

LAPORAN PRAKTIKUM

Modul 07

“Stack”



Disusun Oleh:

Ganesha Rahman Gibran -2211104058

Kelas S1SE-07-02

Dosen :

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

**PROGRAM STUDI S1 SOFTWARE ENGINEERING
FAKULTAS INFORMATIKA
TELKOM UNIVERSITY
PURWOKERTO
2024**

A. TUJUAN PRAKTIKUM

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

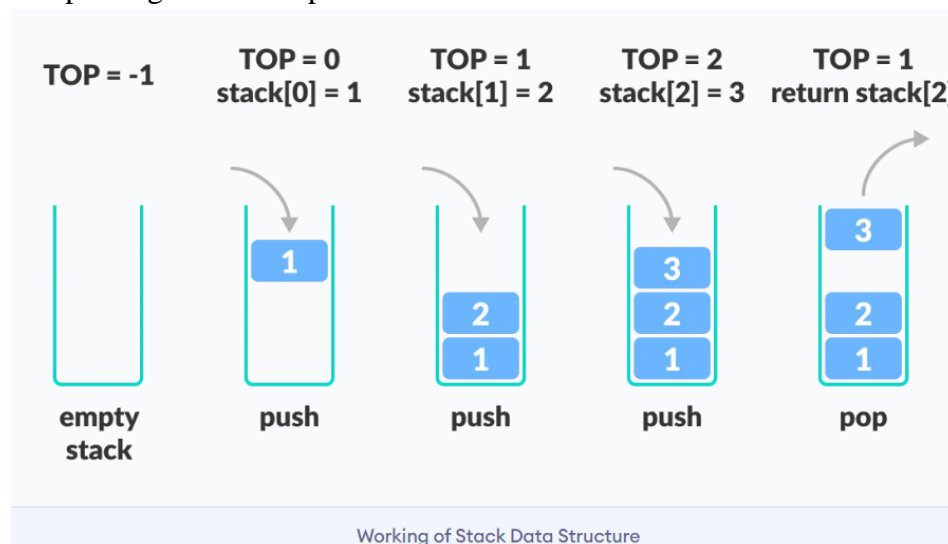
B. DASAR TEORI

Stack, atau tumpukan, adalah sebuah struktur data yang dirancang untuk menyimpan dan mengelola data dengan prinsip Last-In, First-Out (LIFO). Sebuah analogi yang sering digunakan untuk memahami konsep ini adalah tumpukan piring di kafetaria, di mana piring yang terakhir ditambahkan akan menjadi yang pertama diambil.

Dalam implementasinya, stack direpresentasikan sebagai struktur data terurut yang mendukung dua operasi utama, yaitu push dan pop. Operasi push digunakan untuk menambahkan elemen baru ke dalam stack, sementara operasi pop berfungsi untuk menghapus elemen teratas dari stack.

Karena menerapkan prinsip LIFO, stack memiliki berbagai kegunaan dalam aplikasi praktis, seperti manajemen memori komputer, evaluasi ekspresi aritmatika, dan pengelolaan panggilan fungsi dalam pemrograman. Sebagai contoh, dalam manajemen memori, stack sering digunakan untuk menyimpan alamat-alamat memori yang dialokasikan untuk variabel atau fungsi.

Struktur data ini memungkinkan akses data yang efisien, di mana elemen yang terakhir dimasukkan dapat diakses terlebih dahulu. Dengan karakteristik tersebut, stack menjadi salah satu komponen penting dalam pengembangan perangkat lunak dan pemrograman komputer.



