LAPORAN PRAKTIKUM

PERTEMUAN 7

STACK

Nama :
Alvin Bagus Firmansyah - 2311104070

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

PROGRAM STUDI S1 REKAYASA PERANGKAT LUNAK

FAKULTAS INFORMATIKA

TELKOM UNIVERSITY PURWOKERTO

2024

I. TUJUAN

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

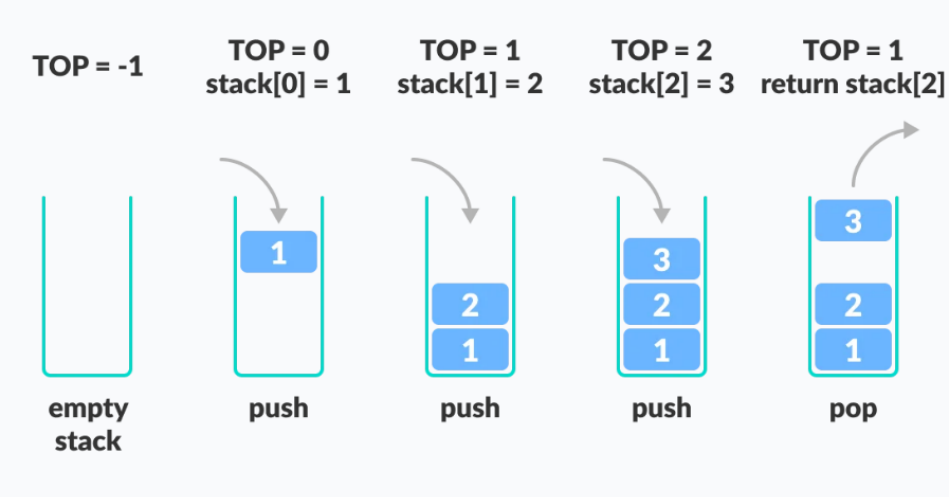
II. TOOL

IDE/Editor: Program ini dapat dijalankan di IDE seperti Visual Studio Code,

Code::Blocks, atau Dev-C++.

III. DASAR TEORI

Stack adalah struktur data linier yang mengikuti prinsip Last In, First Out (LIFO). Artinya, elemen terakhir yang dimasukkan ke dalam stack akan menjadi elemen pertama yang dikeluarkan.



Working of Stack Data Structure

LIFO: Prinsip dasar stack adalah LIFO. Elemen yang paling baru ditambahkan akan berada di atas dan akan menjadi yang pertama dihapus.

Operasi:

- Push: Menambahkan elemen baru ke atas stack.
- Pop: Menghapus elemen paling atas dari stack.
- Peek: Melihat elemen paling atas tanpa menghapusnya.
- IsEmpty: Mengecek apakah stack kosong.
- IsFull: Mengecek apakah stack penuh (jika stack memiliki kapasitas terbatas).

Representasi:

- Array: Stack dapat diimplementasikan menggunakan array dengan indeks sebagai penunjuk ke elemen teratas.
- Linked List: Stack juga dapat diimplementasikan menggunakan linked list, di mana setiap node menyimpan data dan pointer ke node berikutnya.

IV. GUIDE

1.Codingan

main.cpp | main.cpp | main.cpp | include\stack.h | stack.cpp | main.cpp | main.cpp

```cpp
#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 << "Elemen stack: ";
    s.display();

    cout << "Elemen teratas: " << s.peek() << "\n";

    s.pop();
    s.pop();
    cout << "Setelah pop, elemen stack: ";
    s.display();

    return 0;
}
```

Hasil Outputnya:

```
"C:\Users\alvin\OneDrive\Dol
Elemen stack: 30 20 10
Elemen teratas: 30
Setelah pop, elemen stack: 10

Process returned 0 (0x0)   execution time : 1.401 s
Press any key to continue.
```

2.Codingan

main.cpp

```cpp
#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;
    }
```

main.cpp

```cpp
    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 << "Elemen stack: ";
    s.display();

    cout << "Elemen teratas: " << s.peek() << "\n";

    s.pop();
    cout << "Setelah pop, elemen stack: ";
    s.display();

    return 0;
}
```

Hasil Outputnya:

```
"C:\Users\alvin\OneDrive\Dok
Elemen stack: 30 20 10
Elemen teratas: 30
Setelah pop, elemen stack: 20 10

Process returned 0 (0x0)   execution time : 10.590 s
Press any key to continue.
```

