Single Linked List

Agung Kurniman Putra
185100036
Fakultas Komputer
agungkurnimanputra.student@umitra.ac.id

Abstract

Single Linked List

Linked List adalah salah satu bentuk struktur data, berisi kumpulan data (node) yang tersusun secara sekuensial, saling sambungmenyambung, dinamis dan terbatas.
- Linked List sering disebut juga Senarai Berantai
- Linked List saling terhubung dengan bantuan variabel pointer
- Masing-masing data dalam Linked List disebut dengan node (simpul) yang menempati alokasi memori secara dinamis dan biasanya berupa struct yang terdiri dari beberapa field.

Single Linked List adalah sebuah LINKED LIST yang menggunakan sebuah variabel pointer saja untuk menyimpan banyak data dengan metode LINKED LIST, suatu daftar isi yang saling berhubungan.
Single Linked List adalah sekumpulan dari node yang saling terhubung dengan node lain melalui sebuah pointer.
Single Linked List hanya memiliki satu arah dan tidak memiliki dua arah atau bulak balik, dua arah tersebut disebut dengan double linked list.

Pada Implementasinya, Single Linked List terdapat dua variasi yaitu circular dan non-circular.

Kata Kunci : **Single Linked List**

A. PENDAHULUAN

Apabila setiap kali anda ingin menambahkan data anda selalu menggunakan pointer yang baru, anda akan membutuhkan banyak sekali variable pointer (penunjuk). Oleh karena itu, ada baiknya jika anda hanya menggunakan satu variabel pointer saja untuk menyimpan banyak data dengan metode yang kita sebut linked list. Jika kita terjemahkan, ini berarti suatu daftar isi yang saling berhubungan.

Struktur data adalah cara menyimpan atau merepresentasikan data di dalam komputer agar bisa dipakai secara efisien Sedangkan data adalah representasi dari fakta dunia nyata. Fakta atau keterangan tentang kenyataan yang disimpan, direkam atau direpresentasikan dalam bentuk tulisan, suara, gambar, sinyal atau symbol. Secara garis besar type data dapat dikategorikan menjadi :

1. Type data sederhana
   a. Type data sederhana tunggal, misalnya Integer, real, boolean dan karakter
   b. Type data sederhana majemuk, misalnya String
2. Struktur Data, meliputi
   a. Struktur data sederhana, misalnya array dan record
   b. Struktur data majemuk, yang terdiri dari Linier : Stack, Queue, serta List dan Multilist Non Linier : Pohon Biner dan Graph

Pemakaian struktur data yang tepat di dalam proses pemrograman akan menghasilkan algoritma yang lebih jelas dan tepat, sehingga menjadikan program secara keseluruhan lebih efisien dan sederhana. Struktur data yang ″standar″ yang biasanya digunakan dibidang informatika adalah :

- List linier (Linked List) dan variasinya
- Multilist
- Stack (Tumpukan)
- Queue (Antrian)
- Tree (Pohon)
- Graph (Graf)

Salah satu bentuk struktur data yang berisi kumpulan data yang tersusun secara sekuensial, saling bersambungan, dinamis dan terbatas adalah linked list (senarai berkait). Suatu linked list adalah suatu simpul (node) yang dikaitkan dengan simpul yang lain dalam suatu urutan tertentu. Suatu simpul dapat berbentuk suatu struktur atau class. Simpul harus mempunyai satu atau lebih elemen struktur atau class yang berisi data.

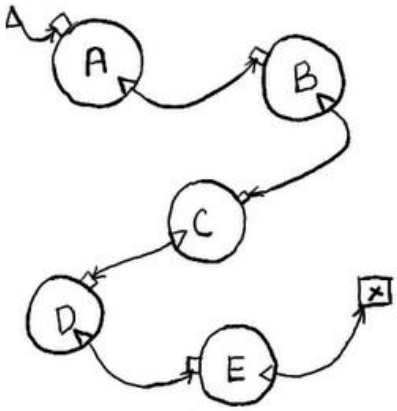
Secara teori, linked list adalah sejumlah node yang dihubungkan secara linier dengan bantuan pointer.

B. PEMBAHASAN / STUDI KASUS

Single Linked List

Linked List adalah salah satu bentuk struktur data, berisi kumpulan data
(node) yang tersusun secara sekuensial, saling sambungmenyambung,
dinamis dan terbatas.
- Linked List sering disebut juga Senarai Berantai
- Linked List saling terhubung dengan bantuan variabel pointer
- Masing-masing data dalam Linked List disebut dengan node (simpul) yang
menempati alokasi memori secara dinamis dan biasanya berupa struct
yang terdiri dari beberapa field.

