

# **LAPORAN PRAKTIKUM**

## **MODUL 6**

### **STACK**



**Disusun oleh:**

**Nabila Shasya Sabrina**

**NIM: 2311102039**

**Dosen Pengampu:**

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

**PROGRAM STUDI TEKNIK INFORMATIKA**

**FAKULTAS INFORMATIKA**

**INSTITUT TEKNOLOGI TELKOM PURWOKERTO**

**PURWOKERTO**

**2024**

## **BAB I**

### **TUJUAN PRAKTIKUM**

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

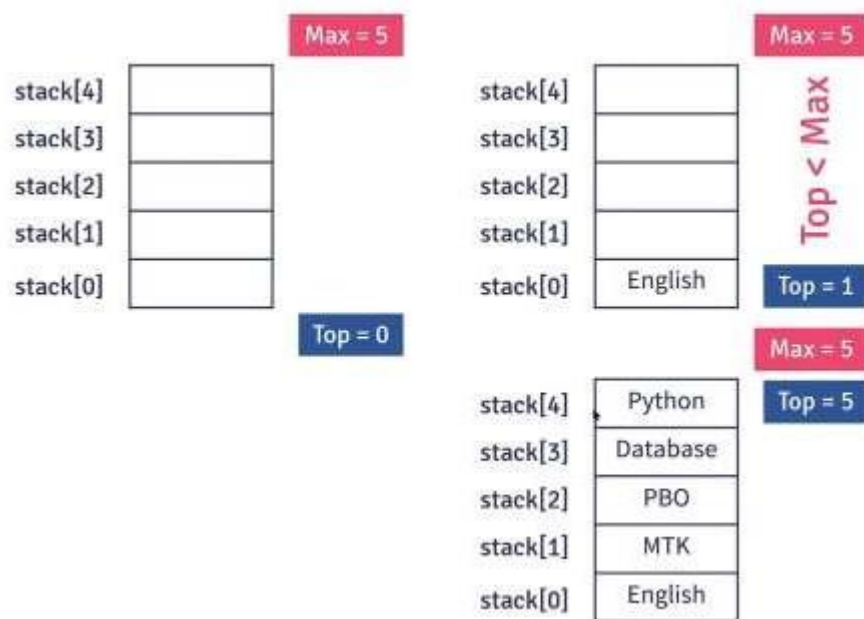
## BAB II

### DASAR TEORI

#### a. Pengertian Stack

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

### GUIDED

#### 1. Guided 1

##### 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;
```

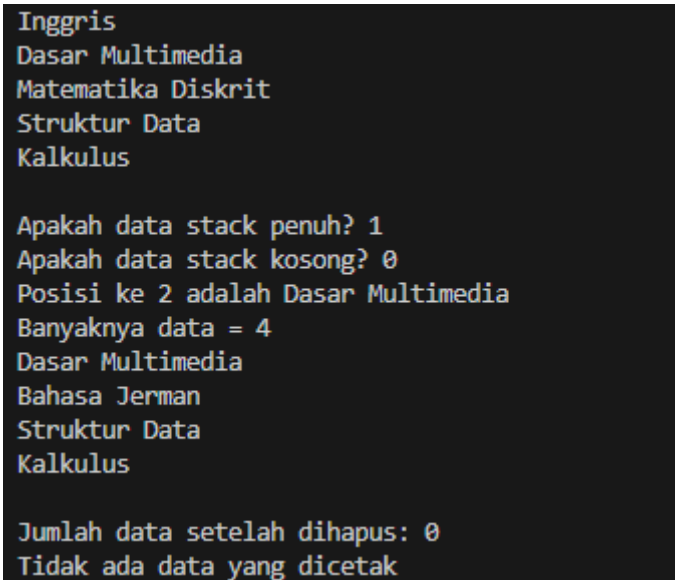
```

    }
}
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:



```
Inggris  
Dasar Multimedia  
Matematika Diskrit  
Struktur Data  
Kalkulus  
  
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
```

