LAPORAN PRAKTIKUM STRUKTUR DATA

MODUL 6
PENGENALAN CODE BLOCKS

Disusun Oleh :
NAMA : Maulana Ananta Piliang
NIM : 103112400156

Dosen
FAHRUDIN MUKTI WIBOWO

PROGRAM STUDI STRUKTUR DATA
FAKULTAS INFORMATIKA
TELKOM UNIVERSITY PURWOKERTO
2025

A. Dasar Teori

C++ adalah pengembangan dari dari bahasa c yang dibuat oleh Bjarne Stroustrup sekitar tahun 1980-an. C++ disebut bahasa multi-paradigma, artinya bisa dipakai dengan gaya prosedural (pakai fungsi biasa), beriorientasi objek (pakai class dan object), atau bahkan gabungan keduanya. C++ punya dasar-dasar seperti variabel, operator percabangan (if, switch), perulangan (for, while), dan bisa memakai class untuk membuat objek.

B. Guided (berisi screenshot source code & output program disertai penjelasannya)

Guided 1

H

```
#ifndef DOUBLYLIST_H
#define DOUBLYLIST_H
#include <iostream>
#include <string> using
namespace std;

struct infotype {
string nopol;
string warna;     int
thnBuat;
};

typedef struct ElmList *address;

struct ElmList {
infotype info;
address next;
address prev;
};

struct List {
```

```
    address First;     address Last;
};

void CreateList(List &L); address alokasi(infotype x); void dealokasi(address P); void
printInfo(List L); void insertLast(List &L, address P); address findElm(List L, string
x); void deleteFirst(List &L, address &P); void deleteLast(List &L, address &P); void
deleteAfter(address Prec, address &P);

#endif
```

Cpp

```
#include "Doublylist.h"


void CreateList(List &L){
L.First = NULL;
    L.Last = NULL;
}


address alokasi(infotype x){
address P = new ElmList;     P-
>info = x;
    P->next = NULL;
```

```
    P->prev = NULL;
    return P;
}

void dealokasi(address P){
    delete P;
}

void printInfo(List L){     address P = L.First;     while(P !=
NULL){         cout << "No Polisi : " << P->info.nopol << endl;
cout << "Warna     : " << P->info.warna << endl;         cout <<
"Tahun     : " << P->info.thnBuat << endl << endl;         P = P-
>next;
    }
}

void insertLast(List &L, address P){
if(L.First == NULL){
        L.First = P;
        L.Last = P;
    } else {
        L.Last->next = P;
        P->prev = L.Last;
        L.Last = P;
    }
}

address findElm(List L, string x){
```

```
   address P = L.First;
while(P != NULL){        if(P-
>info.nopol == x){
return P;
      }
      P = P->next;
   }
   return NULL;
}

void deleteFirst(List &L, address &P){
if(L.First != NULL){        P = L.First;
if(L.First == L.Last){           L.First =
NULL;
         L.Last = NULL;
      } else {
         L.First = L.First->next;
         L.First->prev = NULL;
      }
   }
}

void deleteLast(List &L, address
&P){    if(L.Last != NULL){        P =
L.Last;        if(L.First == L.Last){
L.First = NULL;
         L.Last = NULL;
      } else {
```

```
            L.Last = L.Last->prev;
            L.Last->next = NULL;
        }
    }
}

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

Main

```
#include "Doublylist.h"

int main(){
    List L;
    CreateList(L);
infotype x;     string
cari, hapus;
address P, Q;

    for(int i=0; i<4; i++){         cout <<
"Masukkan nomor polisi: ";
```

```
        cin >> x.nopol;         if(findElm(L, x.nopol) !=
NULL){             cout << "Nomor polisi sudah
terdaftar!\n\n";             continue;
        }
        cout << "Masukkan warna kendaraan:
";         cin >> x.warna;         cout <<
"Masukkan tahun kendaraan: ";         cin >>
x.thnBuat;         P = alokasi(x);
insertLast(L, P);         cout << endl;
    }