Single Linked List adalah sebuah LINKED LIST yang menggunakan sebuah variabel pointer saja untuk menyimpan banyak data dengan metode LINKED LIST, suatu daftar isi yang saling berhubungan.
Ilustrasi single LINKED LIST:



Linked List

Pada gambar di atas, data terletak pada sebuah lokasi dalam sebuah memory, tempat yang disediakan memory untuk menyimpan data disebut node ? simpul, setiap node memiliki pointer (penunjuk) yang menunjuk ke node berikutnya sehingga terbentuk suatu untaian yang disebut single LINKED LIST. Bila dalam single LINKED LIST pointer hanya dapat bergerak ke satu arah saja, maju / mundur, kanan / kiri, sehingga pencarian datanya juga hanya satu arah saja.

SINGLE LINKED LIST MENGGUNAKAN HEAD DAN TAIL
- Dibutuhkan dua buah variabel pointer: head dan tail
- Head akan selalu menunjuk pada node pertama, sedangkan tail akan
selalu menunjuk pada node terakhir.

Inisialisasi LinkedList

```
TNode *head, *tail;
```

Fungsi Inisialisasi LinkedList

```
void init(){
head = NULL;
tail = NULL;
}
```

Function untuk mengetahui kosong tidaknya LinkedList

```
int isEmpty(){
if(tail == NULL) return 1;
else return 0;
}
```

PENAMBAHAN DATA
Pengkaitan node baru ke linked list di depan
Penambahan data baru di depan akan selalu menjadi head.

```
void insertDepan(int databaru){
TNode *baru;
baru = new TNode;
baru->data = databaru;
baru->next = baru;
if(isEmpty()==1){
head=baru;
tail=baru;
head->next=head;
tail->next=tail;
}
else {
baru->next = head;
head = baru;
tail->next = head;
}
cout<<"Data masuk\n";
}
```

Penambahan Data di belakang
Pada penambahan data di belakang, data akan selalu dikaitkan dengan tail, karena tail terletak di node paling belakang. Setelah dikaitkan dengan node baru, maka node baru tersebut akan menjadi tail.

```
void tambahBelakang(int databaru){ TNode *baru;
baru = new TNode;
baru->data = databaru;
baru->next = baru;
if(isEmpty()==1){
head=baru;
tail=baru;
head->next=head;
tail->next=tail;
}
else
{
tail->next = baru;
tail = baru;
tail->next = head;
}
cout<<"Data masuk\n";
}
```

Kelebihan dari Single Linked List dengan Head & Tail adalah pada penambahan data di belakang, hanya dibutuhkan tail yang mengikat node baru saja tanpa harus menggunakan perulangan pointer bantu.

Function untuk menampilkan isi linked list:

```
void tampil(){ TNode *b;
b = head; if(isEmpty()==0)
{
do
{ cout<data<<" ";
```

```
b=b->next;
}
while(b!=tail->next);
cout<<<"Masih kosong\n";
}
```

Pada prinsipnya sama dengan function tampil sebelumnya.

Function untuk menghapus data di depan

```
void hapusDepan(){ TNode
*hapus;
if (isEmpty()==0){ int d;
hapus = head;
d = head->data;
if(head != tail){
hapus = head;
head = head->next;
tail->next = head;
delete hapus;
}else{
head=NULL;
tail=NULL;
}
cout<<<" terhapus\n";
}
else cout<<"Masih kosong\n";
}
```

- Function di atas akan menghapus data terdepan (pertama) yang ditunjuk oleh head pada linked list
- Penghapusan node tidak boleh dilakukan jika keadaan node sedang ditunjuk oleh pointer, maka harus dilakukan penunjukkan terlebih dahulu dengan variabel hapus pada head, kemudian dilakukan pergeseran ke node berikutnya sehingga data setelah head menjadi head baru, kemudian menghapus variabel hapus dengan menggunakan perintah delete.
- Jika tail masih NULL maka berarti data masih kosong!

Function untuk menghapus data di belakang:

Dengan menggunakan Single Linked List ber-Head dan Tail, pengahapusan data di belakang akan mudah dilakukan, tidak seperti pada Single Linked List hanya ber-Head saja.

```
void hapusBelakang(){ TNode
*hapus,*bantu;
if (isEmpty()==0){ int d;
if(head == tail){ d = tail->data;
head = NULL;
tail = NULL;
}
else
{
bantu = head;
while(bantu->next != tail){
bantu = bantu->next;
}
hapus = tail;
tail = bantu;
d = hapus->data;
tail->next = head;
delete hapus;
}
cout<<<" terhapus\n";
}
else cout<<"Masih kosong\n";
}
```

- Function di atas akan menghapus data terbelakang (terakhir) yang ditunjuk oleh tail pada linked list

- Penghapusan node tidak boleh dilakukan jika keadaan node sedang ditunjuk oleh pointer, maka harus dilakukan penunjukkan terlebih dahulu dengan variabel hapus pada tail, kemudian dibutuhkan pointer bantu untuk membantu pergeseran dari head ke node berikutnya sampai sebelum tail, sehingga tail dapat ditunjukkan ke bantu tersebut, dan bantu tersebut akan menjadi tail yang baru. Setelah itu hapus variabel hapus dengan menggunakan perintah delete.
- Jika tail masih NULL maka berarti data masih kosong!

