

LAPORAN PRAKTIKUM
Modul 07
“STACK”



Disusun Oleh:
Aji Prasetyo Nugroho - 2211104049
S1SE-07-2

Assisten Praktikum :
Aldi Putra
Andini Nur Hidayah

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

PROGRAM STUDI S1 SOFTWARE ENGINEERING
FAKULTAS INFORMATIKA
TELKOM UNIVERSITY
PURWOKERTO
2024

A. Tujuan

1. Memahami konsep stack.
2. Mengimplementasikan stack dengan menggunakan representasi pointer dan tabel.
3. Memahami konsep queue.
4. Mengimplementasikan queue dengan menggunakan pointer dan tabel.

B. Landasan Teori

1. Pengertian Stack

Stack adalah salah satu bentuk struktur data yang menggunakan prinsip LIFO (Last In, First Out), di mana elemen yang terakhir dimasukkan akan menjadi elemen pertama yang dikeluarkan. Struktur ini menyerupai tumpukan, seperti tumpukan piring, di mana hanya elemen di bagian atas yang dapat diakses. Implementasi stack dapat dilakukan dengan dua pendekatan utama, yaitu menggunakan pointer dan array (tabel).

2. Komponen Stack

Dalam stack, terdapat beberapa komponen utama:

- TOP: Penunjuk elemen teratas dari stack. Elemen ini digunakan untuk menambah (push) atau mengambil (pop) elemen.
- Elemen: Data yang disimpan dalam stack, yang dapat berupa tipe data sederhana seperti integer atau tipe data kompleks.

3. Operasi Dasar pada Stack

Operasi utama yang dilakukan pada stack meliputi:

- Push: Menambahkan elemen baru di bagian atas stack.
 - Contoh langkah:
 1. Buat elemen baru.
 2. Sambungkan elemen baru ke elemen yang saat ini berada di TOP.
 3. Perbarui TOP menjadi elemen baru.

- Pop: Menghapus elemen teratas dari stack.
 - Contoh langkah:
 1. Akses elemen yang berada di TOP.
 2. Pindahkan penunjuk TOP ke elemen berikutnya.
 3. Kembalikan elemen terhapus ke pengguna.

4. Primitif-Primitif dalam Stack

Primitif adalah fungsi dasar yang digunakan untuk mengelola stack. Beberapa primitif utama meliputi:

- `createStack()`: Membuat stack baru yang kosong.
- `isEmpty()`: Memeriksa apakah stack kosong.
- `push()`: Menambahkan elemen ke stack.
- `pop()`: Menghapus elemen dari stack.
- `printInfo()`: Menampilkan elemen-elemen yang ada di stack.

5. Implementasi Stack

Implementasi stack dapat dilakukan dengan dua cara:

- Pointer: Menggunakan struktur linked list untuk menyimpan elemen-elemen stack. Keunggulannya adalah ukuran stack fleksibel, tetapi membutuhkan manajemen memori yang lebih kompleks.
- Array (Tabel): Menggunakan array dengan indeks untuk menyimpan elemen. Cara ini lebih sederhana namun memiliki batasan pada ukuran maksimum stack.

6. Aplikasi Stack

Stack digunakan dalam berbagai aplikasi, seperti:

- Implementasi algoritma rekursif.
- Penyimpanan ekspresi matematika dan evaluasinya (misalnya dalam infix ke postfix).
- Pemrograman compiler untuk fungsi call stack.

