

LAPORAN PRAKTIKUM

MODUL VI STACK



Disusun oleh:
Aryo Tegar Sukarno
NIM: 2311102018

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

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

BAB I

TUJUAN PRAKTIKUM

1. Mahasiswa memahami perbedaan konsep dan definisi Hash Code
2. Mahasiswa mampu menerapkan Hash Code dalam pemrograman

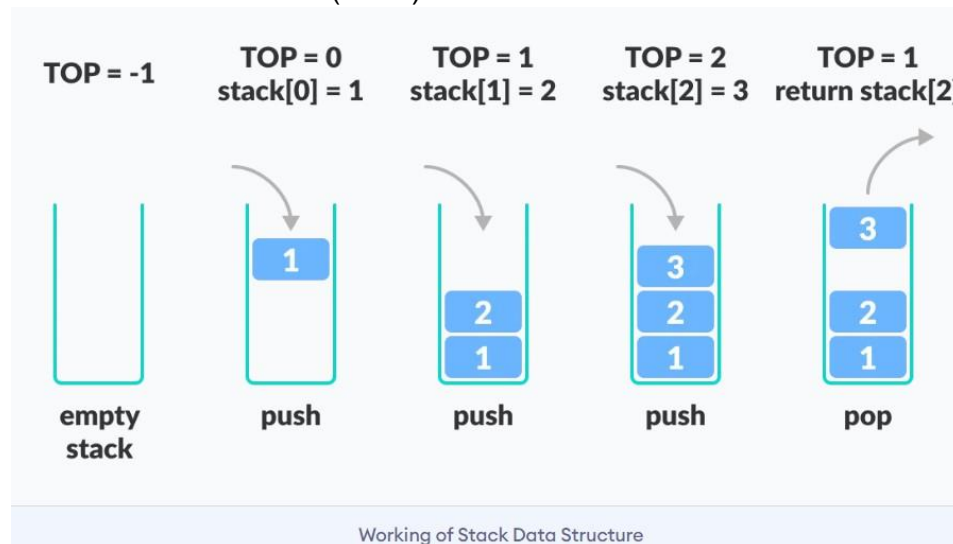
BAB II

DASAR TEORI

a. Pengertian Stack

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:

- Push (Masukkan):** Menambahkan elemen ke dalam tumpukan pada posisi paling atas atau ujung.
- Pop (Keluarkan):** Menghapus elemen dari posisi paling atas atau ujung tumpukan.
- Top (Atas):** Mendapatkan nilai atau melihat elemen teratas pada tumpukan tanpa menghapusnya.
- IsEmpty (Kosong):** Memeriksa apakah tumpukan kosong atau

tidak. Memperbarui data dalam hash table dengan mencari data menggunakan fungsi hash, dan kemudian memperbarui data yang ditemukan.

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

1. 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");
}

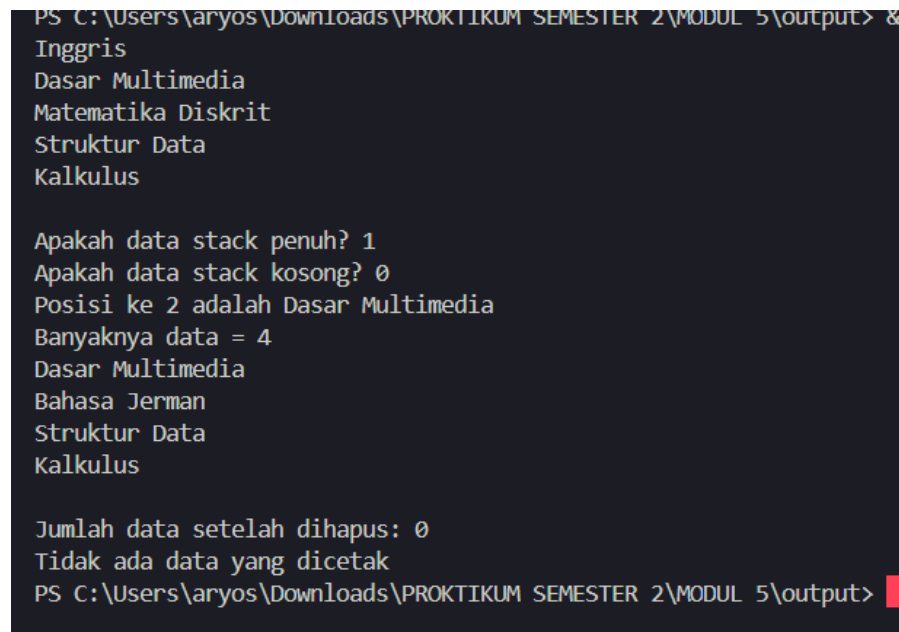
```

```

    cetakArrayBuku();
    cout << "\n";
    destroyArraybuku();
    cout << "Jumlah data setelah dihapus: " << top << endl;
    cetakArrayBuku();
    return 0;
}

```

Screenshoot program



