

**LAPORAN PRAKTIKUM  
STRUKTUR DATA**

**MODUL 7  
STACK**



**Disusun Oleh :**

NAMA : Muhammad Omar Nadiv

NIM : 103112430063

**Dosen**

Fahrudin Mukti Wibowo, S.Kom., M.Eng.

**PROGRAM STUDI STRUKTUR DATA  
FAKULTAS INFORMATIKA  
TELKOM UNIVERSITY PURWOKERTO  
2025**

## A. Dasar Teori

Stack adalah salah satu struktur data linier yang mengikuti prinsip Last In, First Out (LIFO), di mana elemen yang terakhir dimasukkan akan menjadi yang pertama dikeluarkan. Struktur data ini hanya memungkinkan akses pada elemen yang berada di posisi paling atas (top). Oleh karena itu, operasi yang dilakukan pada stack terbatas pada dua fungsi utama, yaitu push (menambahkan elemen kedalam stack) dan pop (mengambil elemen dari stack).

Keunggulan dari stack adalah kesederhanaannya dalam implementasi dan penggunaannya pada berbagai aplikasi seperti pemrosesan ekspresi matematika, pengelolaan undo/redo dalam aplikasi, serta pengelolaan tugas dalam sistem komputer. Dengan menggunakan struktur data seperti array atau linked list, stack dapat diimplementasikan dengan efisien. Operasi pada stack memiliki kompleksitas waktu yang konstan ( $O(1)$ ), sehingga sangat cepat dalam penambahan dan penghapusan elemen. Struktur data ini membantu memecahkan masalah yang memerlukan pengelolaan data secara terurut dan efisien, dengan hanya mengakses elemen yang paling atas.

## 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!" << endl;
        return 0;
    }

    int poppedData = top->data;
    Node *temp = top;
    top = top->next;

    delete temp;
    return poppedData;
}

void show(Node *top)
{
    if(isEmpty(top))
    {
        cout << "Stack kosong." << endl;
        return;
    }

    cout << "TOP -> ";
    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 sisa stack: " << endl;
    show(stack);

    return 0;
}

```

#### Screenshots Output

```

● nadv@omarnadip Guided % ./stack
Menampilkan isi stack:
TOP -> 30 -> 20 -> 10 -> NULL
Pop: 30
Menampilkan sisa stack:
TOP -> 20 -> 10 -> NULL
○ nadv@omarnadip Guided % █

```

### Deskripsi:

Program ini digunakan untuk mengelola data dalam struktur data Stack menggunakan bahasa C++. Dalam program ini, digunakan struktur data Node yang menyimpan dua informasi penting, yaitu data (bertipe integer) yang menyimpan nilai yang dimasukkan ke dalam stack, dan next yang merupakan pointer yang menunjuk ke elemen berikutnya dalam stack. Fungsi isEmpty(top) digunakan untuk mengecek apakah stack kosong dengan memeriksa apakah top bernilai nullptr. Fungsi push(top, data) digunakan untuk menambahkan elemen ke dalam stack, tepatnya di posisi paling atas (top), sedangkan fungsi pop(top) digunakan untuk menghapus elemen dari posisi paling atas stack, serta mengembalikan nilai data yang telah dihapus, dengan menampilkan pesan error jika stack kosong. Fungsi show(top) digunakan untuk menampilkan semua elemen dalam stack, dimulai dari posisi paling atas hingga paling bawah, dan jika stack kosong, fungsi ini akan menampilkan pesan bahwa stack kosong.

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

#### Unguided 1 (file stack.h)

```
#ifndef STACK_H_INCLUDED
#define STACK_H_INCLUDED

typedef int infotype;
typedef struct
{
    infotype info[20];
    int top;
} Stack;

void createStack(Stack &S);
void push(Stack &S, infotype x);
infotype pop(Stack &S);
void printInfo(Stack S);
void balikStack(Stack &S);

#endif
```

#### Unguided 1 (file stack.cpp)

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

using namespace std;

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

void push(Stack &S, infotype x) {
    if (S.top < 19) {
        S.top++;
        S.info[S.top] = x;
    } else {
        cout << "Stack penuh!" << endl;
    }
}
```

```

    }
}

infotype pop(Stack &S) {
    if (S.top == -1) {
        cout << "Stack kosong!" << endl;
        return -1;
    } else {
        return S.info[S.top--];
    }
}

void printInfo(Stack S) {
    if (S.top == -1) {
        cout << "Stack kosong!" << endl;
    } else {
        cout << "Isi Stack: ";
        for (int i = S.top; i >= 0; i--) {
            cout << S.info[i] << " ";
        }
        cout << endl;
    }
}

void balikStack(Stack &S) {
    Stack temp;
    createStack(temp);
    while (S.top != -1) {
        push(temp, pop(S));
    }
    S = temp;
}

