

**LAPORAN PRAKTIKUM STRUKTUR  
DATA DAN ALGORITMA**

**MODUL VI  
STACK**



**Disusun Oleh :**

**NAMA : ANISA YASAROH**

**NIM : 2311102063**

**Dosen :**

**WAHYU ANDI SAPUTRA, S.Pd., M.Eng.**

**PROGRAM STUDI S1 TEKNIK INFORMATIKA  
FAKULTAS INFORMATIKA  
INSTITUT TEKNOLOGI TELKOM PURWOKERTO  
2024**

## A. DASAR TEORI

### a. Pengertian Stack

Stack adalah struktur data LIFO (Last-In, First-Out) yang berarti elemen terakhir yang dimasukkan akan menjadi elemen pertama yang diambil. Ini digunakan dalam pemrograman untuk mengelola urutan operasi atau pemanggilan fungsi, di mana setiap operasi baru ditempatkan di atas tumpukan dan operasi terakhir diambil terlebih dahulu. Stack sangat berguna dalam manajemen pemanggilan fungsi dan pelacakan konteks eksekusi program, memungkinkan program untuk kembali ke titik eksekusi sebelumnya dengan lancar. Selain itu, stack digunakan dalam algoritma seperti rekursi, validasi tumpukan, dan konversi notasi, memberikan struktur data yang efisien untuk menangani operasi berurutan.

### b. Operasi-Operasi Dasar pada Stack

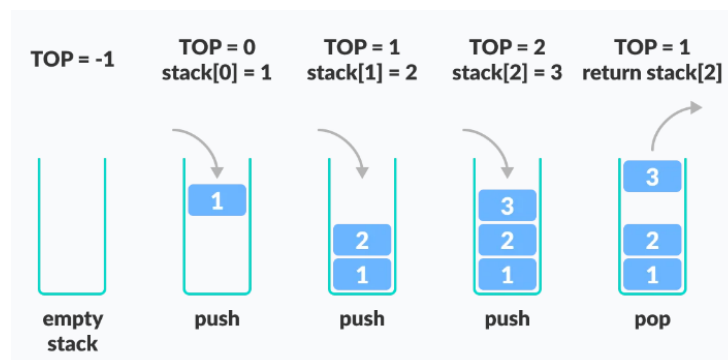
Berikut adalah beberapa operasi dasar yang memungkinkan kita untuk melakukan berbagai tindakan pada stack:

- Push: Menambahkan elemen ke bagian atas stack
- Pop: Menghapus elemen dari bagian atas stack
- IsEmpty: Memeriksa apakah stack kosong
- IsFull: Memeriksa apakah stack penuh
- Peek: Mendapatkan nilai elemen teratas tanpa menghapusnya

### c. Cara Kerja Stack

Operasi-operasi bekerja sebagai berikut:

- a) Sebuah pointer yang disebut TOP digunakan untuk melacak elemen teratas dalam stack.
- b) Saat menginisialisasi stack, kita mengatur nilainya menjadi -1 sehingga kita dapat memeriksa apakah stack kosong dengan membandingkan  $TOP == -1$ .
- c) Saat menambahkan (push) sebuah elemen, kita meningkatkan nilai TOP dan menempatkan elemen baru di posisi yang ditunjuk oleh TOP.
- d) Saat menghapus (pop) sebuah elemen, kita mengembalikan elemen yang ditunjuk oleh TOP dan mengurangi nilainya.
- e) Sebelum menambahkan elemen, kita memeriksa apakah stack sudah penuh.
- f) Sebelum menghapus elemen, kita memeriksa apakah stack sudah kosong.



#### **d. Implementasi Stack**

Stack dapat diimplementasikan menggunakan struktur data seperti array atau linked list.

- **Implementasi Stack dengan Array**  
Dalam implementasi ini, stack diwakili oleh array dengan ukuran tetap. Elemen-elemen stack ditempatkan secara berurutan dalam array, dan indeks terakhir menunjukkan elemen teratas stack. Operasi push dan pop dilakukan dengan menambah dan menghapus elemen pada indeks terakhir array, memberikan akses cepat dan konstan ke elemen teratas. Namun, kelemahannya adalah ukuran stack terbatas pada kapasitas tetap array, sehingga sulit untuk mengubah ukuran stack secara dinamis jika array penuh.
- **Implementasi Stack dengan Linked List**  
Dalam implementasi ini, stack diwakili oleh linked list di mana setiap simpul berisi elemen stack dan tautan ke simpul sebelumnya. Elemen teratas stack berada pada simpul pertama (head). Operasi push dan pop dilakukan dengan menambah atau menghapus simpul di awal linked list. Keuntungan dari implementasi ini adalah fleksibilitas ukuran stack yang dapat berubah secara dinamis, sedangkan kelemahannya adalah memerlukan alokasi memori tambahan untuk tautan antar simpul.

