

LAPORAN PRAKTIKUM

MODUL VI STACK



Disusun oleh:
Nadhif Atha Zaki
NIM: 2311102007

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

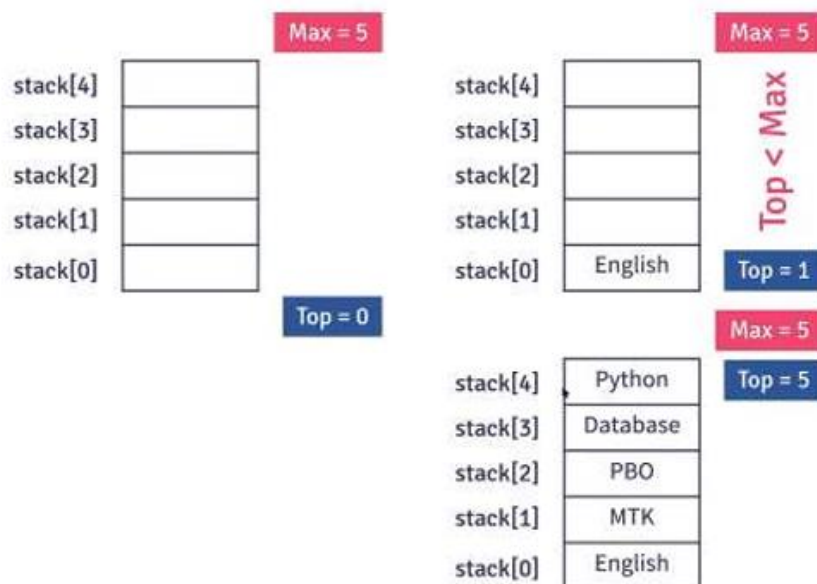
- a. Mampu memahami konsep stack pada struktur data dan algoritma
- b. Mampu mengimplementasikan operasi-operasi pada stack
- c. Mampu memecahkan permasalahan dengan Solusi stack

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).

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

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;
}

```

}

Screenshot program:

```

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 D:\Nathhhh\matkul\smst 2\praktikum struktur data\modul 6 - stack>

```

Deskripsi program:

1. Program ini mengimplementasikan operasi stack menggunakan array dengan ukuran maksimum 5 elemen.
2. `arrayBuku` digunakan untuk menyimpan data buku dalam bentuk string.
3. Fungsi `isFull` mengecek apakah stack sudah penuh dengan mengembalikan nilai `true` jika `top` sama dengan `maksimal`.
4. Fungsi `isEmpty` mengecek apakah stack kosong dengan mengembalikan nilai `true` jika `top` sama dengan 0.
5. Fungsi `pushArrayBuku` menambahkan data buku ke dalam stack jika stack belum penuh, jika penuh mencetak pesan bahwa data telah penuh.

6. Fungsi ``popArrayBuku`` menghapus data buku dari stack jika tidak kosong, jika kosong mencetak pesan bahwa tidak ada data yang dihapus.
7. Fungsi ``peekArrayBuku`` menampilkan data buku pada posisi tertentu dari atas stack, posisi dihitung dari 1.
8. Fungsi ``countStack`` mengembalikan jumlah data buku yang ada dalam stack.
9. Fungsi ``changeArrayBuku`` mengganti data buku pada posisi tertentu dalam stack dengan data baru jika posisi valid.
10. Fungsi ``destroyArraybuku`` menghapus semua data dalam stack dan mengatur ulang ``top`` ke 0, sementara fungsi ``cetakArrayBuku`` menampilkan semua data dalam stack dari atas ke bawah.

.

LATIHAN KELAS - 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.

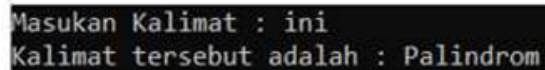
Contoh:

Kalimat : ini

Kalimat tersebut adalah polindrom

Kalimat : telkom

Kalimat tersebut adalah bukan polindrom



```
Masukan Kalimat : ini
Kalimat tersebut adalah : Palindrom
```

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 D:\Nathhh\matkul\smst 2\praktikum struktur data\modul 6 - stack> cd "d:\
ul\smst 2\praktikum struktur data\modul 6 - stack\" ; if ($?) { g++ unguide
nguided1 } ; if ($?) { .\unguided1 }
Masukkan Kalimat: apa
Kalimat tersebut adalah: Palindrom
PS D:\Nathhh\matkul\smst 2\praktikum struktur data\modul 6 - stack>

```

Deskripsi dan Penjelasan Program:

1. Input Kalimat:

- Program meminta pengguna untuk memasukkan sebuah kalimat menggunakan **getline(cin, kalimat)**.

2. Normalisasi Kalimat:

- Kalimat yang diinputkan dinormalisasi dengan menghapus semua karakter non-alfanumerik dan mengubah semua huruf

menjadi huruf kecil. Ini dilakukan dalam loop pertama di dalam fungsi **adalahPalindrom**.

3. **Mendorong Karakter ke Stack:**

- Setiap karakter dalam string yang telah dinormalisasi dimasukkan ke dalam stack **tumpukanKarakter**.

4. **Membalikkan Kalimat:**

- Menggunakan stack, karakter-karakter tersebut diambil satu per satu dari stack dan ditambahkan ke string **kalimatTerbalik**, sehingga menghasilkan string yang merupakan versi terbalik dari kalimat asli yang dinormalisasi.

5. **Membandingkan Kalimat:**

- String asli yang dinormalisasi (**kalimatNormal**) dibandingkan dengan string yang dibalik (**kalimatTerbalik**). Jika kedua string tersebut sama, maka kalimat tersebut adalah palindrom.

