

# **LAPORAN PRAKTIKUM**

## **MODUL VI “Stack (Tumpukan)”**



### **Disusun oleh:**

Shiva Indah Kurnia  
NIM 2211102195

### **Dosen Pengampu:**

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

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

## **TUJUAN PRAKTIKUM**

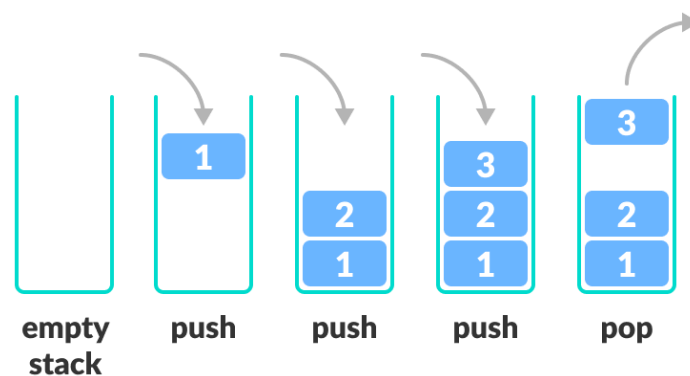
Setelah melakukan praktikum ini diharapkan mahasiswa dapat:

1. Mahasiswa diharapkan mampu menjelaskan dan memahami konsep dari Stack (Tumpukan) itu sendiri.
2. Mahasiswa diharapkan mampu mengaplikasikan Stack kedalam pemrograman C++.
3. Mahasiswa diharapkan mampu menyelesaikan studi kasus menggunakan penyelesaian program Stack.

## DASAR TEORI

### A. Struktur Data Stack

Dalam bahasa pemrograman C++, stack adalah salah satu struktur data yang berguna untuk menyimpan dan mengelola elemen-elemen data. Stack mengikuti konsep LIFO (Last In, First Out), yang berarti elemen yang terakhir dimasukkan ke dalam stack akan menjadi elemen pertama yang dikeluarkan.



Gambar 1. Struktur Data Stack

Pada dasarnya, stack terdiri dari ::

1. Push: Digunakan untuk memasukkan elemen baru ke dalam stack. Elemen baru ini akan ditempatkan di atas elemen-elemen yang sudah ada sebelumnya.
2. Pop: Digunakan untuk mengeluarkan elemen teratas (top) dari stack. Elemen ini dihapus dari stack, dan elemen yang berada di bawahnya menjadi elemen teratas baru.
3. isFull: berguna untuk memeriksa apakah ruang stack sudah penuh atau belum
4. isEmpty: berguna untuk memeriksa apakah ruang stack kosong atau tidak.
5. Peek: berguna untuk meninjau kembali data diposisi tertentu.
6. Count: berguna untuk menghitung banyak data pada stack.
7. Change: berguna untuk merubah posisi data ke posisi tertentu.
8. Display: berguna untuk menampilkan semua data pada stack
9. Destroy: menghapus atau membersihkan semua data pada stack.

Selain itu, ada juga operasi lain yang sering digunakan pada stack, yaitu:

1. Top: Mengembalikan nilai dari elemen teratas (top) pada stack tanpa menghapusnya.
2. Empty: Memeriksa apakah stack kosong atau tidak.
3. Size: Mengembalikan jumlah elemen yang ada dalam stack.

## **B. Implementasi Stack**

Dalam C++, stack dapat diimplementasikan menggunakan struktur data yang tersedia di library `<stack>`. Terdapat dua jenis implementasi stack yang umum digunakan:

- Stack menggunakan array: Stack diimplementasikan menggunakan array dengan ukuran tetap. Namun, penggunaan array ini memiliki batasan pada ukuran maksimum stack.
- Stack menggunakan linked list: Stack diimplementasikan menggunakan linked list, di mana setiap elemen stack (node) memiliki pointer yang menunjuk ke elemen berikutnya. Implementasi ini tidak memiliki batasan pada ukuran stack.

## **C. Jenis-Jenis Stack**

Terdapat beberapa jenis stack yang umum digunakan, di antaranya:

- Stack dengan tipe data dasar: Stack yang menyimpan elemen-elemen dengan tipe data dasar seperti integer, float, karakter, dll.
- Stack dengan tipe data objek: Stack yang menyimpan elemen-elemen dengan tipe data objek, seperti string, struktur, atau kelas. Objek-objek ini dapat diubah menjadi elemen-elemen stack.
- Stack dengan alokasi dinamis: Stack yang menggunakan alokasi dinamis untuk menyesuaikan ukuran stack saat runtime. Implementasinya dapat menggunakan linked list atau array dengan pengelolaan memori yang fleksibel.
- Stack dengan alokasi statis: Stack yang menggunakan alokasi statis dengan ukuran tetap yang ditentukan pada saat kompilasi. Implementasinya menggunakan array dengan ukuran tetap.