V. UNGUIDED

1.Program untuk Menentukan Apakah Kalimat Merupakan Palindrom

Codinganya:

main.cpp

```cpp
#include <iostream>
#include <cctype> // Untuk fungsi tolower dan isalpha
#include <stack>  // Untuk stack
#include <string>
using namespace std;

bool isPalindrome(const string &str) {
    stack<char> s;

    // Menyimpan hanya karakter alphanumeric dalam stack
    for (char c : str) {
        if (isalnum(c)) {
            s.push(tolower(c));  // Masukkan karakter ke stack dalam huruf kecil
        }
    }

    // Memeriksa karakter dari depan dan belakang
    int i = 0;
    while (i < str.size()) {
        if (isalnum(str[i])) {
            if (s.top() != tolower(str[i])) {
                return false;  // Jika karakter tidak sama, bukan palindrom
            }
            s.pop();  // Hapus elemen dari stack jika cocok
        }
        i++;
    }

    return true;  // Jika seluruh karakter cocok
}

int main() {
    string input;

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

    if (isPalindrome(input)) {
        cout << "Kalimat tersebut adalah palindrom.\n";
    } else {
        cout << "Kalimat tersebut bukan palindrom.\n";
    }

    return 0;
}
```

Hasil Outputnya:

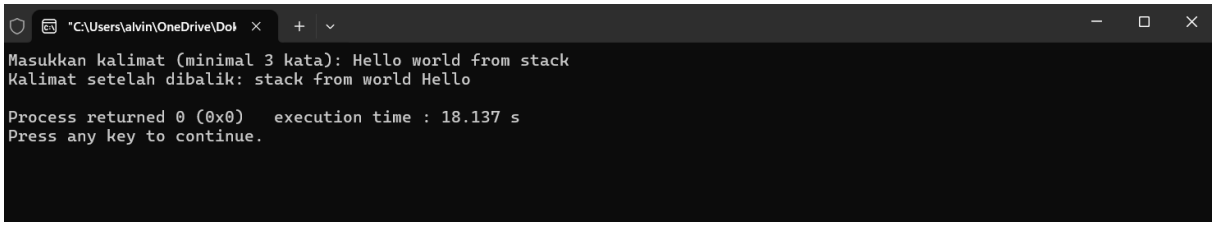
```
"C:\Users\alvin\OneDrive\Dok
Masukkan kalimat: A man a plan a canal Panama
Kalimat tersebut adalah palindrom.

Process returned 0 (0x0)   execution time : 20.250 s
Press any key to continue.
```

2. Program untuk Membalikkan Kalimat Menggunakan Stack.

main.cpp

```cpp
#include <iostream>
#include <stack>
#include <sstream>  // Untuk istringstream
#include <vector>
using namespace std;

void reverseSentence(string sentence) {
    stack<string> words;
    string word;

    // Memecah kalimat menjadi kata-kata dan memasukkan ke stack
    istringstream iss(sentence);
    while (iss >> word) {
        words.push(word);
    }

    // Mencetak kata-kata dari stack (dalam urutan terbalik)
    cout << "Kalimat setelah dibalik: ";
    while (!words.empty()) {
        cout << words.top() << " ";
        words.pop();
    }
    cout << endl;
}

int main() {
    string input;

    cout << "Masukkan kalimat (minimal 3 kata): ";
    getline(cin, input);

    reverseSentence(input);  // Memanggil fungsi untuk membalikkan kalimat

    return 0;
}
```

Hasil Outputnya:



VI. KESIMPULAN

Konsep Stack: Stack adalah struktur data yang mengikuti prinsip Last In, First Out (LIFO). Elemen terakhir yang dimasukkan adalah yang pertama dikeluarkan.

Implementasi Operasi Stack: Dengan menggunakan array atau linked list, kita dapat mengimplementasikan operasi dasar stack, seperti push, pop, peek, isEmpty, dan isFull.

Palindrom dengan Stack: Program untuk memeriksa apakah kalimat adalah palindrom bekerja dengan membandingkan karakter yang dibaca dari depan dengan karakter yang dibaca dari belakang menggunakan stack.

Pembalikan Kalimat dengan Stack: Pembalikan kalimat menggunakan stack dilakukan dengan memasukkan setiap kata ke dalam stack, dan kemudian mengeluarkannya satu per satu untuk mendapatkan urutan yang terbalik.