JOBSHEET

LINKED LIST DAN OPERASI DASAR PADA *LINKED LIST*

Oleh:

HENDRAWATY, ST., MT.

PROGRAM STUDI TEKNIK INFORMATIKA
JURUSAN TEKNOLOGI INFORMASI DAN KOMPUTER
POLITEKNIK NEGERI LHOKSEUMAWE
2020

HALAMAN PENGESAHAN INSTITUSI

LINKED LIST DAN OPERASI DASAR PADA*LINKED LIST*

Kegiatan Pengembangan Jobsheet ini Dibiayai dengan Sumber Dana DIPA Politeknik Negeri Lhokseumawe Tahun Anggaran 2020

Mengetahui,
Ketua Jurusan Teknologi Informasi
dan Komputer,

Muhammad Arhami, S.Si., M.Kom.
Nip. 197410292000031001

Penulis,

Hendrawaty, ST., MT.
Nip. 197002261998022001

Mengetahui/Mengesahkan:
Wakil Direktur Bidang Akademik, Kemahasiswaan, dan Alumni
Politeknik Negeri Lhokseumawe

Zamzami, ST., M.Eng
Nip. 197911122003121003

HALAMAN PENGESAHAN REVIEWER

Jobsheet *Linked List* dan Operasi Dasar pada *Linked List* yang disusun oleh:

Nama : Hendrawaty, ST., MT.
NIP : 197002261998022001
Jurusan : Teknologi Informasi dan Komputer

Telah memenuhi syarat-syarat penulisan Jobsheet yang dibiayai dengan sumber dana DIPA Politeknik Negeri Lhokseumawe Tahun Anggaran 2020.

Reviewer :

1. Salahuddin, ST, M.Cs.
   NIP. 197404242002121001 .....
2. Muhammad Rizka, SST, M. Kom.
   NIP. 198810091015041001 .....

Mengetahui,
Kepala Pusat Pengembangan
Pembelajaran dan Penjaminan Mutu

Ir. Herri Mahyar, MT.
NIP. 19621201 198902 1001

Menyetujui,
Ketua Departemen Pendidikan
Dan Pengembangan Pembelajaran

Ir. Jufriadi, MT.
NIP. 19641102 199303 1002

LABORATORIUM : TELEMATIKA
POLITEKNIK NEGERI LHOKSEUMAWE
PENGUJIAN: *LINKED LIST* DAN OPERASI DASAR PADA *LINKED LIST*

I. Capaian Praktikum/Kompetensi

- Menjelaskan konsep *linked list* (*single Linked list*)
- Menjelaskan operasi *insertion* dan *deletion* pada *linked list*.
- Mengimplementasikan *linked list* dan operasi dasar *linked list* dalam bentuk program menggunkan bahasa C .

II. Keselamatan Kerja

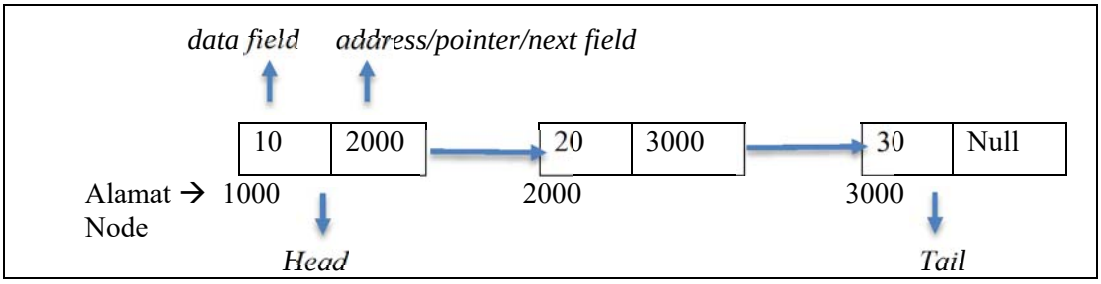
- Praktikum hanya dapat dilakukan atas petunjuk dosen pengasuh atau instruktur.
- Pastikan semua peralatan dalam keadaan baik sebelum dan sesudah melakukan praktikum serta menjaga semua peralatan yang digunakan.
- Perlu mematuhi semua peraturan di laboratorium yang digunakan
- Sebelum melakukan praktikum agar selalu mengawali dengan berdoa

III. Teori

Linked list adalah sekumpulan elemen yang mempunyai keterurutan tertentu, yang terhubung bersama melalui *link.* Berbeda dengan *array* dimana data/elemen akan disimpan dalam memori yang berurutan. Pada *linked list*, elemen bisa disimpan pada lokasi memori mana saja.

Pada *linked list* elemen direpresentasikan sebagai *node. Node* terdiri dari dua *field* yaitu: *data field* dan *address/next field*. *Data field* berisi nilai atau data, sedangkan *address/next field* berisi *link* dari *node* berikutmya (alamat *node* berikutnya) yang disebut dengan istilah *next.* Satu *node* dengan *node* lainnya terkait satu sama lain melalui *link* yang terdapat pada *address/pointer/next field*, seperti yang diperlihatkan Gambar 1. Node pertama berisi data dan berisi *next* (alamat *node* ke-2), *node* kedua berisi data dan berisi *next* (alamat *node* ke-3), demikian seterusnya hingga *node* terakhir. Khusus *node* terakhir , *address/next field*-nya berisi *NULL,* yang menunjukkan bahwa *node* tersebut adalah akhir dari *linked list*. *Node* pertama pada *linked list* disebut

dengan ***head*** dan *node* terakhir disebut tail.



