LAPORAN PRAKTIKUM
STRUKTUR DATA

MODUL 7
STACK

Disusun Oleh :
NAMA : RAIHAN DZAKY MUFLIH
NIM : 103112430029

Dosen
FAHRUDIN MUKTI WIBOWO

PROGRAM STUDI STRUKTUR DATA
FAKULTAS INFORMATIKA
TELKOM UNIVERSITY PURWOKERTO
2025

A. Dasar Teori

Stack adalah struktur data yang cara kerjanya seperti tumpukan barang, di mana data yang terakhir dimasukkan akan keluar lebih dulu. Operasi utama pada stack ada dua, yaitu push untuk menambahkan data ke atas tumpukan dan pop untuk mengambil data teratas. Selain itu ada juga operasi untuk mengecek apakah stack kosong atau penuh, serta fungsi tambahan seperti membalik urutan isi stack. Stack sering digunakan dalam proses pemrograman seperti pembalikan data, pengecekan tanda kurung, dan sistem pemanggilan fungsi.

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 menunjukkan cara kerja struktur data stack menggunakan pointer. Pertama, program menambahkan beberapa angka ke dalam stack menggunakan fungsi push, lalu menampilkan seluruh isinya dari atas ke bawah. Setelah itu, program mengambil satu data dari stack dengan fungsi pop , kemudian menampilkan kembali isi stack setelah data teratas dihapus. Program juga menampilkan pesan jika stack kosong agar pengguna tahu tidak ada data yang bisa diambil.

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

Unguided 1

stack.h

```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

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

int main() {
    Stack S;


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

    push(S, 9);
    push(S, 2);
    push(S, 4);
    push(S, 3);

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

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

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

    push(S, 9);
    push(S, 8);
    push(S, 4);
    push(S, 3);
    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 (balikStack)

```
PS D:\C++\MODUL 7\UNGUIDED> & 'c:\Users\asus\.vscode\extensions\ms-vscode.cpptools-1.28.3-win32-x64\debugAdapters\bin\WindowsDebugLauncher.exe' '--stdin=Microsoft-MIEngine-In-vppdinvg.ch4' '--stdout=Microsoft-MIEngine-Out-5iuifmnt.nk5' '--stderr=Microsoft-MIEngine-Error-o1s0mogl.ftm' '--pid=Microsoft-MIEngine-Pid-p4pavqc5.5bb' '--dbgExe=D:\C++\mingw64\bin\gdb.exe' '--interpreter=mi'
Hello world!
[TOP] 9 2 4 3
balik stack
[TOP] 3 4 2 9
PS D:\C++\MODUL 7\UNGUIDED>
```

Screenshots Output 2 (pushAscending)

```
Hello world!
5839247
[TOP] 233489
balik stack
[TOP] 984332
```

Screenshots Output 3 (getInputStream)

```
Hello world!
1 0 6 9 2 7 4
[TOP] 4729601
balik stack
[TOP] 1069274
PS D:\C++\MODUL 7\UNGUIDED>
```

Deskripsi:
Program ini menampilkan cara kerja stack dengan tiga percobaan. Pada bagian pertama dan kedua, program langsung menambahkan beberapa angka ke dalam stack dan menampilkan hasilnya sebelum dan sesudah dibalik. Pada bagian ketiga, program meminta pengguna untuk memasukkan deretan angka, lalu menampilkan isi stack tersebut dan hasilnya setelah dibalik. Setiap bagian diawali dengan tulisan “Hello world!” agar hasilnya terlihat terpisah dan mudah dibaca.

D. Kesimpulan

Dari percobaan ini dapat disimpulkan bahwa stack sangat berguna untuk mengatur data dengan urutan masuk dan keluar yang terbalik. Dengan memahami cara kerja push, pop, dan pembalikan stack, kita bisa lebih mudah memproses data yang memerlukan urutan tertentu. Program yang dibuat juga menunjukkan bahwa operasi stack dapat dilakukan baik secara otomatis maupun dengan input dari pengguna.

E. Referensi

Drozdek, A. (2012). *Data Structures and Algorithms in C++*. Cengage Learning.

Malik, D. S. (2010). *Data Structures Using C++*. Course Technology.

GeeksforGeeks. (n.d.). *Stack Data Structure (Introduction and Program)*. Diakses dari https://www.geeksforgeeks.org/stack-data-structure/