrol ini dibalikan menuju proses penggunaan tanpa menunggu proses input output berakhir . Sehingga proses input output dan proses pengguna bisa digunakan secara berbarengan.

3) Proteksi Input Output

Proteksi input output dikatakan selesai, jika pemakai dapat dipastikan tidak akan menyentuh cara monitor. Jika terjadi proteksi input output dapat di gunakan.

4) Manajemen Sistem Input Output

Biasa disebut device manager pada sistem komputer. Menyiapkan perangkat driver yang umum, sehingga proses input output bisa bersama membuka, membaca, menulis, menutup. Contoh : user meemakai proses yang percis untuk melihat data di hardisk, compact dis (cd)-room dan floopy diskk

Komponen Sistem Operasi untuk input output :

1) Buffers : menyimpan selama data dari/ke perangkat input output.

2) Spoolingg : membuat jadwal pemakaian input output sistem supaya Bisa menjadi efisien.

3) Menyiapkan driver untuk dapat melakukan proses operasi perangkat keras input output tertentu.

4) Memanajemen peripheral alat input output merupakan sebuah program sistem operasi dan kongkrit dengan beragam perangkat dan aplikasinya.

2. Pengelompokan Perangkat Input Output

Periperal alat Input Output bisa dikelompokan menjadi :

1) Sifat Aliran Datanya
   a. Perangkat Beroritentasi blok
      Contoh : Disk, Tape, CD ROM, Optical Disk.
   b. Perangkat berorientasi aliran karakter
      Contoh : Terminal, Line Printer, Pita Kertas, Kartu-kartu berlubang, Interface jaringan, Mouse

2) Sasaran Komunikasi
   a. Perangkat yang terbaca oleh manusia
      Contoh : VDT (Video Display Terminal) : monitor, keyboard, mouse
   b. Perangkat yang terbaca oleh mesin
      Contoh : Disk, tape, sensor, dan controller
   c. Perangkat Komunikasi
      Contoh : Modem

Faktor-faktor yang membedakan antar perangkat :

1) Kecepatan konversi data (kecepatan data)

2) Model aplikasi yang digunakan

3) Kontrol kompleksitas • Jumlah unit yang ditransfer

4) Representasi atau refleksi data

5) Kondisi kesalahan teknis untuk pemrograman perangkat I / O

Teknik Pemograman Input Output

Input Output Terprogram

Software pengatur perangkat (driver) diproses harus mentransfer data ke/dari pengatur diperangkat dan menunggu sampai operasi yang dikerjakan sampai perangkat selesai.

Driver berisikan beberapa intruksi :

1) Pengeoprasian

   Berguna untuk membukakan perangkat luar dan memberitahu yang perlu dilakukan.

   Contoh kasus : Unit disk magnetic di instrusikan guna kembali ke tempat awal, bergerak ke record berikut, dan sebagainya.

2) Percobaan

   Berguna mengecek status perangkat keras berkaitan dengan alat input output

3) Pembacaan/Penulisan

   Berguna membaca/menulis guna mengirim data dengan pemroses dan peripheral alat luar. Hambatan pokok input output terprogram adalah pemproses dibuat lama untuk menunggu proses input output.

Input Output diatur oleh interupsi

Metode input output diarahkan pada interupsi untuk mempunyai mekanisme kerja sebagai berikut :

1) Pemproses membantu interuksi menuju perangkat input output, setelah itu melaksanakan tugas lainnya.

2) Alat Iinput output untuk menginterupsi menjalankan fasilitas layanan saat alat telah sedia berpindah data dengan pemproses.

3) Ketika menerima interupsi perangkat keras, pemproses segera mengeksekusikan transfer data.

Kelebihan : Pemproses tidak disibukan menunggu dan memelihara perangkat input output untuk mengecek status perangkat.

Kekurangan : kecepatan mengirim data input output dibatasi kecepatan pengujian dan menjalankan perangkat operasi.

DMA (Direct Memory Access)

DMA memiliki fungsi melepaskan prosesor untuk menunggu perangkat input dan output mentransfer data. Ketika prosesor ingin membaca atau menulis data, prosesor memerintahkan pengontrol DMA dengan mengirimkan informasi berikut :

1) Menulis / membaca perintah
2) Alamat perangkat I / O
3) Mulai dari lokasi penyimpanan tulisan / bacaan
4) Jumlah kata yang ditulis / dibaca (byte)

Setelah mengirim pesan ke DMA controller, pemproses dapat melanjutkan kerja lain. Direct Memory Access (DMA) mengirimkan seluruh data yang diminta memori secara langsung tanpa melewati pemproses. Pada saat mengirimkan data sudah selesaoi, Direct Memory Access (DMA) mentransfer sinyal interupsi ke pemproses. Sampai pemproses dicantumkan saat awal dan akhir transfer data.

Keunggulan : Penghematan waktu pemproses dan peningkatan kinerja Input Output

3. Prinsip Manajemen Perangkat Input Output

Terdapat dua tujuan perancangan input output, yaitu :

1) Efisiensi

   Aspek penting dalam perangcangan manajemen perangkat, karena operasi input output sering menimbulkan bottleneck.

2) Generalitas

   Selain sederhana dan bebas kesalahan, pengelompokan perangkat input dan output juga dapat menangani perangkat secara seragam dalam proses memanggang dan cara sistem operasi mengelola perangkat input dan output serta operasi. Perangkat lunak ini diatur secara hierarki. Pemrosesan tingkat rendah menyembunyikan kesulitan perangkat keras tingkat tinggi. Lapisan atas transaksi memberikan pemakainya antarmuka yang indah, bersih, nyaman dan seragam.

Masalah-masalah manajemen Input adalah :

1) Penamaan yang seragam (Uniform Naming)
2) Penanganan kesalahan (error handling)
3) Transfer sinkron dan asinkron

4) Dedicated dengan sharable

Hirarki Manajemen perangkat Input Output

1) Interrupt Handler

   Interupsi harus disembunyikan sehingga rutinitas berikutnya tidak dapat melihatnya. Setelah perintah input dan output, driver perangkat akan diblokir dan menunggu gangguan.

   Ketika terjadi interupsi, proses penanganan interupsi akan membuat driver perangkat keluar dari status pemblokiran.

2) Device drivers

   Semua kode terkait perangkat ditempatkan di driver perangkat. Setiap driver perangkat menyediakan layanan untuk satu jenis (kelas) perangkat, dan bertanggung jawab untuk menerima permintaan abstrak untuk perangkat lunak perangkat independen di dalamnya dan menjalankan permintaan layanan.

3) Perangkat Lunak device independent

   Ini bertujuan untuk menetapkan fungsi masukan dan keluaran yang berlaku untuk semua perangkat dan menyediakan antarmuka terpadu untuk perangkat lunak tingkat pengguna.Fungsi-fungsi lain yang dilakukan :

Perangkat lunak level pemakai

Pada umumnya, perangkat Lunak input output terdapat di sistem operasi. Satu bagian kecil berisi pustaka-pustaka yang di kaitkan pada program pemakai dan berjalan diluar kernel. Kumpulan prosedur pustaka input output merupakan bagian peraturan-peraturan pustaka. Bagian penting ialah system spooling. Spooling ialah cara khusus berurusan dengan perangkat input output yang harus didedikasikan kepada system multi programming.

Perangkat Input

Perangkat Input yaitu beberapa komponen atau alat yang pergunakan pemakai guna memasukan data ke dalam computer untuk diproses lebih lanjut supaya mengciptalam informasi yang di butuhkan.

