

**LAPORAN PRAKTIKUM STRUKTUR DATA DAN  
PEMROGRAGAMAN**

**MODUL 6  
STACK**



**Disusun oleh :**

**ZEFANYA.B.T.TARIGAN**

**2311102028**

**IF-11-A**

**Dosen Pengampu :**

**Wahyu Andi Saputra, S. Pd., M. Eng**

**PROGRAM STUDI TEKNIK INFORMATIKA  
FAKULTAS INFORMATIKA  
INSTITUT TEKNOLOGI TELKOM PURWOKERTO  
PURWOKERTO**

**2024**

# BAB I

## A. TUJUAN PRAKTIKUM

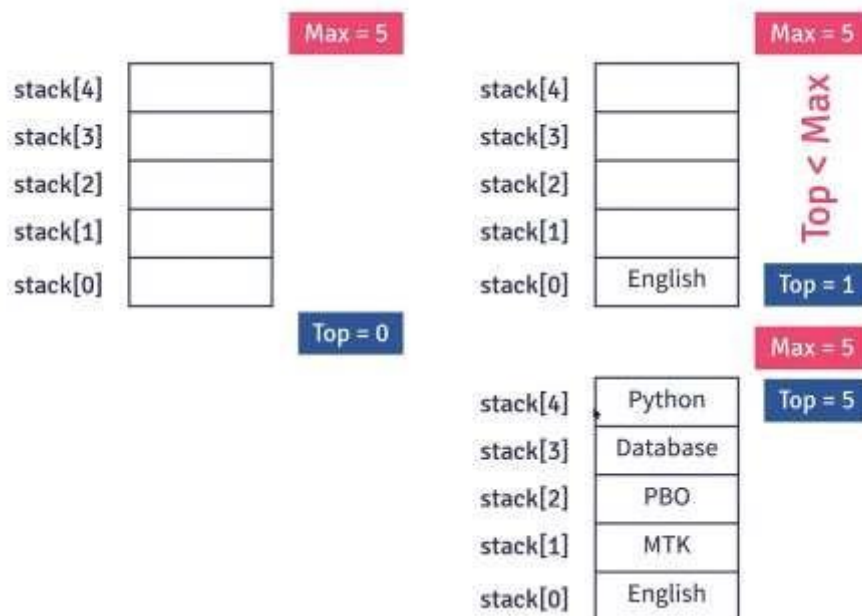
- Mampu memahami konsep stack pada struktur data dan algoritma
- Mampu mengimplementasikan operasi-operasi pada stack
- Mampu memecahkan permasalahan dengan solusi stack

# BAB II

## B. DASAR TEORI

Stack adalah struktur data sederhana yang digunakan untuk menyimpan data (mirip dengan Linked Lists). Dalam tumpukan, urutan kedatangan data penting. Sebuah tumpukan piring di kafetaria adalah contoh bagus dari tumpukan. Piring ditambahkan ke tumpukan saat mereka dibersihkan dan ditempatkan di bagian atas. Ketika sebuah piring dibutuhkan, diambil dari bagian atas tumpukan. Piring pertama yang ditempatkan di tumpukan adalah yang terakhir digunakan.

Definisi: Sebuah tumpukan adalah daftar terurut di mana penyisipan dan penghapusan dilakukan di satu ujung, disebut atas. Elemen terakhir yang dimasukkan adalah yang pertama dihapus. Oleh karena itu, disebut daftar Last in First out (LIFO).



Operasi pada stack melibatkan beberapa fungsi dasar yang dapat dilakukan pada struktur data ini. Berikut adalah beberapa operasi umum pada stack:

- Push (Masukkan):** Menambahkan elemen ke dalam tumpukan pada posisi paling atas atau ujung.
- Pop (Keluarkan):** Menghapus elemen dari posisi paling atas atau ujung tumpukan.

- c. Top (Atas): Mendapatkan nilai atau melihat elemen teratas pada tumpukan tanpa menghapusnya.
- d. IsEmpty (Kosong): Memeriksa apakah tumpukan kosong atau tidak.
- e. IsFull (Penuh): Memeriksa apakah tumpukan penuh atau tidak (terutama pada implementasi tumpukan dengan kapasitas terbatas).
- f. Size (Ukuran): Mengembalikan jumlah elemen yang ada dalam tumpukan.
- g. Peek (Lihat): Melihat nilai atau elemen pada posisi tertentu dalam tumpukan tanpa menghapusnya.
- h. Clear (Hapus Semua): Mengosongkan atau menghapus semua elemen dari tumpukan.
- i. Search (Cari): Mencari keberadaan elemen tertentu dalam tumpukan.

