LAPORAN PRAKTIKUM
STRUKTUR DATA
Modul 7
“STACK”

Disusun Oleh:
Satria Ariq Adelard Dompas/
2211104033 SE-07-2

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

PROGRAM STUDI S1 SOFTWARE ENGINEERING
FAKULTAS INFORMATIKA
TELKOM UNIVERSITY
PURWOKERTO

2024

1. **Tujuan**
   a. Mahasiswa mampu memahami konsep stack
   b. Mahasiswa mampu mengimplementasikan stack dengan menggunakan representasi pointer pada tabel.
   c. Mahasiswa dapat memecahkan permasalahan dengan solusi stack.
2. **Landasan Teori**

   Stack

   Stack (disebut stack dalam bahasa Indonesia) adalah struktur data linier yang menerapkan prinsip last-in-first-out (LIFO).
   Prinsip ini berarti item terakhir yang dimasukkan adalah item pertama yang diambil.
   Contohnya termasuk fungsi undo/redo dan penumpukan piring di restoran.
   Dalam implementasinya, tumpukan direpresentasikan sebagai struktur data terurut dengan dua operasi utama

   a. Push: Menambahkan elemen ke tumpukan.
   b. Pop: menghapus elemen dari tumpukan dimulai dengan elemen pertama.

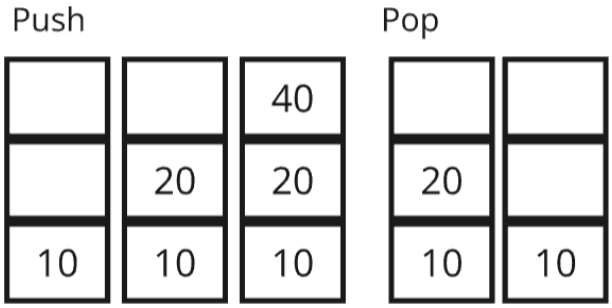
   Prinsip LIFO memungkinkan tumpukan mengakses data secara efisien, menjadikannya salah satu struktur data penting dalam pengembangan perangkat lunak dan pemrograman komputer.

3. **Guided**
   a. Membuat stack

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

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

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

    return 0;
}
```

Jadi penerapan stack adalah First In Last Out (FILO) dan Last In First Out (LIFO).



Output:

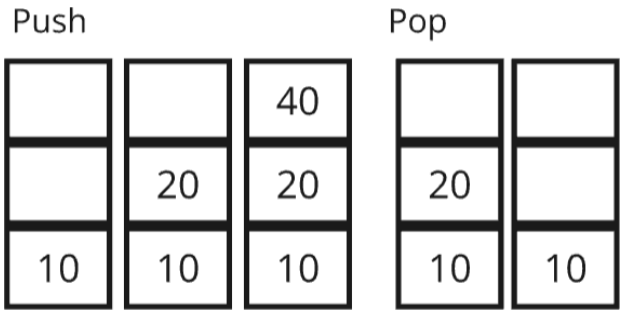
```
OUTPUT  DEBUG CONSOLE  PROBLEMS  TERMINAL  PORTS  COMMENTS

PS D:\Praktikum Struktur Data\Pertemuan8> cd 'd:\Praktikum Struktur Data\Pertemuan8\GUIDED\output'
PS D:\Praktikum Struktur Data\Pertemuan8\GUIDED\output> & .\'Guided.exe'
Stack elements: 30 20 10
Top element: 30
After popping, stack elements: 10
PS D:\Praktikum Struktur Data\Pertemuan8\GUIDED\output>
```

b. Membuat stack menggunakan pointer

```
#include <iosstream>

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 isEmpety() { return top == nullptr; }

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

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

    int peek() {
        if (!isEmpety()) {
            return top->data;
        }
        cout << "Stack is Empety\n";
        return -1;
    }
```

```
    void display() {
        if (isEmpety()) {
            cout << "Stack is Empety\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();
    s.pop();
    cout << "After popping, stack elements: ";
    s.display();

    return 0;
}
```

Jadi penerapan stack adalah First In Last Out (FILO) dan Last In First Out (LIFO).



Output:

```
OUTPUT  DEBUG CONSOLE  PROBLEMS  TERMINAL  PORTS  COMMENTS

PS D:\Praktikum Struktur Data\Pertemuan8> cd 'd:\Praktikum Struktur Data\Pertemuan8\GUIDED\output'
PS D:\Praktikum Struktur Data\Pertemuan8\GUIDED\output> & .\'Guided.exe'
Stack elements: 30 20 10
Top element: 30
After popping, stack elements: 10
PS D:\Praktikum Struktur Data\Pertemuan8\GUIDED\output> cd 'd:\Praktikum Struktur Data\Pertemuan8\GUIDED\output'
PS D:\Praktikum Struktur Data\Pertemuan8\GUIDED\output> & .\'Guided2.exe'
```

4. **Unguided**

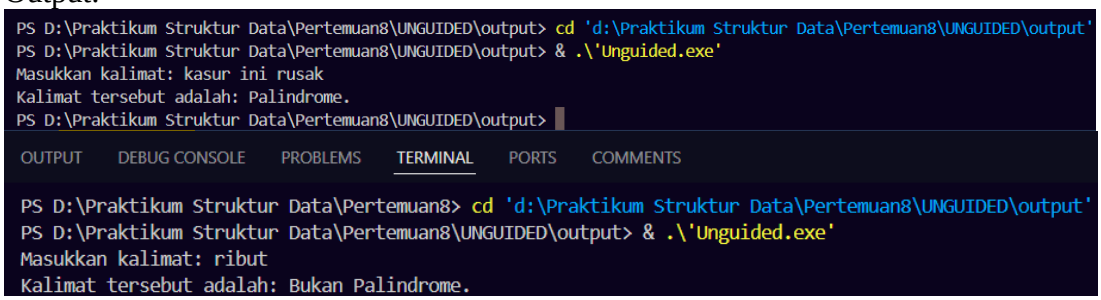
a. Buatlah program untuk menentukan apakah kalimat tersebut yang diinputkan dalam program stack adalah polindrom/tidak. Polindrom kalimat yang dibaca dari depan dan belakang sama. Jelaskan juga bagaimana cara kerja programnya secara singkat. Code:

```
#include <bits/stdc++.h>
using namespace std;

void isPalindrome(string str) {

    string filtered = "";
    for (char ch : str) {
        if (isalnum(ch)) {
            filtered += tolower(ch);
        }
    }

    string rev = filtered;
    reverse(rev.begin(), rev.end());


    if (filtered == rev)
        cout << "Palindrome." << endl;
    else
        cout << "Bukan Palindrome." << endl;
}

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

    cout << "Kalimat tersebut adalah: ";
    isPalindrome(inputString);

    return 0;
}
```

Penjelasan : Program ini memeriksa apakah sebuah string merupakan palindrom, yaitu string yang terbaca sama dari depan maupun belakang. Input string disaring untuk hanya menyertakan karakter alfanumerik dan diubah menjadi huruf kecil. Setelah itu, string yang sudah difilter dibandingkan dengan versi terbaliknya. Jika

keduanya sama, program mencetak "Palindrome."; jika tidak, mencetak "Bukan Palindrome.". Program ini mengabaikan spasi, tanda baca, dan kapitalisasi saat melakukan pemeriksaan.

Output:

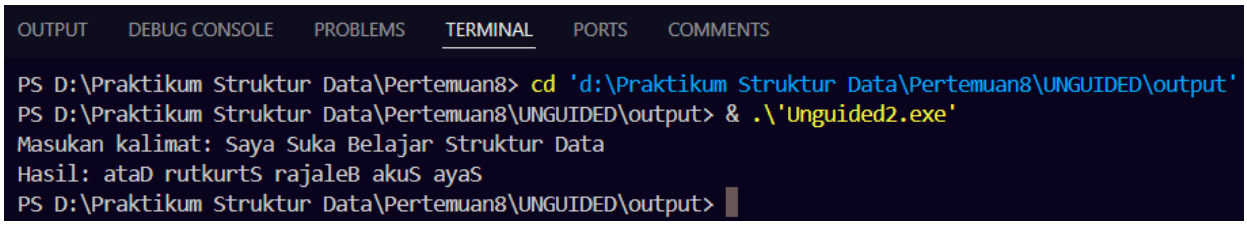


b. Buatlah program untuk melakukan pembalikan terhadap kalimat menggunakan stack dengan minimal 3 kata, jelaskan output program nya beserta code dan fungsin nya.

Code:

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

using namespace std; int main(){
string inputString;
cout << "Masukan kalimat: "; getline(cin, inputString);
reverse(inputString.begin(), inputString.end()); cout<<"Hasil:
"<<inputString;
return 0;
}
```

Penjelasan: Program ini membaca sebuah kalimat dari pengguna, kemudian membalikkan urutan seluruh karakter dalam kalimat tersebut menggunakan fungsi reverse dari pustaka <algorithm>. Input kalimat diambil menggunakan getline(cin, inputString) untuk menangkap keseluruhan kalimat, termasuk spasi. Setelah itu, string yang diinput diproses dengan membalik urutan karakter dari awal hingga akhir. Hasil akhirnya dicetak ke layar dengan format "Hasil: [kalimat terbalik]". Program ini sederhana dan hanya fokus pada pembalikan urutan karakter tanpa modifikasi lainnya.



Output:

5. **Kesimpulan**

   Stack adalah struktur data linear yang mengikuti prinsip **LIFO (Last In, First Out)**, di mana elemen yang terakhir dimasukkan adalah elemen pertama yang akan dikeluarkan. Operasi utama pada stack meliputi **push** (menambahkan elemen ke atas stack), **pop** (menghapus elemen teratas), dan **peek** atau **top** (melihat elemen teratas tanpa menghapusnya). Stack sering digunakan dalam berbagai aplikasi seperti pemrosesan ekspresi matematika, pengelolaan fungsi rekursif, undo-redo dalam aplikasi, dan navigasi di browser. Sederhana namun kuat, stack adalah fondasi dari banyak algoritma dan struktur data lainnya.