```

## Unguided 1 (file main.cpp)

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

using namespace std;

int main() {
    Stack S;
    createStack(S);

    int choice, value;

    while (true) {
        cout << "\nMenu: " << endl;
        cout << "1. Push (Masukkan elemen ke stack)" << endl;
        cout << "2. Pop (Ambil elemen dari stack)" << endl;
        cout << "3. Tampilkan Stack" << endl;
        cout << "4. Balik Stack" << endl;
        cout << "5. Keluar" << endl;
        cout << "Pilih menu (1-5): ";
        cin >> choice;

        switch (choice) {
            case 1:
                cout << "Masukkan nilai yang ingin dimasukkan ke stack: ";
                cin >> value;
                push(S, value);
                cout << value << " berhasil dimasukkan ke stack." << endl;
                break;
            case 2:
                cout << "Pop: " << pop(S) << endl;
                break;
            case 3:
                printInfo(S);
                break;
            case 4:
                cout << "Balik stack" << endl;
                balikStack(S);
                printInfo(S);
                break;
            case 5:
```



```

        cout << "Keluar dari program." << endl;
        return 0;
    default:
        cout << "Pilihan tidak valid. Coba lagi." << endl;
    }
}

return 0;
}

```

## Screenshots Output

```

● nadiv@omarnadip Unguided % ./stack_program

Menu:
1. Push (Masukkan elemen ke stack)
2. Pop (Ambil elemen dari stack)
3. Tampilkan Stack
4. Balik Stack
5. Keluar
Pilih menu (1-5): 3
Stack kosong!

Menu:
1. Push (Masukkan elemen ke stack)
2. Pop (Ambil elemen dari stack)
3. Tampilkan Stack
4. Balik Stack
5. Keluar
Pilih menu (1-5): 1
Masukkan nilai yang ingin dimasukkan ke stack: 12
12 berhasil dimasukkan ke stack.

Menu:
1. Push (Masukkan elemen ke stack)
2. Pop (Ambil elemen dari stack)
3. Tampilkan Stack
4. Balik Stack
5. Keluar
Pilih menu (1-5): 3
Isi Stack: 12

Menu:
1. Push (Masukkan elemen ke stack)
2. Pop (Ambil elemen dari stack)
3. Tampilkan Stack
4. Balik Stack
5. Keluar
Pilih menu (1-5): 2
Pop: 12

Menu:
1. Push (Masukkan elemen ke stack)
2. Pop (Ambil elemen dari stack)
3. Tampilkan Stack
4. Balik Stack
5. Keluar
Pilih menu (1-5): 5
Keluar dari program.
○ nadiv@omarnadip Unguided % █

```

#### Deskripsi:

Program ini mengimplementasikan struktur data Stack menggunakan Array dalam bahasa C++. Stack adalah struktur data dengan prinsip Last In, First Out (LIFO), di mana elemen yang terakhir dimasukkan akan menjadi elemen pertama yang dikeluarkan. Program ini mencakup operasi dasar seperti push untuk menambahkan elemen ke stack, pop untuk mengeluarkan elemen dari stack, printInfo untuk menampilkan elemen stack, dan balikStack untuk membalik urutan elemen dalam stack. Pengguna dapat berinteraksi dengan program melalui menu untuk memasukkan data, mengeluarkan data, dan menampilkan atau membalik stack. Dengan menggunakan array sebagai representasi stack, program ini memberikan pemahaman tentang pengelolaan data yang berurutan dengan prinsip LIFO.

#### D. Kesimpulan

Pada modul 7 ini, saya belajar tentang penggunaan struktur data Stack dalam pemrograman C++. Saya memahami prinsip dasar stack yang menerapkan konsep Last In, First Out (LIFO), di mana elemen yang terakhir dimasukkan akan menjadi yang pertama dikeluarkan. Saya juga belajar cara mengimplementasikan stack menggunakan array, serta operasi dasar seperti push, pop, printInfo, dan balikStack. Selain itu, saya memperoleh pemahaman tentang bagaimana stack digunakan untuk mengelola data secara efisien dalam aplikasi tertentu. Pembelajaran ini membantu saya memahami konsep dasar struktur data stack yang sederhana namun sangat berguna dalam pengolahan data.

#### E. Referensi

[https://www.w3schools.com/cpp/cpp\\_stacks.asp](https://www.w3schools.com/cpp/cpp_stacks.asp)

<https://medium.com/@RobuRishabh/understanding-how-to-use-stack-queues-c-9f1fc06d1c5e>

<https://www.trivusi.web.id/2022/07/struktur-data-stack.html>