PRAKTIKUM STRUKTUR DATA

TUGAS PENDAHULUAN 4

Nama :

Aulia Ahmad Ghaus Adzam (2311104028)

Dosen :

Yudha Islami Sulistya, S.
Kom.,M.Kom.

PROGRAM STUDI S1 REKAYASA PERANGKAT LUNAK

FAKULTAS INFORMATIKA

TELKOM UNIVERSITY PURWOKERTO

2024

B. Soal Tugas Pendahuluan

1. Membuat Deklarasi Tipe List:

```
#ifndef LIST_H
#define LIST_H
#include <iostream>
using namespace std;
#define Nil NULL
typedef int infotype;
typedef struct element *address;

struct element {
    infotype info;
    address next;
};

struct List {
    address first;
};
```

Buat File list.cpp dan ketik sintak berikut:

```
#include <iostream>
#include "list.h"
using namespace std;
```

2. Membuat List Kosong, yaitu procedure createList

Tambahkan pada file list.h:

```
void createList(List &L);
```

Tambahkan pada list.cpp implementasi dari procedure createList:

```
void createList(List &L) {
    L.first = Nil;
}
```

3. **Setelah list sudah ada, selanjutnya buatlah elemen dengan menggunakan fungsi allocate.**

   Tambahkan pada list.h primitif dari fungsi allocate:

```
address allocate(infotype x);
```

   Tambahkan pada list.cpp implementasi dari fungsi allocate, sintak c++ seperti ini:

```
address allocate(infotype x) {
    address P = new element;
    P->info = x;
    P->next = Nil;
    return P;
}
```

4. **Setelah List dan elemen sudah ada, maka selanjutnya elemen tersebut harus diinsert ke List agar bisa menjadi elemen list.** Proses insert dapat menggunakan procedure Insert First, procedure Insert Last, atau procedure insert After. Pada Tugas Pendahuluan kali ini, akan dicontohkan menggunakan insert first. Tambahkan pada list.h primitif procedure insertFirst:

```
void insertFirst(List &L, address P);
```

   Tambahkan pada list.cpp implementasi dari procedure insertFirst sesuai sintak berikut :

```
void insertFirst(List &L, address P) {
    P->next = L.first;
    L.first = P;
}
```

5. **Setelah proses insert elemen, maka agar bisa mengetahui apakah elemen berhasil diinsertkan, maka kita perlu menampilkan isi list.**
   Tambahkan pada list.h primitif procedure printInfo:

```
void printInfo(List L);
```

   Tambahkan pada list.cpp implementasi dari proc printInfo, sintak C++ sebagai berikut :

```
void printInfo(List L) {
    address P = L.first;
    while (P != Nil) {
        cout << P->info << " ";
        P = P->next;
    }
    cout << endl;
}
```

6. **Sekarang, setelah ADT List sudah terisi dengan beberapa fungsi Procedur di atas, maka mari buat sebuah List berisi 3 elemen yang berisi 3 digit nim terakhir Anda di main.cpp Adapun gambaran isi dari main.cpp nya adalah sbb :**

```
#include "list.h"

int main() {
    List L;
    createList(L);

    address P1 = allocate(0);
    address P2 = allocate(2);
    address P3 = allocate(8);

    insertFirst(L, P1);
    insertFirst(L, P2);
    insertFirst(L, P3);

    printInfo(L);

    return 0;
}
```

   Semua fungsi di panggil disini createList(), address allocate(), insertFirst(), printInfo(), lalu sesuai NIM saya yaitu 028:

```
PS D:\Documents\ngoding\New folder> ./linkedlist
8 2 0
```

