LAPORAN PRAKTIKUM STRUKTUR DATA
LAPORAN AWAL PRAKTIKUM

Pertemuan ke-09

Linked List

Disusun Oleh:

Nama Lengkap	:	Nova Ardiansyah
NIM	:	211011401309
Kelas	:	04-TPLE008

TEKNIK INFORMATIKA
FAKULTAS TEKNIK
UNIVERSITAS PAMULANG

Jl. Raya Puspitek No. 11 Buaran, Serpong Telp. (021) 7412566, Fax. (021) 7412566
Tangerang Selatan - Banten

A. RANGKUMAN MATERI

Linked List adalah struktur berupa rangkaian elemen saling berkait dimana tiap elemen dihubungkan ke elemen yang lain melalui pointer. Pointer adalah alamat elemen. Penggunaan pointer untuk mengacu elemen berakibat elemen-elemen bersebelahan secara logic walaupun tidak bersebelahan secara fisik di memori. Linked List merupakan kumpulan komponen yang saling berkaitan satu dengan yang lain melalui pointer. Masing-masing komponen sering disebut dengan simpul atau verteks.

Singly Linked List Merupakan Linked List yang paling sederhana. Setiap simpul dibagi menjadi dua bagian yaitu bagian isi dan bagian pointer. Bagian isi merupakan bagian yang berisi data yang disimpan oleh simpul, sedangkan bagian pointer merupakan bagian yang berisi alamat dari simpul berikutnya.

Doubly Linked List merupakan Linked List dimana setiap simpul dibagi menjadi tiga bagian, yaitu bagian isi, bagian pointer kiri, dan bagian pointer kanan. Bagian isi merupakan bagian yang berisi data yang disimpan oleh simpul, bagian pointer kiri merupakan bagian yang berisi alamat dari simpul sebelumnya dan bagian pointer kanan merupakan bagian yang berisi alamat dari simpul berikutnya. Deklarasi Doubly Linked List.

B. TUGAS PENDAHULUAN

1. Apa yang dimaksud dengan Linked List!
   Jawab :

   Linked List adalah struktur berupa rangkaian elemen saling berkait dimana tiap elemen dihubungkan ke elemen lain melalui pointer. Pointer juga merupakan alamat dari sebuah elemen.

2. Jelaskan perbedaan antara singly linked list, doubly linked list, dan circular linked list!
   Jawab :

Singly Linked List merupakan suatu linked list yang hanya memiliki satu variabel pointer saja. Dimana pointer tersebut menunjuk ke node selanjutnya, biasanya field pada tail menunjuk ke NULL.

Doubly Linked List merupakan suatu linked list yang memiliki dua variabel pointer yaitu pointer yang menunjuk ke node selanjutnya dan pointer yang menunjuk ke node sebelumnya. Setiap head dan tailnya juga menunjuk ke NULL.

Circular Linked List merupakan suatu linked list dimana tail (node terakhir) menunjuk ke head (node pertama). Jadi tidak ada pointer yang menunjuk NULL.

3. Jelaskan operasi-operasi pada singly linked list!
   Jawab :

   - **Insert** Istilah Insert berarti menambahkan sebuah simpul baru ke dalam suatu linked list.
   - **Konstruktor** Fungsi ini membuat sebuah linked list yang baru dan masih kosong.
   - **IsEmpty** Fungsi ini menentukan apakah linked list kosong atau tidak.
   - **Find First** Fungsi ini mencari elemen pertama dari linked list.
   - **Find Next** Fungsi ini mencari elemen sesudah elemen yang ditunjuk now.
   - **Retrieve** Fungsi ini mengambil elemen yang ditunjuk oleh now. Elemen tersebut lalu dikembalikan oleh fungsi.
   - **Update** Fungsi ini mengubah elemen yang ditunjuk oleh now dengan isi dari sesuatu.
   - **Delete Now** Fungsi ini menghapus elemen yang ditunj uk oleh now. Jika yang dihapus adalah elemen pertama dari linked list (head), head akan berpindah ke elemen berikut.