## BAB III

### C. GUIDED

#### Source code

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

string arrayBuku[5];
int maksimal = 5, top = 0;
bool isFull()
{
    return (top == maksimal);
}
bool isEmpty()
{
    return (top == 0);
}
void pushArrayBuku(string data)
{
    if (isFull())
    {
        cout << "Data telah penuh" << endl;
```

```

}

else
{
    arrayBuku[top] = data;
    top++;
}
}

void popArrayBuku()
{
    if (isEmpty())
    {
        cout << "Tidak ada data yang dihapus" << endl;
    }
    else
    {
        arrayBuku[top - 1] = "";
        top--;
    }
}

void peekArrayBuku(int posisi)
{
    if (isEmpty())
    {
        cout << "Tidak ada data yang bisa dilihat" << endl;
    }
    else
    {
        int index = top;
        for (int i = 1; i <= posisi; i++)
        {
            index--;
        }
    }
}

```

```
cout << "Posisi ke " << posisi << " adalah " << arrayBuku[index] << endl;
    }
}

int countStack()
{
    return top;
}

void changeArrayBuku(int posisi, string data)
{
    if (posisi > top)
    {
        cout << "Posisi melebihi data yang ada" << endl;
    }
    else
    {
        int index = top;
        for (int i = 1; i <= posisi; i++)
        {
            index--;
        }
        arrayBuku[index] = data;
    }
}

void destroyArraybuku()
{
    for (int i = top; i >= 0; i--)
    {
        arrayBuku[i] = "";
    }
}
```

```

top = 0;
}
void cetakArrayBuku()
{
    if (isEmpty())
    {
        cout << "Tidak ada data yang dicetak" << endl;
    }
    else
    {
        for (int i = top - 1; i >= 0; i--)
        {
            cout << arrayBuku[i] << endl;
        }
    }
}
int main()
{
    pushArrayBuku("Kalkulus");
    pushArrayBuku("Struktur Data");
    pushArrayBuku("Matematika Diskrit");
    pushArrayBuku("Dasar Multimedia");
    pushArrayBuku("Inggris");
    cetakArrayBuku();
    cout << "\n";

    cout << "Apakah data stack penuh? " << isFull() << endl;
    cout << "Apakah data stack kosong? " << isEmpty() << endl;
    peekArrayBuku(2);
    popArrayBuku();

    cout << "Banyaknya data = " << countStack() << endl;
}

```

```

changeArrayBuku(2, "Bahasa Jerman");

cetakArrayBuku();

cout << "\n";

destroyArraybuku();

cout << "Jumlah data setelah dihapus: " << top << endl;

cetakArrayBuku();

return 0;
}

```

## SCREENSHOOT PROGRAM

```

Apakah data stack penuh? 1
Apakah data stack kosong? 0
Posisi ke 2 adalah Dasar Multimedia
Banyaknya data = 4
Dasar Multimedia
Bahasa Jerman
Struktur Data
Kalkulus

Jumlah data setelah dihapus: 0
Tidak ada data yang dicetak
PS D:\KULIAH\SEMESTER2\struktur data praktikum>

```

## DESKRIPSI PROGRAM

codingan di atas mengimplementasikan sebuah tumpukan (stack) . Dan program tersebut menggunakan

-Fungsi pushArrayBuku(string data):

Menambahkan data ke dalam stack jika belum penuh, dan mengubah top. Jika penuh, mencetak pesan bahwa data telah penuh.

-Fungsi popArrayBuku():

Menghapus elemen teratas dari stack jika tidak kosong, dan mengurangi top. Jika kosong, mencetak pesan bahwa tidak ada data yang dihapus.

-Fungsi countStack():

Mengembalikan jumlah elemen dalam stack (nilai top).

-Fungsi destroyArraybuku():

Menghapus semua elemen dalam stack dan mengatur top menjadi 0,dll.

## **BAB IV**

### **UNGUIDED**

#### **UNGUIDED1**

##### **SOURCE CODE**