Operasi pada stack melibatkan beberapa fungsi dasar yang dapat digunakan untuk mengelola elemen dalam struktur data ini. Berikut adalah operasi-operasi umum pada stack:

- a. **Push (Masukkan):** Menambahkan elemen baru ke tumpukan pada posisi paling atas.
- b. **Pop (Keluarkan):** Menghapus elemen dari posisi paling atas tumpukan.
- c. **Top (Atas):** Mengakses elemen teratas dari tumpukan tanpa menghapusnya.
- d. **IsEmpty (Kosong):** Memeriksa apakah tumpukan dalam keadaan kosong.
- e. **IsFull (Penuh):** Memeriksa apakah tumpukan telah mencapai kapasitas maksimum (relevan pada stack dengan kapasitas terbatas).
- f. **Size (Ukuran):** Mengembalikan jumlah elemen yang saat ini ada dalam tumpukan.
- g. **Peek (Lihat):** Melihat elemen pada posisi tertentu dalam tumpukan tanpa menghapusnya.
- h. **Clear (Hapus Semua):** Menghapus semua elemen yang ada dalam tumpukan sehingga menjadi kosong.
- i. **Search (Cari):** Mencari elemen tertentu dalam tumpukan untuk menentukan apakah elemen tersebut ada dan di mana posisinya.

Operasi-operasi ini memungkinkan stack berfungsi secara fleksibel dan efisien dalam berbagai aplikasi pemrograman.

C. GUIDED 1

Sourcecode

```
#include <iostream>
#define MAX 100

using namespace std;

class Stack{
private:
    int top;
    int arr[MAX];
public:
    Stack() { top = -1;}

    bool isFull() {return top == MAX -1;}
    bool isEmpty() {return top == -1;}

    void push (int x){
        if (isFull()){
            cout << "Stack Overflow\n";
            return;
        }
        arr[++top]=x;
    }

    void pop(){
        if(isEmpty()){
            cout << "Stack Underflow\n";
            return;
        }
        top--;
    }
}
```

```
int peek(){
    if(!isEmpty()){
        return arr[top];
    }
    cout << "Stack is Empty\n";
    return -1;
}

void display(){
    if(isEmpty()){
        cout << "Stack is Empty\n";
        return;
    }
    for (int i = top; i >=0; i--){
        cout << arr[i] << " ";
    }
    cout << endl;
}
};
```

```
int main(){
    Stack s;
    s.push(10);
    s.push(20);
    s.push(30);

    cout << "Stack Elements: ";
    s.display();

    cout << "Top Element: " << s.peek() << endl;

    s.pop();
    s.pop();
    cout << "After popping stack Elements: ";
    s.display();

    return 0;
}
```

Output

```
i'
Stack Elements: 30 20 10
Top Element: 30
After popping stack Elements: 10
PS E:\Struktur Data\2211104058_Ganesha_Rahman_Gibran_SE-06-02>
```

GUIDED 2

Sourcecode

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

class Node{
public:
    int data;
    Node* next;
    Node(int value){
        data = value;
        next = nullptr;
    }
};

class Stack {
private:
    Node* top;

public:
    Stack(){top=nullptr;}
    bool isEmpty(){ return top == nullptr;}

    void push(int x){
        Node* newNode = new Node(x);
        newNode->next = top;
        top = newNode;
    }
}
```

```
void pop(){
    if(isEmpty()){
        cout << "Stack Underflow\n";
        return;
    }
    Node* temp = top;
    top = top->next;
    delete temp;
}

int peek(){
    if(isEmpty()){
        return top ->data;
    }
    cout << "Stack is empty\n";
    return -1;
}

void display(){
    if (isEmpty()){
        cout << "Stack is empty\n";
        return;
    }
    Node* current = top;
    while (current){
        cout << current->data << " ";
        current = current -> next;
    }
    cout << "\n";
}
};
```

```
int main(){
    Stack s;
    s.push(10);
    s.push(20);
    s.push(30);

    cout << "Stack Elements: ";
    s.display();

    cout << "Top Element: " << s.peek() << endl;

    s.pop();
    s.pop();
    cout << "After popping stack Elements: ";
    s.display();

    return 0;
}
```

Output

```
i'
Stack Elements: 30 20 10
Top Element: Stack is empty
-1
After popping stack Elements: 10
PS E:\Struktur Data\2211104058_Ganesha_Rahman_Gibran_SE-06-02>
```

D. 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
```

Input :