7. **SESI HAVE FUN. Rekan-rekan dapat mencoba hal di bawah ini agar memudahkan saat praktikum:**

i. Tambahkan procedure insertLast, insertAfter, deleteLast, deleteAfter pada list.h dan list.cpp

list.h

```
// SOAL NO 7
void deleteLast(List &L, address &P);
address searchInfo(List L, infotype x);
void insertLast(List &L, address P);
void insertAfter(address Prec, address P);
void deleteAfter(address Prec, address &P);
```

list.cpp

```
// SOAL NO 7
void insertLast(List &L, address P) {
    if (L.first == Nil) {
        insertFirst(L, P);
    } else {
        address Q = L.first;
        while (Q->next != Nil) {
            Q = Q->next;
        }
        Q->next = P;
    }
}

void insertAfter(address Prec, address P) {
    if (Prec != Nil) {
        P->next = Prec->next;
        Prec->next = P;
    }
}

void deleteLast(List &L, address &P) {
    if (L.first != Nil) {
        if (L.first->next == Nil) {
            P = L.first;
            L.first = Nil;
        } else {
            address Q = L.first;
            while (Q->next->next != Nil) {
                Q = Q->next;
            }
            P = Q->next;
            Q->next = Nil;
        }
    }
}

void deleteAfter(address Prec, address &P) {
    if (Prec != Nil && Prec->next != Nil) {
        P = Prec->next;
        Prec->next = Prec->next->next;
        P->next = Nil;
    }
}

address searchInfo(List L, infotype x) {
    address P = L.first;
    while (P != Nil) {
        if (P->info == x) {
            return P;
        }
        P = P->next;
    }
    return Nil;
}
```

ii. Tambahkan Function searchInfo pada list.h dan list.cpp

list.h

```
address searchInfo(List L, infotype x);
```

List.cpp

```
address searchInfo(List L, infotype x) {
    address P = L.first;
    while (P != Nil) {
        if (P->info == x) {
            return P;
        }
        P = P->next;
    }
    return Nil;
}
```

iii. Ubah main.cpp agar proses insert N data tidak satu persatu, tapi sesuai dengan jumlah digit NIM yaitu 10 data (clue : gunakan looping). Dan NIM yang diinput, saat di show tidak boleh terurut terbalik (clue : gunakan insert Last) Tampilan

main.cpp

```
#include "list.h"

int main() {
    List L;
    createList(L);
    int digit;

    for (int x = 0; x < 10; x++) {
        cout << "Masukkan digit ke-" << (x + 1) << " : ";
        cin >> digit;
        insertLast(L, allocate(digit));
    }
    cout << "HASIL NIM : ";
    printInfo(L);

    return 0;
}
```

Output:

```
Masukkan digit ke-1 : 2
Masukkan digit ke-2 : 3
Masukkan digit ke-3 : 1
Masukkan digit ke-4 : 1
Masukkan digit ke-5 : 1
Masukkan digit ke-6 : 0
Masukkan digit ke-4 : 1
Masukkan digit ke-5 : 1
Masukkan digit ke-6 : 0
Masukkan digit ke-5 : 1
Masukkan digit ke-6 : 0
Masukkan digit ke-6 : 0
Masukkan digit ke-7 : 4
Masukkan digit ke-7 : 4
Masukkan digit ke-8 : 0
Masukkan digit ke-8 : 0
Masukkan digit ke-9 : 2
Masukkan digit ke-10 : 8
HASIL NIM : 2 3 1 1 1 0 4 0 2 8
```

C. Latihan Unguided

1. Membuat Single Linked List Buatlah program C++ untuk membuat sebuah single linked list dengan operasi dasar sebagai berikut:

- Insert Node di Depan: Fungsi untuk menambah node baru di awal linked list.
- Insert Node di Belakang: Fungsi untuk menambah node baru di akhir linked list.
- Cetak Linked List: Fungsi untuk mencetak seluruh isi linked list.