4. Jelaskan operasi-operasi pada doubly linked list!
   Jawab :

- **Insert Tail** Fungsi insert tail berguna untuk menambah simpul di belakang (sebelah kanan) pada sebuah linked list.
- **Insert Head** Sesuai dengan namanya, fungsi Insert Head berguna untuk menambah simpul di depan (sebelah kiri). Fungsi ini tidak berada jauh dengan fungsi Insert Tail yang telah dijelaskan sebelumnya.
- **DeleteTail** Fungsi Delete Tail berguna untuk menghapus simpul dari belakang. Fungsi ini merupakan kebalikan dari fungsi Insert Tail yang menambah simpul dibelakang. Fungsi Delete Tail akan mengarahkan Now kepada Tail dan kemudian memanggil fungsi Delete Now.
- **Delete Head** Fungsi Delete Head merupakan kebalikan dari fungsi Delete Tail yang menghapus simpul dari belakang, sedangkan Delete Head akan menghapus simpul dari depan (sebelah kiri). Fungsi Delete Head akan mengarahkan Now kepada Head dan kemudian memanggil fungsi Delete Now.

C. TUGAS PRAKTIKUM

- Lat10_1

```
#include <iostream>
using namespace std;

struct node {
  char Isi;
  node* Next;
};

typedef node* simpul;

void Sisip_Depan(simpul& L, char elemen);
void Sisip_Belakang(simpul& L, char elemen);
void Sisip_Tengah1(simpul& L, char elemen1, char elemen2);
void Sisip_Tengah2(simpul& L, char elemen1, char elemen2);
void Hapus_Depan(simpul& L);
void Hapus_Belakang(simpul& L);
void Hapus_Tengah(simpul& L, char elemen);
void Cetak(simpul L);

int main()
{
  cout << "Nama \t: Nova Ardiansyah\n";
  cout << "NIM \t: 211011401309\n";
  cout << "========================\n\n";

  char huruf, huruf2;
  simpul L = NULL;

  cout << "=============== Operasi SLL ===============\n\n";

  cout << "1. Sisip Depan\n";
  cout << "==============\n\n";

  cout << "Masukkan huruf : ";
  cin >> huruf;
  Sisip_Depan(L, huruf);

  cout << "Masukkan huruf : ";
  cin >> huruf;
  Sisip_Depan(L, huruf);
  Cetak(L);

  cout << "\n\n2. Sisip Belakang\n";
  cout << "=================\n\n";

  cout << "Masukkan huruf : ";
  cin >> huruf;
  Sisip_Belakang(L, huruf);

  cout << "Masukkan huruf : ";
  cin >> huruf;
  Sisip_Belakang(L, huruf);
  Cetak(L);

  cout << "\n\n3. Sisip setelah simpul tertentu\n";
  cout << "================================\n\n";

  cout << "Masukkan huruf : ";
```

```
  cin >> huruf;

  cout << "Masukkan huruf setelah huruf " << huruf << " : ";
  cin >> huruf2;
  Sisip_Tengah1(L, huruf, huruf2);
  Cetak(L);

  cout << "\n\n4. Sisip sebelum simpul tertentu\n";
  cout << "================================\n\n";

  cout << "Masukkan huruf : ";
  cin >> huruf;
  Sisip_Tengah2(L, huruf, huruf2);
  Cetak(L);

  cout << "\n\n5. Hapus simpul depan\n";
  cout << "=====================\n\n";

  Hapus_Depan(L);
  Cetak(L);

  cout << "6. Hapus simpul belakang\n";
  cout << "========================\n\n";

  Hapus_Belakang(L);
  Cetak(L);

  cout << "\n\n7. Hapus simpul tertentu\n";
  cout << "========================\n\n";

  cout << "Masukkan huruf : ";
  cin >> huruf;

  Hapus_Tengah(L, huruf);
  Cetak(L);

  return 0;
}

void Sisip_Depan(simpul& L, char elemen)
{
  simpul baru = new node;
  baru->Isi = elemen;
  baru->Next = L;

  L = baru;
}

void Sisip_Tengah1(simpul& L, char elemen1, char elemen2)
{
  if (L == NULL)
  {
    cout << "List kosong\n";