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



```
bool isPalindrome(const std::string &input) {
    std::stack<char> charStack;
    std::string cleanedInput;

    for (char ch : input) {
        if (isalnum(ch)) {
            cleanedInput += tolower(ch);
        }
    }

    for (char ch : cleanedInput) {
        charStack.push(ch);
    }

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

    return true;
}

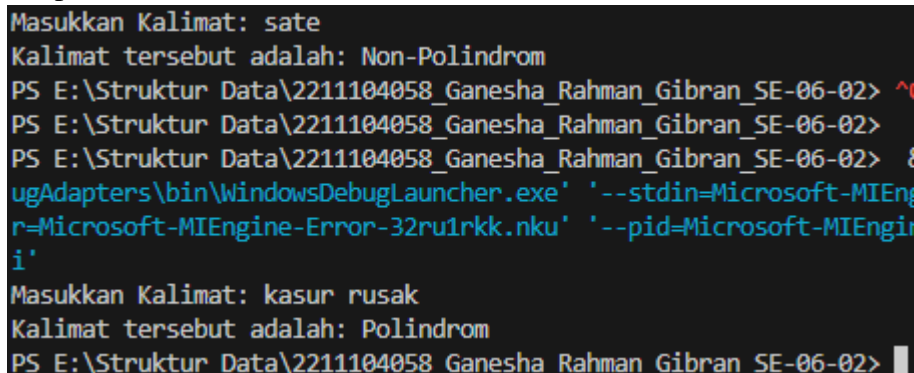
int main() {
    std::string input;

    std::cout << "Masukkan Kalimat: ";
    std::getline(std::cin, input);

    if (isPalindrome(input)) {
        std::cout << "Kalimat tersebut adalah: Polindrom" << std::endl;
    } else {
        std::cout << "Kalimat tersebut adalah: Non-Polindrom" << std::endl;
    }

    return 0;
}
```

Output :



```
Masukkan Kalimat: sate
Kalimat tersebut adalah: Non-Polindrom
PS E:\Struktur Data\2211104058_Ganesha_Rahman_Gibran_SE-06-02> ^C
PS E:\Struktur Data\2211104058_Ganesha_Rahman_Gibran_SE-06-02>
PS E:\Struktur Data\2211104058_Ganesha_Rahman_Gibran_SE-06-02> 8
ugAdapters\bin\WindowsDebugLauncher.exe' '--stdin=Microsoft-MIEng
r=Microsoft-MIEngine-Error-32ru1rkk.nku' '--pid=Microsoft-MIEngin
i'
Masukkan Kalimat: kasur rusak
Kalimat tersebut adalah: Polindrom
PS E:\Struktur Data\2211104058_Ganesha_Rahman_Gibran_SE-06-02> █
```

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
```

Input :

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

std::string reverseString(const std::string &input) {
    std::stack<char> charStack;
    std::string reversed;

    for (char ch : input) {
        charStack.push(ch);
    }

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

    return reversed;
}

int main() {
    std::string input;

    std::cout << "Masukkan Kata: ";
    std::getline(std::cin, input);

    std::string reversed = reverseString(input);

    std::cout << "Data Array: ";
    for (char ch : reversed) {
        std::cout << ch << " ";
    }
    std::cout << "\nData: " << reversed << std::endl;

    return 0;
}
```

Output :

```
Masukkan Kata: Telkom University Purwokerto
Data Array: o t r e k o w r u P   y t i s r e v i n U   m o k l e T
Data: otrekowruP ytisrevinU mokleT
PS E:\Struktur Data\2211104058 Ganesha Rahman Gibran SE-06-02>
```

E. KESIMPULAN

Stack merupakan struktur data yang beroperasi berdasarkan prinsip Last-In, First-Out (LIFO), di mana elemen yang terakhir dimasukkan akan menjadi elemen

pertama yang diakses. Dengan mendukung berbagai operasi seperti push, pop, top, isEmpty, dan lainnya, stack menjadi alat yang sangat fleksibel untuk pengelolaan data. Kemampuan stack dalam menyimpan, mengakses, dan menghapus data secara efisien membuatnya banyak digunakan dalam berbagai aplikasi pemrograman, termasuk manajemen memori, evaluasi ekspresi aritmatika, dan pengelolaan panggilan fungsi. Dengan karakteristik dan fungsionalitasnya yang spesifik, stack merupakan salah satu struktur data dasar yang sangat penting dalam pengembangan perangkat lunak.