Gambar 1. Representasi *node* pada *linked list*

Deklarasi *node*

Node terdiri dari 2 *field*. Data yang ada pada *data field* bisa saja bertipe *int, float*, atau *char*. Sedangkan pada *address field* bertipe *pointer*. Gambar 2. Memperlihatkan deklarasi sebuah *node* dengan *structure*.

```
struct node
{
   int data;
   int key;
   struct node *next;
};
```

Gambar 2. Deklarsi *node*

Membuat sebuah *link.*

Telah dijelaskan sebelumnya bahwa elemen bisa disimpan pada lokasi memori mana saja. Oleh karena itu alokasi node di memori dilakukan secara dinamis menggunakan fungsi *melloc().contoh:*

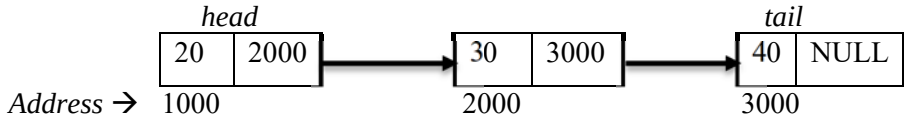
```
struct node*link=(struct node*)malloc(sizeof(struct node));
```

Operasi pada *Linked List*

Operasi dasar pada *linked list* yaitu : *Insertion* - menambah elemen pada *linked list*, *Deletion*-menghapus elemen pada linked list, *Display*-menampilkan elemen-elemen pada *linked list* secara lengkap, dan *Search*-mencari elemen pada *linked list* .

Insertion Operation

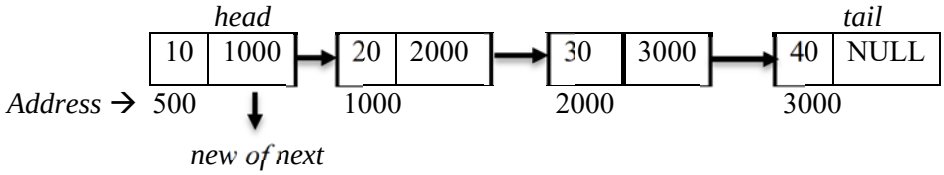
Insertion - menambah sebuah elemen pada *linked list*. Tiga jenis operasi *insertion* yaitu: *insertion* element di awal *linked list, insertion* elemen di akhir *linked list*, dan *insertion* elemen di posisi tertentu dari *linked list.* Misal terdapat 3 buah *node* pada *linked list* seperti pada Gambar 3., yang menggambarkan keadaan *linked list* sebelum dilakukan operasi *insertion.*



Gambar 3. *Linked list* sebelum operasi insertion

***insertion* di awal**

Misal di inginkan untuk menambah elemen10 di awal linked list pada Gambar 3. *Output linked list* setelah insertion menjadi seperti pada Gambar 4.



Gambar 4. *Output* operasi *insertion* pada awal *linked list*

Setiap node harus dialokasikan ke memori secara dinamis menggunakan fungsi *malloc()*. Contoh:

```
new = (struct node *) malloc( sizeof (struct node));
```

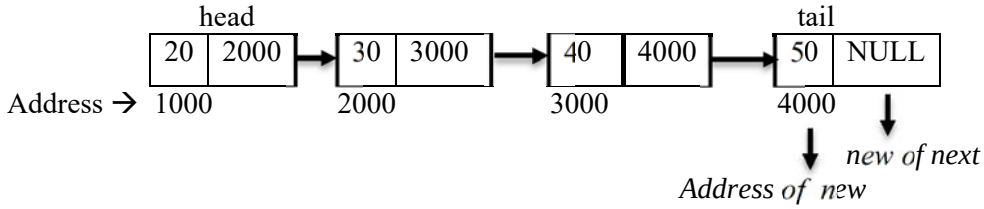
Logika sederhana dari program untuk operasi insertion tersebut tampak pada Gambar 5.

```
scanf("%d",&value); → value =10
new → data = value;
new → next = head;
head = new;
```

Gambar 5. *Simple code* insertion di awal *list*.

insertion* di akhir *linked list

misalkan di-inginkan untuk menginsert data 50 di akhir linked list tersebut(Gambar 3.). *Output linked list* setelah operasi *insertion* menjadi seperti pada Gambar 6.



