**LAPORAN PRAKTIKUM STRUKTUR DATA**

**LAPORAN AWAL PRAKTIKUM**

**Pertemuan ke-10**

**Linked List (Lanjut)**

****

**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

1. **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.

1. **TUGAS PENDAHULUAN**
2. Jelaskan Fungsi antara Singly Linked List, Doubly dan Linked List!

**Jawab :**

Sinkly Linked List menggunakan setiap simpul dibagi menjadi 2 bagian yaitu bagian isi dan bagian pointer. Doubly Liked List menggunakan setiap simpul dibagi menjadi 3 bagian, yaitu bagian isi, bagian pointer kiri ddan bagian pointer kanan. Sedangkan Circular Linked List menggunakan linked list yang tidak memiliki nili nil/NULL untuk medan sambungannya.

1. Jelaskan Operasi-Operasi pada Circular Linked List!

**Jawab :**

Simpul depan Simpul yang disisipkan selalu berada di posisi depan dari Linked List (CL). Misalkan kita memiliki suatu Linked List CL dengan empat simpul di mana masing-masing berisi informasi A, B, C, dan D, dan kita juga memiliki suatu simpul baru yang berisi informasi E yang akan disisipkan di posisi depan dari Linked List CL. Langkah-langkah penyisipan simpul baru dapat dilakukan Simpul belakang Penyisipan belakang sebenarnya hampir sama dengan penyisipan depan. Yang membedakan hanyalah pemindahan CL ke simpul yang ditunjuk oleh baru untuk penyisipan belakang tidak ada.

Walaupun dalam penyisipan depan juga tidak harus dilakukan pemindahan CL, karena dalam Circular semua simpul berhak menjadi depan. Langkah-langkah penyisipan simpul baru dapat dilakukan sebagai berikut: Simpul tengah Dalam melakukan penyisipan simpul tengah, maka Linked List harus dijamin tidak boleh kosong. Penyisipan tengah dapat dilakukan pada sebelum atau setelah simpul tertentu. Yang akan dilakukan di sini adalah hanya penyisipan simpul setelah simpul tertentu. Penghapusan Simpul penghapusan simpul adalah operasi penghapusan simpul dari Linked List.

Jika kita akan melakukan penghapusan simpul maka pastikan bahwa Linked List tidak boleh kosong. Penghapusan pada Circular Singly Linked List juga dapat dilakukan pada simpul yang berada di posisi depan, posisi tengah, maupun yang berada pada posisi belakang..

1. Jelaskan Operasi-Operasi pada Doubly Linked List!

**Jawab :**

* **Insert First** Penyisipan di awal list, sehingga pointer head juga akan berpindah ke elemen baru.
* **Insert Last** Penyisipan di akhir list, sehingga pointer tail juga akan berpindah ke elemen baru
* **Insert After / Before** Penyisipan after/before kurang lebih sama satu sama lain. Pada kasus diatas berlaku juga insert before 3
* **Delete First** Penghapusan di awal list, pointer head akan berpindah ke node selanjutnya, sementara node awal akan di dealokasi.
* **Delete Last** Penghapusan di akhir list, pointer tail akan berpindah ke node sebelumnya,sementara node akhir akan di dealokasi.
* **Delete Node** Penghapusan node dengan data tertentu, pada kasus diatas yaitu delete node 2.

1. Jelaskan Jenis Program Yang menggunakan Linked List!

**Jawab :**

Linked list dapat dibagi ke dalam 4 jenis, yakni: Singly linked list, Doubly linked list, Circular linked list, dan Circular doubly linked list.