Contoh perangkat input antara lain.

1) Papan ketik (key board)

2) Tombol Jostcik

3) Pointer Mouse

4) Barcode Scanner

5) Touchscreen

6) Treck ball

Perangkat Output

Perangkat alat keluaran adalah perangkat yang di pakai dalam membawa data keluar computer atau untuk mengalihkan data dari komputer ke perangkat yang berbeda lainnya.

Berdasarkan bentuk outputnya, unit output terdiri dari :

1) Printer

2) Layar Monitors

3) Printers Cetak

4) Projektor

5) Mic Audio

Peralatan Input/Output :

1) Hard disk

2) Flash drive

3) Router Internet

4) Hub Internet

5) Switch Internet

6) Server Print

7) CD-Reads and Writer

8) DVD-Read and Writer

Definisi dan Persyaratan Kontrol I/O

Sistem kontrol input-output dirancang untuk membantu pengguna sehingga mereka dapat mengakses file terlepas dari karakteristik detail dan waktu penyimpanan mereka. Kontrol input dan output melibatkan manajemen file dan alat manajemen yang merupakan bagian dari sistem operasi.

Tugas dari sistem kendali input dan output adalah :

1) Memelihara direktori dari berkas dan lokasi informasi

2) Menentukan jalan bagi aliran data antara main memori dan alat penyimpanan sekunder

3) Mengkoordinasi komunikasi antara CPU dan alat penyimpanan sekunder

4) Menyiapkan berkas penggunaan input atau output telah selesai

Channel

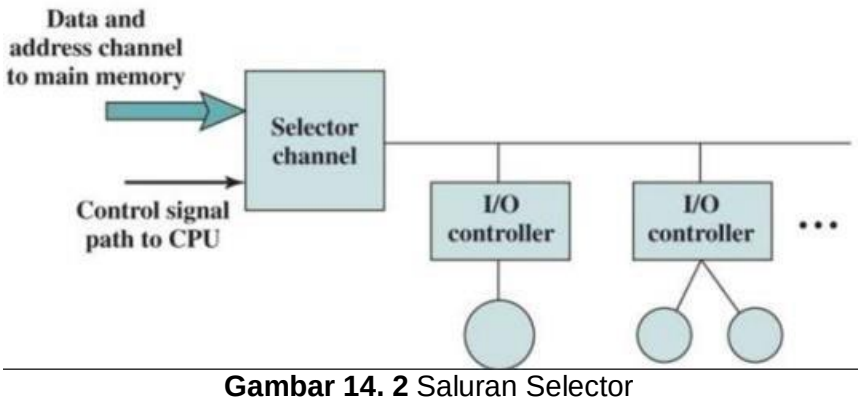
Di sebagian besar sistem komputer, central processing unit (CPU) tidak perlu melakukan tugas-tugas yang berkaitan dengan input dan output, tetapi tanggung jawab pengendalian perangkat diserahkan kepada prosesor input dan output yang disebut I / O Channel. Saluran input dan output itu sendiri adalah prosesor yang telah diprogram sebelumnya. Program yang dijalankan disebut program saluran. Perencana program menentukan operasi, akses peralatan yang diperlukan, dan kontrol jalur data.

Jenis-jenis Saluran (channel)

Saluran Selector

Bisa mengelola aliran data antara memori utama pada sebuah perangkat pada saat pemprosesan. Karena saluran itu adalah prosesor-prosesor yang cepat maka selector biasanya hanya memakai input output dengan kecepatan tinggi.

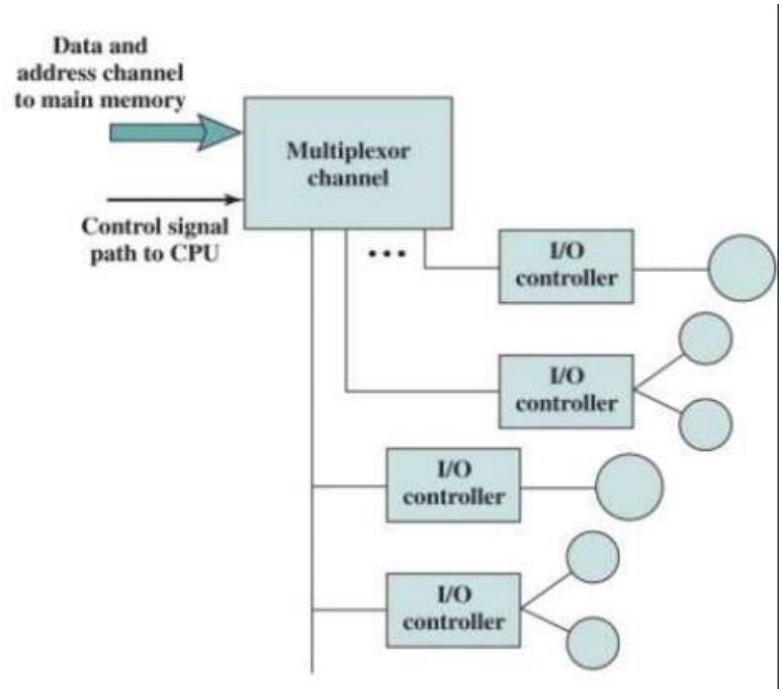
Contoh : pembaca kartu (Card Reader)



Gambar 14. 2 Saluran Selector

Saluran Multi plexor

Beberapa perangkat dapat digunakan untuk mengatur aliran data memori utama. Dibandingkan dengan pemilih saluran, saluran multiplekser lebih efektif saat menggunakan perangkat berkecepatan rendah.



Gambar 14. 3 Saluran Multi Plexor

Saluran Block Multi plexor

Aliran data ke berbagai perangkat. Channel multiplexer blok dapat menjalankan instruksi dari saluran program satu perangkat, dan kemudian dapat mentransfer instruksi dari saluran program ke perangkat lain.

Macan-macam Device :

1) Dedicated Device

2) Shared Device

Aktifitas Saluran

Beberapa saluran akan memberi perintah :

q Test I/O, untuk menentukan apakah jalur (pathway) yang menuju peralatan sedang sibuk.

q Start I/O, pada peralatan tertentu.

q Halt I/O, pada peralatan tertentu.

4. Manajemen Buffer

Buffering adalah melembutkan lonjakan-lonjakan kebutuhan pengaksesan I/O, sehingga meningkatkan efisiensi dan kinerja sistem operasi.

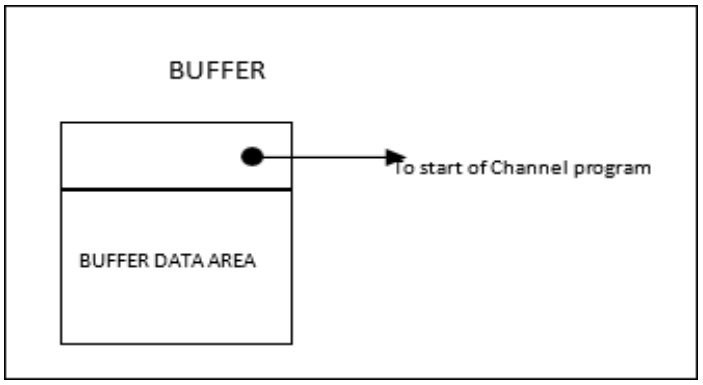
Buffering terbagi menjadi 4 jenis manajemen yaitu :

Single Buffering

Single buffering dapat diterapkan dengan dua mode, yaitu :

1) Mode line at a time.