Function untuk menghapus semua elemen LinkedList

```
void clear(){ TNode *bantu,*hapus;
if(isEmpty() == 0){ bantu = head;
while(bantu->next!=head){
hapus = bantu;
bantu = bantu->next;
delete hapus;
}
head = NULL;
tail = NULL;
}
}
```

- Menggunakan pointer bantu yang digunakan untuk bergerak sepanjang
list, dan menggunakan pointer hapus yang digunakan untuk menunjuk
node-node yang akan dihapus.
- Pada saat pointer hapus menunjuk pada node yang akan dihapus, pointer
bantu akan bergerak ke node selanjutnya, dan kemudian pointer hapus
akan di delete.

Dikatakan single (singly) linked apabila hanya ada satu pointer yang menghubungkan setiap node. single artinya field pointer-nya hanya satu buah saja dan satu arah.
Linked list adalah struktur data yang paling dasar. Linked list terdiri atas sejumlah unsur-unsur dikelompokkan, atau terhubung, bersama-sama di suatu deret yang spesifik. Linked list bermanfaat di dalam memelihara koleksi-koleksi data, yang serupa dengan array.
Bagaimanapun juga, linked list dan array mempunyai perbedaan. Memakai Linked list lebih bagus dibandingkan dengan array/larik baik dalam banyak hal. Secara rinci, linked list lebih efisien di dalam melaksanakan penyisipan-penyisipan dan penghapusan-penghapusan. Linked list juga menggunakan alokasi penyimpanan secara dinamis, yang merupakan penyimpanan yang dialokasikan pada runtime. Karena di dalam banyak aplikasi, ukuran dari data itu tidak diketahui pada saat kompile, hal ini bisa merupakan suatu atribut yang baik juga. Setiap node akan berbentuk struct dan memiliki

satu buah field bertipe struct yang sama, yang berfungsi sebagai pointer. Dalam menghubungkan setiap node, kita dapat menggunakan cara first-create-first-access ataupun first-create-last-access. Yang berbeda dengan deklarasi struct sebelumnya adalah satu field bernama next, yang bertipe struct tnode. Hal ini sekilas dapat membingungkan. Namun, satu hal yang jelas, variabel next ini akan menghubungkan kita dengan node di sebelah kita, yang juga bertipe struct tnode.

•Single : artinya field pointer-nya hanya satu buah saja dan satu arah serta pada akhir node, pointernya menunjuk NULL
•Linked List : artinya node-node tersebut saling terhubung satu sama lain.
•Setiap node pada linked list mempunyai field yang berisi pointer ke node berikutnya, dan juga memiliki field yang berisi data.
•Node terakhir akan menunjuk ke NULL yang akan digunakan sebagai kondisi berhenti pada saat pembacaan isi linked list..

C. ID SECURITY

QWTD4452377-ASP-5244107

D. KESIMPULAN

Simpul adalah semacam tipe data yang kita buat sendiri sebagaimanan nama tipe yang telah disediakan oleh bahasa pemrograman.

Info adalah tempat penyimpanan data dengan tipe yang berbeda-beda sesuai keinginan.

Link adalah tempat penyimpanan alamat simpulnya (pointer).

Linked list adalah sebuah struktur untuk menyimpan data yang bersifat dinamik

Beberapa operasi dapat diterapkan pada linked list seperti
sisip(insert),hapus(delete)

Operasi-operasi yang ada pada linked list relatif lebih sulit jika dibandingkan dengan operasi-operasi pada struktur yang statis

Null adalah suatu kondisi khusus dimana pointer itu belum di set dengan sebuah address tertentu, artinya pointer tidak mrnunjuk ke alamat manapun.

E. DISKUSI

Apakah artikel ini sangat membantu ?

Dwi : Ya sangat membantu karna artikel ini sudah cukup lengkap

Dwi : Kalo begitu saya akan bertanya tentang pengertian **single linked list**?

Saya**;** single linked list adalah salah satu bentuk struktur data, berisi kumpulan data
(node) yang tersusun secara sekuensial, saling sambung menyambung,
dinamis dan terbatas.

