LAPORAN PRAKTIKUM STRUKTUR DATA DAN ALGORITMA

MODUL VI

STACK

Disusun Oleh :

NAUFAL THORIQ MUZHAFFAR

2311102078

Dosen

WAHYU ANDI SAPUTRA, S.Pd., M.Eng

PROGRAM STUDI S1 TEKNIK INFORMATIKA

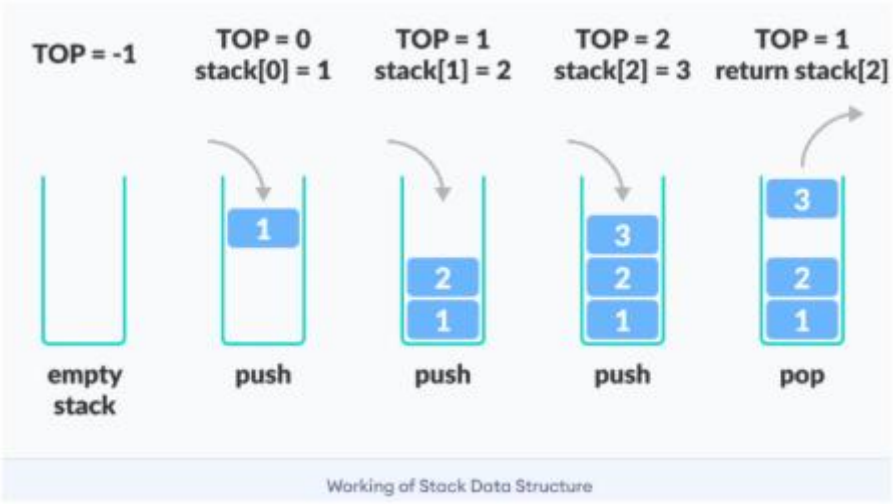
FAKULTAS INFORMATIKA

INSTITUT TEKNOLOGI TELKOM PURWOKERTO

2024

A. Dasar Teori

Stack adalah struktur data LIFO (Last In, First Out) yang mirip dengan daftar terurut. Data baru ditambahkan ke "atas" tumpukan, dan data yang terakhir ditambahkan adalah yang pertama kali dihapus. Hal ini mirip dengan tumpukan piring di kafetaria, di mana piring terakhir yang ditumpuk adalah yang pertama kali diambil.



Operasi stack adalah fungsi dasar yang dapat dilakukan pada struktur data stack. Berikut adalah beberapa operasi umum pada stack:

Operasi Dasar:

1. Push (Masukkan): Menambahkan elemen baru ke bagian atas stack’
2. Pop (Keluarkan): Menghapus elemen teratas dari stack dan mengembalikannya
3. Top (Atas): Mendapatkan nilai elemen teratas pada stack tanpa menghapusnya
4. IsEmpty (Kosong): Memeriksa apakah stack kosong atau tidak

Operasi Tambahan (Opsional):

1. IsFull (Penuh): Memeriksa apakah stack penuh atau tidak
2. Size (Ukuran): Mengembalikan jumlah elemen yang ada dalam stack
3. Peek (Lihat): Melihat nilai elemen pada posisi tertentu dalam stack tanpa menghapusnya.
4. Clear (Hapus Semua): Mengosongkan atau menghapus semua elemen dari stack
5. Search (Cari): Mencari keberadaan elemen tertentu dalam stack.