1. **TUGAS PRAKTIKUM**

* Lat11\_1

|  |
| --- |
| #include <iostream>  #include <conio.h>  #include <stdlib.h>  #define true 1  #define false 0  using namespace std;  typedef struct node \*simpul;  struct node  {  char Isi;  simpul kanan;  simpul kiri;  };  void Sisip\_Depan (simpul &DL, char elemen);  void Sisip\_Belakang (simpul &DL, char elemen);  void Sisip\_Tengah1 (simpul &DL, char elemen1, char elemen2);  void Sisip\_Tengah2 (simpul &DL, char elemen1, char elemen2);  void Hapus\_Depan (simpul &DL);  void Hapus\_Belakang (simpul &DL);  void Hapus\_Tengah (simpul &DL, char elemen);  void Cetak (simpul DL);  int main()  {  cout << "Nama \t: Nova Ardiansyah\n";  cout << "NIM \t: 211011401309\n";  cout << "=========================\n\n";  char huruf, huruf2;  simpul DL = NULL;  int i;  cout << "Operasi Pada Double Linked List\n\n";  cout << "Penyisipan Simpul Di Depan\n\n";  for (i = 1; i <= 4; i++) {  cout << "Masukkan huruf : ";  cin >> huruf;  Sisip\_Depan(DL, huruf);  }  Cetak(DL);  cout << "\n\nPenyisipan Simpul Di Belakang\n\n";  for (i = 1; i <= 4; i++) {  cout << "Masukkan huruf : ";  cin >> huruf;  Sisip\_Belakang(DL, huruf);  }  Cetak(DL);  // \* Sisip simpul setelah simpul  cout << "\n\nPenyisipan Simpul Setelah Simpul Tertentu\n\n";  cout << "Masukkan Huruf : ";  cin >> huruf;  cout << "Sisipkan Setelah Huruf : ";  cin >> huruf2;  cout << huruf << "Disisipkan setalah " << huruf2 << endl;  Sisip\_Tengah1(DL, huruf, huruf2);  Cetak(DL);  // \* Sisip simpul sebelum simpul  cout << "\n\nPenyisipan Simpul Sebelum Simpul Tertentu\n\n";  cout << "Masukkan Huruf : ";  cin >> huruf;  cout << "Sisipkan Sebelum Huruf : ";  cin >> huruf2;  cout << huruf << "Disisipkan Sebelum " << huruf2 << endl;  Sisip\_Tengah2(DL, huruf, huruf2);  Cetak(DL);  // \* Hapus simpul belakang  cout << "Setelah Hapus Simpul Belakang\n\n";  Hapus\_Belakang(DL);  Cetak(DL);  // \* Hapus simpul tengah  cout << "\n\nMasukkan huruf tengah yang akan dihapus : ";  cin>>huruf;  Hapus\_Tengah(DL, huruf);  Cetak(DL);  return 0;  }  void Sisip\_Depan (simpul &DL, char elemen)  {  simpul baru;  baru = (simpul) malloc (sizeof(simpul));  baru->Isi = elemen;  baru->kanan = NULL;  baru->kiri = NULL;  if (DL == NULL) {  DL = baru;  } else {  baru->kanan = DL;  DL->kiri = baru;  DL = baru;  }  }  void Sisip\_Tengah1(simpul &DL, char elemen1, char elemen2)  {  simpul bantu, baru;  baru = (simpul) malloc(sizeof(node));  baru->Isi = elemen1;  baru->kanan = NULL;  baru->kiri = NULL;  if (DL == NULL) {  cout << "List Kosong.........\n";  } else {  bantu = DL;  while (bantu->Isi != elemen2) {  bantu = bantu->kanan;  }  baru->kanan = bantu->kanan;  baru->kiri = bantu;  bantu->kanan->kiri = baru;  bantu->kanan = baru;  }  }  void Sisip\_Tengah2(simpul &DL, char elemen1, char elemen2)  {  simpul bantu, baru;  baru = (simpul) malloc(sizeof(node));  baru->Isi = elemen1;  baru->kanan = NULL;  baru->kiri = NULL;  if (DL == NULL) {  cout << "List Kosong.........\n";  } else {  bantu = DL;  while (bantu->kanan->Isi != elemen2) {  bantu = bantu->kanan;  }  baru->kanan = bantu->kanan;  baru->kiri = bantu;  bantu->kanan->kiri = baru;  bantu->kanan = baru;  }  }  void Sisip\_Belakang(simpul &DL, char elemen)  {  simpul bantu, baru;  baru = (simpul) malloc(sizeof(simpul));  baru->Isi = elemen;  baru->kanan = NULL;  baru->kiri = NULL;  if (DL == NULL) {  DL = baru;  } else {  bantu = DL;  while (bantu->kanan != NULL) {  bantu = bantu->kanan;  }  bantu->kanan = baru;  baru->kiri = bantu;  }  }  void Cetak(simpul DL)  {  simpul bantu;  if (DL == NULL) {  cout << "List Kosong.........\n";  } else {  bantu = DL;  cout << "Isi linked list : ";  while (bantu->kanan != NULL) {  cout << bantu->Isi << "<= =>";  bantu = bantu->kanan;  }  cout << bantu->Isi;  }  }  void Hapus\_Depan(simpul &DL)  {  simpul hapus;  if (DL == NULL) {  cout << "List Kosong.........\n";  } else {  hapus = DL;  DL = DL->kanan;  DL->kiri = NULL;  hapus->kanan = NULL;  free(hapus);  }  }  void Hapus\_Belakang(simpul &DL)  {  simpul bantu, hapus;    if (DL == NULL) {  cout << "List Kosong.........\n";  } else {  bantu = DL;  while (bantu->kanan->kanan != NULL) bantu = bantu->kanan;  hapus = bantu->kanan;  bantu->kanan = NULL;  hapus->kiri = NULL;  free(hapus);  }  }  void Hapus\_Tengah(simpul &DL, char elemen)  {  simpul bantu, hapus;    if (DL == NULL) {  cout << "List Kosong.........\n";  } else {  bantu = DL;  while (bantu->kanan->Isi != elemen)  bantu = bantu->kanan;  hapus = bantu->kanan;  bantu->kanan->kanan->kiri = bantu;  bantu->kanan = bantu->kanan->kanan;  hapus->kanan = NULL;  hapus->kiri = NULL;  free(hapus);  }  } |
|  |