F. REFERENCE

[1] O. M. Febriani and A. S. Putra, "Sistem Informasi Monitoring Inventori Barang Pada Balai Riset Standardisasi Industri Bandar Lampung," *J. Inform.*, vol. 13, no. 1, pp. 90–98, 2014.

[2] A. S. Putra, "Paperplain: Execution Fundamental Create Application With Borland Delphi 7.0 University Of Mitra Indonesia," 2018.

[3] A. S. Putra, "2018 Artikel Struktur Data, Audit Dan Jaringan Komputer," 2018.

[4] A. S. Putra, "ALIAS MANAGER USED IN DATABASE DESKTOP STUDI CASE DB DEMOS."

[5] A. S. Putra, "COMPREHENSIVE SET OF PROFESSIONAL FOR DISTRIBUTE COMPUTING."

[6] A. S. Putra, "DATA ORIENTED RECOGNITION IN BORLAND DELPHI 7.0."

[7] A. S. Putra, "EMBARCADERO DELPHI XE 2 IN GPU-POWERED FIREMONKEY APPLICATION."

[8] A. S. Putra, "HAK ATAS KEKAYAAN INTELEKTUAL DALAM DUNIA TEKNOLOGY BERBASIS REVOLUSI INDUSTRI 4.0."

[9] A. S. Putra, "IMPLEMENTASI PERATURAN PERUNDANGAN UU. NO 31 TAHUN 2000 TENTANG DESAIN INDUSTRI BERBASIS INFORMATION TECHNOLOGY."

[10] A. S. Putra, "IMPLEMENTATION OF PARADOX DBASE."

[11] A. S. Putra, "IMPLEMENTATION OF TRADE SECRET CASE STUDY SAMSUNG MOBILE PHONE."

[12] A. S. Putra, "IMPLEMENTATION PATENT FOR APPLICATION WEB BASED CASE STUDI WWW. PUBLIKLAMPUNG. COM."

[13] A. S. Putra, "IMPLEMENTATION SYSTEM FIRST TO INVENT IN DIGITALLY INDUSTRY."

[14] A. S. Putra, "MANUAL REPORT & INTEGRATED DEVELOPMENT ENVIRONMENT BORLAND

DELPHI 7.0."
[15] A. S. Putra, "PATENT AS RELEVAN SUPPORT RESEARCH."
[16] A. S. Putra, "PATENT FOR RESEARCH STUDY CASE OF APPLE. Inc."
[17] A. S. Putra, "PATENT PROTECTION FOR APPLICATION INVENT."
[18] A. S. Putra, "QUICK REPORT IN PROPERTY PROGRAMMING."
[19] A. S. Putra, "REVIEW CIRCUIT LAYOUT COMPONENT REQUIREMENT ON ASUS NOTEBOOK."
[20] A. S. Putra, "REVIEW TRADEMARK PATENT FOR INDUSTRIAL TECHNOLOGY BASED 4.0."
[21] A. S. Putra, "TOOLBAR COMPONENT PALLETTE IN OBJECT ORIENTED PROGRAMMING."
[22] A. S. Putra, "WORKING DIRECTORY SET FOR PARADOX 7."
[23] A. S. Putra, "ZQUERY CONNECTION IMPLEMENTED PROGRAMMING STUDI CASE PT. BANK BCA Tbk."
[24] A. S. Putra, D. R. Aryanti, and I. Hartati, "Metode SAW (Simple Additive Weighting) sebagai Sistem Pendukung Keputusan Guru Berprestasi (Studi Kasus: SMK Global Surya)," in *Prosiding Seminar Nasional Darmajaya*, 2018, vol. 1, no. 1, pp. 85–97.
[25] A. S. Putra and O. M. Febriani, "Knowledge Management Online Application in PDAM Lampung Province," in *Prosiding International conference on Information Technology and Business (ICITB)*, 2018, pp. 181–187.
[26] A. S. Putra, O. M. Febriani, and B. Bachry, "Implementasi Genetic Fuzzy System Untuk Mengidentifikasi Hasil Curian Kendaraan Bermotor Di Polda Lampung," *SIMADA (Jurnal Sist. Inf. dan Manaj. Basis Data)*, vol. 1, no. 1, pp. 21–30, 2018.
[27] A. S. Putra, H. Sukri, and K. Zuhri, "Sistem Monitoring Realtime Jaringan Irigasi Desa (JIDES) Dengan Konsep Jaringan Sensor Nirkabel," *IJEIS (Indonesian J. Electron. Instrum. Syst.*, vol. 8, no. 2, pp. 221–232.
[28] D. P. Sari, O. M. Febriani, and A. S. Putra, "Perancangan Sistem Informasi SDM Berprestasi pada SD Global Surya," in *Prosiding Seminar Nasional Darmajaya*, 2018, vol. 1, no. 1, pp. 289–294.