    cout << "\nDATA LIST\n";
    printInfo(L);

    cout << "Masukkan Nomor Polisi yang dicari: ";
cin >> cari;
    Q = findElm(L, cari);     if(Q != NULL){         cout <<
"\nNomor Polisi : " << Q->info.nopol;         cout <<
"\nWarna        : " << Q->info.warna;         cout << "\nTahun
: " << Q->info.thnBuat << endl;
    } else {         cout << "Data tidak
ditemukan\n";
    }

    cout << "\nMasukkan Nomor Polisi yang akan dihapus: ";
cin >> hapus;
```

```
    Q = findElm(L, hapus);

    if(Q == L.First){
deleteFirst(L, Q);
dealokasi(Q);
    }
    else if(Q == L.Last){
deleteLast(L, Q);         dealokasi(Q);
    }     else {         address
Prec = Q->prev;
deleteAfter(Prec, Q);
dealokasi(Q);
    }

    cout << "\nData setelah dihapus:\n";
printInfo(L);

    return 0;
}
```

Screenshots Output

No.1

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS
PS C:\Users\ASUS\Documents\Struktur Data\Laporan\modul6> ./app
>>
Masukkan nomor polisi: D001
Masukkan warna kendaraan: hitam
Masukkan tahun kendaraan: 90

Masukkan nomor polisi: D003
Masukkan warna kendaraan: putih
Masukkan tahun kendaraan: 70

Masukkan nomor polisi: D002
Masukkan warna kendaraan: merah
Masukkan tahun kendaraan: 80

Masukkan nomor polisi: D004
Masukkan warna kendaraan: kuning
Masukkan tahun kendaraan: 90


DATA LIST
No Polisi : D001
Warna     : hitam
Tahun     : 90

No Polisi : D003
Warna     : putih
Tahun     : 70

No Polisi : D002
Warna     : merah
Tahun     : 80

No Polisi : D004
Warna     : kuning
Tahun     : 90
```

No.2

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\Users\ASUS\Documents\Struktur Data\Laporan\modul6> ./app
>>
Masukkan Nomor Polisi yang dicari: D001

Nomor Polisi : D001
Warna        : hitam
Tahun        : 90
```

No.3

Deskripsi:

Program ini memanfaatkan struktur data Doubly Linked List untuk menyimpan informasi kendaraan berupa nomor polisi, warna, dan tahun pembuatan. Setiap node memiliki pointer ke node sebelum dan sesudahnya, sehingga proses penambahan data, pencarian, serta penghapusan dapat dilakukan dengan lebih mudah dan efisien.

C. Kesimpulan

Kesimpulan dari keempat program C++ tersebut adalah bahwa masing-masing program memperkenalkan konsep dasar pengelolaan struktur data *singly linked list* menggunakan pointer. Program pertama menunjukkan cara membuat list dan menambahkan elemen di bagian awal. Program kedua melanjutkan dengan menampilkan isi list secara keseluruhan. Program ketiga menambahkan fitur pencarian elemen dalam list, sedangkan program keempat memperluas fungsionalitas dengan menghitung total nilai dari seluruh node. Secara keseluruhan, keempat program ini membantu memahami konsep pointer, alokasi memori dinamis, traversal node, serta operasi dasar pada linked list dalam pemrograman C++.

D. Referensi

- Alhazmi, A.,& Huang, W.(2020). Teaching loops concept through visualization construction. International Journal of Instruction. 13(4),399-114.

- Sáez-López, J. M., Román-González, M., & Vázquez-Cano, E. (2020). Dampak penggunaan bahasa pemrograman terhadap pemahaman konsep loop pada siswa K-12. Pendidikan dan Teknologi Informasi, 25(1), 987-1000.
- Stroustrup, B. (1999). Tinjauan umum bahasa pemrograman C++. ACM SIGPLAN Notices, 34(4), 7-18.
- Stroustrup, B. (1999). Tinjauan umum bahasa pemrograman C++. ACM SIGPLAN Notices, 34(4), 7-18.