2) Mode byte at a time.



Gambar 14. 4 Single Buffering

Struktur dasar dari channel program untuk mengisi buffer adalah :

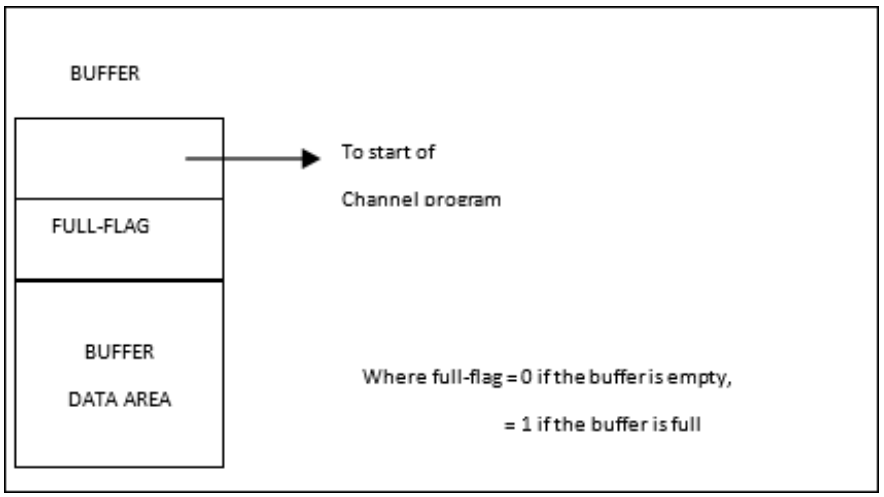
1) Tunggu instruksi READ dari program

2) Memberitahukan instruksi start I/O ke unit control

3) Tunggu hingga buffer dikosongkan

4) Memberitahukan interupsi pada program sehingga dapat mulai membaca dari buffer.

Anticipatory Buffering

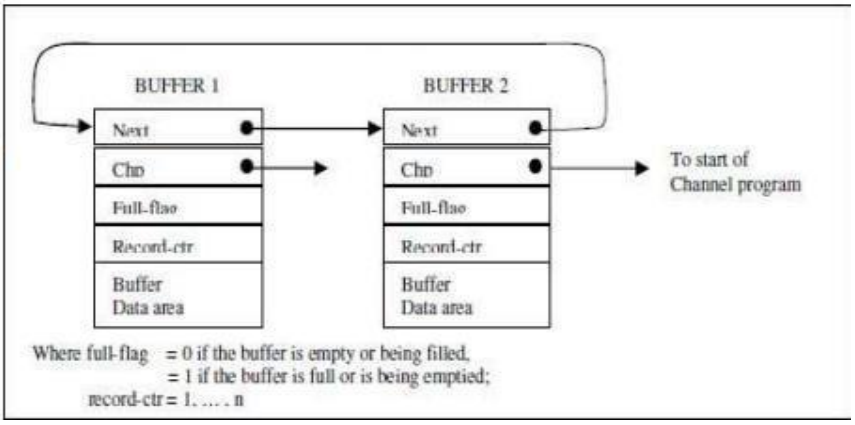


Gambar 14. 5 Anticipatory Buffering

Double Buffering.

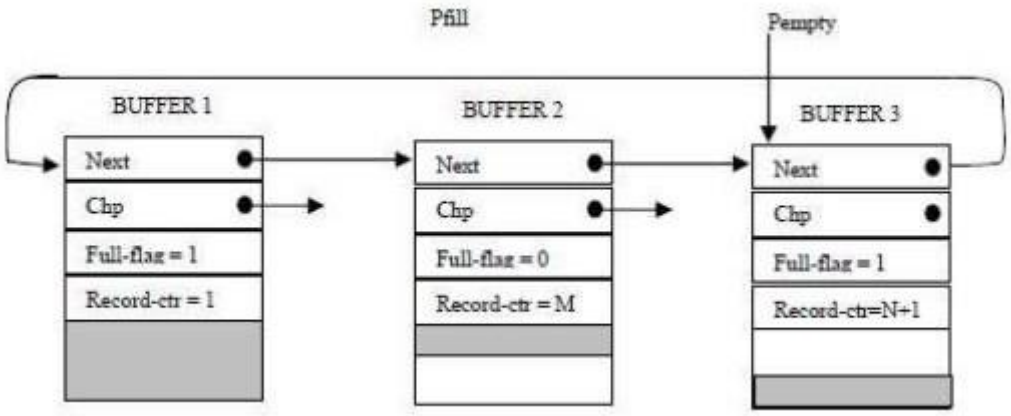
Double buffering mempunyai 2 mode alternatif, yaitu :

1. Mode line at a time
2. Mode byte at a time.



Gambar 14. 6 Double Buffering

Three Buffers



Gambar 14. 7 Three Buffering

Keterangan :

Pfill : yang menunjukkan buffer berikutnya akan diisi atau sedang diisi

Pempty : yang menunjukkan buffer berikutnya akan dikosongkan atau sedang dikosongkan

Algoritma Penjadwalan Disk

Terdapat dua tipe penjadwalan disk, yaitu :

1. Optimasi seek.
2. Optimasi rotasi (rotational latency)

Beberapa kriteria penjadwalan disk, yaitu :

1. Throughput
2. Variansi waktu tanggap, diusahakan minimum.

Beberapa algoritma penjadwalan disk, antara lain :

1. First come first serve (FCFS)
2. Shortest seek first (SSF)
3. Elevator (SCAN)
4. Elevator dimodifikasi (C-SCAN)
5. N-step scan
6. Exchenbach scheme

Penanganan masalah operasi disk

Beberapa tipe kesalahan saat operasi disk dikategorikan sebagai berikut:

1. Programming error
2. Transient checksum error
3. Permanent checksum error
4. Seek error
5. Controller error
6. Track at time caching

Kesalahan ini ditanggulangi dengan mengkalibrasi disk supaya berfungsikembali.

Metode pemrograman PIT

Terdapat dua mode pemograman PIT

1. One shoot mode
2. Square wave mode

C. Soal Latihan

1. Jelaskan pengertian dan fungsi dari Input/Output?
2. Ada berapa macam-macam Input/Output, Sebutkan satu-satunya?
3. Ada berapa teknik pemograman Input/Output?
4. Jelaskan pengertian Channel dan sebutkan macam-macamnya?
5. Coba gambarkan struktur dari double buffering dan three buffering?