Operasi-operasi ini memungkinkan manipulasi data yang efisien dalam stack, membuatnya menjadi struktur data yang serbaguna dengan berbagai aplikasi dalam

pemrograman. Kesimpulannya adalah bahwa stack itu merupakan tumpukkan data dimana data tersebut saling bertumpuk antar satu dengan yang lainnya yang mana stack ini memiliki sifat LIFO (Last In First Out) yang mana maksud dari sifat ini adalah diman data yang terakhir masuk dan data tersebut yang pertama kali keluar Stack sendiri memiliki istilah-istilah yang sangat penting didalam stack tersebut. Berikut merupakan istilah-istilah yang terdapat dalam stack:

1. Create

Operasi untuk membuat sebuah stack kosong.

```
struct Stack {
 int top;
 int data[n];
}myStack;
```

2. InitStack

Operasi untuk menginisiakisasikan nilai awal stack (Stack Kosong).

```
void initStack(){
 myStack.top = -1
}
```

3. isEmpty

Operasi untuk menginisialisasikan nilai awal stack (stack kosong). Stack kosong ditandai dengan nilai Top kurang dari nol (-1).

```
int isEmpty()
{
 if (myStack.top == -1)
 return 1; //true
 else
return 0; //false
}
```

4. isFull

Operasi untuk memeriksa apakah stack yang ada sudah penuh. Stack akan mengindikasikan penuh jika puncak stack (top) terletak tepat di bawah jumlah maksimum (MAX) yang dapat ditampung stack.

```
int isFull()
{
 if (myStack.top == MAX-1)
 return 1; //true
 else
 return 0; //false
 }
```

5. Push (Insert)

Operasi untuk menambahkan satu elemen ke dalam stack dan tidak dapat dilakukan jika stack dalam keadaan penuh.

```
void push (int data
{
 if (isEmpty() == 1){
 myStack.top++;
 myStack.data[myStack.top] = Data;
 }
 else if (isFull() == 0){
 myStack.top++;
 myStack.data[myStack.top] = Data;
 }
 else{
 cout << "Stack sudah penuh!";
 }
}
```

6. Pop

Operasi untuk mengambil atau menghapus data teratas (top) dari stack.

```
void pop() {
 if (isEmpty() == 0){
 myStack.top--;
 cout << " Data teratas terambil" << endl;
 } else{
 cout << " Stack masih kosong!" << endl;
 }
}
```

7. Clear

Operasi untuk menghapus atau mengosongkan seluruh data stack. Jika Top bernilai -1 maka stack dianggap kosong.

```
void clear(){
 myStack.top = -1;
 }
```

8. Display

Operasi untuk menampilkan seluruh data stack.

```
void display() {
 if (isEmpty() == 0){
 for (int i= myStack.top; i>=0; i--){
 cout << myStack.data[i] << endl;
 }
 }
 else{
 cout << "Stack masih kosong!" << endl;
 }
}
```