Jawab:

```cpp
#include <iostream>
using namespace std;
struct Node {
    int data;
    Node* next;
};

void insertAtFront(Node** head, int newData) {
    Node* newNode = new Node();
    newNode->data = newData;
    newNode->next = *head;
    *head = newNode;
}

void insertAtEnd(Node** head, int newData) {
    Node* newNode = new Node();
    newNode->data = newData;
    newNode->next = nullptr;

    if (*head == nullptr) {
        *head = newNode;
        return;
    }

    Node* temp = *head;
    while (temp->next != nullptr) {
        temp = temp->next;
    }
    temp->next = newNode;
}

void printList(Node* node) {
    while (node != nullptr) {
        cout << node->data;
        if (node->next != nullptr) {
            cout << " -> ";
        }
        node = node->next;
    }
    cout << endl;
}

int main() {
    Node* head = nullptr;
    insertAtFront(&head, 10);
    insertAtEnd(&head, 20);
    insertAtFront(&head, 5);

    cout << "Linked List: ";
    printList(head);

    return 0;
}
```

Output:

```
Linked List: 5 -> 10 -> 20
PS D:\documents\ngoding\STD_Aulia_Ahmad_Ghaus_Adzam_2311104028\04_Single_Linked_List_Bagian_1>
```

2. Menghapus Node pada Linked List Buatlah program C++ yang dapat menghapus node tertentu dalam single linked list berdasarkan nilai yang diberikan oleh pengguna. Tugas ini mencakup operasi:

- Delete Node dengan Nilai Tertentu: Fungsi untuk menghapus node yang memiliki nilai tertentu.
- Cetak Linked List: Setelah penghapusan, cetak kembali isi linked list.

Jawab:

Kode program sama dengan no 1 cuma ditambahkan void deleteNode kayak gini:

```
void deleteNode(Node** head, int key) {
    Node* temp = *head;
    Node* prev = nullptr;

    if (temp != nullptr && temp->data == key) {
        *head = temp->next;
        delete temp;
        return;
    }

    while (temp != nullptr && temp->data != key) {
        prev = temp;
        temp = temp->next;
    }

    if (temp == nullptr) {
        cout << "Node dengan nilai " << key << " tidak ditemukan." << endl;
        return;
    }
    prev->next = temp->next;
    delete temp;
}
```

```
int main() {
    Node* head = nullptr;
    insertAtFront(&head, 10);
    insertAtEnd(&head, 20);
    insertAtFront(&head, 5);

    cout << "Linked List Sebelum Dihapus: ";
    printList(head);

    deleteNode(&head, 10);
    cout << "Linked List Setelah Dihapus: ";
    printList(head);

    return 0;
}
```

```
Linked List Sebelum Dihapus: 5 -> 10 -> 20
Linked List Setelah Dihapus: 5 -> 20
PS D:\documents\ngoding\STD_Aulia_Ahmad_Ghaus_Adzam_2311104028\04_Single_Linked_List_Bagian_1>
```

3. Mencari dan Menghitung Panjang Linked List

Buatlah program C++ yang dapat melakukan operasi berikut:

- Cari Node dengan Nilai Tertentu: Fungsi untuk mencari apakah sebuah nilai ada di dalam linked list.
- Hitung Panjang Linked List: Fungsi untuk menghitung jumlah node yang ada di dalam linked list.

Jawab:
Kode Program nya masih sama dengan No 1 cuman ditambahkan searchNode bertipe Boolean, sama hitungNodes tipe data integer.

```
bool searchNode(Node* head, int key) {
    Node* current = head;
    while (current != nullptr) {
        if (current->data == key) {
            return true;
        }
        current = current->next;
    }
    return false;
}

int hitungNodes(Node* head) {
    int count = 0;
    Node* current = head;
    while (current != nullptr) {
        count++;
        current = current->next;
    }
    return count;
}
```

```
int main() {
    Node* head = nullptr;
    insertAtFront(&head, 10);
    insertAtEnd(&head, 20);
    insertAtFront(&head, 5);

    int cariValue = 20;
    if (searchNode(head, cariValue)) {
        cout << "Node dengan nilai " << cariValue << " ditemukan." << endl;
    } else {
        cout << "Node dengan nilai " << cariValue << " tidak ditemukan." << endl;
    }

    cout << "Panjang linked list: " << hitungNodes(head) << endl;

    return 0;
}
```

Output:

```
Node dengan nilai 20 ditemukan.
Panjang linked list: 3
PS D:\documents\ngoding\STD_Aulia_Ahmad_Ghaus_Adzam_2311104028\04_Single_Linked_List_Bagian_1>
```