LAPORAN PRAKTIKUM

PERTEMUAN 4

SINGLE LINKED LIST

Nama :

RIFKI TAUFIKURROHMAN (2311104033)

Dosen :

YUDHA ISLAMI SULISTYA, S.Kom., M.Kom.

PROGRAM STUDI S1 REKAYASA PERANGKAT LUNAK

FAKULTAS INFORMATIKA

TELKOM UNIVERSITY PURWOKERTO

2024

Soal TP

1. Soal 1 – 6

   List.cpp :

```cpp
#include <iostream>
#include "list.h"

using namespace std;

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

void printInfo(List L) {
  address p = first(L);
  while (p != NULL) {
    cout << info(p) << ", ";
    p = next(p);
  }
  cout << endl;
}

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

  for (int i = 0; i < 3; ++i) {
    infoType number;
    cout << "Masukkan angka ke-" << (i + 1) << ": ";
    cin >> number;

    address P = allocate(number);
    insertFirst(L, P);

    cout << "List setelah menambahkan angka ke-" << (i + 1) << ": ";
    printInfo(L);
  }
}
```

List.h :

```
#include <iostream>
#define first(L) L.first
#define next(P) P -> next
#define info(P) p -> info

using namespace std;
typedef int infoType;
typedef struct elmlist *address;

struct elmlist {
  infoType info;
  address next;
};

struct List {
  address first;
};

void createList(List &L){
  first(L) = NULL;
}

address allocate(infoType x) {
  address p = new elmlist;
  info(p) = x;
  next(p) = NULL;

  return p;
}
```

Hasil kode :

```
Masukkan angka ke-1: 6
List setelah menambahkan angka ke-1: 6,
Masukkan angka ke-2: 2
List setelah menambahkan angka ke-2: 2, 6,
Masukkan angka ke-3: 3
List setelah menambahkan angka ke-3: 3, 2, 6,
```

2. Soal 7

```
#include "list2.h"
using namespace std;

void createList(List &L) {
    first(L) = NULL;
}

address allocate(infoType x) {
    address p = new elmlist;
    info(p) = x;
    next(p) = NULL;
    return p;
}

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

void insertLast(List &L, address P) {
    if (first(L) == NULL) {
        first(L) = P;
    } else {
        address temp = first(L);
        while (next(temp) != NULL) {
            temp = next(temp);
        }
        next(temp) = P;
    }
}

void insertAfter(List &L, address Prec, address P) {
    if (Prec != NULL) {
        next(P) = next(Prec);
        next(Prec) = P;
    }
}
```

```
void deleteLast(List &L) {
    if (first(L) != NULL) {
        address temp = first(L);
        if (next(temp) == NULL) {
            first(L) = NULL;
            delete temp;
        } else {
            address prev = NULL;
            while (next(temp) != NULL) {
                prev = temp;
                temp = next(temp);
            }
            next(prev) = NULL;
            delete temp;
        }
    }
}

void deleteAfter(List &L, address Prec) {
    if (Prec != NULL && next(Prec) != NULL) {
        address temp = next(Prec);
        next(Prec) = next(temp);
        delete temp;
    }
}

bool searchInfo(List L, infoType x) {
    address temp = first(L);
    while (temp != NULL) {
        if (info(temp) == x) {
            return true;
        }
        temp = next(temp);
    }
    return false;
}
```

```
void printInfo(List L) {
    address p = first(L);
    while (p != NULL) {
        cout << info(p);
        p = next(p);
    }
    cout << endl;
}

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

    int totalDigits = 10;
    infoType nimDigit;

    cout << "Masukkan NIM perdigit:" << endl;

    for (int i = 1; i <= totalDigits; ++i) {
        cout << "Digit " << i << " : ";
        cin >> nimDigit;

        address P = allocate(nimDigit);
        insertLast(L, P);
    }

    cout << "Isi list: ";
    printInfo(L);

    return 0;
}
```

Hasil kode :

```
Masukkan NIM perdigit:
Digit 1 : 2
Digit 2 : 3
Digit 3 : 1
Digit 4 : 1
Digit 5 : 1
Digit 6 : 0
Digit 7 : 4
Digit 8 : 0
Digit 9 : 3
Digit 10 : 3
Isi list: 2311104033
```

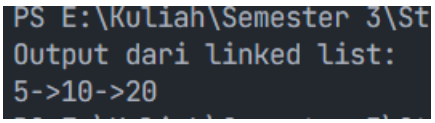
UNGUIDED

1. Latihan 1

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

struct Node {
  int info;
  Node* next;
};

void insertNodeDepan(Node** head, int nilai) {
  Node* newNode = new Node();
  newNode -> info = nilai;
  newNode -> next = *head;
  *head = newNode;
}

void insertNodeBelakang(Node** head, int nilai) {
  Node* newNode = new Node();
  newNode -> info = nilai;
  newNode -> next = nullptr;