Setiap jenis stack memiliki kegunaan yang berbeda tergantung pada kebutuhan aplikasi. Pemilihan jenis stack yang tepat penting untuk memastikan efisiensi dan keandalan program.

## 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 yang dicetak
```

## Deskripsi Program

Program source code di atas merupakan implementasi stack menggunakan array dalam bahasa C++. Program ini memiliki beberapa fungsi dan operasi yang digunakan untuk memanipulasi stack yang menyimpan data buku.

Berikut beberapa operasi yang digunakan dalam program tersebut: ``isFull``, ``isEmpty``, ``pushArrayBuku(string data)``, ``popArrayBuku()``, ``peekArrayBuku(int posisi)``, ``countStack()``, ``changeArrayBuku(int posisi, string data)``, ``destroyArraybuku()``, ``cetakArrayBuku()``

Dalam fungsi program ``main()``, terdapat serangkaian operasi yang dilakukan pada stack buku:

1. ``pushArrayBuku("Kalkulus")``: Data buku "Kalkulus" dimasukkan ke dalam stack menggunakan fungsi ``pushArrayBuku()``.
2. ``pushArrayBuku("Struktur Data")``: Data buku "Struktur Data" dimasukkan ke dalam stack.
3. ``pushArrayBuku("Matematika Distit")``: Data buku "Matematika Distit" dimasukkan ke dalam stack.
4. ``pushArrayBuku("Dasar Multimedia")``: Data buku "Dasar Multimedia" dimasukkan ke dalam stack.
5. ``pushArrayBuku("Inggris")``: Data buku "Inggris" dimasukkan ke dalam stack.

Setelah itu, dilakukan operasi sebagai berikut:

1. ``cetakArrayBuku()``: Seluruh data buku dalam stack dicetak menggunakan fungsi ``cetakArrayBuku()``.
2. Cetak baris kosong ("``\n``").
3. ``isFull()``: Dilakukan pengecekan apakah stack penuh menggunakan fungsi ``isFull()``, dan hasilnya dicetak.



4. ``isEmpty()``: Dilakukan pengecekan apakah stack kosong menggunakan fungsi ``isEmpty()``, dan hasilnya dicetak.
5. ``peekArrayBuku(2)``: Melihat data buku pada posisi 2 dalam stack menggunakan fungsi ``peekArrayBuku()``. Data buku pada posisi tersebut dicetak.
6. ``popArrayBuku()``: Menghapus data buku teratas dari stack menggunakan fungsi ``popArrayBuku()``.
7. ``countStack()``: Menghitung jumlah data buku dalam stack menggunakan fungsi ``countStack()``, dan hasilnya dicetak.
8. ``changeArrayBuku(2, "Bahasa Jerman")``: Mengubah data buku pada posisi 2 menjadi "Bahasa Jerman" menggunakan fungsi ``changeArrayBuku()``.
9. ``cetakArrayBuku()``: Seluruh data buku dalam stack dicetak kembali setelah perubahan. Cetak baris kosong ("  
").
10. ``destroyArraybuku()``: Menghapus semua data buku dalam stack dan mengosongkan stack menggunakan fungsi ``destroyArraybuku()``.
11. Cetak nilai ``top``.
12. ``cetakArrayBuku()``: Seluruh data buku dalam stack dicetak setelah penghapusan.

Program ini menunjukkan contoh penggunaan fungsi-fungsi pada stack dan operasi-operasi yang dapat dilakukan dengan menggunakan stack dalam implementasi array.

## UNGUIDED

### 1. Unguided 1

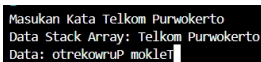
```
#include <stdio.h>
#include <conio.h>
#include <string.h>

char balik(char x[50]);
char cek(char y[50], char z[50]);

int main()
{
    char m[50], o[50];
    printf("Masukan Kata Telkom Purwokerto\n");
    printf("Data Stack Array: ");
    gets(m);
    strcpy(o,m);
    balik(o);
    getch();
}

char balik(char x[50])
{
    strrev(x);
    printf("Data: %s", x);
}
```

### Screenshot program