### Deskripsi program:

Program di atas adalah implementasi stack menggunakan array untuk menyimpan data buku. Stack diinisialisasi dengan kapasitas maksimum lima buku dan memiliki fungsi-fungsi untuk mengelola stack tersebut. Fungsi `isFull()` dan `isEmpty()` digunakan untuk memeriksa apakah stack penuh atau kosong. Buku dapat ditambahkan ke stack menggunakan `pushArrayBuku()` dan dihapus menggunakan `popArrayBuku()`. Fungsi `peekArrayBuku()` menampilkan buku pada posisi tertentu dari atas stack, sedangkan `countStack()` mengembalikan jumlah buku dalam stack. Buku pada posisi tertentu dapat diubah menggunakan `changeArrayBuku()`, dan semua buku dalam stack dapat dihapus menggunakan `destroyArraybuku()`. Fungsi `cetakArrayBuku()` digunakan untuk mencetak semua buku dalam stack. Pada fungsi `main()`, lima buku ditambahkan ke stack dan berbagai operasi dilakukan untuk menunjukkan fungsi-fungsi yang ada, termasuk mencetak isi stack, mengecek apakah stack penuh atau kosong, menampilkan buku pada posisi tertentu, mengubah buku, dan menghapus semua buku dari stack.

## BAB IV

### UNGUIDED

1. Buatlah program untuk menentukan apakah kalimat tersebut yang diinputkan dalam program stack adalah palindrom/tidak. Palindrom kalimat yang dibaca dari depan dan belakang sama. Jelaskan bagaimana cara kerja programnya.

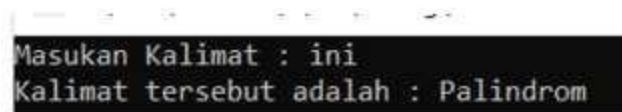
Contoh:

Kalimat : ini

Kalimat tersebut adalah polindrom

Kalimat : telkom

Kalimat tersebut adalah bukan polindrom



```
Masukan Kalimat : ini
Kalimat tersebut adalah : Palindrom
```

**Source code:**

```
#include <iostream>
#include <stack>
#include <string>
#include <algorithm>

using namespace std;

bool apakahPolindrom(string str) {
    string cleanedStr = "";
    for (char c : str) {
        if (isalnum(c)) {
            cleanedStr += tolower(c);
        }
    }

    stack<char> charStack;

    for (char c : cleanedStr) {
        charStack.push(c);
    }

    for (char c : cleanedStr) {
        if (c != charStack.top()) {
            return false;
        }
        charStack.pop();
    }

    return true;
}

int main() {
    string input;
    cout << "Masukkan sebuah kalimat: ";
    getline(cin, input);
```



```

    if (apakahPolindrom(input)) {
        cout << "Kalimat tersebut adalah palindrom." << endl;
    } else {
        cout << "Kalimat tersebut bukan palindrom." << endl;
    }

    return 0;
}

```

### Screenshot Output:

```

Masukkan sebuah kalimat: ini
Kalimat tersebut adalah palindrom.

```

```

Masukkan sebuah kalimat: kasur rusak
Kalimat tersebut adalah palindrom.

```

```

Masukkan sebuah kalimat: telkom
Kalimat tersebut bukan palindrom.

```

```

Masukkan sebuah kalimat: semangat
Kalimat tersebut bukan palindrom.

```

### Deskripsi Program:

Program ini bertujuan untuk menentukan apakah sebuah kalimat adalah palindrom atau tidak. Dalam program ini, kalimat yang dimasukkan oleh pengguna pertama-tama difilter untuk hanya menyisakan karakter alfanumerik dan mengubahnya menjadi huruf kecil. Proses ini dilakukan dalam fungsi `apakahPolindrom`, di mana setiap karakter alfanumerik dari kalimat input ditambahkan ke sebuah string baru, `cleanedStr`. Setelah itu, karakter-karakter dalam `cleanedStr` dimasukkan ke dalam sebuah stack. Program kemudian membandingkan setiap karakter dalam `cleanedStr` dengan karakter yang dikeluarkan dari stack satu per satu. Jika ada ketidaksesuaian antara karakter tersebut, fungsi akan mengembalikan nilai `false`, menunjukkan bahwa kalimat tersebut bukan palindrom. Jika semua karakter sesuai, fungsi akan mengembalikan nilai `true`, menunjukkan bahwa kalimat tersebut adalah palindrom. Program utama meminta pengguna untuk memasukkan sebuah kalimat, memanggil fungsi `apakahPolindrom`, dan mencetak hasilnya apakah kalimat tersebut adalah palindrom atau tidak.