C. Guided

1. Guided 1

Source Code :

```
#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 << "\n";
    }
};

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

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

    cout << "Top element: " << s.peek() << "\n";

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

    return 0;
}
```

Output :

```
Stack elements: 30 20 10  
Top element: 30  
After popping, stack elements: 10  
PS D:\Praktikum STD_2211104049>
```

2. Guided 2

Source Code :

```
#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())
        {
            cout << "Stack is empty\n";
            return -1;
        }
        return top->data;
    }

    void display()
    {
        if (isEmpty())
        {
            cout << "Stack is empty\n";
            return;
        }
        Node *current = top;
        while (current != nullptr)
        {
            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() << "\n";

    s.pop();
    cout << "Stack after pop: ";
    s.display();

    return 0;
}
```

Output :

```
Stack elements: 30 20 10  
Top element: 30  
Stack after pop: 20 10  
PS D:\Praktikum STD 2211104049>
```

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
```

Source Code :

- a. main.cpp

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

bool isPalindrome(string kalimat) {
    stack<char> s;
    string cleaned = "";

    for (char c : kalimat) {
        if (isalnum(c)) {
            cleaned += tolower(c);
        }
    }

    for (char c : cleaned) {
        s.push(c);
    }

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

    return true;
}

int main() {
    string kalimat;

    cout << "Masukkan kalimat: ";
    getline(cin, kalimat);

    if (isPalindrome(kalimat)) {
        cout << "Kalimat tersebut adalah: Palindrom" << endl;
    } else {
        cout << "Kalimat tersebut adalah: Bukan Palindrom" << endl;
    }

    return 0;
}
```


Output :

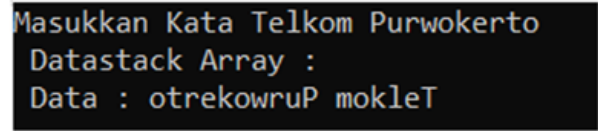
```
Masukkan kalimat: itu
Kalimat tersebut adalah: Bukan Palindrom
PS D:\Praktikum STD_2211104049> ^C
PS D:\Praktikum STD_2211104049>
PS D:\Praktikum STD_2211104049> & 'c:\Users\
crosoft-MIEngine-In-2hvvsfeo.0ts' '--stdout=M
xgfg2pi.qeg' '--dbgExe=D:\TDM-GCC-64\bin\gdb.
Masukkan kalimat: ini
Kalimat tersebut adalah: Palindrom
PS D:\Praktikum STD_2211104049> █
```

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

a. main.cpp

Source Code :

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

void reverseWordsWithStack(string kalimat) {
    stack<char> s;
    string reversed = "";

    for (char c : kalimat) {
        if (c == ' ') {
            while (!s.empty()) {
                reversed += s.top();
                s.pop();
            }
            reversed += ' ';
        } else {
            s.push(c);
        }
    }

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

    cout << "Data: " << reversed << endl;
}

int main() {
    string kalimat;

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

    cout << "Datastack Array : " << endl;
    reverseWordsWithStack(kalimat);

    return 0;
}
```

Output :

```
Masukkan Kata: ini
Datastack Array :
Data: ini
PS D:\Praktikum STD_2211104049> ^C
PS D:\Praktikum STD_2211104049>
PS D:\Praktikum STD_2211104049> & 'c:\U
crosoft-MIEngine-In-anhgq2rw.tfo' '--std
uqu0bvi.nui' '--dbgExe=D:\TDM-GCC-64\bin
Masukkan Kata: itu
Datastack Array :
Data: uti
PS D:\Praktikum STD_2211104049> █
```

E. Kesimpulan

Stack merupakan salah satu struktur data penting yang menerapkan prinsip LIFO (Last In, First Out), di mana elemen yang terakhir dimasukkan akan menjadi yang pertama dikeluarkan. Struktur ini sangat berguna dalam berbagai aplikasi pemrograman seperti pengolahan ekspresi matematika, pengelolaan memori, atau implementasi algoritma rekursif. Operasi dasar stack meliputi push untuk menambahkan elemen dan pop untuk mengambil elemen, dengan implementasi yang dapat dilakukan menggunakan pointer (linked list) atau array (tabel). Pemahaman tentang stack memberikan dasar yang kuat untuk mengelola data secara efisien dan terstruktur sesuai kebutuhan aplikasi.