**LAPORAN PRAKTIKUM STRUKTUR DATA**

**LAPORAN AKHIR PRAKTIKUM**

**Pertemuan ke-10**

**Linked List (Lanjut)**

****

**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

1. **TUGAS AKHIR**
2. Modifikasi program diatas dengan ditambahkan linked List!

**Jawab :**

|  |
| --- |
| #include <iostream>  #include <conio.h>  #include <stdlib.h>  #define true 1  #define false 0  using namespace std;  typedef struct node \*simpul;  struct node  {  char Isi;  simpul kanan;  simpul kiri;  };  void Sisip\_Depan (simpul &DL, char elemen);  void Sisip\_Belakang (simpul &DL, char elemen);  void Sisip\_Tengah1 (simpul &DL, char elemen1, char elemen2);  void Sisip\_Tengah2 (simpul &DL, char elemen1, char elemen2);  void Hapus\_Depan (simpul &DL);  void Hapus\_Belakang (simpul &DL);  void Hapus\_Tengah (simpul &DL, char elemen);  void Cetak (simpul DL);  int main()  {  cout << "Nama \t: Nova Ardiansyah\n";  cout << "NIM \t: 211011401309\n";  cout << "=========================\n\n";  char huruf, huruf2;  simpul DL = NULL;  int i;  cout << "Operasi Pada Double Linked List\n\n";  cout << "Penyisipan Simpul Di Depan\n\n";  for (i = 1; i <= 4; i++) {  cout << "Masukkan huruf : ";  cin >> huruf;  Sisip\_Depan(DL, huruf);  }  Cetak(DL);  cout << "\n\nPenyisipan Simpul Di Belakang\n\n";  for (i = 1; i <= 4; i++) {  cout << "Masukkan huruf : ";  cin >> huruf;  Sisip\_Belakang(DL, huruf);  }  Cetak(DL);  // \* Sisip simpul setelah simpul  cout << "\n\nPenyisipan Simpul Setelah Simpul Tertentu\n\n";  cout << "Masukkan Huruf : ";  cin >> huruf;  cout << "Sisipkan Setelah Huruf : ";  cin >> huruf2;  cout << huruf << " Disisipkan setelah " << huruf2 << endl;  Sisip\_Tengah1(DL, huruf, huruf2);  Cetak(DL);  // \* Sisip simpul sebelum simpul  cout << "\n\nPenyisipan Simpul Sebelum Simpul Tertentu\n\n";  cout << "Masukkan Huruf : ";  cin >> huruf;  cout << "Sisipkan Sebelum Huruf : ";  cin >> huruf2;  cout << huruf << " Disisipkan Sebelum " << huruf2 << endl;  Sisip\_Tengah2(DL, huruf, huruf2);  Cetak(DL);  // \* Hapus simpul belakang  cout << "Setelah Hapus Simpul Belakang\n\n";  Hapus\_Belakang(DL);  Cetak(DL);  // \* Hapus simpul tengah  cout << "\n\nMasukkan huruf tengah yang akan dihapus : ";  cin >> huruf;  Hapus\_Tengah(DL, huruf);  Cetak(DL);  return 0;  }  void Sisip\_Depan (simpul &DL, char elemen)  {  simpul baru;  baru = (simpul) malloc (sizeof(node));  baru->Isi = elemen;  baru->kanan = NULL;  baru->kiri = NULL;  if (DL == NULL) {  DL = baru;  } else {  baru->kanan = DL;  DL->kiri = baru;  DL = baru;  }  }  void Sisip\_Tengah1(simpul &DL, char elemen1, char elemen2)  {  simpul bantu, baru;  baru = (simpul) malloc(sizeof(node));  baru->Isi = elemen1;  baru->kanan = NULL;  baru->kiri = NULL;  if (DL == NULL) {  cout << "List Kosong.........