    return;
  }

  simpul bantu = L;
  while (bantu != NULL && bantu->Isi != elemen2)
  {
    bantu = bantu->Next;
  }

  if (bantu == NULL)
```

```
  {
   cout << "Simpul dengan elemen " << elemen2 << " tidak ditemukan\n";
   return;
  }

  simpul baru = new node;
  baru->Isi = elemen1;
  baru->Next = bantu->Next;

  bantu->Next = baru;
}

void Sisip_Tengah2(simpul& L, char elemen1, char elemen2)
{
  if (L == NULL)
  {
   cout << "List kosong\n";
   return;
  }

  if (L->Isi == elemen2)
  {
   Sisip_Depan(L, elemen1);
   return;
  }

  simpul bantu = L;
  while (bantu->Next != NULL && bantu->Next->Isi != elemen2)
  {
   bantu = bantu->Next;
  }

  if (bantu->Next == NULL)
  {
   cout << "Simpul dengan elemen " << elemen2 << " tidak ditemukan\n";
   return;
  }

  simpul baru = new node;
  baru->Isi = elemen1;
  baru->Next = bantu->Next;

  bantu->Next = baru;
}

void Sisip_Belakang(simpul& L, char elemen)
{
  simpul baru = new node;
  baru->Isi = elemen;
  baru->Next = NULL;

  if (L == NULL)
  {
   L = baru;
  }
  else
  {
   simpul bantu = L;
   while (bantu->Next != NULL)
   {
    bantu = bantu->Next;
   }
   bantu->Next = baru;
```

```
  }
}

void Cetak(simpul L)
{
  if (L == NULL)
  {
    cout << "List kosong\n";
  }
  else
  {
    simpul bantu = L;
    while (bantu != NULL)
    {
      cout << bantu->Isi << "-->";
      bantu = bantu->Next;
    }
    cout << endl;
  }
}

void Hapus_Depan(simpul& L)
{
  if (L == NULL)
  {
    cout << "List kosong\n";
    return;
  }

  simpul hapus = L;
  L = L->Next;
  delete hapus;
}

void Hapus_Belakang(simpul& L)
{
  if (L == NULL)
  {
    cout << "List kosong\n";
    return;
  }

  if (L->Next == NULL)
  {
    delete L;
    L = NULL;
    return;
  }

  simpul bantu = L;
  while (bantu->Next->Next != NULL)
  {
    bantu = bantu->Next;
  }

  delete bantu->Next;
  bantu->Next = NULL;
}

void Hapus_Tengah(simpul& L, char elemen)
{
  if (L == NULL)
  {
```

```
  cout << "List kosong\n";
  return;
 }

 if (L->Isi == elemen)
 {
  Hapus_Depan(L);
  return;
 }

 simpul bantu = L;
 while (bantu->Next != NULL && bantu->Next->Isi != elemen)
 {
  bantu = bantu->Next;
 }

 if (bantu->Next == NULL)
 {
  cout << "Simpul dengan elemen " << elemen << " tidak ditemukan\n";
  return;
 }

 simpul hapus = bantu->Next;
 bantu->Next = hapus->Next;
 delete hapus;
}
```

```
Nama    : Nova Ardiansyah
NIM     : 211011401309
=========================

=============== Operasi SLL ===============

1. Sisip Depan
==============

Masukkan huruf : a
Masukkan huruf : b
b-->a-->


2. Sisip Belakang
=================

Masukkan huruf : c
Masukkan huruf : d
b-->a-->c-->d-->

3. Sisip setelah simpul tertentu
================================

Masukkan huruf : e
Masukkan huruf setelah huruf e : a
b-->a-->e-->c-->d-->


4. Sisip sebelum simpul tertentu
================================

Masukkan huruf : f
b-->f-->a-->e-->c-->d-->


5. Hapus simpul depan
=====================
```

```
6. Hapus simpul belakang
========================

f-->a-->e-->c-->


7. Hapus simpul tertentu
========================

Masukkan huruf : e
f-->a-->c-->

--------------------------------
Process exited after 25.16 seconds with return value 0
Press any key to continue . . .
```