```
#include <iostream>
#include <stack>
#include <string>
#include <algorithm>
using namespace std;
bool isPalindrome(string str)
{
    stack<char> charStack;
    string originalStr = str;
    // Menghapus spasi dan tanda baca dari kalimat
    str.erase(remove_if(str.begin(), str.end(), ::isspace), str.end());
    str.erase(remove_if(str.begin(), str.end(), ::ispunct), str.end());
    // Memasukkan setiap karakter ke dalam stack
    for (char c : str)
    {
        charStack.push(c);
    }
    // Membandingkan karakter pada stack dengan karakter asli dari
    belakang
    for (char c : originalStr)
    {

```



```
if (isspace(c) || ispunct(c))
{
    continue; // Lewati spasi dan tanda baca
}
if (c != charStack.top())
{
    return false; // Tidak palindrom
}
charStack.pop();
}
return true; // Palindrom
}
int main()
{
    string kalimat;
    cout << "Masukkan kalimat: ";
    getline(cin, kalimat);
    if (isPalindrome(kalimat))
    {
        cout << "Kalimat merupakan palindrom." << endl;
    }
    else
    {
        cout << "Kalimat bukan palindrom." << endl;
    }
    return 0;
}
```

## SCREENSHOOT PROGRAM

```
PS D:\KULIAH\SEMESTER2\struktur data praktikum> ./Unguided1m6
Masukkan kalimat: ini
Kalimat merupakan palindrom.
PS D:\KULIAH\SEMESTER2\struktur data praktikum> █
```

## DESKRIPSI PROGRAM

Program ini digunakan untuk memeriksa apakah sebuah kalimat merupakan palindrom atau tidak. Sebuah palindrom adalah sebuah kata, frasa, angka, atau rangkaian karakter lainnya yang memiliki sifat dapat dibaca sama baik dari depan maupun dari belakang. Contoh palindrom adalah "katak", "level", dan "radar".

## UNGUIDED 2

## SOURCE CODE

```
#include <iostream>
#include <stack>
#include <string>
using namespace std;
void reverseSentence(string sentence) {
    stack<string> wordStack;
    string word = "";
    // Memisahkan kalimat menjadi kata-kata
    for (char c : sentence) {
        if (c == ' ') {
            wordStack.push(word);
            word = "";
        } else {
            word += c;
        }
    }
    wordStack.push(word);
}
```

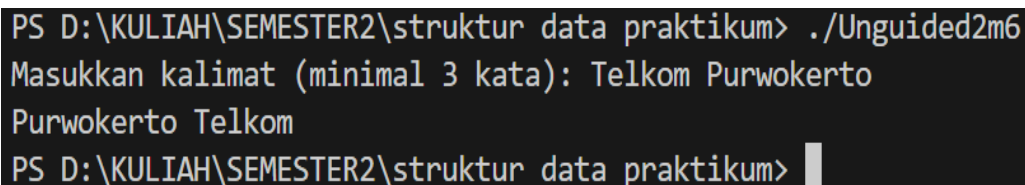
```

// Memasukkan kata terakhir setelah loop
// Membalikkan urutan kata-kata
while (!wordStack.empty()) {
    cout << wordStack.top() << " ";
    wordStack.pop();
}
cout << endl;
}

int main() {
    string kalimat;
    cout << "Masukkan kalimat (minimal 3 kata): ";
    getline(cin, kalimat);
    reverseSentence(kalimat);
    return 0;
}

```

### SCREENSHOOT PROGRAM



```

PS D:\KULIAH\SEMESTER2\struktur data praktikum> ./Unguided2m6
Masukkan kalimat (minimal 3 kata): Telkom Purwokerto
Purwokerto Telkom
PS D:\KULIAH\SEMESTER2\struktur data praktikum>

```

### DESKRIPSI PROGRAM

Program ini digunakan untuk membalik urutan kata-kata dalam sebuah kalimat. Program tersebut menggunakan struktur data stack untuk menyimpan setiap kata dalam kalimat secara terpisah, kemudian mengeluarkan kata-kata tersebut dari stack dalam urutan terbalik untuk mencetak kalimat yang telah dibalik.

## **BAB V**

### **KESIMPULAN**

Stack berfungsi dengan prinsip LIFO yang berguna dalam berbagai aplikasi, seperti undo/redo dalam aplikasi, pemrosesan ekspresi aritmatika, dan pemanggilan fungsi rekursif. Implementasi stack yang baik juga melibatkan pengecekan kondisi overflow (untuk array) dan underflow, serta memastikan operasi aman dan tidak menyebabkan kesalahan memori.

## **BAB VI**

### **DAFTAR PUSTAKA**

Karumanchi, N. (2016). Data Structures and algorithms made easy: Concepts, problems, Interview Questions. CareerMonk Publications.