LAPORAN PRAKTIKUM
STRUKTUR DATA

MODUL 7
STACK

Disusun Oleh :
NAMA : M.IRFAN ADIB
NIM : 103112400257

Dosen
FAHRUDIN MUKTI WIBOWO

PROGRAM STUDI STRUKTUR DATA
FAKULTAS INFORMATIKA
TELKOM UNIVERSITY PURWOKERTO
2025

A. Dasar Teori

Stack adalah salah satu struktur data linear yang cara kerjanya mengikuti prinsip LIFO (Last In, First Out), yaitu elemen yang terakhir dimasukkan akan menjadi elemen yang pertama dikeluarkan. Stack dapat dianalogikan seperti tumpukan piring: piring yang terakhir diletakkan di atas adalah piring yang pertama kali diambil. Stack memiliki dua operasi utama, yaitu push, yaitu proses menambahkan data ke dalam stack, dan pop, yaitu proses menghapus data dari stack. Selain itu, pada beberapa implementasi juga tersedia operasi seperti peek/top untuk melihat data paling atas tanpa menghapusnya. Stack sering digunakan dalam berbagai proses komputasi, seperti penyimpanan sementara data, proses rekursi, pengurutan ekspresi matematika, hingga sistem *undo-redo* pada aplikasi. Karena keterbatasan ruang penyimpanan, stack juga mengenal kondisi stack overflow ketika data melebihi kapasitas, dan stack underflow ketika dilakukan penghapusan pada stack yang kosong. Dengan karakteristik tersebut, stack menjadi struktur data dasar yang penting dalam pemrograman karena sederhana, efisien, dan memiliki banyak aplikasi dalam pemecahan masalah.

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

Guided 1

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

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

bool isEmpty(Node *top)
{
    return top == nullptr;
}

void push(Node *&top, int data)
{
    Node *newNode = new Node();
    newNode->data = data;
    newNode->next = top;
    top = newNode;
}

int pop(Node *&top)
```

```
{
    if (isEmpty(top))
    {
        cout << "Stack kosong, tidak bisa pop\n";
        return 0;
    }
    Node *temp = top;
    int poppedData = top->data;
    top = top->next;

    delete temp;
    return poppedData;
}

void show(Node *top)
{
    if (isEmpty(top))
    {
        cout << "Stack kosong\n";
        return;
    }
    cout << "Elemen teratas adalah " << top->data << endl;
    Node *temp = top;

    while (temp != nullptr)
    {
        cout << temp->data << " -> ";
        temp = temp->next;
    }
    cout << "NULL" << endl;
}

int main()
{
    Node *stack = nullptr;

    push(stack, 10);
    push(stack, 20);
    push(stack, 30);

    cout << "menampilkan isi stack : " << endl;
    show(stack);
```

```
    cout << "Pop : " << pop(stack) << endl;
    cout << "menampilkan isi stack setelah pop: " << endl;
    show(stack);

    return 0;
}
```

Screenshots Output

```
menampilkan isi stack :
Elemen teratas adalah 30
30 -> 20 -> 10 -> NULL
Pop : 30
menampilkan isi stack setelah pop:
Elemen teratas adalah 20
20 -> 10 -> NULL
```

Deskripsi:
Program ini adalah contoh sederhana dari struktur data stack yang bekerja dengan prinsip *terakhir masuk, pertama keluar* (LIFO). Data disimpan dalam bentuk node yang saling terhubung, dan setiap data baru selalu ditaruh di bagian paling atas menggunakan fungsi push(). Ketika ingin mengambil data, program menggunakan fungsi pop() yang akan menghapus dan mengembalikan elemen teratas dari stack. Program juga menyediakan fungsi show() untuk menampilkan isi stack mulai dari elemen paling atas hingga paling bawah. Pada bagian utama program, beberapa data dimasukkan ke dalam stack, kemudian ditampilkan, lalu satu data dikeluarkan menggunakan pop, dan isi stack ditampilkan kembali untuk melihat perubahannya. Program ini membantu kita memahami bagaimana stack bekerja secara sederhana dan mudah.

C. Unguided/Tugas (berisi screenshot source code & output program disertai penjelasannya)

Unguided 1

stack.h