- Lat10_2

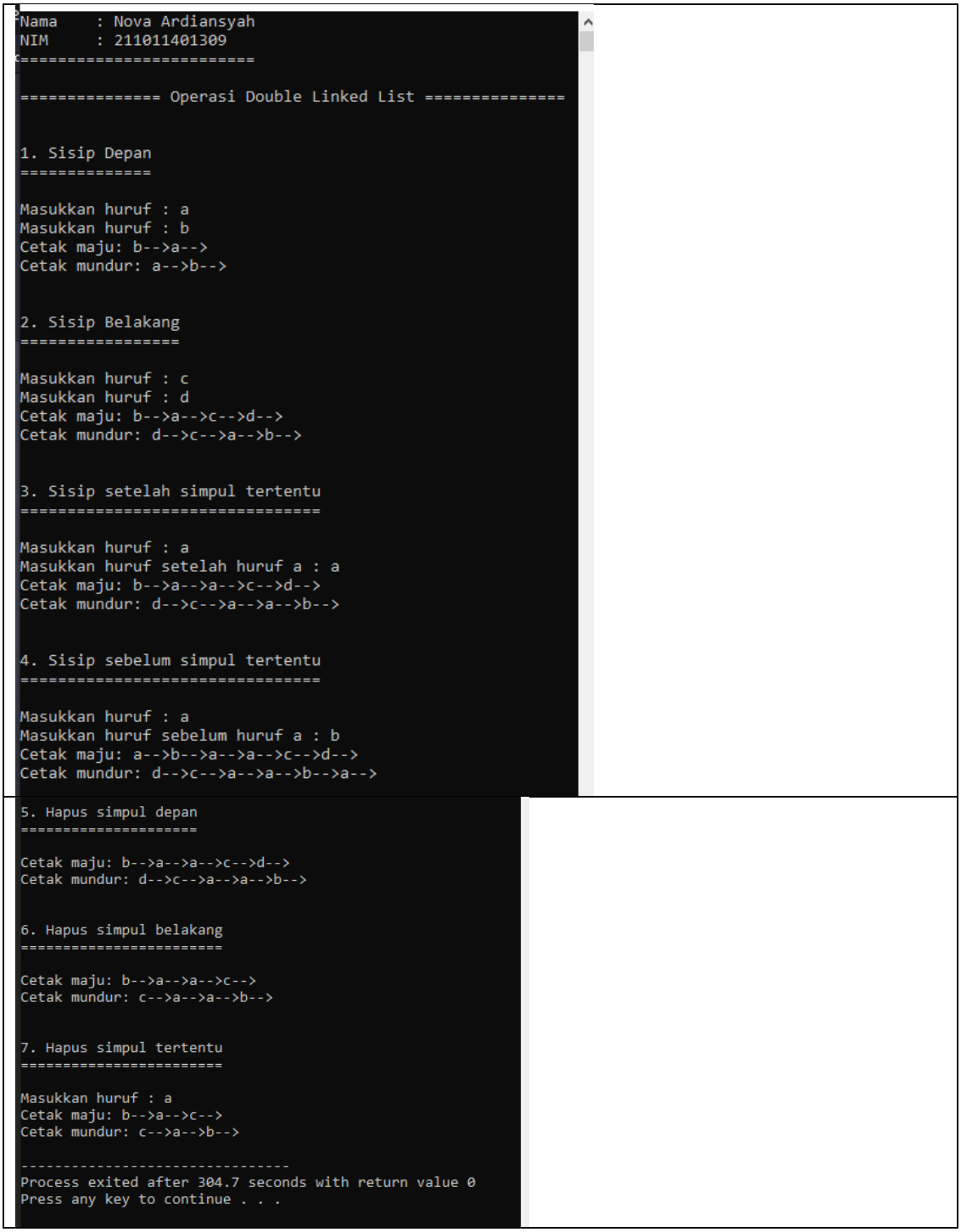
```
#include <iostream>
using namespace std;

struct node {
  char Isi;
  node* Prev;
  node* Next;
};

typedef node* simpul;

void Sisip_Depan(simpul& L, char elemen);
void Sisip_Belakang(simpul& L, char elemen);
void Sisip_Tengah1(simpul& L, char elemen1, char elemen2);
void Sisip_Tengah2(simpul& L, char elemen1, char elemen2);
void Hapus_Depan(simpul& L);
void Hapus_Belakang(simpul& L);
void Hapus_Tengah(simpul& L, char elemen);
void Cetak_Maju(simpul L);
void Cetak_Mundur(simpul L);

int main()
{
  cout << "Nama \t: Nova Ardiansyah\n";
  cout << "NIM \t: 211011401309\n";
  cout << "=========================\n\n";

  char huruf, huruf2;
  simpul L = NULL;

  cout << "=============== Operasi Double Linked List ===============\n";

  cout << "\n\n1. Sisip Depan\n";
  cout << "==============\n\n";

  cout << "Masukkan huruf : ";
  cin >> huruf;
  Sisip_Depan(L, huruf);

  cout << "Masukkan huruf : ";
  cin >> huruf;
  Sisip_Depan(L, huruf);
  Cetak_Maju(L);
  Cetak_Mundur(L);

  cout << "\n\n2. Sisip Belakang\n";
  cout << "=================\n\n";
```

```
  cout << "Masukkan huruf : ";
  cin >> huruf;
  Sisip_Belakang(L, huruf);

  cout << "Masukkan huruf : ";
  cin >> huruf;
  Sisip_Belakang(L, huruf);

  Cetak_Maju(L);
  Cetak_Mundur(L);

  cout << "\n\n3. Sisip setelah simpul tertentu\n";
  cout << "================================\n\n";

  cout << "Masukkan huruf : ";
  cin >> huruf;

  cout << "Masukkan huruf setelah huruf " << huruf << " : ";
  cin >> huruf2;
  Sisip_Tengah1(L, huruf, huruf2);

  Cetak_Maju(L);
  Cetak_Mundur(L);

  cout << "\n\n4. Sisip sebelum simpul tertentu\n";
  cout << "================================\n\n";

  cout << "Masukkan huruf : ";
  cin >> huruf;
  cout << "Masukkan huruf sebelum huruf " << huruf << " : ";
  cin >> huruf2;
  Sisip_Tengah2(L, huruf, huruf2);

  Cetak_Maju(L);
  Cetak_Mundur(L);

  cout << "\n\n5. Hapus simpul depan\n";
  cout << "=====================\n\n";

  Hapus_Depan(L);
  Cetak_Maju(L);
  Cetak_Mundur(L);

  cout << "\n\n6. Hapus simpul belakang\n";
  cout << "========================\n\n";

  Hapus_Belakang(L);
  Cetak_Maju(L);
  Cetak_Mundur(L);

  cout << "\n\n7. Hapus simpul tertentu\n";
