Ketentuan Tugas Pendahuluan

- Untuk soal teori **JAWABAN DIKETIK DENGAN RAPI** dan untuk soal algoritma **SERTAKAN SCREENSHOOT CODINGAN DAN HASIL OUTPUT.**
- Deadline pengumpulan TP Modul 6 adalah **Senin, 21 Oktober 2024 pukul 06.00 WIB.**
- **TIDAK ADA TOLERANSI KETERLAMBATAN, TERLAMBAT ATAU TIDAK MENGUMPULKAN TP ONLINE MAKA DIANGGAP TIDAK MENGERJAKAN.**
- **DILARANG PLAGIAT (PLAGIAT = E).**
- Kerjakan TP dengan jelas agar dapat dimengerti.
- Untuk setiap soal nama fungsi atau prosedur **WAJIB** menyertakan **NIM**, contoh: **insertFirst_130122xxxx**.
- File diupload di LMS menggunakan format **PDF** dengan ketentuan **:** **TP_MODX_NIM_KELAS.pdf**

```
Contoh:
    int searchNode_130122xxxx (List L, int X);
```

CP:

- Raihan (089638482851)
- Kayyisa (085105303555)
- Abiya (082127180662)
- Rio (081210978384)

SELAMAT MENGERJAKAN^^

SOAL TP
Soal 1: Menambahkan Elemen di Awal dan Akhir DLL

Deskripsi Soal:
Buatlah program yang mengizinkan pengguna menambahkan elemen ke dalam Doubly Linked List di awal dan di akhir list.

Instruksi:
1. Implementasikan fungsi `insertFirst` untuk menambahkan elemen di awal list.
2. Implementasikan fungsi `insertLast` untuk menambahkan elemen di akhir list.
3. Tampilkan seluruh elemen dalam list dari depan ke belakang setelah penambahan dilakukan.

Contoh Input:
- Input: Masukkan elemen pertama = 10
- Input: Masukkan elemen kedua di awal = 5
- Input: Masukkan elemen ketiga di akhir = 20

Output:
- DAFTAR ANGGOTA LIST: 5 <-> 10 <-> 20

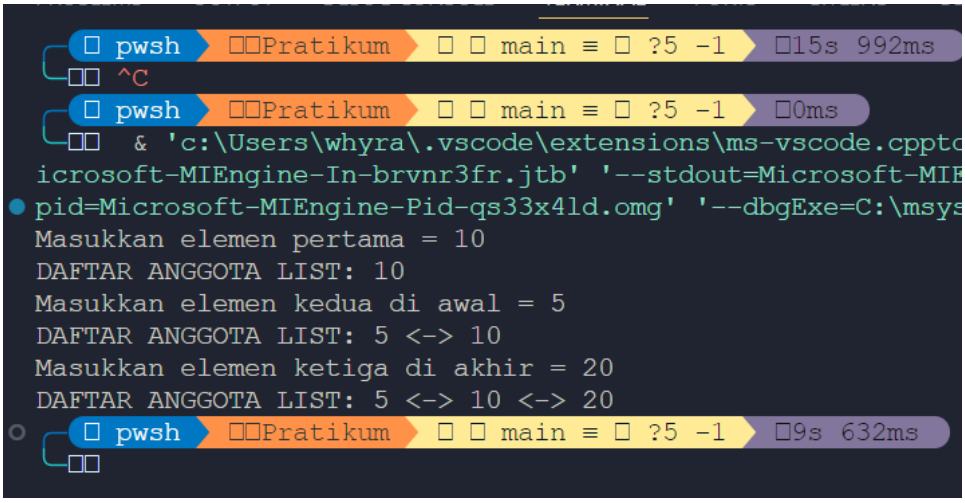
Kode Program:

Double_Linked_List_Bagian_1 > TP > C++ soal_1.cpp > ...