B. Guided

Guided 1

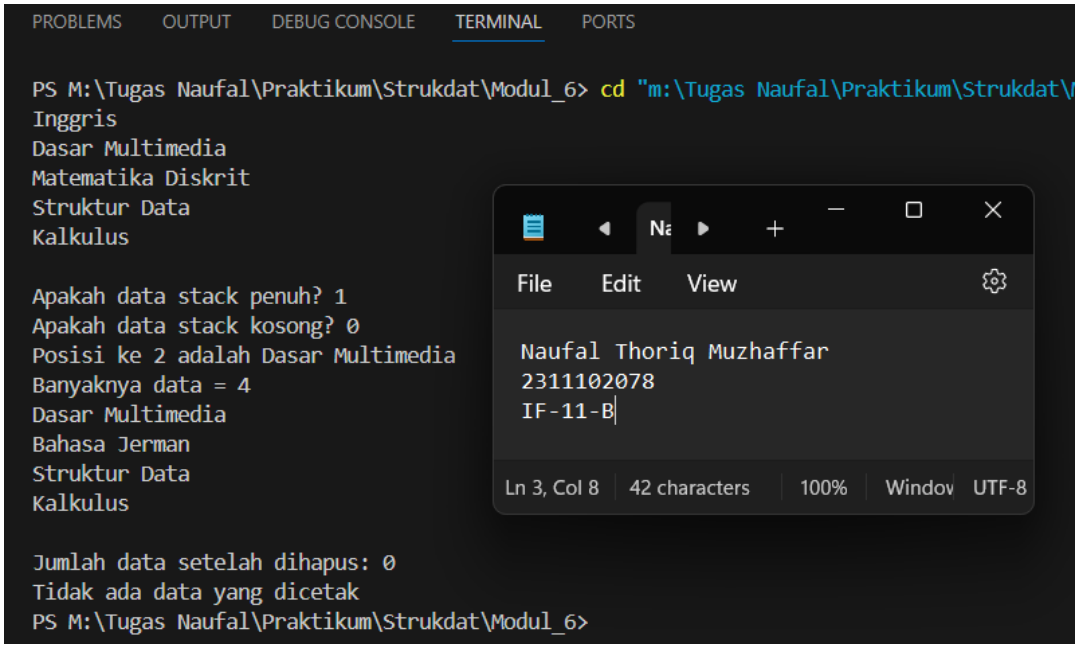
Source Code

```
#include <iostream>
using namespace std;
string arrayBuku[5];
int maksimal = 5, top = 0;
bool isFull()
{
    return (top == maksimal);
}
bool isEmpty()
{
    return (top == 0);
}
void pushArrayBuku(string data)
{
    if (isFull())
    {
        cout << "Data telah penuh" << endl;
    }
    else
    {
        arrayBuku[top] = data;
        top++;
    }
}
void popArrayBuku()
{
    if (isEmpty())
    {
        cout << "Tidak ada data yang dihapus" << endl;
    }
    else
    {
        arrayBuku[top - 1] = "";
        top--;
    }
}
void peekArrayBuku(int posisi)
{
    if (isEmpty())
    {
        cout << "Tidak ada data yang bisa dilihat" << endl;
    }
    else
    {
```

```
        int index = top;
        for (int i = 1; i <= posisi; i++)
        {
            index--;
        }
        cout << "Posisi ke " << posisi << " adalah " << arrayBuku[index] <<
endl;
    }
}
int countStack()
{
    return top;
}
void changeArrayBuku(int posisi, string data)
{
    if (posisi > top)
    {
        cout << "Posisi melebihi data yang ada" << endl;
    }
    else
    {
        int index = top;
        for (int i = 1; i <= posisi; i++)
        {
            index--;
        }
        arrayBuku[index] = data;
    }
}
void destroyArraybuku()
{
    for (int i = top; i >= 0; i--)
    {
        arrayBuku[i] = "";
    }
    top = 0;
}
void cetakArrayBuku()
{
    if (isEmpty())
    {
        cout << "Tidak ada data yang dicetak" << endl;
    }
    else
    {
        for (int i = top - 1; i >= 0; i--)
        {
            cout << arrayBuku[i] << endl;
```

```
        }
    }
}
int main()
{
    pushArrayBuku("Kalkulus");
    pushArrayBuku("Struktur Data");
    pushArrayBuku("Matematika Diskrit");
    pushArrayBuku("Dasar Multimedia");
    pushArrayBuku("Inggris");
    cetakArrayBuku();
    cout << "\n";
    cout << "Apakah data stack penuh? " << isFull() << endl;
    cout << "Apakah data stack kosong? " << isEmpty() << endl;
    peekArrayBuku(2);
    popArrayBuku();
    cout << "Banyaknya data = " << countStack() << endl;
    changeArrayBuku(2, "Bahasa Jerman");
    cetakArrayBuku();
    cout << "\n";
    destroyArraybuku();
    cout << "Jumlah data setelah dihapus: " << top << endl;
    cetakArrayBuku();
    return 0;
}
```

Sreenshots Output



Deskripsi Program

Program tersebut merupakan implementasi stack dengan array dalam bahasa C++. Stack adalah struktur data yang bekerja berdasarkan prinsip LIFO (Last In, First Out). Program ini memiliki beberapa fungsi untuk mengelola stack, seperti menambahkan elemen (push), menghapus elemen (pop), melihat elemen tertentu (peek), mengubah data, menghitung jumlah elemen dalam stack, menghapus seluruh elemen dalam stack, dan mencetak elemen-elemen dalam stack.