```

PS C:\Users\aryos\Downloads\PROKTIKUM SEMESTER 2\MODUL 5\output> &
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:\Users\aryos\Downloads\PROKTIKUM SEMESTER 2\MODUL 5\output>

```

Deskripsi program

Program diatas merupakan implementasi dari struktur data stack menggunakan array. Dengan menggunakan struktur ini, program dapat menyimpan daftar judul buku dan melakukan berbagai operasi dasar seperti menambahkan buku ke dalam stack, menghapus buku dari stack, melihat judul buku pada posisi tertentu, mengubah judul buku pada posisi tertentu, menghitung jumlah buku dalam stack, menghapus semua buku dari stack, dan mencetak seluruh daftar buku yang tersimpan dalam stack. Struktur data stack ini memberikan cara yang efisien untuk mengatur dan mengakses informasi dalam daftar buku.

LATIHAN KELAS - UNGUIDED

1. Unguided 1

Source code

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

//aryo tegar sukarno
//2311102018
//IF-11-A

using namespace std;

bool cekPalindrom(const string& kalimat) {
    stack<char> tumpukan;

    for (char ch : kalimat) {
        if (ch != ' ') {
            tumpukan.push(tolower(ch));
        }
    }

    for (char ch : kalimat) {
        if (ch != ' ') {
            if (tumpukan.top() != tolower(ch)) {
                return false;
            }
            tumpukan.pop();
        }
    }
}
```

```

    }
    return true;
}

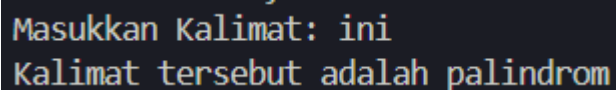
int main() {
    string kalimat;
    cout << "Masukkan Kalimat: ";
    getline(cin, kalimat);

    if (cekPalindrom(kalimat)) {
        cout << "Kalimat tersebut adalah palindrom\n";
    } else {
        cout << "Kalimat tersebut bukan palindrom\n";
    }

    return 0;
}

```

Screenshoot program



```

Masukkan Kalimat: ini
Kalimat tersebut adalah palindrom

```

Deskripsi Program

Program ini merupakan implementasi dari algoritma pengecekan palindrom menggunakan stack. Algoritma ini memanfaatkan struktur data stack untuk menyimpan karakter-karakter dari kalimat yang dimasukkan oleh pengguna. Setelah karakter-karakter disimpan, program melakukan pengecekan karakter dari kedua ujung, yaitu depan dan belakang, dengan menggunakan stack. Jika karakter-karakter tersebut sesuai, maka kalimat tersebut dianggap sebagai palindrom. Pada akhirnya, program akan memberikan output apakah kalimat yang dimasukkan merupakan palindrom atau bukan. Algoritma ini memberikan cara yang efisien untuk mengecek palindromitas sebuah kalimat.

2. Unguided 1

Source code

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

//Aryo Tegar Sukarno
//2311102018
//IF-11-A

using namespace std;

int main() {
    stack<char> s;
    string kalimat, hasil;

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

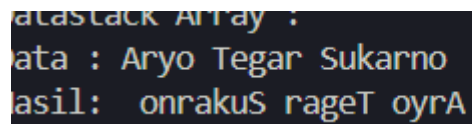
    for (char c : kalimat) {
        s.push(c);
    }

    while (!s.empty()) {
        hasil += s.top();
        s.pop();
    }

    cout << "Datastack Array :" << endl;
    cout << "Data : " << kalimat << endl;
    cout << "Hasil: " << hasil << endl;
```

```
    return 0;  
}
```

Screenshoot program



```
ataStack Array :  
ata : Aryo Tegar Sukarno  
asil: onrakuS rageT oyra
```

Deskripsi Program

Program diatas menggunakan struktur data stack yang dimana untuk melakukan pembalikan karakter-karakter dari sebuah kata atau kalimat yang dimasukkan oleh pengguna. Setiap karakter dari input kalimat dimasukkan ke dalam stack satu per satu. Kemudian, karakter-karakter tersebut dikeluarkan dari stack satu per satu sehingga menghasilkan kalimat yang telah terbalik. Proses pembalikan karakter dilakukan dengan menggunakan operasi pop dan top pada stack. Setelah proses pembalikan selesai, program mencetak hasil pembalikan tersebut. Input awal, yaitu kata atau kalimat yang dimasukkan oleh pengguna, dicetak bersama dengan hasil pembalikan karakter.

BAB IV

KESIMPULAN

Single dan double linked list adalah dua struktur data penting dalam pemrograman, yang digunakan untuk menyimpan dan mengorganisir kumpulan data secara terurut. Kesimpulannya, pemilihan antara single dan double linked list tergantung pada kebutuhan dan kinerja aplikasi yang diinginkan. Single linked list cenderung lebih hemat ruang memori dan cocok untuk aplikasi yang hanya memerlukan penambahan dan penghapusan di bagian depan atau belakang list. Sementara itu, double linked list lebih fleksibel dalam operasi penambahan, penghapusan, dan traversal maju-mundur, namun membutuhkan lebih banyak ruang memori.