

**LAPORAN PRAKTIKUM STRUKTUR
DATA DAN ALGORITMA**

**MODUL VI
STACK**



Disusun Oleh :

**NAMA : WISNU RANANTA RADITYA
PUTRA
NIM : 2311102013**

Dosen :

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

**PROGRAM STUDI S1 TEKNIK INFORMATIKA
FAKULTAS INFORMATIKA
INSTITUT TEKNOLOGI TELKOM PURWOKERTO
2024**

BAB I

TUJUAN PRAKTIKUM

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

BAB II

DASAR TEORI

Stack adalah struktur data sederhana yang digunakan untuk menyimpan data (mirip dengan Linked Lists). Dalam tumpukan, urutan kedatangan data penting. Sebuah tumpukan piring di kafetaria adalah contoh bagus dari tumpukan. Piring ditambahkan ke tumpukan saat mereka dibersihkan dan ditempatkan di bagian atas. Ketika sebuah piring dibutuhkan, diambil dari bagian atas tumpukan. Piring pertama yang ditempatkan di tumpukan adalah yang terakhir digunakan.

Definisi: Sebuah tumpukan adalah daftar terurut di mana penyisipan dan penghapusan dilakukan di satu ujung, disebut atas. Elemen terakhir yang dimasukkan adalah yang pertama dihapus. Oleh karena itu, disebut daftar Last in First out (LIFO).

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

- a. Push (Masukkan): Menambahkan elemen ke dalam tumpukan pada posisi paling atas atau ujung.
- b. Pop (Keluarkan): Menghapus elemen dari posisi paling atas atau ujung tumpukan.
- c. Top (Atas): Mendapatkan nilai atau melihat elemen teratas pada tumpukan tanpa menghapusnya.
- d. IsEmpty (Kosong): Memeriksa apakah tumpukan kosong atau tidak.
- e. IsFull (Penuh): Memeriksa apakah tumpukan penuh atau tidak (terutama pada implementasi tumpukan dengan kapasitas terbatas).
- f. Size (Ukuran): Mengembalikan jumlah elemen yang ada dalam tumpukan.
- g. Peek (Lihat): Melihat nilai atau elemen pada posisi tertentu dalam tumpukan tanpa menghapusnya.
- h. Clear (Hapus Semua): Mengosongkan atau menghapus semua elemen dari tumpukan.
- i. Search (Cari): Mencari keberadaan elemen tertentu dalam tumpukan.

BAB III

GUIDED

Guided 1

```
#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;
}

```

Screenshots Output

```
PS C:\Semester 2\Praktikum Strudur Data\Modul_6> & 'c
ndowsDebugLauncher.exe' '--stdin=Microsoft-MIEngine-In
ne-Error-sa5e11xj.4t5' '--pid=Microsoft-MIEngine-Pid-b

Inggris
Dasar Multimedia
Matematika Diskrit
Struktur Data
Kalkulus

Apakah data stack penuh? 1
Apakah data stack kosong? 0
Posisi ke 2 adalah Dasar Multimedia
Banyaknya data = 4
Dasar Multimedia
Bahasa Jerman
Struktur Data
Kalkulus

Jumlah data setelah dihapus: 0
Tidak ada data yang dicetak
PS C:\Semester 2\Praktikum Strudur Data\Modul_6> █
```

Deskripsi:

Program diatas merupakan implementasi dari struktur data stack. Program di atas menggunakan prinsip Last In, First Out (LIFO) untuk mengelola stack buku. Program di atas menggunakan beberapa operasi dasar stack ``isFull`` untuk mengecek apakah stack sudah penuh. Mengembalikan true jika top sama dengan maksimal, dan false jika tidak, ``IsEmpty`` untuk mengecek apakah stack kosong. Mengembalikan true jika top sama dengan 0, dan false jika tidak, ``pushArrayBuku(string data)`` untuk menambahkan buku ke dalam stack, ``popArrayBuku()`` unruk Menghapus buku dari stack, ``peekArrayBuku(int posisi)`` untuk Melihat buku pada posisi tertentu dalam stack tanpa menghapusnya, ``countStack()`` untuk menghitung jumlah buku dalam stack saat ini, yaitu nilai dari top, ``changeArrayBuku(int posisi, string data)`` untuk mengganti buku pada posisi tertentu dalam stack dengan data buku baru, ``destroyArraybuku()`` untuk menghapus semua buku dalam stack dengan mengosongkan array dan mengatur top menjadi 0, ``cetakArrayBuku()`` untuk Mencetak semua buku dalam stack dari atas ke bawah.