D. Referensi

Handayani, Dewi. 2001. Sistem Berkas. Yogyakarta : J&J Learning
Hariyanto, Bambang. 2003. Pengarsipan Dan Akses Pada Sistem Berkas. Bandung : Informatika
Wahyuni, 2013. Sistem Berkas. Yogyakarta : Penerbit Andi
Devilzc0de, Yudhie. (2015, April 05). Pengertian dan Klasifikasi Sistem Berkas
Akhmad. Februari 2012. http://teknikgufron.blogspot.com/2012/02/media-penyimpanan-berkas (diakses 06 Desember2014).
Kurniawan, P. (2015, Maret 15). Kelebihan Dan Kekurangan Magnetic Disk. Diambil kembali dari http://id.scribd.com/doc/175034256/Kelebihan-Dan-Kekurangan-Magnetic-Disk#scribd

GLOSARIUM

Add	:	Adalah operasi matematika yang mengambil dua atau lebih angka yang berbeda dan menghasilkan nilai totalnya.
Algorithm	:	Algorithm atau Algoritma merupakan prosedur untuk memecahkan masalah matematika (seperti menemukan pembagi umum terbesar) dalam sejumlah langkah terbatas yang sering melibatkan pengulangan operasi.
Ambiguitas	:	Kata atau ekspresi yang dapat dipahami dengan dua atau lebih cara yang mungkin: kata atau ekspresi ambigu.
Array	:	Struktur data di mana elemen data yang serupa disusun dalam tabel.
Ascending	:	Disusun dari yang terkecil hingga terbesar, meningkat.
Attribut	:	Adalah kualitas, karakter, atau karakteristik yang ditujukan kepada seseorang atau sesuatu.
BASIC	:	Merupakan Singkatan dari (Beginner's All-purpose Symbolic Instruction Code) BASIC adalah bahasa pemrograman komputer yang dikembangkan pada pertengahan 1960-an untuk menyediakan cara bagi siswa untuk menulis program komputer sederhana.
Batch	:	Sebuah proses yang dilakukan secara dikelompokkan.
Binary	:	Binary atau Biner merupakan matematika : sistem angka hanya berdasarkan angka 0 dan 1.
BIOS	:	Merupakan elemen perangkat lunak dari sistem operasi komputer yang memungkinkan CPU untuk berkomunikasi dengan perangkat input dan output yang terhubung (seperti keyboard atau monitor).
Block	:	Merupakan potongan atau bagian bahan substansial terutama ketika bekerja atau diubah untuk melayani tujuan tertentu.
Chip	:	Merupakan bahan semikonduktor berukuran kecil yang membentuk dasar untuk sirkuit terintegrasi.
Collision	:	Kumpulan perngkat jaringan yang data framenya dapat salling bertabrakan.
CPU	:	CPU (Central Processor Unit) Unit pemrosesan pusat yang dibuat pada satu atau beberapa chip, berisi elemen aritmatika, logika, dan kontrol dasar komputer yang diperlukan untuk memproses data.
Data Access	:	Merupakan proses program yang sedang mengakses record-record yang terdapat dari suatu file.
Data Storage	:	Data Storage (Penyimpanan Data) merupakan perangkat keras (hardware) yang dapat digunakan sebagai tempat penyimpanan data pada sebuah hardware komputer sehingga bisa dibuka dan dibaca kembali untuk dilakukan proses ulang.
Database	:	Merupakan sekumpulan besar data yang biasanya digunakan terutama untuk pencarian dan pengambilan cepat (seperti pada olah komputer).
Descending	:	Disusun dari yang terbesar hingga terkecil, menurun.
Directory	:	Merupakan sebuah daftar berbentuk alfabet atau rahasia (sesuai nama dan alamat).

Domain	:	Domain berisi sekelompok komputer yang dapat diakses dan dikelola dengan sekumpulan aturan umum.
Entitas	:	Merupakan keberadaan sebuah hal yang kontras dengan atributnya.
Efisiensi	:	Ukuran dalam membandingkan input yang direncanakan atau penggunaan input dengan penggunaan aktual yang terealisasikan.
FTP	:	Protokol yang digunakan untuk menjembatani pertukaran informasi di komputer.
Field	:	Field merupakan satu item data dalam database atau program perangkat lunak.
File	:	Perangkat (seperti folder, kotak, atau kabinet) melalui kertas mana yang disimpan secara berurutan.
File System	:	File system adalah proses yang mengelola bagaimana dan di mana data pada disk penyimpanan, biasanya hard disk drive (HDD), disimpan, diakses, dan dikelola.
Firmware	:	Firmware adalah komponen elektronik nyata dengan instruksi perangkat lunak tertanam, seperti BIOS.
Foreign Key	:	Foreign Key (Kunci Asing) adalah bidang (atau kumpulan bidang) dalam satu tabel yang merujuk ke kunci utama di tabel lain.
Fstab	:	File yang berisi tabel sistem file di Linux.
GigaByte / GB	:	Sama dengan 1.000 megabita dan mendahului satuan pengukuran terabyte.
Hardware	:	Perangkat keras komputer mengacu pada bagian fisik komputer dan perangkat terkait.
Hash	:	Hash adalah fungsi yang mengonversi satu nilai ke nilai lainnya. Hashing data adalah praktik umum dalam ilmu komputer dan digunakan untuk beberapa tujuan yang berbeda.
HDD	:	HDD (Hard Disk Drive) merupakan perangkat penyimpanan data yang bersifat non volatile. Umumnya diinstal secara internal di komputer, dilampirkan langsung ke pengontrol disk motherboard komputer.
Index	:	Daftar item (seperti topik atau nama) yang diperlakukan dalam karya cetak yang memberikan untuk setiap item nomor halaman tempat item tersebut ditemukan.
Insert	:	Perintah insert digunakan untuk menyisipkan satu atau beberapa baris ke dalam tabel database dengan nilai kolom tabel tertentu.
Integer	:	Integer adalah bilangan bulat (bukan pecahan) yang bisa positif, negatif, atau nol.
Inversi	:	Perubahan urutan ketentuan proporsi matematika yang disebabkan oleh membalikkan setiap rasio.
Inverted File	:	erupakan sebuah kunci pada Indeks terbalik dengan semua nilai kunci, setiap nilai kunci memiliki penunjuk ke rekaman terkait.
Iterative	:	Sebuah proses yang dilakukan secara satu persatu.
Kapasitor	:	Merupakan komponen listrik yang menyimpan energi potensial.
Key	:	Sarana untuk mendapatkan atau mencegah pintu masuk, kepemilikan, atau kontrol.

Konduktor	:	Merupakan bahan yang memungkinkan listrik mengalir melalui mereka dengan mudah.
Linear	:	Linear mengacu pada apa pun yang berorientasi atau terjadi dalam urutan tertentu, tanpa penyimpangan.
Linked List	:	Merupakan struktur data yang digunakan untuk menyimpan koleksi data.
Linux	:	Unix dan sistem operasi lain dengan kernel linux sebagai intinya, Disertakan aplikasi dan modul pendukung lainnya agar dapat berjalan normal dan dapat digunakan secara keseluruhan layaknya sebuah sistem operasi.
Maintenance	:	Merupakan pembaruan sistem operasi dan program aplikasi untuk menambahkan fungsi baru dan mengubah format data.
MegaByte / MB	:	1024 kilobyte atau 1.048.576 byte.
Megahertz / MHz	:	Merupakan ukuran frekuensi per satuan waktu, atau jumlah siklus per detik.
Merge	:	Proses penggabungan data hingga terurut.
Modulus	:	Merupakan sebuah operasi matematika yang digunakan untuk menemukan sisa pembagian satu angka dengan angka lainnya.
Mounting	:	Proses menautkan sistem file yang baru ditemukan (dalam perangkat) ke direktori home.
Multi List	:	Merupakan sebuah pendekatan yang memiliki indeks untuk tiap key sekunder.
Non Volatile	:	Merupakan kondisi dimana file data atau program tidak hilang jika arus listrik terputus atau padam.
Null	:	Berkaitan dengan metode pengukuran di mana jumlah yang tidak diketahui (pada arus listrik) dibandingkan dengan jumlah yang diketahui dari jenis yang sama.
Overflow	:	Sebuah proses mengisi ruang untuk kapasitas dan menyebar di luar batasnya.
Partisi	:	Merupakan kumpulan dari sektor yang bersebelahan pada disk.
Pascal	:	Pascal adalah bahasa pemrograman prosedural yang mendukung pemrograman terstruktur dan struktur data untuk mendorong praktik pemrograman yang baik.
Pile	:	Pile merupakan sebuah tipe data abstrak yang berfungsi untuk menyimpan data dengan cara berurutan yang longgar.
Pointer	:	Alamat memori komputer yang berisi alamat lain (sebagai data yang diinginkan).
Primary Key	:	Primary Key (Kunci Utama) Berfungsi secara unik mengidentifikasi setiap rekaman dalam tabel.
Primary Memory	:	Primary Memory (Memori Primer) merupakan memori yang langsung diakses oleh CPU untuk menyimpan dan mengambil informasi maupun data.
Probe	:	Probe adalah program atau perangkat lain yang dimasukkan pada juncture kunci dalam jaringan untuk tujuan memantau atau mengumpulkan data tentang aktivitas jaringan.
Radix	:	Radix adalah istilah yang digunakan untuk menggambarkan jumlah digit yang digunakan dalam sistem angka posisi sebelum bergulir ke tempat digit berikutnya.