  cout << "========================\n\n";

  cout << "Masukkan huruf : ";
  cin >> huruf;

  Hapus_Tengah(L, huruf);
  Cetak_Maju(L);
  Cetak_Mundur(L);

  return 0;
}
```

```
// FUNCTION

void Sisip_Depan(simpul& L, char elemen)
{
  simpul baru = new node;
  baru->Isi = elemen;
  baru->Prev = NULL;
  baru->Next = L;

  if (L != NULL)
    L->Prev = baru;

  L = baru;
}

void Sisip_Tengah1(simpul& L, char elemen1, char elemen2)
{
  if (L == NULL)
  {
    cout << "List kosong\n";
    return;
  }

  simpul bantu = L;
  while (bantu != NULL && bantu->Isi != elemen2)
    bantu = bantu->Next;

  if (bantu == NULL)
  {
    cout << "Simpul dengan elemen " << elemen2 << " tidak ditemukan\n";
    return;
  }

  simpul baru = new node;
  baru->Isi = elemen1;
  baru->Prev = bantu;
  baru->Next = bantu->Next;

  if (bantu->Next != NULL)
    bantu->Next->Prev = baru;

  bantu->Next = baru;
}

void Sisip_Tengah2(simpul& L, char elemen1, char elemen2)
{
  if (L == NULL)
  {
    cout << "List kosong\n";
    return;
  }

  simpul bantu = L;
  while (bantu != NULL && bantu->Isi != elemen2)
    bantu = bantu->Next;

  if (bantu == NULL)
  {
    cout << "Simpul dengan elemen " << elemen2 << " tidak ditemukan\n";
    return;
  }

  simpul baru = new node;
```

```
  baru->Isi = elemen1;
  baru->Prev = bantu->Prev;
  baru->Next = bantu;

  if (bantu->Prev != NULL)
    bantu->Prev->Next = baru;
  else
    L = baru;

  bantu->Prev = baru;
}

void Sisip_Belakang(simpul& L, char elemen)
{
  simpul baru = new node;
  baru->Isi = elemen;
  baru->Prev = NULL;
  baru->Next = NULL;

  if (L == NULL)
  {
    L = baru;
    return;
  }

  simpul bantu = L;
  while (bantu->Next != NULL)
    bantu = bantu->Next;

  bantu->Next = baru;
  baru->Prev = bantu;
}

void Hapus_Depan(simpul& L)
{
  if (L == NULL)
  {
    cout << "List kosong\n";
    return;
  }

  simpul hapus = L;
  L = L->Next;

  if (L != NULL)
    L->Prev = NULL;

  delete hapus;
}

void Hapus_Belakang(simpul& L)
{
  if (L == NULL)
  {
    cout << "List kosong\n";
    return;
  }

  simpul bantu = L;
  while (bantu->Next != NULL)
    bantu = bantu->Next;

  if (bantu->Prev != NULL)
```

```
    bantu->Prev->Next = NULL;
  else
    L = NULL;