## 1. GUIDED

### 1. Guided 1 – Hash Table

#### 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 - 1; 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;
}
```

## Screenshot Output

```
1mr.oil' '--dbgExe=C:\Program Files\CodeBlocks\mingw\bin\gdb.exe' '--interpreter=mi'
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
PS D:\Struktur Data smt 2>
```

## Deskripsi Program

Program di atas yaitu arrayBuku[5] menyimpan maksimal 5 elemen string, dengan top sebagai penunjuk elemen teratas. Fungsi-fungsinya meliputi isFull() dan isEmpty() untuk memeriksa status stack, pushArrayBuku() untuk menambahkan data, popArrayBuku() untuk menghapus data teratas, peekArrayBuku() untuk melihat elemen pada posisi tertentu, countStack() untuk menghitung jumlah elemen, changeArrayBuku() untuk mengubah elemen pada posisi tertentu, destroyArraybuku() untuk mengosongkan stack, dan cetakArrayBuku() untuk mencetak elemen stack. Dalam main(), program menambahkan lima buku ke stack, mencetak seluruh elemen, memeriksa status penuh atau kosong, menampilkan elemen tertentu, menghapus elemen teratas,

menghitung jumlah elemen, mengubah elemen, dan akhirnya mengosongkan serta mencetak elemen stack yang sudah kosong. Program ini menggambarkan operasi dasar stack (push, pop, peek, isFull, isEmpty) menggunakan array.

## 2. UNGUIDED

- a. 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.

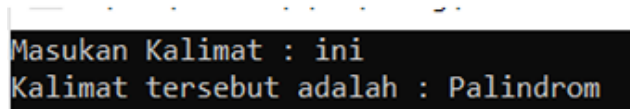
Contoh:

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>
using namespace std;

bool isPalindrome(const string& input) {
    stack<char> charStack;
    int lengthInput = input.length();

    for (int a_63 = 0; a_63 < lengthInput; a_63++) {
        charStack.push(input[a_63]);
    }

    for (int a_63 = 0; a_63 < lengthInput; a_63++) {
        if (input[a_63] != charStack.top()) {
            return false;
        }
        charStack.pop();
    }

    return true;
}

int main() {
    string input;
```

```

char choice;

do {
    cout << "\nMasukkan Kalimat: ";
    cin >> input;

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

    cout << "Lanjutkan? (y/n): ";
    cin >> choice;

} while (choice == 'y' || choice == 'Y');

return 0;
}

```

## Screenshoot Output

```

PS D:\Struktur Data smt 2> & 'c:\Users\ASUS\.vscode\extensions\ms-vscode.cpptools-1.20.5-win32-x64\def
ft-MIEngine-In-hamtq04m.zqd' '--stdout=Microsoft-MIEngine-Out-fdvkgy3s.vms' '--stderr=Microsoft-MIEngi
nri.oxs' '--dbgExe=C:\Program Files\CodeBlocks\MingW\bin\gdb.exe' '--interpreter=mi'

Masukkan Kalimat: ini
Kalimat ini adalah palindrom
Lanjutkan? (y/n): y

Masukkan Kalimat: telkom
Kalimat telkom bukan palindrom
Lanjutkan? (y/n): y

Masukkan Kalimat: malam
Kalimat malam adalah palindrom
Lanjutkan? (y/n): y

Masukkan Kalimat: menara
Kalimat menara bukan palindrom
Lanjutkan? (y/n): n
PS D:\Struktur Data smt 2>

```

Notepad++ window content:

```

Nama : ANISA YASAROH
NIM : 2311102063

```

Notepad++ status bar: Ln 2, Col 18 | 38 characters | 100% | Window | UTF-8

## Deskripsi Program

Program di atas yaitu memeriksa apakah sebuah string (kalimat) adalah palindrom. Pertama, pengguna diminta untuk memasukkan sebuah kalimat. Program kemudian menggunakan fungsi `isPalindrome` untuk memeriksa apakah kalimat tersebut adalah palindrom dengan memasukkan setiap karakter ke dalam sebuah stack dan kemudian membandingkannya satu per satu dengan karakter asli dalam urutan terbalik. Jika semua karakter cocok, kalimat tersebut adalah palindrom. Hasilnya



ditampilkan di konsol. Setelah itu, pengguna ditanya apakah ingin melanjutkan (memeriksa kalimat lain) dengan memasukkan 'y' atau 'Y'. Jika pengguna memilih 'y' atau 'Y', program akan mengulang proses tersebut, dan akan berhenti jika pengguna memasukkan 'n' atau 'N'.