RAM	:	RAM (Random Access Memory) merupakan tempat dimana data disimpan sementara disaat komputer sedang bekerja dan bisa diakses secara acak.
Record	:	Merupakan (suara, gambar visual, data, dll.) yang disimpan / didaftarkan pada sesuatu (seperti cakram atau pita magnetik) dalam bentuk yang dapat direproduksi
ROM	:	ROM (Read Only Memory) merupakan jenis memory yang hanya dapat dibaca.
Representasi	:	Suatu proses dimana suatu objek ditangkap oleh indera seseorang, maka masuk akal untuk mengolah hasilnya adalah suatu konsep / gagasan yang dalam bahasa akan disampaikan / diungkapkan kembali.
Secondary Key	:	Kunci Sekunder adalah kunci yang belum dipilih untuk menjadi kunci utama.
Secondary Memory	:	Secondary Memory (Memori Sekunder) merupakan perangkat penyimpanan yang tidak dapat diakses langsung oleh CPU dan digunakan sebagai perangkat penyimpanan permanen yang menyimpan data bahkan setelah power dimatikan.
Semantics	:	Cabang linguistik yang mempelajari arti atau makna.
Sequential	:	Berkaitan dengan atau berdasarkan metode pengujian hipotesis statistik yang melibatkan pemeriksaan urutan sampel yang masing-masing keputusan dibuat untuk menerima atau menolak hipotesis atau untuk melanjutkan pengambilan sampel.
Server	:	Suatu sistem yang terdapat dalam sebuah komputer yang dapat memberikan berbagai macam layanan untuk ditampilkan kepada klien pada suatu sistem jaringan komputer.
Software	:	Perangkat lunak adalah program yang memungkinkan komputer untuk melakukan tugas tertentu, dibandingkan dengan komponen fisik sistem (perangkat keras).
Sort	:	Proses pemilihan data yang sebelumnnya disusun secara acak sehingga dapat diatur secara terurut sesuai aturan tertentu.
String	:	String adalah tipe data yang digunakan dalam pemrograman, seperti bilangan bulat dan unit titik pecahan, tetapi digunakan untuk mewakili teks daripada angka.
Synomin	:	Adalah dua atau lebih key yang berbeda pada hash ke home address yang sama.
Table	:	Susunan data yang sistematis biasanya dalam baris dan kolom untuk referensi.
TI	:	Teknologi Informasi merupakan istilah yang digunakan untuk teknologi yang berguna untuk mempermudah pekerjaan manusia dalam membuat, mengubah, menyimpan, dan menyebarkan informasi.
Track	:	Merupakan sebuah rangkaian jalur paralel atau konsentris di mana materi (seperti musik atau informasi) direkam (seperti pada rekaman fonograf atau pita magnetik).
Transistor	:	Merupakan perangkat semikonduktor yang mentransfer sinyal lemah dari sirkuit resistensi rendah ke sirkuit resistensi tinggi.
Unix	:	Sistem operasi digunakan sebagai sistem operasi default pada berbagai jenis komputer, terutama komputer kecil sebagai

		workstation atau server (sistem yang menyediakan layanan dalam suatu jaringan).
Update	:	Merupakan informasi saat ini untuk memperbarui sesuatu.
Utility	:	Merupakan program atau rutinitas yang dirancang untuk melakukan atau memfasilitasi operasi rutin (seperti menyalin file atau mengedit teks) di komputer.
Variable	:	Berupa nilai dan nama symbol dari nilai yang dapat diubah.
Volatile	:	Merupakan kondisi dimana file data atau program akan hilang ketika arus listrik terputus atau padam.
Volume	:	jumlah ruang yang ditempati oleh objek tiga dimensi yang diukur dalam unit kubik.
Www	:	Ruang informasi di Internet, pengenal global yang disebut URL digunakan untuk mengidentifikasi sumber daya yang berguna.
Add	:	Adalah operasi matematika yang mengambil dua atau lebih angka yang berbeda dan menghasilkan nilai totalnya.
Algorithm	:	Algorithm atau Algoritma merupakan prosedur untuk memecahkan masalah matematika (seperti menemukan pembagi umum terbesar) dalam sejumlah langkah terbatas yang sering melibatkan pengulangan operasi.
Ambiguitas	:	Kata atau ekspresi yang dapat dipahami dengan dua atau lebih cara yang mungkin: kata atau ekspresi ambigu.
Array	:	Struktur data di mana elemen data yang serupa disusun dalam tabel.
Ascending	:	Disusun dari yang terkecil hingga terbesar, meningkat.
Attribut	:	Adalah kualitas, karakter, atau karakteristik yang ditujukan kepada seseorang atau sesuatu.
BASIC	:	Merupakan Singkatan dari (Beginner's All-purpose Symbolic Instruction Code) BASIC adalah bahasa pemrograman komputer yang dikembangkan pada pertengahan 1960-an untuk menyediakan cara bagi siswa untuk menulis program komputer sederhana.
Batch	:	Sebuah proses yang dilakukan secara dikelompokkan.
Binary	:	Binary atau Biner merupakan matematika : sistem angka hanya berdasarkan angka 0 dan 1.
BIOS	:	Merupakan elemen perangkat lunak dari sistem operasi komputer yang memungkinkan CPU untuk berkomunikasi dengan perangkat input dan output yang terhubung (seperti keyboard atau monitor).
Block	:	Merupakan potongan atau bagian bahan substansial terutama ketika bekerja atau diubah untuk melayani tujuan tertentu.
Chip	:	Merupakan bahan semikonduktor berukuran kecil yang membentuk dasar untuk sirkuit terintegrasi.
CPU	:	CPU (Central Processor Unit) Unit pemrosesan pusat yang dibuat pada satu atau beberapa chip, berisi elemen aritmatika, logika, dan kontrol dasar komputer yang diperlukan untuk memproses data.
Data Access	:	Merupakan proses program yang sedang mengakses record-record yang terdapat dari suatu file.