  delete bantu;
}

void Hapus_Tengah(simpul& L, char elemen)
{
  if (L == NULL)
  {
    cout << "List kosong\n";
    return;
  }

  simpul bantu = L;
  while (bantu != NULL && bantu->Isi != elemen)
    bantu = bantu->Next;

  if (bantu == NULL)
  {
    cout << "Simpul dengan elemen " << elemen << " tidak ditemukan\n";
    return;
  }

  if (bantu->Prev != NULL)
    bantu->Prev->Next = bantu->Next;
  else
    L = bantu->Next;

  if (bantu->Next != NULL)
    bantu->Next->Prev = bantu->Prev;

  delete bantu;
}

void Cetak_Maju(simpul L)
{
  cout << "Cetak maju: ";
  simpul bantu = L;
  while (bantu != NULL)
  {
    cout << bantu->Isi << "-->";
    bantu = bantu->Next;
  }
  cout << endl;
}

void Cetak_Mundur(simpul L)
{
  cout << "Cetak mundur: ";
  simpul bantu = L;
  while (bantu != NULL && bantu->Next != NULL)
    bantu = bantu->Next;

  while (bantu != NULL)
  {
    cout << bantu->Isi << "-->";
    bantu = bantu->Prev;
  }
  cout << endl;
}
```



```
Nama    : Nova Ardiansyah
NIM     : 211011401309
=========================

=============== Operasi Double Linked List ===============


1. Sisip Depan
==============

Masukkan huruf : a
Masukkan huruf : b
Cetak maju: b-->a-->
Cetak mundur: a-->b-->


2. Sisip Belakang
=================

Masukkan huruf : c
Masukkan huruf : d
Cetak maju: b-->a-->c-->d-->
Cetak mundur: d-->c-->a-->b-->


3. Sisip setelah simpul tertentu
================================

Masukkan huruf : a
Masukkan huruf setelah huruf a : a
Cetak maju: b-->a-->a-->c-->d-->
Cetak mundur: d-->c-->a-->a-->b-->


4. Sisip sebelum simpul tertentu
================================

Masukkan huruf : a
Masukkan huruf sebelum huruf a : b
Cetak maju: a-->b-->a-->a-->c-->d-->
Cetak mundur: d-->c-->a-->a-->b-->a-->


5. Hapus simpul depan
=====================

Cetak maju: b-->a-->a-->c-->d-->
Cetak mundur: d-->c-->a-->a-->b-->


6. Hapus simpul belakang
========================

Cetak maju: b-->a-->a-->c-->
Cetak mundur: c-->a-->a-->b-->


7. Hapus simpul tertentu
========================

Masukkan huruf : a
Cetak maju: b-->a-->c-->
Cetak mundur: c-->a-->b-->

--------------------------------
Process exited after 304.7 seconds with return value 0
Press any key to continue . . .
```