- b. 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
```

### Source Code

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

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

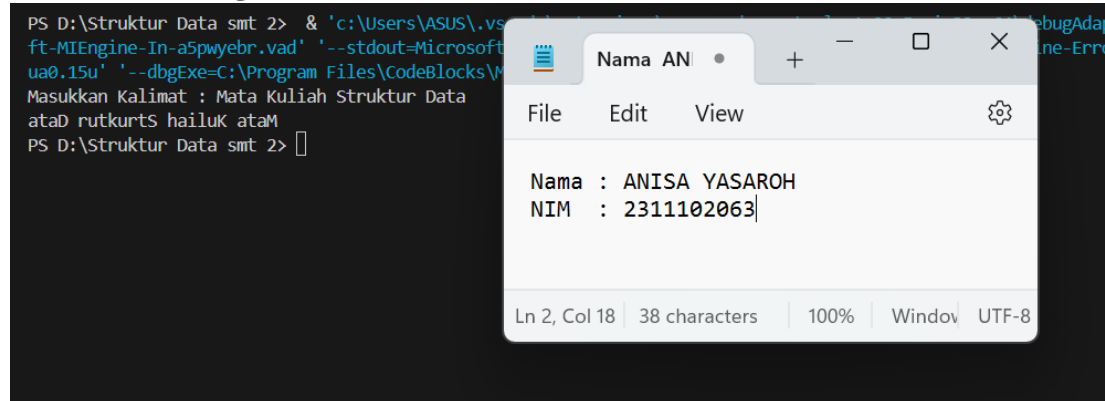
    stack<char> charStack;
    int lengthInput = input.length();

    for (int a_63 = 0; a_63 < lengthInput; a_63++)
    {
        charStack.push(input[a_63]);
    }

    for (int a_63 = 0; a_63 < lengthInput; a_63++)
    {
        cout << charStack.top();
        charStack.pop();
    }

    return 0;
}
```

## Screenshoot Program



The screenshot shows a Windows environment. In the background, a terminal window displays the following text:

```
PS D:\Struktur Data smt 2> & 'c:\Users\ASUS\.vscode\bin\DebugAdapt...ne-Err...  
ft-MIEngine-In-a5pwyebr.vad' '--stdout=Microsoft...  
ua0.15u' '--dbgExe=C:\Program Files\CodeBlocks\M...  
Masukkan Kalimat : Mata Kuliah Struktur Data  
ataD rutkurTS hailuK ataM  
PS D:\Struktur Data smt 2> █
```

In the foreground, a text editor window titled 'Nama ANI' is open. It contains the following text:

```
Nama : ANISA YASAROH  
NIM : 2311102063█
```

The text editor's status bar at the bottom indicates 'Ln 2, Col 18 | 38 characters | 100% | Window | UTF-8'.

## Deskripsi Program

Program di atas membalik urutan sebuah kalimat yang dimasukkan oleh pengguna menggunakan stack. Setelah meminta pengguna untuk memasukkan sebuah kalimat dengan getline, setiap karakter dari kalimat tersebut dimasukkan ke dalam stack. Selanjutnya, program mengeluarkan karakter satu per satu dari puncak stack dan mencetaknya, yang menghasilkan urutan karakter yang terbalik dari kalimat asli. Dengan menggunakan stack, program ini mendemonstrasikan bagaimana LIFO (Last In, First Out) bekerja untuk membalikkan string.

## 3. KESIMPULAN

Berdasarkan praktikum yang telah dilakukan, dapat disimpulkan bahwa :

1. Stack adalah struktur data linear yang mengikuti prinsip Last-In First-Out (LIFO), di mana elemen terakhir yang dimasukkan menjadi elemen pertama yang dikeluarkan.
2. Operasi dasar pada stack meliputi push (menambahkan elemen ke stack), pop (menghapus elemen paling atas dari stack), dan peek (melihat nilai elemen paling atas tanpa menghapusnya). Operasi ini membantu dalam manipulasi dan penggunaan data dalam stack.
3. Penerapan stack dapat membantu dalam penyelesaian masalah yang memerlukan penanganan data secara terbalik atau dalam urutan terbalik, serta membantu mengatur aliran eksekusi program.

#### **4. REFERENSI**

Anita Sindar, R. M. S. (2019). *Struktur Data Dan Algoritma Dengan C++* (Vol. 1). CV. AA. RIZKY.

Astuti, I. K. (2019). STRUKTUR DATA LINKED LIST.

Erkamim, E., Abdurrohman, I., Yuliyanti, S., Karim, R., Rahman, A., Admira, T. M. A., & Ridwan, A. (2024). *Buku Ajar Algoritma dan Struktur Data*. PT. Sonpedia Publishing Indonesia.

Sulasmoro, A. H. (2022). *Buku ajar algoritma dan pemrograman I*. Penerbit P4I.

UNITY, O. D. C. D., & GAME, B. O. O. O. (2024). DASAR-DASAR PEMROGRAMAN GIM BERORIENTASI.