Data Storage	:	Data Storage (Penyimpanan Data) merupakan perangkat keras (hardware) yang dapat digunakan sebagai tempat penyimpanan data pada sebuah hardware komputer sehingga bisa dibuka dan dibaca kembali untuk dilakukan proses ulang.
Database	:	Merupakan sekumpulan besar data yang biasanya digunakan terutama untuk pencarian dan pengambilan cepat (seperti pada olah komputer).
Descending	:	Disusun dari yang terbesar hingga terkecil, menurun.
Directory	:	Merupakan sebuah daftar berbentuk alfabet atau rahasia (sesuai nama dan alamat).
Domain	:	Domain berisi sekelompok komputer yang dapat diakses dan dikelola dengan sekumpulan aturan umum.
Entitas	:	Merupakan keberadaan sebuah hal yang kontras dengan atributnya.
Field	:	Field merupakan satu item data dalam database atau program perangkat lunak.
File	:	Perangkat (seperti folder, kotak, atau kabinet) melalui kertas mana yang disimpan secara berurutan.
File System	:	File system adalah proses yang mengelola bagaimana dan di mana data pada disk penyimpanan, biasanya hard disk drive (HDD), disimpan, diakses, dan dikelola.
Firmware	:	Firmware adalah komponen elektronik nyata dengan instruksi perangkat lunak tertanam, seperti BIOS.
Foreign Key	:	Foreign Key (Kunci Asing) adalah bidang (atau kumpulan bidang) dalam satu tabel yang merujuk ke kunci utama di tabel lain.
GigaByte / GB	:	Sama dengan 1.000 megabita dan mendahului satuan pengukuran terabyte.
Hardware	:	Perangkat keras komputer mengacu pada bagian fisik komputer dan perangkat terkait.
Hash	:	Hash adalah fungsi yang mengonversi satu nilai ke nilai lainnya. Hashing data adalah praktik umum dalam ilmu komputer dan digunakan untuk beberapa tujuan yang berbeda.
HDD	:	HDD (Hard Disk Drive) merupakan perangkat penyimpanan data yang bersifat non volatile. Umumnya diinstal secara internal di komputer, dilampirkan langsung ke pengontrol disk motherboard komputer.
Index	:	Daftar item (seperti topik atau nama) yang diperlakukan dalam karya cetak yang memberikan untuk setiap item nomor halaman tempat item tersebut ditemukan.
Insert	:	Perintah insert digunakan untuk menyisipkan satu atau beberapa baris ke dalam tabel database dengan nilai kolom tabel tertentu.
Integer	:	Integer adalah bilangan bulat (bukan pecahan) yang bisa positif, negatif, atau nol.
Inversi	:	Perubahan urutan ketentuan proporsi matematika yang disebabkan oleh membalikkan setiap rasio.
Inverted File	:	erupakan sebuah kunci pada Indeks terbalik dengan semua nilai kunci, setiap nilai kunci memiliki penunjuk ke rekaman terkait.
Iterative	:	Sebuah proses yang dilakukan secara satu persatu.
Kapasitor	:	Merupakan komponen listrik yang menyimpan energi potensial.

Key	:	Sarana untuk mendapatkan atau mencegah pintu masuk, kepemilikan, atau kontrol.
Konduktor	:	Merupakan bahan yang memungkinkan listrik mengalir melalui mereka dengan mudah.
Linear	:	Linear mengacu pada apa pun yang berorientasi atau terjadi dalam urutan tertentu, tanpa penyimpangan.
Linked List	:	Merupakan struktur data yang digunakan untuk menyimpan koleksi data.
Maintenance	:	Merupakan pembaruan sistem operasi dan program aplikasi untuk menambahkan fungsi baru dan mengubah format data.
MegaByte / MB	:	1024 kilobyte atau 1.048.576 byte.
Megahertz / MHz	:	Merupakan ukuran frekuensi per satuan waktu, atau jumlah siklus per detik.
Modulus	:	Merupakan sebuah operasi matematika yang digunakan untuk menemukan sisa pembagian satu angka dengan angka lainnya.
Multi List	:	Merupakan sebuah pendekatan yang memiliki indeks untuk tiap key sekunder.
Non Volatile	:	Merupakan kondisi dimana file data atau program tidak hilang jika arus listrik terputus atau padam.
Null	:	Berkaitan dengan metode pengukuran di mana jumlah yang tidak diketahui (pada arus listrik) dibandingkan dengan jumlah yang diketahui dari jenis yang sama.
Overflow	:	Sebuah proses mengisi ruang untuk kapasitas dan menyebar di luar batasnya.
Partisi	:	Merupakan kumpulan dari sektor yang bersebelahan pada disk.
Pascal	:	Pascal adalah bahasa pemrograman prosedural yang mendukung pemrograman terstruktur dan struktur data untuk mendorong praktik pemrograman yang baik.
Pile	:	Pile merupakan sebuah tipe data abstrak yang berfungsi untuk menyimpan data dengan cara berurutan yang longgar.
Pointer	:	Alamat memori komputer yang berisi alamat lain (sebagai data yang diinginkan).
Primary Key	:	Primary Key (Kunci Utama) Berfungsi secara unik mengidentifikasi setiap rekaman dalam tabel.
Primary Memory	:	Primary Memory (Memori Primer) merupakan memori yang langsung diakses oleh CPU untuk menyimpan dan mengambil informasi maupun data.
Probe	:	Probe adalah program atau perangkat lain yang dimasukkan pada juncture kunci dalam jaringan untuk tujuan memantau atau mengumpulkan data tentang aktivitas jaringan.
Radix	:	Radix adalah istilah yang digunakan untuk menggambarkan jumlah digit yang digunakan dalam sistem angka posisi sebelum bergulir ke tempat digit berikutnya.
RAM	:	RAM (Random Access Memory) merupakan tempat dimana data disimpan sementara disaat komputer sedang bekerja dan bisa diakses secara acak.
Record	:	Merupakan (suara, gambar visual, data, dll.) yang disimpan / didaftarkan pada sesuatu (seperti cakram atau pita magnetik) dalam bentuk yang dapat direproduksi

ROM	:	ROM (Read Only Memory) merupakan jenis memory yang hanya dapat dibaca.
Secondary Key	:	Kunci Sekunder adalah kunci yang belum dipilih untuk menjadi kunci utama.
Secondary Memory	:	Secondary Memory (Memori Sekunder) merupakan perangkat penyimpanan yang tidak dapat diakses langsung oleh CPU dan digunakan sebagai perangkat penyimpanan permanen yang menyimpan data bahkan setelah power dimatikan.
Sequential	:	Berkaitan dengan atau berdasarkan metode pengujian hipotesis statistik yang melibatkan pemeriksaan urutan sampel yang masing-masing keputusan dibuat untuk menerima atau menolak hipotesis atau untuk melanjutkan pengambilan sampel.
Software	:	Perangkat lunak adalah program yang memungkinkan komputer untuk melakukan tugas tertentu, dibandingkan dengan komponen fisik sistem (perangkat keras).
String	:	String adalah tipe data yang digunakan dalam pemrograman, seperti bilangan bulat dan unit titik pecahan, tetapi digunakan untuk mewakili teks daripada angka.
Table	:	Susunan data yang sistematis biasanya dalam baris dan kolom untuk referensi.
TI	:	Teknologi Informasi merupakan istilah yang digunakan untuk teknologi yang berguna untuk mempermudah pekerjaan manusia dalam membuat, mengubah, menyimpan, dan menyebarkan informasi.
Track	:	Merupakan sebuah rangkaian jalur paralel atau konsentris di mana materi (seperti musik atau informasi) direkam (seperti pada rekaman fonograf atau pita magnetik).
Transistor	:	Merupakan perangkat semikonduktor yang mentransfer sinyal lemah dari sirkuit resistensi rendah ke sirkuit resistensi tinggi.
Update	:	Merupakan informasi saat ini untuk memperbarui sesuatu.
Utility	:	Merupakan program atau rutinitas yang dirancang untuk melakukan atau memfasilitasi operasi rutin (seperti menyalin file atau mengedit teks) di komputer.
Volatile	:	Merupakan kondisi dimana file data atau program akan hilang ketika arus listrik terputus atau padam.
Volume	:	jumlah ruang yang ditempati oleh objek tiga dimensi yang diukur dalam unit kubik.