C. Unduided

Unguided 1

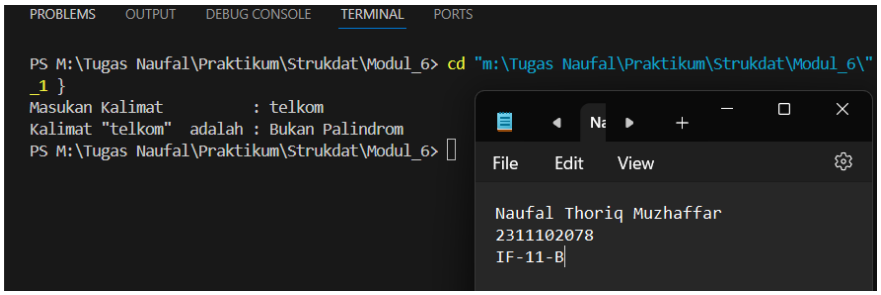
Buatlah program untuk menentukan apakah kalimat tersebut yang diinputkan dalam program stack adalah palindrom/tidak. Palindrom kalimat yang dibaca dari depan dan belakang sama. Jelaskan bagaimana cara kerja programnya.

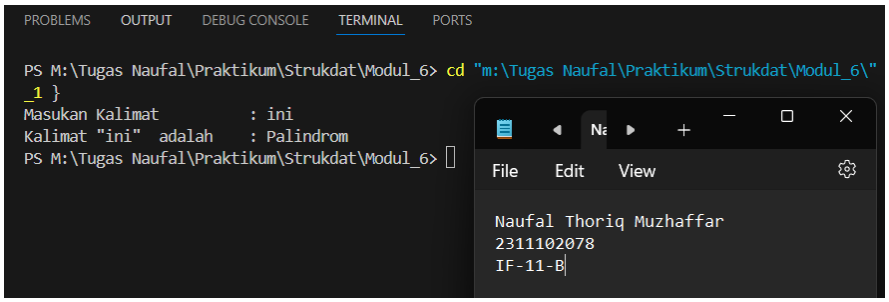
Source Code

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

using namespace std;

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

    for (char ch : str) {
        if (isalpha(ch)) {
            cleanedStr += tolower(ch);
        }
    }


    for (char ch : cleanedStr) {
        s.push(ch);
    }


    for (char ch : cleanedStr) {
```

```
        if (ch != s.top()) {
            return false;
        }
        s.pop();
    }

    return true;
}

int main() {
    string input;

    cout << "Masukan Kalimat\t\t : ";
    getline(cin, input);

    if (isPalindrome(input)) {
        cout << "Kalimat \"" << input << "\"  adalah\t : Palindrom" << endl;
    } else {
        cout << "Kalimat \"" << input << "\"  adalah : Bukan Palindrom" <<
endl;
    }

