# UDT Double Linked List

## Waktu Pelaksanaan Praktikum

Durasi kegiatan praktikum **= 170 menit,** dengan rincian sebagai berikut:

1. 15 menit untuk pengerjaan Tes Awal atau wawancara Tugas Pendahuluan
2. 60 menit untuk penyampaian materi
3. 45 menit untuk pengerjaan tugas / Studi Kasus
4. 50 menit **Pengayaan**

## Tujuan

Setelah mengikuti praktikum ini mahasiswa diharapkan dapat:

1. Mampu memahami konsep double linked list
2. Mampu mengaplikasikan double linked list

## Alat & Bahan

1. Komputer
2. Dev C++ IDE

## Dasar Teori

Double Linked List merupakan representasi data yang disimpan dalan suatu rangkaian node dimana setiap node mempunyai penunjuk ke node sebelumnya dan sesudahnya. Double : artinya field pointer-nya dua buah dan dua arah, yang menunjuk ke node sebelum dan sesudahnya. Berguna bila perlu melakukan pembacaan linkedlist dari dua arah. Double linked list memiliki 2 buah pointer yaitu pointer next dan prev. Pointer next : mengarah ke node belakang (tail). Pointer prev : mengarah ke node depan (head). Ilstrasi double Linked List dapat digambarkan berikut :

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| pHead |  |  | Node |  |  |  | Node |  |  |  | Node |  |  | pTail |
|  |  |  | Data |  |  |  | Data |  |  |  | Data |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |

Ketika masih ada satu node maka kedua pointer (next dan prev) akan menunjuk ke NULL. Ketika double linked list memiliki banyak node maka node yang paling depan pointer prev-nya menunjuk ke NULL. Sedangkan node yang paling belakang pointer next-nya yang menunjuk ke NULL. Pada double linked dibutuhkan pointer bantu yaitu head dan tail. Head : menunjuk pada node yang paling depan. Tail : menunjuk pada node yang paling belakang. Dari ilustrasi gambar di atas ADT double Linked-list dapat direpresentasikan sebagai berikut :

|  |
| --- |
| struct node {  **//bagian data**  tipedata data 1;  tipedata data 2;  …  tipedata data n;  **//pointer ke node selanjutnya**  struct node \*next;  struct node \*prev;  };  typedef struct node node; |

## Prosedur Praktikum

1. **Percobaan Pertama: Memahami *Node* yang digunakan pada *Double Linked List***
   * + 1. Buatlah file percobaan\_1.cpp di IDE Dev C++ dan tuliskan kode di bawah :

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27 | #include <cstdlib>  #include <iostream>  #include <conio.h>  using namespace std;  struct node {  //bagian data  char data;  //pointer ke node selanjutnya  struct node \*next;  struct node \*prev;  };  typedef struct node node;  node \*head, \*tail;  int main()  {            getch();  return EXIT\_SUCCESS;  } |

* + - 1. Lakukan beberapa langkah berikut untuk memahami konsep Node pada Double Linked List:
  1. Tambahkan kode head = (node \*) malloc(sizeof(node)); pada baris ke 19.
  2. Tambahkan kode cout << "Data:" << head->data <<endl; pada baris ke 22, kemudian jalankan program.
  3. Tambahkan kode cout << "Pointer next:" << head->next <<endl; pada baris ke 23, dan cout << "Pointer prev:" << head->prev <<endl; pada baris ke 24 kemudian jalankan program.
  4. Tambahkan kode head->data='A'; pada baris ke 20, kemudian jalankan program.
  5. Tambahkan kode head->next=head->prev=NULL; pada baris ke 21, kemudian jalankan program.
     + 1. Tuliskan hasil percobaan, analisis hasil dan kesimpulannya.