DAFTAR PUSTAKA

(1) Buku

Bach, Maurice J. (2016). *The Design of the UNIX Operating System*. United States: Prentice Hall.

Bhattacharya, Arnab. (2015). *Fundamentals of Database Indexing and Searching*. New York: CRC Press book publisher.

Coronel, Carlos. (2010). *Database Systems: Design, Implementation, and Management*. California: Course Technology.

Deitel, Paul, & Deitel, Harvey. (2016). *C how to program : with an introduction to C++*. Boston: Pearson.

Ferreira Filho, Wladston. (2017). *Computer Science Distilled*. Las Vegas: Code Energy LLC.

Fuxi, Gan, & Yang, Wang. (2015). *Data Storage at the Nanoscale Advances and Applications*. United States: Taylor & Francis Group, LLC.

Handayani, Dewi. (2001). *Sistem Berkas*. Yogyakarta : J&J Learning.

Hariyanto, Bambang. (2003). *Pengarsipan Dan Akses Pada Sistem Berkas*. Bandung : Informatika.

Hariyanto, Bambang. (2007). *Sistem Pengarsipan Dan Metode Akses*. Bandung: Informatika.

Indonusa Esa Unggul. (2003). *Materi Kuliah Sistem Operasi*. Jakarta: Universitas Indonusa Esa Unggul.

Karumanchi, Narasimha. (2017). *Data Structures and Algorithms Made Easy*. Hyderabad, Andhra Pradesh, India: CareerMonk Publications.

Krogh, Einar. (2020). *An Introduction to Windows Operating System*. Copenhagen, Denmark: Einar Krogh & bookboon.

L. Tharp, Alan. (1988). *File Organization and Processing*. New York, United States: John Wiley & Sons, Inc.

M., Rahmat, Ibrahim, Samik, & Masyarakat Digital Gotong Royong (MDGR). (2008). *Pengantar Sistem Operasi Komputer: Plus Studi Kasus Kernel Linux*. Yogyakarta: Ardi Publishing.

O'Reilly, James. (2017). *Network Storage Tools and Technologies for Storing Your Company's Data*. United States: Elsevier Inc.

Silberschatz, Abraham. (2002). *Operation System Concepts, sixth edition*. United

States: John Wiley & Sons, Inc.

Silberschatz, Galvin, & Gagne. (2002). *Operating System Concepts, 6th ed*. United States: John Wiley & Sons, Inc.

Stallings, Williem. (2000). *Operating System 4th ed*. Engrewood cliffs, New Jersey: Prentice Hall Inc.

Tananbaum, & Andrew S. (1992). *Modern Operating System 2nd edition*. Engrewood cliffs, New Jersey: Prentice Hall Inc.

(2) Sumber Website

Akhmad. (2012, Februari). Media Penyimpanan Berkas. Diakses pada Agustus 5, 2020, dari http://teknikgufron.blogspot.com/2012/02/media-penyimpanan-berkas

Devilzc0de, Yudhie. (2015, April 5). Pengertian dan Klasifikasi Sistem Berkas. Diakses pada Agustus 5, 2020, dari http://nurlianapulungan.blogspot.com/2017/02/sistem-berkas.html

Kurniawan, Pungky. (2015, Maret 15). Kelebihan Dan Kekurangan Magnetic Disk. Diakses pada Agustus 5, 2020, dari https://www.scribd.com/document/374413059/Tugas-II-Sistem-Berkas

Kurniawan, Pungky. Kelebihan dan Kekurangan Magnetic Disk. Diakses pada Agustus, 5, 2020, dari http://id.scribd.com/doc/175034256/Kelebihan-Dan- Kekurangan-Magnetic-Disk#scribd

Rahariansyah, Rizki. Macam-macam Metode Sorting. Diakses pada Agustus 5,2020, dari https://rizkirahadiansyah.wordpress.com/2018/03/28/macam-macam-metode-sorting/

Rizkinawawi. Pengurutan Rekaman. Diakses pada pada Agustus 5, 2020, dari https://rizkinawawi.wordpress.com/2020/07/08/pertemuan-12-pengurutan-rekaman/

S. Rumetna. Pengurutan Rekaman. Diakses pada Agustus 5, 2020, dari http://matheusrumetna.com/materi/sistem_berkas/BAB_VII_PENGURUTAN_REKAMAN.pdf

FORMAT RENCANA PEMBELAJARAN SEMESTER (RPS)

Program Studi	: S1 Teknik Informatika	***SKS***	: 2 SKS
Mata Kuliah/Kode	: Sisten Berkas	***Prasyarat***	: -
Semester	: 3	***Kurikulum***	: -
Deskripsi Mata Kuliah	: Mata kuliah ini merupakan mata kuliah wajib program studi teknik informatika yang membahas Tentang media penyimpanan file; dasar dasar organisasi file :berkas primer, sekunder,langsung dengan banyak kunci & key,relatif dan index sekuensial; majemen kolasi,pengurutan rekaman, sort & marge,perangkat kontrol input dan output Konsep umum organisasi dan arsitektur komputer, pengkodean, gerbang logika, memory, input/ output,	***Capaian Pembelajaran***	: Setelah menyelesaikan mata kuliah ini mahasiswa mampu membuat struktur berkas pada file dan mengimplementasikan ke dalam program
Penyusun	: 1. Samsoni S.Kom., M.Kom 2. Khoirunnisya S.Kom., M.Kom 3. Aniq Astofa S.Kom., M.Kom 4. Erdi Sutriyatna S.Kom., M.Kom		

PERTEMUAN KE-	KEMAMPUAN AKHIR YANG DIHARAPKAN	POKOK BAHASAN	METODE PEMBELAJARAN	PENGALAMAN BELAJAR	KRITERIA PENILAIAN	BOBOT NILAI
(1)	(2)	(3)	(4)	(5)	(6)	(7)
1.	**Mampu memahami Konsep dasar sistem berkas**	**Pengantar sistem file.**	**Diskusi**	**Menjelaskan mengenai atribut pada file,klasifikasi file dan gambaran file dan organisasi pada file**	**Dapat menjelaskan kembali dan mengambarkan gambaran file**	**5%**
2.	**1. Mahasiswa mampu menjelaskan jenis- jenis media penyimpanan 2. Mahasiswa mampu melakukan perhitungan untuk organisasi berkas pada magnetic tape**	**MEDIA PENYIMPANAN FILE**	**Diskusi, simulasi dan penugasan**	**Menjelaskan media penyimpanan file dan mempraktekan cara menghitung parity dan error control pada magnetic tape**	**Dapat membandingkan dan menjelaskan macam" jenis penyimpanan file sesuai dengan fungsinya dan memeriksa kesalahan pada magnetik tape**	**7%**
3.	**Mahasiswa memahami dan Mampu menjelaskan konsep dasar organisasi berkas**	**ORGANISASI BERKAS PRIMER**	**Diskusi dan penugasan**	**Menjelaskan lebih detail dan memberikan contoh mengenai berkas primer**	**Dapat merancang dalam pembuatan berkas primer**	**7%**
4.	**Mahasiswa memahami dan dapat menjelaskan organisasi berkas sekuensial**	**ORGANISASI BERKAS SEKUENSIAL**	**Diskusi, simulator dan penugasan**	**Menjelaskan lebih detail dan memberikan contoh mengenai**	**Dapat membuat berksa squensial dengan mengulang**	**7,5%**