LAPORAN PRAKTIKUM STRUKTUR DATA
LAPORAN AKHIR PRAKTIKUM

Pertemuan ke-09

Linked List

Disusun Oleh:

Nama Lengkap	:	Nova Ardiansyah
NIM	:	211011401309
Kelas	:	04-TPLE008

TEKNIK INFORMATIKA
FAKULTAS TEKNIK
UNIVERSITAS PAMULANG

Jl. Raya Puspitek No. 11 Buaran, Serpong Telp. (021) 7412566, Fax. (021) 7412566
Tangerang Selatan - Banten

A. TUGAS AKHIR

1. Buatlah Program Menu untuk menampilkan program diatas!

   Jawab :

```
#include<iostream>
#include<cstdlib>

using namespace std;
typedef struct node *simpul;

struct node
{
  char isi;
  simpul next;
};

void sisipDepan (simpul &l, char elemen);
void sisipBelakang(simpul &l, char elemen);
void sisipTengah1 (simpul &l, char elemen1, char elemen2);
void sisipTengah2 (simpul &l, char elemen1, char elemen2);
void hapusDepan (simpul &l);
void hapusBelakang(simpul &l);
void hapusTengah (simpul &l, char elemen);
void cetak (simpul l);

int main()
{
  cout << "Nama \t: Nova Ardiansyah\n";
  cout << "NIM \t: 211011401309\n";
  cout << "========================\n\n";

  char huruf, huruf2;
  simpul l = NULL;
  int menu;
  cout << "OPERASI PADA SINGLE LINKED LIST" << endl << endl;

    do {
      cout << "Menu : " << endl;
      cout << "1. sisip depan" << endl;
      cout << "2. sisip belakang" << endl;
      cout << "3. sisip tengah1" << endl;
      cout << "4. sisip tengah2" << endl;
      cout << "5. hapus depan" << endl;
      cout << "6. hapus belakang" << endl;
      cout << "7. hapus tengah" << endl;
      cout << "8. cetak" << endl;
      cout << "9. keluar" << endl;

      cout << "Pilih menu : ";
      cin     >> menu;
      cout << endl;

      switch(menu) {
        case 1 :
          cout << "## Masukan huruf : ";
          cin >> huruf;
          sisipDepan(l, huruf);
          cout << endl;
          break;
```

```
      case 2 :
       cout << "## Masukan huruf : ";
       cin >> huruf;
       sisipBelakang(l, huruf);
       cout << endl;
       break;
      case 3 :
       cout << "## Masukan huruf : ";
       cin >> huruf; cout << endl;
       cout << "## Disisip setelah huruf : ";
       cin  >> huruf2; cout << endl;
       sisipTengah1(l, huruf, huruf2);
       break;
      case 4 :
       cout << "## Masukan huruf : ";
       cin  >> huruf; cout << endl;
       cout << "## Disisip sebelum huruf : ";
       cin  >> huruf2; cout << endl;
       sisipTengah2(l, huruf, huruf2);
       break;
      case 5 :
       hapusDepan(l);
       cout << "## Simpul depan dihapus" << endl << endl;
       break;
      case 6 :
       hapusBelakang(l);
       cout << "## Simpul belakang dihapus" << endl << endl;
       break;
      case 7 :
       cout << "## Masukan huruf tengah yang akan dihapus : ";
       cin >> huruf;
       hapusTengah(l, huruf);
       cout << endl;
       break;
      case 8 :
       cetak(l);
       cout << endl << endl;
       break;
      case 9 :
       cout << "## Keluar program..." << endl;
       break;
     default :
      cout << "## kode salah, coba lagi" << endl << endl;
      break;
     }
   }

  while(menu != 9);
  return 0;
}

void sisipDepan (simpul &l, char elemen)
{
   simpul baru;
   baru = (simpul) malloc(sizeof(simpul));
   baru->isi = elemen;
   baru-> next = NULL;
   if(l == NULL)
      l = baru;
   else
   {
      baru->next = l;
      l = baru;
```

```
   }
}