Gambar 6. *Output* operasi *insertion* di akhir *linked list*

Simple code untuk operasi insertion tersebut diperlihatkan pada Gambar 7.

```
scanf ("%d",&value); → value = 50
new → data = value
tail→ next = new
new → next = null
 tail = new
```

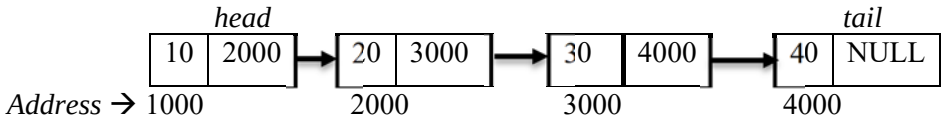
Gambar 7. *Simple code* insertion di akhir *list*.

Deletion operation

Deletion berfungsi untuk menghapus elemen pada *linked list*. Tiga jenis operasi *deletion* pada *linked list* yaitu: *delete* element di awal *list*, di akhir *list,* dan *delete* elemen pada posisi tertentu. *Deleting node* dilakukan dengan cara memutus *node* yang ingin di-*delete*, dengan cara mengisi *next field* dari *node* tersebut dengan NULL.

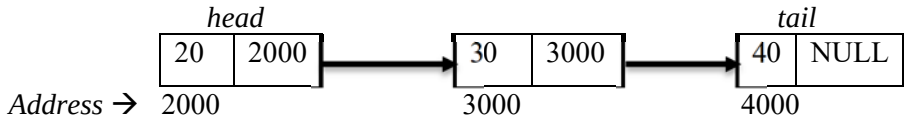
Deletion* operasi di awal *linked list

Misal terdapat *linked list* seperti pada gambar 10. Dan di-inginkan untuk menghapus elemen pertama dari *linked list*.



Gambar 10. *Linked list* sebelum operasi *deletion*

Output linked list setelah *deleting* elemen awal list diperlihatkan pada Gambar 11.



Gambar 11. *Linked list* setealah *deletion* pada awal *list.*

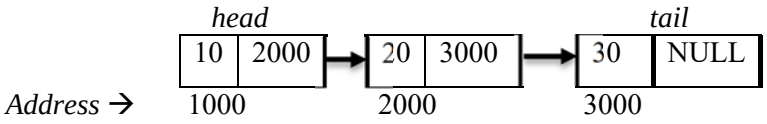
Logika sederhana dari program untuk operasi *deletion* diperlihatkan pada Gambar 12.

```
temp=head;
head = head → next
temp → next = NULL
```

Gambar 12. *Simple code deletion* di awal *list*.

***Deletion* operasi di akhir linked list**

di-inginkan untuk menghapus elemen terakhir dari *linked list* (pada Gambar 10). *Output* dari operasi ini dapat dilihat pada Gambar 13.



Gambar 13. *Output* operasi *insertion* pada awal *linked list*

Simple code untuk operasi *deletion* tersebut diperlihatkan oleh Gambar 14.

```
temp = head ;
while(temp → next != tail)
{
    temp = temp →next;
}
temp → next = NULL
tail = temp ;
```

Gambar 13. *Simple code deletion* di akhir *list.*

Display operation

Display operation menampilkan elemen *linked list* satu persatu secara keseluruhan. Gambar 16. memperlihatkan *simple code* untuk *display.*

```
//display the list
void printList()
{
    struct node *ptr = head;
    printf("\n[ ");

    //start from the beginning
    while(ptr != NULL)
    {
      printf("(%d,%d) ",ptr->key,ptr->data);
      ptr = ptr->next;
    }

    printf(" ]");
}
```

Gambar 15. *Simple code* operasi *display*

IV. Alat/Bahan

1. Komputer Pribadi (*Personal Computer*)
2. Perangkat Lunak Dev-C++, gcc, vim atau pico

3. Sistem operasi *windows* atau *linux*

V. Prosedur Praktikum

1. Berdasarkan langkah yang telah dijelaskan diatas, buatlah program *linked list* yang terdiri dari 4 *node*, kemudian tampilkan seluruh elemen linked list tersebut
2. Berdasarkan langkah yang telah dijelaskan diatas, buatlah program untuk mendemontrasikan operasi insert di awal *linked list*. Kemudian tampilkan seluruh elemen *linked list* sebelum dan sesudah operasi *insertion* dilakukan.
3. Berdasarkan langkah-langkah yang telah dijelaskan diatas, buatlah program untuk mendemontrasikan operasi *delete* pada awal *linked list.* Tampilkan seluruh elemen *linked list* sebelum dan sesudah operasi *deletion* dilakukan.

VI. Data Percobaan

Buat program seperti tugas diatas dan hasil eksekusinya dalam borang berikut:

Tabel1. *Program* dan hasil eksekusi

No Perc.	Program (Source Code)	Hasil (Output)
1.		
2.		
3.		
4.		

VII. Analisa dan Kesimpulan

1. Analisa hasil dari eksekusi program diatas, berikan maksud dari setiap baris program yang diberikan.
2. Dari hasil percobaan yang dilakukan, buat kesimpulan dari masing-masing percobaan diatas.

VIII. Daftar Pustaka

Jain, H. (2017). Problem Solving in Data Structures & Algorithms Using C. Bhopal, India.

Tutorialspoint. (2016). Data Structures & Algorithms Simply Easy Learning, Tutorial points Ltd, www.tutorialspoint.com.