```
#ifndef STACK_H
#define STACK_H

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

```
const int MAX = 20;
typedef int infotype;

struct Stack {
    infotype info[MAX];
    int top;
};

void createStack(Stack &S);
bool isEmpty(Stack S);
bool isFull(Stack S);
void push(Stack &S, infotype x);
infotype pop(Stack &S);
void printInfo(Stack S);
void balikStack(Stack &S);
void getInputStream(Stack &S);

#endif
```

stack.cpp

```
#include "stack.h"
#include <cctype>

void createStack(Stack &S) {
    S.top = -1;
}

bool isEmpty(Stack S) {
    return (S.top == -1);
}

bool isFull(Stack S) {
    return (S.top == MAX - 1);
}

void push(Stack &S, infotype x) {
    if (!isFull(S)) {
        S.top++;
```

```
        S.info[S.top] = x;
    }
}

infotype pop(Stack &S) {
    if (!isEmpty(S)) {
        int x = S.info[S.top];
        S.top--;
        return x;
    } else {
        return -1;
    }
}

void printInfo(Stack S) {
    if (isEmpty(S)) {
        cout << "[TOP] (kosong)" << endl;
    } else {
        cout << "[TOP] ";
        for (int i = S.top; i >= 0; i--) {
            cout << S.info[i];
        }
        cout << endl;
    }
}

void balikStack(Stack &S) {
    Stack temp;
    createStack(temp);
    while (!isEmpty(S)) {
        push(temp, pop(S));
    }
    S = temp;
}

void getInputStream(Stack &S) {
    char c;
    while (true) {
        c = cin.get();
        if (c == '\n') break;
        if (isdigit(c)) {
            int num = c - '0';
            push(S, num);
```

```
        }
    }
}
```

main.cpp

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

int main() {
    Stack S;

    cout << "Hello world!" << endl;
    createStack(S);

    push(S, 12);
    push(S, 15);
    push(S, 17);
    push(S, 19);

    cout << "4729601" << endl;
    printInfo(S);

    cout << "balik stack" << endl;
    balikStack(S);
    printInfo(S);
    cout << endl;

    cout << "Hello world!" << endl;
    createStack(S);

    push(S, 12);
    push(S, 15);
    push(S, 17);
    push(S, 19);
    push(S, 3);
    push(S, 2);

    cout << "5839247" << endl;
    printInfo(S);
```

```
    cout << "balik stack" << endl;
    balikStack(S);
    printInfo(S);
    cout << endl;

    cout << "Hello world!" << endl;
    createStack(S);

    getInputStream(S);
    printInfo(S);

    cout << "balik stack" << endl;
    balikStack(S);
    printInfo(S);
    return 0;
}
```

Screenshots Output 1

```
Hello world!
4729601
[TOP] 19171512
balik stack
[TOP] 12151719
```

Screenshots Output 2

```
Hello world!
5839247
[TOP] 2319171512
balik stack
[TOP] 1215171932
```

Screenshots Output 3

```
Hello world!
123456
[TOP] 654321
balik stack
[TOP] 123456
```

Deskripsi:

Program ini dibuat untuk mengelola sebuah stack berbasis array, mulai dari membuat stack kosong, mengecek apakah stack penuh atau kosong, menambah data dengan push, menghapus data dengan pop, hingga menampilkan isi stack dari elemen teratas. Program juga menyediakan fungsi untuk membalik urutan elemen stack dengan cara memindahkan semua data ke stack sementara, kemudian mengembalikannya lagi sehingga urutannya terbalik. Selain itu, ada fungsi khusus untuk membaca input berupa karakter satu per satu, lalu hanya menyimpan angka ke dalam stack dengan mengubah karakter angka menjadi bilangan integer. Secara keseluruhan, program ini membantu kita memahami cara kerja stack sebagai struktur data yang menggunakan prinsip LIFO (Last In, First Out) dengan operasi yang sederhana dan mudah dipahami.

D. Kesimpulan

Program stack ini menunjukkan bagaimana struktur data LIFO (Last In, First Out) bekerja dalam C++. Melalui operasi push, pop, dan tampilkan, kita bisa melihat bagaimana data selalu keluar dari elemen yang terakhir masuk. Program juga menambahkan fitur membalik urutan stack serta memasukkan karakter angka sebagai integer, sehingga memberikan gambaran nyata tentang fleksibilitas stack dalam memproses data. Secara keseluruhan, program ini memperkuat pemahaman tentang konsep dasar stack dan penerapannya dalam pemrograman.

E. Referensi

Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C. (2022). *Introduction to Algorithms* (4th ed.). The MIT Press.

Stroustrup, B. (2014). *Programming: Principles and Practice Using C++* (2nd ed.). Addison-Wesley Professional.