1. **Percobaan Kedua: Memahami Operasi sederhana pada Double Linked List**
2. Buatlah file *percobaan\_2.cpp* di IDE Dev C++ dan Tuliskan kode di bawah :

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43 | #include <cstdlib>  #include <iostream>  #include <conio.h>  using namespace std;  struct node {  //bagian data  char data;  //pointer ke node selanjutnya  struct node \*next;  struct node \*prev;  };  typedef struct node node;  node \*head, \*tail;  void AddLast(char item){ // Add Node After Tail  struct node \*pNew;  pNew = (node \*) malloc(sizeof(node));  pNew->data = item;  if (head == NULL){  head= pNew;  }  else {  tail->next = pNew;  pNew->prev=tail;  }  tail=pNew;  }  void AddFirst(char item){ // AddNodeBeforeHead  struct node \*pNew;  pNew = (node \*) malloc(sizeof(node));  pNew->data = item;  if (head == NULL){  tail= pNew;  }  else {  pNew->next = head;  head->prev=pNew;  }  head=pNew;  } |

1. Tambahkan method main pada file *percobaan\_2.cpp* dengan dengan kode sebagai berikut kemudian jalankan.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19 | int main()  {  head=tail=NULL;  AddFirst('A');  cout << "head : " << head->data <<endl;  cout << "tail : " << tail->data <<endl;  AddFirst('B');  cout << "head : " << head->data <<endl;  cout << "tail : " << tail->data <<endl;  AddFirst ('C');  cout << "head : " << head->data <<endl;  cout << "tail : " << tail->data <<endl;  AddFirst ('D');  cout << "head : " << head->data <<endl;  cout << "tail : " << tail->data <<endl;  getch();  return EXIT\_SUCCESS;  } |

1. Ganti isi dari method main pada file *percobaan\_2.cpp* dengan kode sebagai berikut kemudian jalankan.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19 | int main()  {  head=tail=NULL;  AddLast ('A');  cout << "head : " << head->data <<endl;  cout << "tail : " << tail->data <<endl;  AddLast ('B');  cout << "head : " << head->data <<endl;  cout << "tail : " << tail->data <<endl;  AddLast ('C');  cout << "head : " << head->data <<endl;  cout << "tail : " << tail->data <<endl;  AddLast ('D');  cout << "head : " << head->data <<endl;  cout << "tail : " << tail->data <<endl;  getch();  return EXIT\_SUCCESS;  } |

1. Tuliskan hasil percobaan, analisis hasil dan kesimpulannya!

## Hasil Percobaan

1. Tuliskan hasil dari percobaan pertama di atas!
2. Tuliskan hasil dari percobaan kedua di atas!

## Analisis Hasil

1. Tuliskan Analisis hasil dari percobaan pertama di atas!
2. Tuliskan Analisis hasil dari percobaan kedua di atas!

## Kesimpulan

1. Tuliskan kesimpulan dari percobaan pertama di atas!
2. Tuliskan kesimpulan dari percobaan kedua di atas!

## Latihan

Operasi pada Double Linked List secara lengkap adalah sebagai berikut:

1. Penambahan( awal, akhir, indeks)
2. Penghapusan (awal, akhir, indeks, setelah data x, sebelum data x, hapus semua)
3. Penyisipan (setelah data x, sebelum data x)
4. Pencarian (data a berada pada indeks x)
5. Pengaksesan (mulai dari head, tail, indeks tertentu hingga akhir, data tertentu hingga akhir)

Lengkapi kode pada file *percobaan\_2.cpp* diatas untuk operasi-operasi (method) yang belum ada pada Double Linked List!

## Tugas

1. Modifikasilah program pada file *percobaan\_2.cpp* di atas sehingga dapat menampung data struct Mahasiswa sebagai berikut!

|  |
| --- |
| **Mahasiswa** |
| **String nim**  **String nama**  **double ipk** |

1. Gunakan dan modifikasi method penyisipan data pada tugas di modul 3 yang bisa membentuk linked list urut sejak awal dengan pengurutan berdasarkan ipk!
2. Lengkapi method pada file *percobaan\_2.cpp* yang bisa menampilkan data secara *ascending* dan *descending* berdasarkan ipk!

**DAFTAR PUSTAKA**

|  |
| --- |
| * Goodrich, M.T., Tamassia, R., dan Mount, D., 2003, “Data Structures and Algorithms in C++ 2nd edition”, John Wiley & Sons. * Sahni, S., 2005, “Data Structures, Algorithms and Applications in C++ 2nd edition”, Universities Press. * Sanjaya, D., 2005, “Asyiknya Belajar Struktur Data di Planet C++”, PT Elex Media Komputindo. |