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

// Struktur node untuk Doubly Linked List
struct Node {
    int data;
    Node* next;
    Node* prev;
};

// Fungsi untuk membuat node baru
Node* createNode(int data) {
    Node* newNode = new Node();
    newNode->data = data;
    newNode->next = nullptr;
    newNode->prev = nullptr;
    return newNode;
}

// Fungsi untuk menambahkan elemen di awal list
void insertFirst(Node*& head, int data) {
    Node* newNode = createNode(data);
    if (head == nullptr) {
        head = newNode;
    } else {
        newNode->next = head;
        head->prev = newNode;
        head = newNode;
    }
}

// Fungsi untuk menambahkan elemen di akhir list
void insertLast(Node*& head, int data) {
    Node* newNode = createNode(data);
    if (head == nullptr) {
        head = newNode;
    } else {
        Node* temp = head;
        while (temp->next != nullptr) {
            temp = temp->next;
        }
        temp->next = newNode;
        newNode->prev = temp;
    }
}
```

```cpp
// Fungsi untuk menampilkan elemen dari depan ke belakang
void displayList(Node* head) {
    Node* temp = head;
    cout << "DAFTAR ANGGOTA LIST: ";
    while (temp != nullptr) {
        cout << temp->data;
        if (temp->next != nullptr) cout << " <-> ";
        temp = temp->next;
    }
    cout << endl;
}

int main() {
    Node* head = nullptr;
    int data;

    // Masukkan elemen pertama
    cout << "Masukkan elemen pertama = ";
    cin >> data;
    insertLast(head, data);
    displayList(head);

    //  Masukkan elemen di awal
    cout << "Masukkan elemen kedua di awal = ";
    cin >> data;
    insertFirst(head, data);
    displayList(head);

    // Masukkan elemen di akhir
    cout << "Masukkan elemen ketiga di akhir = ";
    cin >> data;
    insertLast(head, data);
    displayList(head);

    return 0;
}
```

Output Program:



Soal 2: Menghapus Elemen di Awal dan Akhir DLL

Deskripsi Soal:
Buatlah program yang memungkinkan pengguna untuk menghapus elemen pertama dan elemen terakhir dalam Doubly Linked List.

Instruksi:

1. Implementasikan fungsi `deleteFirst` untuk menghapus elemen pertama.
2. Implementasikan fungsi `deleteLast` untuk menghapus elemen terakhir.
3. Tampilkan seluruh elemen dalam list setelah penghapusan dilakukan.

Contoh Input:

- Input: Masukkan elemen pertama = 10
- Input: Masukkan elemen kedua di akhir = 15
- Input: Masukkan elemen ketiga di akhir = 20
- Hapus elemen pertama dan terakhir.

Output:

- DAFTAR ANGGOTA LIST SETELAH PENGHAPUSAN: 15

Kode Program:

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

// Struktur node untuk Doubly Linked List
struct Node {
    int data;
    Node* next;
    Node* prev;
};

// Fungsi untuk membuat node baru
Node* createNode(int data) {
    Node* newNode = new Node();
    newNode->data = data;
    newNode->next = nullptr;
    newNode->prev = nullptr;
    return newNode;
}

// Fungsi untuk menambahkan elemen di akhir list
void insertLast(Node*& head, int data) {
    Node* newNode = createNode(data);
    if (head == nullptr) {
        head = newNode;
    } else {
        Node* temp = head;
        while (temp->next != nullptr) {
            temp = temp->next;
        }
        temp->next = newNode;
        newNode->prev = temp;
    }
}

// Fungsi untuk menghapus elemen pertama
void deleteFirst(Node*& head) {
    if (head == nullptr) {
        cout << "List kosong, tidak ada elemen yang bisa dihapus.\n";
        return;
    }
    Node* temp = head;
    if (head->next == nullptr) {  // Jika hanya ada satu elemen
        head = nullptr;
    } else {
        head = head->next;
        head->prev = nullptr;
    }
    delete temp;
}

// Fungsi untuk menghapus elemen terakhir
void deleteLast(Node*& head) {
    if (head == nullptr) {
        cout << "List kosong, tidak ada elemen yang bisa dihapus.\n";
        return;
    }
    Node* temp = head;
    if (head->next == nullptr) {  // Jika hanya ada satu elemen
        head = nullptr;
    } else {
        while (temp->next != nullptr) {
            temp = temp->next;
        }
        temp->prev->next = nullptr;
    }
    delete temp;
}
```

```cpp
// Fungsi untuk menampilkan elemen dari depan ke belakang
void displayList(Node* head) {
    Node* temp = head;
    cout << "DAFTAR ANGGOTA LIST: ";
    if (temp == nullptr) {
        cout << "List kosong";
    } else {
        while (temp != nullptr) {
            cout << temp->data;
            if (temp->next != nullptr) cout << " <-> ";
            temp = temp->next;
        }
    }
    cout << endl;
}

int main() {
    Node* head = nullptr;
    int data;

    // Contoh input pertama: Masukkan elemen pertama
    cout << "Masukkan elemen pertama = ";
    cin >> data;
    insertLast(head, data);
    displayList(head);

    // Contoh input kedua: Masukkan elemen di akhir
    cout << "Masukkan elemen kedua di akhir = ";
    cin >> data;
    insertLast(head, data);
    displayList(head);

    // Contoh input ketiga: Masukkan elemen di akhir
    cout << "Masukkan elemen ketiga di akhir = ";
    cin >> data;
    insertLast(head, data);
    displayList(head);

    // Hapus elemen pertama
    cout << "Menghapus elemen pertama...\n";
    deleteFirst(head);
    displayList(head);

    // Hapus elemen terakhir
    cout << "Menghapus elemen terakhir...\n";
    deleteLast(head);
    displayList(head);

    return 0;
}
```

