LAPORAN PRAKTIKUM

Modul 07

“Stack”

Disusun Oleh:

Marvel Sanjaya Setiawan (2311104053)

SE-07-02

Dosen :

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

PROGRAM STUDI S1 SOFTWARE ENGINEERING

FAKULTAS INFORMATIKA

TELKOM UNIVERSITY

PURWOKERTO

2024

1. **Tujuan**
    - Memahami konsep stack.
    - Mengimplementasikan operasi stack
    - Memecahkan masalah dengan stack.
2. **Landasan Teori**

    Stack

    Stack (Tumpukan): Stack adalah struktur data dengan prinsip Last-In, First-Out (LIFO), mirip dengan tumpukan piring di kafetaria. Dua operasi utama pada stack adalah:

    Operasi Dasar Double Linked List.

    a. Push: Menambahkan elemen.

    b. Pop: Menghapus elemen teratas.

    Kegunaan:

    a. Manajemen memori komputer.

    b. Evaluasi ekspresi aritmatika.

    c. Manajemen panggilan fungsi dalam pemrograman.

Operasi pada Stack:

a. Push: Tambah elemen.

b. Pop: Hapus elemen teratas.

c. Top: Lihat elemen teratas tanpa menghapus.

d. IsEmpty: Periksa apakah stack kosong.

e. IsFull: Periksa apakah stack penuh.

f. Size: Jumlah elemen dalam stack.

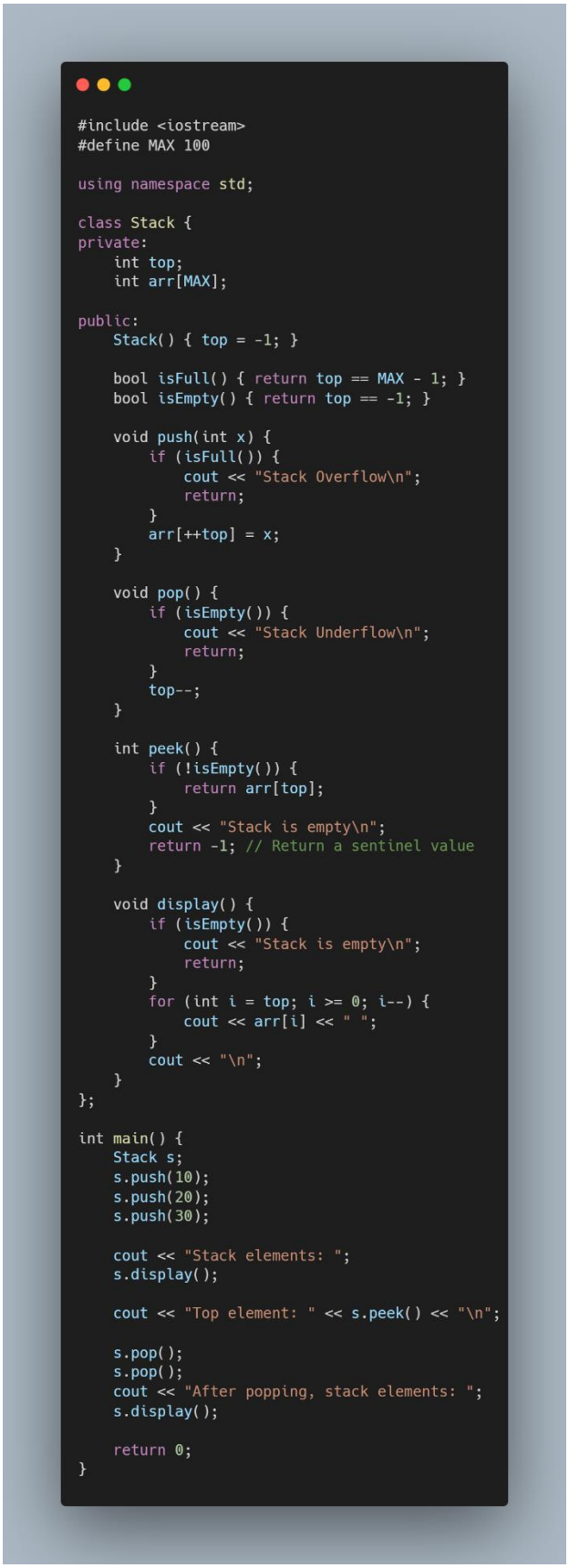
g. Peek: Lihat elemen pada posisi tertentu tanpa menghapus.

h. Clear: Hapus semua elemen.

i. Search: Cari elemen dalam stack.