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

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

```cpp
void cetakLinkedList(Node* head) {
  Node* temp = head;
  while (temp != nullptr) {
    cout << temp ->info;
    if (temp -> next != nullptr) {
      cout << "->";
    }
    temp = temp -> next;
  }
  cout << endl;
}

int main() {
  Node* head = nullptr;

  insertNodeDepan(&head, 10);
  insertNodeBelakang(&head, 20);
  insertNodeDepan(&head, 5);

  cout << "Output dari linked list: " << endl;
  cetakLinkedList(head);
}
```

Hasil kode :



Penjelasan :

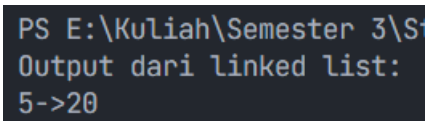
1. Buat struct Node dengan property int info dan Node* next
2. Buat fungsi insertNodeDepan untuk membuat node baru dan menempatkannya pada awal list dengan parameter Node** head dan int nilai.
3. Buat fungsi insertNodeBelakang untuk membuat node baru dan menempatkannya pada posisi akhir list dengan parameter Node** head dan int nilai.
4. Buat fungsi cetakLinkedList untuk mencetak isi dari list yang sudah dimasukkan
5. Fungsi main untuk menjalankan program c++ dan memanggil semua fungsi yang sudah dibuat untuk dijalankan.

2. Latihan 2

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

struct Node {
  int info;
  Node* next;
};

void insertNodeDepan(Node** head, int nilai) {
  Node* newNode = new Node();
  newNode -> info = nilai;
  newNode -> next = *head;
  *head = newNode;
}

void insertNodeBelakang(Node**head, int nilai) {
  Node* newNode = new Node();
  newNode -> info = nilai;
  newNode -> next = nullptr;

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

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

```cpp
void hapusNode(Node** head, int nilai) {
  Node* temp = *head;
  Node* prev = nullptr;

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

  while(temp != nullptr && temp -> info != nilai) {
    prev = temp;
    temp = temp -> next;
  }

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

void cetakLinkedList(Node* head) {
  Node* temp = head;
  while (temp != nullptr) {
    cout << temp ->info;
    if (temp -> next != nullptr) {
      cout << "->";
    }
    temp = temp -> next;
  }
  cout << endl;
}

int main() {
  Node* head = nullptr;

  insertNodeDepan(&head, 10);
  insertNodeBelakang(&head, 20);
  insertNodeDepan(&head, 5);

  hapusNode(&head, 10);
  cout << "Output dari linked list: " << endl;
  cetakLinkedList(head);
}
```

Hasil kode :



Penjelasan kode :

1. Buat struct Node dengan property int info dan Node* next
2. Buat fungsi insertNodeDepan untuk membuat node baru dan menempatkannya pada awal list dengan parameter Node** head dan int nilai.
3. Buat fungsi insertNodeBelakang untuk membuat node baru dan menempatkannya pada posisi akhir list dengan parameter Node** head dan int nilai.
4. Buat fungsi hapusNode untuk menghapus node dari list yang sudah ada berdasarkan head dan nilai yang diberikan.
5. Buat fungsi cetakLinkedList untuk mencetak isi dari list yang sudah dimasukkan
6. Fungsi main untuk menjalankan program c++ dan memanggil semua fungsi yang sudah dibuat untuk dijalankan.

3. Latihan 3

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

struct Node {
  int info;
  Node* next;
};

void insertNodeDepan(Node** head, int nilai) {
  Node* newNode = new Node();
  newNode -> info = nilai;
  newNode -> next = *head;
  *head = newNode;
}

void insertNodeBelakang(Node**head, int nilai) {
  Node* newNode = new Node();
  newNode -> info = nilai;
  newNode -> next = nullptr;

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

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

bool cariNode(Node* head, int nilai) {
  Node* temp = head;
  while(temp != nullptr) {
    if(temp -> info == nilai) {
      return true;
    }
    temp = temp -> next;
  }
  return false;
}
```

```cpp
int main() {
  Node* head = nullptr;

  insertNodeDepan(&head, 10);
  insertNodeBelakang(&head, 20);
  insertNodeDepan(&head, 5);

  cout << "Output : " << endl;
  int cariNilai = 20;
  if (cariNode(head, cariNilai)) {
      cout << "Node dengan nilai " << cariNilai << " ditemukan." << endl;
  } else {
      cout << "Node dengan nilai " << cariNilai << " tidak ditemukan." << endl;
  }

  int panjang = hitungPanjang(head);
  cout << "Panjang linked list: " << panjang << endl;
}
```

Hasil kode :

```
Output :
Node dengan nilai 20 ditemukan.
Panjang linked list: 3
```

Penjelasan Kode :

1. Buat struct Node dengan property int info dan Node* next
2. Buat fungsi insertNodeDepan untuk membuat node baru dan menempatkannya pada awal list dengan parameter Node** head dan int nilai.
3. Buat fungsi insertNodeBelakang untuk membuat node baru dan menempatkannya pada posisi akhir list dengan parameter Node** head dan int nilai.
4. Buat fungsi cariNode untuk mencari node dari list yang sudah ada berdasarkan head dan nilai yang diberikan.
5. Fungsi main untuk menjalankan program c++ dan memanggil semua fungsi yang sudah dibuat untuk dijalankan.