Output Program:

```
 pwsh  Pratikum    main ≡  ?5 -1  17s 456ms
  ^C
 pwsh  Pratikum    main ≡  ?5 -1  0ms
  & 'c:\Users\whyra\.vscode\extensions\ms-vscode.cpptools-1.22.10-wi
icrosoft-MIEngine-In-44xl5l0b.qkx' '--stdout=Microsoft-MIEngine-Out-brw3
pid=Microsoft-MIEngine-Pid-ld3e5bm4.ill' '--dbgExe=C:\msys64\ucrt64\bin\
Masukkan elemen pertama = 10
DAFTAR ANGGOTA LIST: 10
Masukkan elemen kedua di akhir = 15
DAFTAR ANGGOTA LIST: 10 <-> 15
Masukkan elemen ketiga di akhir = 20
DAFTAR ANGGOTA LIST: 10 <-> 15 <-> 20
Menghapus elemen pertama...
DAFTAR ANGGOTA LIST: 15 <-> 20
Menghapus elemen terakhir...
DAFTAR ANGGOTA LIST: 15
 pwsh  Pratikum    main ≡  ?5 -1  4s 998ms
```

Soal 3: Menampilkan Elemen dari Depan ke Belakang dan Sebaliknya

Deskripsi Soal: Buatlah program yang memungkinkan pengguna memasukkan beberapa elemen ke dalam Doubly Linked List. Setelah elemen dimasukkan, tampilkan seluruh elemen dalam list dari depan ke belakang, kemudian dari belakang ke depan.

Instruksi:

1. Implementasikan fungsi untuk menampilkan elemen dari depan ke belakang.
2. Implementasikan fungsi untuk menampilkan elemen dari belakang ke depan.
3. Tambahkan 4 elemen ke dalam list dan tampilkan elemen tersebut dalam dua arah.

Contoh Input:

- Input: Masukkan 4 elemen secara berurutan: 1, 2, 3, 4

Output:

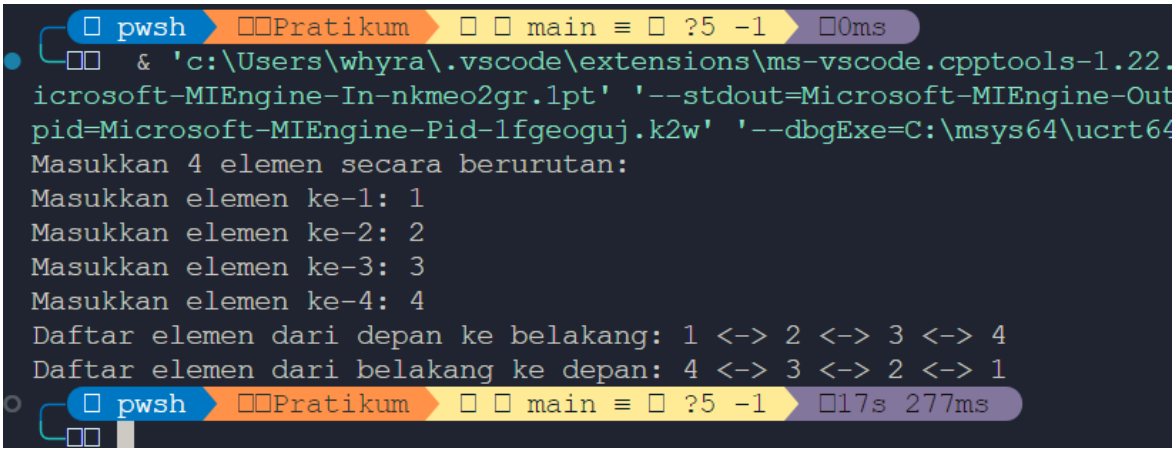
- Daftar elemen dari depan ke belakang: 1 <-> 2 <-> 3 <-> 4
- Daftar elemen dari belakang ke depan: 4 <-> 3 <-> 2 <-> 1

Kode Program:

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

// Struktur node untuk Doubly Linked List
struct Node {
    int data;
    Node* next;
    Node* prev;
};

// Fungsi untuk membuat node baru
Node* createNode(int data) {
    Node* newNode = new Node();
    newNode->data = data;
    newNode->next = nullptr;
    newNode->prev = nullptr;
    return newNode;
}

// Fungsi untuk menambahkan elemen di akhir list
void insertLast(Node*& head, int data) {
    Node* newNode = createNode(data);
    if (head == nullptr) {
        head = newNode;
    } else {
        Node* temp = head;
        while (temp->next != nullptr) {
            temp = temp->next;
        }
        temp->next = newNode;
        newNode->prev = temp;
    }
}

// Fungsi untuk menampilkan elemen dari depan ke belakang
void displayForward(Node* head) {
    Node* temp = head;
    cout << "Daftar elemen dari depan ke belakang: ";
    while (temp != nullptr) {
        cout << temp->data;
        if (temp->next != nullptr) cout << " <-> ";
        temp = temp->next;
    }
    cout << endl;
}
```

```cpp
// Fungsi untuk menampilkan elemen dari belakang ke depan
void displayBackward(Node* head) {
    if (head == nullptr) return;

    Node* temp = head;
    // Bergerak ke node terakhir
    while (temp->next != nullptr) {
        temp = temp->next;
    }

    cout << "Daftar elemen dari belakang ke depan: ";
    while (temp != nullptr) {
        cout << temp->data;
        if (temp->prev != nullptr) cout << " <-> ";
        temp = temp->prev;
    }
    cout << endl;
}

int main() {
    Node* head = nullptr;
    int data;

    // Masukkan 4 elemen secara berurutan
    cout << "Masukkan 4 elemen secara berurutan:\n";
    for (int i = 0; i < 4; i++) {
        cout << "Masukkan elemen ke-" << i + 1 << ": ";
        cin >> data;
        insertLast(head, data);
    }

    // Tampilkan elemen dari depan ke belakang
    displayForward(head);

    // Tampilkan elemen dari belakang ke depan
    displayBackward(head);

    return 0;
}
```

Output Program:


pwsh
Pratikum
main ≡ ?5 -1
0ms
& 'c:\Users\whyra\.vscode\extensions\ms-vscode.cpptools-1.22.
icrosoft-MIEngine-In-nkmeo2gr.1pt' '--stdout=Microsoft-MIEngine-Out
pid=Microsoft-MIEngine-Pid-1fgeoguj.k2w' '--dbgExe=C:\msys64\ucrt64
Masukkan 4 elemen secara berurutan:
Masukkan elemen ke-1: 1
Masukkan elemen ke-2: 2
Masukkan elemen ke-3: 3
Masukkan elemen ke-4: 4
Daftar elemen dari depan ke belakang: 1 <-> 2 <-> 3 <-> 4
Daftar elemen dari belakang ke depan: 4 <-> 3 <-> 2 <-> 1
pwsh
Pratikum
main ≡ ?5 -1
17s 277ms