//fungsi sisip setelah simpul tertentu
void sisipTengah1 (simpul &l, char elemen1, char elemen2)
{
   simpul bantu, baru;
   baru = (simpul) malloc(sizeof(simpul));
   baru->isi = elemen1;
   baru->next = NULL;
   if(l == NULL)
      cout << "List kosong....." << endl;
   else
   {
      bantu = l;
      while(bantu -> isi != elemen2) bantu = bantu -> next;
      baru -> next = bantu -> next;
      bantu -> next = baru;
   }

}

//fungsi sisip simpul sebelum simpul tertentu
void sisipTengah2 (simpul &l, char elemen1, char elemen2)
{
   simpul bantu, baru;
   baru = (simpul) malloc(sizeof(simpul));
   baru -> isi = elemen1;
   baru -> next = NULL;
   if(l == NULL)
      cout << "list kosong....." << endl;
   else
   {
      bantu = l;
      while(bantu -> isi != elemen2) bantu = bantu -> next;
      baru -> next = bantu -> next;
      bantu -> next = baru;
   }
}

//fungsi simpul di belakang
void sisipBelakang(simpul &l, char elemen)
{
   simpul bantu, baru;
   baru = (simpul) malloc(sizeof(simpul));
   baru -> isi = elemen;
   baru -> next = NULL;
   if(l == NULL)
      l = baru;
   else
   {
      bantu = l;
      while(bantu -> next != NULL) bantu = bantu -> next;
      bantu -> next = baru;
   }
}

//fungis mencetak isi liked list
void cetak(simpul l)
{
   simpul bantu;
   if(l == NULL)
      cout << "Linked list kosong....." << endl;
```

```
    else
    {
       bantu = l;
       cout << "isi linked list : ";
       while(bantu -> next != NULL)
       {
          cout << bantu -> isi << "-->";
          bantu = bantu -> next;
       }
       cout << bantu -> isi;
    }
}

//funsi hapus simpul depan
void hapusDepan(simpul &l)
{
    simpul hapus;
    if(l == NULL)
       cout << "linked list kosong....." << endl;
    else
    {
       hapus = l;
       l = l -> next;
       hapus -> next == NULL;
       free(hapus);
    }
}

//fungsi hapus simpul belakang
void hapusBelakang(simpul &l)
{
  simpul bantu, hapus;
  if (l == NULL) {
    cout << "linked list kosong..... " << endl;
  } else {
    bantu = l;
    while(bantu -> next -> next != NULL) bantu = bantu -> next;
    hapus = bantu -> next;
    bantu -> next = NULL;
    free(hapus);
  }
}

void hapusTengah(simpul &l, char elemen)
{
    simpul bantu, hapus;
    if (l == NULL) {
      cout << "Linked list kosong.....";
    } else {
      bantu = l;
      while(bantu -> next -> isi != elemen) bantu = bantu -> next;
      hapus = bantu -> next;
      bantu -> next = bantu -> next -> next;
      hapus -> next = NULL;
      free(hapus);
    }
}
```

```
Nama    : Nova Ardiansyah
NIM     : 211011401309
========================

OPERASI PADA SINGLE LINKED LIST

Menu :
1. sisip depan
2. sisip belakang
3. sisip tengah1
4. sisip tengah2
5. hapus depan
6. hapus belakang
7. hapus tengah
8. cetak
9. keluar
Pilih menu : 1

## Masukan huruf : a

Menu :
1. sisip depan
2. sisip belakang
3. sisip tengah1
4. sisip tengah2
5. hapus depan
6. hapus belakang
7. hapus tengah
8. cetak
9. keluar
Pilih menu : 8

isi linked list : a
```

2. KESIMPULAN

Linked list adalah strukur data linier berbentuk rantai simpul di mana setiap simpul menyimpan 2 item, yaitu nilai data dan pointer ke simpul elemen berikutnya. Berbeda dengan array, elemen linked list tidak ditempatkan dalam alamat memori yang berdekatan melainkan elemen ditautkan menggunakan pointer.