BAB IV

UNGUIDED

Unguided 1

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

using namespace std;

bool isPalindrome(string str) {
    stack<char> charStack;
    int length_2311102013 = str.length();
    int i, mid_2311102013 = length_2311102013 / 2;

    for (i = 0; i < mid_2311102013; i++) {
        charStack.push(str[i]);
    }

    for (i = mid_2311102013 + length_2311102013 % 2; i <
length_2311102013; i++) {
        if (charStack.top() != str[i]) {
            return false;
        }
        charStack.pop();
    }

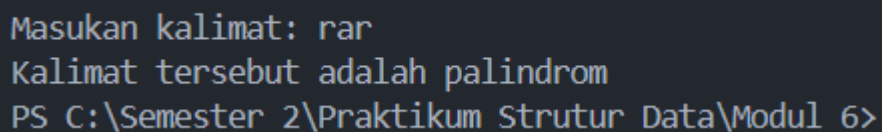
    return true;
}

int main() {
    string input_2311102013;
    cout << "Masukan kalimat: ";
    cin >> input_2311102013;

    if (isPalindrome(input_2311102013)) {
        cout << "Kalimat tersebut adalah palindrom";
    } else {
        cout << "Kalimat tersebut bukan palindrom";
    }

    return 0;
}
```

Screenshots Output:



```
Masukan kalimat: rar
Kalimat tersebut adalah palindrom
PS C:\Semester 2\Praktikum Strutur Data\Modul_6>
```


Deskripsi:

Program di atas merupakan implementasi dari struktur data stack yang memeriksa apakah kalimat tersebut adalah palindrom atau tidak. Stack pada program ini digunakan untuk menyimpan karakter-karakter setengah string, kemudian loop pertama untuk menambahkan karakter ke dalam stack hingga mencapai titik tengah string, selanjutnya loop kedua untuk memulai dari karakter tepat setelah titik tengah (atau melewati tengah jika panjang string ganjil), kemudian membandingkan karakter tersebut dengan karakter yang dikeluarkan dari stack. Jika semua karakter cocok, maka string adalah palindrom.

Unguided 2

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

using namespace std;

string reverseLetters(const string& sentence_2311102013) {
    stack<char> charStack;

    for (char c : sentence_2311102013) {
        charStack.push(c);
    }

    string reversedSentence;
    while (!charStack.empty()) {
        reversedSentence += charStack.top();
        charStack.pop();
    }

    return reversedSentence;
}

int main() {
    string sentence_2311102013;
    cout << "Masukkan kalimat : ";
    getline(cin, sentence_2311102013);

    string reversed_2311102013 =
reverseLetters(sentence_2311102013);
    cout << "Hasil : " << reversed_2311102013 << endl;

    return 0;
}
```

Screenshots Output:

```
Masukkan kalimat : Institut Teknologi Telkom  
Hasil : mokleT igolonkeT tutitsnI  
PS C:\Semester 2\Praktikum Strutur Data\Modul_6>
```

Deskripsi:

Program di atas merupakan implementasi dari struktur data stack. Stack pada program di atas digunakan untuk membalik urutan karakter dalam kalimat. Program ini menggunakan beberapa operasi dasar stack seperti `charStack.push(c)` digunakan untuk menambahkan karakter ke dalam stack, dan `charStack.top()` serta `charStack.pop()` digunakan untuk mengeluarkan karakter dalam urutan terbalik, sehingga menghasilkan kalimat yang terbalik. `charStack.empty()` digunakan untuk memastikan bahwa proses berhenti ketika semua karakter telah dikeluarkan dari stack.

BAB V

KESIMPULAN

Kesimpulan yang dapat saya ambil, Stack adalah struktur data yang mengikuti prinsip Last In First Out (LIFO), di mana elemen terakhir yang dimasukkan adalah yang pertama dikeluarkan. Operasi dasar pada stack meliputi penambahan elemen (push), penghapusan elemen (pop), pemeriksaan elemen teratas tanpa menghapusnya (top), pemeriksaan apakah stack kosong atau penuh (isEmpty dan isFull), mengembalikan jumlah elemen dalam stack (size), melihat nilai pada posisi tertentu tanpa menghapusnya (peek), mengosongkan stack (clear), dan mencari keberadaan elemen tertentu dalam stack (search).

Dengan prinsip dan operasi-operasi ini, stack sangat berguna dalam berbagai aplikasi seperti penanganan ekspresi matematika infix, implementasi rekursi, manajemen memori, dan lainnya. Keuntungan utama dari penggunaan stack adalah sifat LIFO-nya yang dapat digunakan untuk mengelola urutan dan mengakses elemen secara efisien.

BAB VI

DAFTAR PUSTAKA

[1] Asisten Praktikum. 2024. Modul 6 “Stack”. Diakses 20 Mei 2024, 19:00 WIB.
<https://lms.ittelkom-pwt.ac.id/>