6. **Output Hasil:**

- Program mencetak hasil apakah kalimat tersebut adalah palindrom atau bukan berdasarkan hasil perbandingan. Jika palindrom, mencetak "Kalimat tersebut adalah: Palindrom". Jika bukan, mencetak "Kalimat tersebut adalah: Bukan Palindrom".

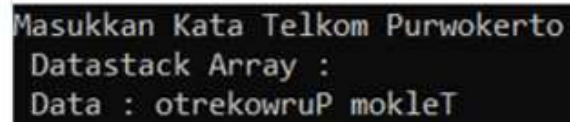
2. .

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

Contoh

Kalimat : Telkom Purwokerto

Hasil : otrekowruP mokleT



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

Source Code:

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

using namespace std;

// Fungsi untuk membalikkan kalimat menggunakan stack
string balikkanKalimat(const string &kalimat)
{
    stack<char> tumpukanKarakter;
    string kalimatTerbalik;

    // Dorong setiap karakter ke dalam stack
    for (char ch : kalimat)
    {
        tumpukanKarakter.push(ch);
    }

    // Ambil karakter dari stack dan bentuk kalimat terbalik
    while (!tumpukanKarakter.empty())
    {
        kalimatTerbalik += tumpukanKarakter.top();
        tumpukanKarakter.pop();
    }

    return kalimatTerbalik;
}
```

```

}

int main()
{
    string kalimat;

    // Meminta pengguna untuk memasukkan kalimat
    cout << "Masukkan Kalimat: ";
    getline(cin, kalimat);

    // Membalikkan kalimat
    string hasil = balikkanKalimat(kalimat);

    // Menampilkan hasil pembalikan
    cout << "Data stack Array : " << endl;
    cout << "Data : " << hasil << endl;

    return 0;
}

```

Screenshot Program:

```

PS D:\Nathhh\matkul\smst 2\praktikum struktur data\modul 6 - stack> cd "d:\Nathhh\matkul\smst 2\praktikum struktur data\modul 6 - stack\" ; if ($?) { g++ unguided2.cpp unguided2 } ; if ($?) { .\unguided2 }
Masukkan Kalimat: apakah dia ni
Data stack Array :
Data : in aid halapa
PS D:\Nathhh\matkul\smst 2\praktikum struktur data\modul 6 - stack>

```

Deskripsi dan Penjelasan Program:

1. Fungsi balikkanKalimat:

- **Tujuan:** Membalikkan urutan karakter dalam string yang diberikan menggunakan stack.
- **Parameter:** Menerima satu parameter **const string &kalimat**, yaitu kalimat yang akan dibalikkan.

- **Operasi:**
 - Mendeklarasikan stack **tumpukanKarakter** untuk menyimpan karakter-karakter dari kalimat.
 - Mengiterasi setiap karakter dalam kalimat dan mendorongnya ke dalam stack (**tumpukanKarakter.push(ch)**).
 - Mengambil karakter satu per satu dari stack (**tumpukanKarakter.top()**) dan menambahkannya ke string **kalimatTerbalik** sambil mengeluarkannya dari stack (**tumpukanKarakter.pop()**).
 - Mengembalikan string **kalimatTerbalik** yang berisi kalimat dengan urutan karakter terbalik.

2. **Fungsi main:**

- **Tujuan:** Mengelola input pengguna dan menampilkan hasil pembalikan kalimat.
- **Operasi:**
 - Menginisialisasi variabel **kalimat** untuk menyimpan input dari pengguna.
 - Meminta pengguna untuk memasukkan sebuah kalimat dan menyimpannya di variabel **kalimat** menggunakan **getline(cin, kalimat)**.
 - Memanggil fungsi **balikkanKalimat** dengan argumen **kalimat** dan menyimpan hasilnya di variabel **hasil**.
 - Menampilkan hasil pembalikan kalimat dengan mencetak **hasil** ke layar.

Ringkasan:

Program ini mengambil input kalimat dari pengguna, membalikkan urutan karakter dalam kalimat tersebut menggunakan stack, dan

kemudian menampilkan kalimat yang sudah dibalik. Fungsi **balikkanKalimat** bertanggung jawab untuk membalikkan kalimat, sedangkan fungsi **main** mengelola alur input-output.

BAB IV

KESIMPULAN

Kesimpulan tentang Stack

Stack adalah struktur data sederhana yang mirip dengan Linked Lists, digunakan untuk menyimpan data dengan urutan kedatangan yang penting. Seperti tumpukan piring di kafetaria, elemen ditambahkan dan dihapus dari bagian atas, menjadikannya contoh nyata dari konsep Last In, First Out (LIFO). Ini berarti elemen terakhir yang dimasukkan ke dalam stack adalah yang pertama dihapus.

Operasi dasar yang dapat dilakukan pada stack meliputi:

Push: Menambahkan elemen ke bagian atas stack.

Pop: Menghapus elemen dari bagian atas stack.

Top: Melihat elemen teratas tanpa menghapusnya.

IsEmpty: Memeriksa apakah stack kosong.

IsFull: Memeriksa apakah stack penuh (pada implementasi dengan kapasitas terbatas).

Size: Mengembalikan jumlah elemen dalam stack.

Peek: Melihat elemen pada posisi tertentu tanpa menghapusnya.

Clear: Menghapus semua elemen dari stack.

Search: Mencari elemen tertentu dalam stack.

Kesimpulannya, stack adalah struktur data yang efisien untuk kasus-kasus di mana elemen yang terakhir ditambahkan adalah elemen pertama yang perlu diakses atau dihapus, memanfaatkan metode LIFO untuk operasi penyimpanan dan penghapusan data. Operasi-operasi pada stack

memastikan manajemen data yang sederhana dan cepat, membuatnya ideal untuk berbagai aplikasi dalam pemrograman dan ilmu komputer.