    return 0;
}
```

Screenshots Output





Deskripsi dan Penjelasan Cara Kerja Program

Program menerima input dari pengguna dan menyimpannya dalam variabel input.Program membersihkan kalimat dengan menghapus spasi, tanda baca, dan mengubah semua huruf

menjadi huruf kecil dalam loop pertama fungsi isPalindrome. Setelah dibersihkan, setiap karakter dimasukkan ke dalam stack yang mengikuti prinsip LIFO (Last In First Out). Program membandingkan karakter satu per satu dari awal kalimat yang dibersihkan dengan karakter yang diambil dari stack. Jika ada karakter yang tidak cocok, program mengembalikan false, menandakan kalimat bukan palindrom. Jika semua karakter cocok, program mengembalikan true, menunjukkan kalimat adalah palindrom. Di fungsi main, berdasarkan hasil dari isPalindrome, program menampilkan apakah kalimat yang diinput adalah palindrom atau bukan.

Unguided 2

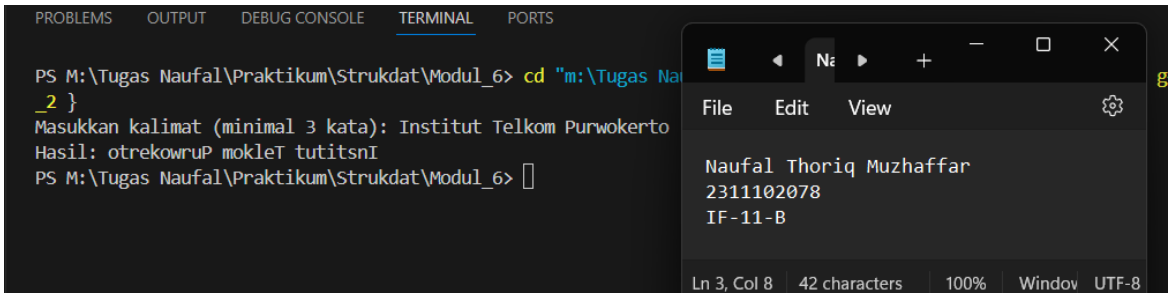
Buatlah program untuk melakukan pembalikan terhadap kalimat menggunakan stack dengan minimal 3 kata. Jelaskan output program dan source codenya beserta operasi/fungsi yang dibuat?

Source Code

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

using namespace std;

string reverseWords(const string &str) {
    stack<char> s;
    string result = "";

    // Membalikkan setiap karakter dalam string menggunakan stack
    for (char ch : str) {
        s.push(ch);
    }

    // Mengambil karakter dari stack untuk membentuk kalimat yang dibalik
    while (!s.empty()) {
        result += s.top();
        s.pop();
    }

    return result;
}

int main() {
    string input;
```

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

    // Memastikan bahwa input memiliki minimal 3 kata
    int wordCount = 0;
    for (char ch : input) {
        if (ch == ' ') {
            wordCount++;
        }
    }
    wordCount++; // Menambah satu untuk kata terakhir

    if (wordCount < 3) {
        cout << "Error: Kalimat harus memiliki minimal 3 kata." << endl;
    } else {
        string reversed = reverseWords(input);
        cout << "Hasil: " << reversed << endl;
    }

    return 0;
}
```

Screenshots Output



Deskripsi dan Penjelasan Program

Program ini dirancang untuk membalikkan urutan karakter dalam sebuah kalimat yang diinputkan oleh pengguna, dengan syarat kalimat tersebut minimal terdiri dari tiga kata. Program ini memanfaatkan stack untuk membalikkan urutan karakter dalam kalimat.

- Stack: Digunakan untuk membalikkan urutan karakter dalam string.
- getline: Digunakan untuk menerima input kalimat dari pengguna.
- Push: Menambahkan karakter ke dalam stack.
- Pop: Menghapus karakter dari stack dan menambahkannya ke string hasil.
- top: Mengambil karakter teratas dari stack tanpa menghapusnya.
- isalpha & amp; tolower: Digunakan dalam contoh program sebelumnya untuk

membersihkan string, tetapi tidak digunakan dalam program ini.

- empty(): Memeriksa apakah stack kosong.

D. Kesimpulan

Stack adalah kumpulan data yang disusun secara bertumpuk, mirip dengan tumpukan fisik. Stack menggunakan prinsip LIFO (Last In First Out), di mana elemen yang terakhir dimasukkan akan menjadi elemen pertama yang diambil. Operasi dasar yang dapat dilakukan pada stack meliputi: Create, initStack, isEmpty, isFull, Push, Pop, clear, dan display. Stack dapat diimplementasikan menggunakan struktur, array, dan single linked list.

E. Referensi

|1| Asprak “Modul 6 Stack”. Learning Management System 2024.

|2| RepublicCode.com – Stack C++. Diakses pada 17 Mei 2024 https://republiccode.com/contoh-program-stack-c/