\n";  } else {  bantu = DL;  while (bantu->Isi != elemen2) {  bantu = bantu->kanan;  }  baru->kanan = bantu->kanan;  if (bantu->kanan != NULL) {  bantu->kanan->kiri = baru;  }  baru->kiri = bantu;  bantu->kanan = baru;  }  }  void Sisip\_Tengah2(simpul &DL, char elemen1, char elemen2)  {  simpul bantu, baru;  baru = (simpul) malloc(sizeof(node));  baru->Isi = elemen1;  baru->kanan = NULL;  baru->kiri = NULL;  if (DL == NULL) {  cout << "List Kosong.........\n";  } else {  bantu = DL;  while (bantu->kanan != NULL && bantu->kanan->Isi != elemen2) {  bantu = bantu->kanan;  }  if (bantu->kanan != NULL) {  baru->kanan = bantu->kanan;  baru->kiri = bantu;  bantu->kanan->kiri = baru;  bantu->kanan = baru;  } else {  cout << "Simpul " << elemen2 << " tidak ditemukan\n";  }  }  }  void Sisip\_Belakang(simpul &DL, char elemen)  {  simpul bantu, baru;  baru = (simpul) malloc(sizeof(node));  baru->Isi = elemen;  baru->kanan = NULL;  baru->kiri = NULL;  if (DL == NULL) {  DL = baru;  } else {  bantu = DL;  while (bantu->kanan != NULL) {  bantu = bantu->kanan;  }  bantu->kanan = baru;  baru->kiri = bantu;  }  }  void Cetak(simpul DL)  {  simpul bantu;  if (DL == NULL) {  cout << "List Kosong.........\n";  } else {  bantu = DL;  cout << "Isi linked list : ";  while (bantu->kanan != NULL) {  cout << bantu->Isi << "<= =>";  bantu = bantu->kanan;  }  cout << bantu->Isi;  }  }  void Hapus\_Depan(simpul &DL)  {  simpul hapus;  if (DL == NULL) {  cout << "List Kosong.........\n";  } else {  hapus = DL;  DL = DL->kanan;  if (DL != NULL) {  DL->kiri = NULL;  }  hapus->kanan = NULL;  free(hapus);  }  }  void Hapus\_Belakang(simpul &DL)  {  simpul bantu, hapus;    if (DL == NULL) {  cout << "List Kosong.........\n";  } else {  bantu = DL;  while (bantu->kanan->kanan != NULL) {  bantu = bantu->kanan;  }  hapus = bantu->kanan;  bantu->kanan = NULL;  hapus->kiri = NULL;  free(hapus);  }  }  void Hapus\_Tengah(simpul &DL, char elemen)  {  simpul bantu, hapus;    if (DL == NULL) {  cout << "List Kosong.........\n";  } else {  bantu = DL;  while (bantu->kanan != NULL && bantu->kanan->Isi != elemen) {  bantu = bantu->kanan;  }  if (bantu->kanan != NULL) {  hapus = bantu->kanan;  bantu->kanan->kanan->kiri = bantu;  bantu->kanan = bantu->kanan->kanan;  hapus->kanan = NULL;  hapus->kiri = NULL;  free(hapus);  } else {  cout << "Simpul " << elemen << " tidak ditemukan\n";  }  }  } |
|  |

1. **KESIMPULAN**

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.