PERTEMUAN KE-	KEMAMPUAN AKHIR YANG DIHARAPKAN	POKOK BAHASAN	METODE PEMBELAJARAN	PENGALAMAN BELAJAR	KRITERIA PENILAIAN	BOBOT NILAI
(1)	(2)	(3)	(4)	(5)	(6)	(7)
				pembuatan berkas primer dan pencarian biner	**proses perbandingan**	
5.	**Mahasiswa memahami dan dapat menjelaskan organisasi berkas langsung**	**ORGANISASI BERKAS LANGSUNG**	**Diskusi,simulator dan penugasan**	**Menjelaskan lebih detail dan memberikan contoh mengenai pembuatan berkas langsung dalam pencarian data**	**Dapat membuat berkasa langsung dengan mencari file data tanpa harus melalui media pencarian**	**5%**
6.	**Mahasiswa memahami dan dapat menjelaskan organisasi berkas banyak kunci**	**ORGANISASI BERKAS BANYAK KUNCI**	**Ceramah dan Diskusi**	**Menjelaskan lebih detai tentang berkas banyak kunci dan membuat record record pada file**	**Dapat memberikan contoh pada kehidupan sehari-hari dan membuat record**	**5%**
7.	**Mahasiswa memahami dan dapat menjelaskan organisasi berkas banyak key**	**ORGANISASI BERKAS BANYAK KEY**	**Ceramah dan Diskusi**			**5%**
8.	**Mahasiswa memahami dan dapat menjelaskan organisasi berkas relatif**	**ORGANISASI BERKAS RELATIF**	**Ceramah dan Diskusi Simulasi dan Responsi**			**7,5%**

PERTEMUAN KE-	KEMAMPUAN AKHIR YANG DIHARAPKAN	POKOK BAHASAN	METODE PEMBELAJARAN	PENGALAMAN BELAJAR	KRITERIA PENILAIAN	BOBOT NILAI
(1)	(2)	(3)	(4)	(5)	(6)	(7)
9.	**1. Mahasiswa memahami dan dapat menjelaskan organisasi berkas indek sequensial 2. Menjelaskan cara pembuatan berkas sekuensial. 3. Menjelaskan pengertian prime dan overflow data area, dan dapat menyebutkan contohnya.**	**ORGANISASI BERKAS INDEX SEQUENSIAL**	**Ceramah dan Diskusi, Simulasi dan Responsi**			**7,5%**
10.	**Mahasiswa memahami dan dapat menjelaskan konsep mounting, sharing dan proteksi**	**KONSEP MOUNTING, SHARING DAN PROTEKSI**	**Simulasi dan Responsi**			**5%**
11.		**MANAJEMEN KOLISI**	**Simulasi dan Responsi**			**5%**
12.		**PENGURUTAN REKAMAN**	**Simulasi dan Responsi**			**5%**
13.	**Mahasiswa dapat: 1. Menjelaskan pengertian natural merge file**	**SORT DAN MERGE FILE**	**Simulasi dan Responsi**			**5%**

PERTEMUAN KE-	KEMAMPUAN AKHIR YANG DIHARAPKAN	POKOK BAHASAN	METODE PEMBELAJARAN	PENGALAMAN BELAJAR	KRITERIA PENILAIAN	BOBOT NILAI
(1)	(2)	(3)	(4)	(5)	(6)	(7)
	dan dapat memberikan contohnya. **2. Menjelaskan pengertian balance merge file dan dapat memberikan contohnya.** **3. Menjelaskan pengertian polyphase merge file dan dapat memberikan contohnya.** **4. Menjelaskan pengertian cascade merge file dan dapat memberikan contohnya.**					
14.	**Mahasiswa dapat:** **1. Menjelaskan definisi dan persyaratan kontrol I/O.** **2. Menjelaskan pengertian direktori berkas dan kontrol informasi.**	**PENGENALAN KONTROL INPUT/OUTPUT**	**Simulasi dan Demonstrasi**			**5%**

PERTEMUAN KE-	KEMAMPUAN AKHIR YANG DIHARAPKAN	POKOK BAHASAN	METODE PEMBELAJARAN	PENGALAMAN BELAJAR	KRITERIA PENILAIAN	BOBOT NILAI
(1)	(2)	(3)	(4)	(5)	(6)	(7)
	3. Menjelaskan pengertian kontrol peralatan. **4. Menjelaskan dan menyebutkan tipe-tipe saluran.** **5. Menjelaskan dan menyebutkan tipe-tipe buffer.**					

Referensi/sumber:

1. William Stalling Computer Organization and Architecture, Prentice Hall, 5 Th ed, 2000
2. Hamacher, Carl, et all, Computer organization, fifth edition, McGraw Hill, 2002

Ketua Program Studi
…….

(…….)
NIDN:……..

Ketua Team Teching
…….

(…….)
NIDN:……..

***Sumber**: Buku Kurikulum pendidikan Tinggi DIKTI 2015*

CARA MENGISI RENCANA PEMBELAJARAN:

NO KOLOM	JUDUL KOLOM	PENJELASAN ISIAN
1	**PERTEMUAN KE-**	Di isi sesuai dengan pertemuan ke-1 s/d selesai (di sesuaikan dengan bobot SKS: 2 SKS=14 kali pertemuan; 3 SKS=18 kali pertemuan; 4 SKS=24 pertemuan)
2	**KEMAMPUAN AKHIR YANG DIHARAPKAN**	Rumusan kemampuan dibidang *kognitif*, *psikomotorik*, dan *afektif* diusahakan lengkap dan utuh (*hard skills & soft skills*). Merupakan tahapan kemampuan yang diharapkan dapat mencapai kompetensi mata kuliah ini diakhir semester.
3	**POKOK BAHASAN**	Materi pokok yang akan diajarkan tiap pertemuan, yang terdiri dari sub pokok bahasan yang sesuai dengan tujuan pembelajaran.
4	**METODE PEMBELAJARAN**	Dapat berupa : ceramah, diskusi, presentasi tugas, seminar, simulasi, responsi, praktikum, latihan, kuliah lapangan, praktek bengkel, survai lapangan, bermain peran,atau gabungan berbagai bentuk. Penetapan bentuk pembelajaran didasarkan pada keniscayaan bahwa kemampuan yang diharapkan diatas akan tercapai dengan metode pembelajaran yang dipilih.
5	**PENGALAMAN BELAJAR**	Kegiatan yang harus dilakukan oleh mahasiswa yang dirancang oleh dosen agar yang bersangkutan memiliki kemampuan yang telah ditetapkan (tugas, suvai, menyusun paper, melakukan praktek, studi banding, dsb)
6	**KRITERIA PENILAIAN**	Berisi: indikator yang dapat menunjukan pencapaian kemampuan yang dicanangkan, atau unsur kemampuan yang dinilai (bisa kualitatif misal ketepatan analisis, kerapian sajian, Kreatifitas ide, kemampuan komunikasi, juga bisa juga yang kuantitatif : banyaknya kutipan acuan/unsur yang dibahas, kebenaran hitungan).8
7	**BOBOT NILAI**	disesuaikan dengan waktu yang digunakan untuk membahas atau mengerjakan tugas, atau besarnya sumbangan suatu kemampuan terhadap pencapaian kompetensi mata kuliah ini