2. Buatlah program untuk melakukan pembalikan terhadap kalimat menggunakan stack dengan minimal 3 kata. Jelaskan output program dan source codenya beserta operasi/fungsi yang dibuat?

Contoh

Kalimat : Telkom Purwokerto

Hasil : otrekowruP mokleT

```

Masukkan Kata Telkom Purwokerto

```

```

Datastack Array :

```

```

Data : otrekowruP mokleT

```

```

#include <iostream>
#include <stack>
#include <string>

using namespace std;

string reverseWords(string sentence) {
    stack<char> charStack;
    string reversedSentence = "", word = "";

    for (char c : sentence) {
        if (c != ' ') {
            charStack.push(c);
        } else {
            while (!charStack.empty()) {
                word += charStack.top();
                charStack.pop();
            }
            reversedSentence += word + " ";
            word = "";
        }
    }

    while (!charStack.empty()) {
        word += charStack.top();
        charStack.pop();
    }
    reversedSentence += word;

    return reversedSentence;
}

int main() {
    string sentence;
    cout << "Masukkan kalimat (minimal 3 kata): ";
    getline(cin, sentence);

    string reversedWords = reverseWords(sentence);

    cout << "Kalimat setelah dibalik kata-katanya: " <<
        reversedWords << endl;

    return 0;
}

```

### Screenshot Output:

```

Masukkan kalimat (minimal 3 kata): Telkom Purwokerto
Kalimat setelah dibalik kata-katanya: mokleT otrekowruP

```

```

Masukkan kalimat (minimal 3 kata): Shasya
Kalimat setelah dibalik kata-katanya: aysahS

```

### Deskripsi Program:

Program ini berfungsi untuk membalikkan setiap kata dalam sebuah kalimat yang dimasukkan oleh pengguna, tanpa mengubah urutan kata-kata tersebut. Program menggunakan fungsi `reverseWords`, di mana setiap karakter dalam kalimat dimasukkan ke dalam sebuah stack hingga ditemukan spasi.

Ketika ditemukan spasi, program mengeluarkan karakter-karakter dari stack dan membentuk kata yang terbalik, lalu menambahkannya ke string hasil `reversedSentence`. Proses ini berlanjut hingga akhir kalimat, memastikan setiap kata dibalik dan ditambahkan ke hasil akhir. Dalam fungsi `main`, program meminta pengguna untuk memasukkan sebuah kalimat, lalu memanggil fungsi `reverseWords` untuk membalikkan setiap kata dalam kalimat tersebut, dan akhirnya mencetak hasilnya ke layar.

## **BAB V**

### **KESIMPULAN**

Stack adalah salah satu struktur data fundamental yang beroperasi berdasarkan prinsip Last In, First Out (LIFO). Artinya, elemen terakhir yang dimasukkan ke dalam stack adalah elemen pertama yang akan dikeluarkan. Struktur data ini terdiri dari dua operasi utama: `push`, untuk menambahkan elemen ke dalam stack, dan `pop`, untuk mengeluarkan elemen dari stack. Stack sering digunakan dalam berbagai aplikasi, seperti pemrograman rekursif, penelusuran pohon, pengelolaan undo-redo pada aplikasi, dan pengecekan keseimbangan tanda kurung dalam ekspresi matematika.

Kesimpulannya, stack adalah alat yang sangat berguna dan fleksibel dalam pemrograman, menyediakan cara sederhana namun efisien untuk menyimpan dan mengelola data sementara. Implementasi stack dapat mempermudah penyelesaian berbagai masalah komputasi, terutama yang melibatkan proses berulang atau pengelolaan data sementara secara bertahap.

## DAFTAR PUSTAKA

Asisten Praktikum. 2024. "*MODUL 5 HASH TABLE*". Learning Management System.

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