Telkom University Purwokerto

3. Guided



```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; // Return a sentinel value
    }

    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;
}
```

Telkom University Purwokerto

```
Stack elements: 30 20 10
Top elements: 30
After popping, stack elements: 10
```

Cara Kerja:

1. Kelas Stack dengan atribut top dan arr.
2. top diinisialisasi ke -1.
3. push(int x) menambahkan elemen, cek penuh.
4. pop() menghapus elemen teratas, cek kosong.
5. peek() mengembalikan elemen teratas.
6. display() menampilkan elemen dari atas ke bawah.
7. main() buat objek Stack s.
8. Tambah elemen 10, 20, 30 ke stack.
9. display() menampilkan elemen stack.
10. peek() menampilkan elemen teratas.
11. Hapus dua elemen teratas.
12. display() menampilkan elemen sisa.

Telkom University Purwokerto

```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;
    }

    int peek() {
        if (!isEmpty()) {
            return top->data;
        }
        cout << "Stack is empty\n";
        return -1; // Return a sentinel value
    }

    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() << "\n";

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

    return 0;
}
```

```
Stack elements: 30 20 10
Top elements: 30
After popping, stack elements: 10
```

Cara Kerja:

1. Kelas Node: Mewakili setiap elemen dalam stack.
2. Kelas Stack: Mengelola operasi stack.
3. push(int x): Menambahkan node ke atas stack.
4. pop(): Menghapus node teratas.
5. peek(): Mengembalikan data node teratas.
6. display(): Menampilkan semua node dari atas ke bawah.
7. main(): Membuat stack dan melakukan operasi: push 10, 20, 30, display, peek, pop, display.

4. Unguided

1. Membuat program Palindrom kalimat.

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

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

    // Membersihkan string dan mengisi stack
    for (char ch : str) {
        if (std::isalnum(ch)) { // Memeriksa apakah karakter adalah alfanumerik
            cleanedStr.push_back(std::tolower(ch)); // Menyimpan dalam bentuk lowercase
            s.push(std::tolower(ch));
        }
    }

    // Membandingkan karakter dari stack dengan string yang dibersihkan
    for (char ch : cleanedStr) {
        if (ch != s.top()) {
            return false; // Jika tidak sama, bukan palindrom
        }
        s.pop();
    }
    return true; // Jika semua karakter sama, maka palindrom
}

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

    if (isPalindrome(input)) {
        std::cout << "Kalimat tersebut adalah Palindrom." << std::endl;
    } else {
        std::cout << "Kalimat tersebut bukan Palindrom." << std::endl;
    }

    return 0;
}
```

```
Masukkan kalimat: ini
Kalimat tersebut adalah Palindrom.
```

Cara Kerja:

1. Membersihkan string: Hanya menyimpan karakter alfanumerik dalam lowercase ke stack.
2. Membandingkan: Bandingkan karakter dari stack dengan string yang dibersihkan.
3. Hasil: Jika semua karakter cocok, kalimat adalah palindrom; jika tidak, bukan palindrom.

2. **Membuat pembalikan terhadap kalimat.**

```cpp
#include <iostream>
#include <stack>
#include <sstream>

using namespace std;

// Fungsi untuk membalikkan huruf dalam setiap kata
void reverseWordsInSentence(const string& sentence) {
    stack<string> wordsStack;
    stringstream ss(sentence);
    string word;

    // Memisahkan kata-kata dan memasukkannya ke dalam stack
    while (ss >> word) {
        wordsStack.push(word);
    }

    // Mengeluarkan kata-kata dari stack dan membalikkan hurufnya
    while (!wordsStack.empty()) {
        string currentWord = wordsStack.top();
        wordsStack.pop();

        // Balikkan huruf dari currentWord
        string reversedWord(currentWord.rbegin(), currentWord.rend());
        cout << reversedWord << " ";
    }
    cout << endl;
}

int main() {
    string sentence;

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

    cout << "Datastack Array :" << endl;
    cout << "Data : ";
    reverseWordsInSentence(sentence);

    return 0;
}
```

```
Masukkan kalimat: Telkom Purwokerto
Datastack Array :
Data : otrekowruP mokleT
```

Cara Kerja:

1. **Pisahkan kalimat** menjadi kata-kata dan masukkan ke stack.

2. **Keluarkan kata** dari stack, balikkan hurufnya, dan tampilkan.

5. Kesimpulan

Praktikum ini mempelajari konsep stack serta penerapannya. Operasi dasar stack seperti push, pop, peek, dan display diimplementasikan dalam program. Stack adalah struktur data dengan prinsip Last-In, First-Out (LIFO) yang berguna dalam manajemen memori, evaluasi ekspresi aritmatika, dan manajemen panggilan fungsi dalam pemrograman. Implementasi juga mencakup pemeriksaan string palindrom dan pembalikan kata dalam kalimat menggunakan stack, menunjukkan fleksibilitas dan kegunaan stack dalam berbagai masalah pemrograman..