```
Masukan Kata Telkom Purwokerto
Data Stack Array: Telkom Purwokerto
Data: otrekowruP moklel
```

### Deskripsi program

Program di atas merupakan contoh implementasi fungsi untuk membalikkan string menggunakan fungsi `strrev()` dalam bahasa C.

Program tersebut memiliki beberapa fungsi dan operasi sebagai berikut:

1. `balik(char x[50])`: Fungsi ini digunakan untuk membalikkan string yang diberikan sebagai argumen. Di dalam fungsi ini, string `x` diubah menjadi string terbalik menggunakan fungsi `strrev()`. Kemudian, string terbalik tersebut dicetak menggunakan `printf()`.

2. ``main()``: Fungsi ``main()`` merupakan fungsi utama dalam program. Pada fungsi ini, string ``m`` digunakan untuk menyimpan input kata dari pengguna menggunakan fungsi ``gets()``. Selanjutnya, string ``o`` diisi dengan nilai yang sama dengan string ``m`` menggunakan ``strcpy()``. Kemudian, fungsi ``balik()`` dipanggil dengan argumen string ``o`` untuk membalikkan string tersebut. Akhirnya, fungsi ``getch()`` digunakan untuk menahan tampilan console agar tidak langsung tertutup setelah program selesai dijalankan.

Program ini bertujuan untuk menerima input kata dari pengguna dan kemudian mencetak kata tersebut dalam bentuk terbalik menggunakan fungsi ``strrev()``.

## 2. Unguided 2

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

using namespace std;

// Fungsi untuk membalikkan huruf-huruf dalam sebuah kalimat
menggunakan stack
string reverseLetters(const string& sentence) {
    stack<char> charStack;

    // Memasukkan setiap huruf ke dalam stack
    for (char c : sentence) {
        charStack.push(c);
    }

    // Membangun kalimat terbalik dari stack
    string reversedSentence;
    while (!charStack.empty()) {
        reversedSentence += charStack.top();
        charStack.pop();
    }

    return reversedSentence;
}

int main() {
    string sentence;
```

```

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


string reversed = reverseLetters(sentence); // Balikkan
huruf-huruf dalam kalimat dari paling akhir ke awal menggunakan
stack

cout << "Hasil pembalikan kalimat: " << reversed << endl;

return 0;
}

```

### Screenshot program



```

Masukkan kalimat (minimal 3 kata): Shiva Indah Kurnia
Hasil pembalikan kalimat: ainruk hadnI avihs

```

### Deskripsi program

Program di atas merupakan contoh implementasi fungsi untuk memeriksa apakah suatu kalimat adalah palindrom atau bukan dalam bahasa C.

Program tersebut memiliki beberapa fungsi dan operasi sebagai berikut:

1. ``balik(char x[50])``: Fungsi ini digunakan untuk membalikkan string yang diberikan sebagai argumen. Di dalam fungsi ini, string ``x`` diubah menjadi string terbalik menggunakan fungsi ``strrev()``. Namun, dalam program ini, fungsi ini tidak mencetak hasil balikkannya.
2. ``cek(char y[50], char z[50])``: Fungsi ini digunakan untuk memeriksa apakah dua string yang diberikan sebagai argumen (string asli dan string terbalik) adalah palindrom atau bukan. Di dalam fungsi ini, dilakukan perbandingan antara string ``y`` dan ``z`` menggunakan fungsi ``strcmp()``. Jika kedua string tersebut sama (nilai return ``strcmp()`` adalah 0), maka dicetak bahwa kalimat tersebut adalah palindrom. Jika tidak, dicetak bahwa kalimat tersebut bukan palindrom.
3. ``main()``: Fungsi ``main()`` merupakan fungsi utama dalam program. Pada fungsi ini, terdapat dua bagian yang hampir identik. Pertama, pengguna diminta untuk memasukkan sebuah kalimat menggunakan fungsi ``gets()`` dan kalimat tersebut disimpan dalam string ``m``. Kemudian, string ``o`` diisi dengan nilai

yang sama dengan string ``m`` menggunakan ``strcpy()``. Selanjutnya, fungsi ``balik()`` dipanggil dengan argumen string ``o`` untuk membalikkan string tersebut. Setelah itu, fungsi ``cek()`` dipanggil dengan argumen string ``m`` dan ``o`` untuk memeriksa apakah kalimat tersebut adalah palindrom atau bukan. Terakhir, fungsi ``getch()`` digunakan untuk menahan tampilan console agar tidak langsung tertutup setelah program selesai dijalankan.

Program ini bertujuan untuk menerima input sebuah kalimat dari pengguna, kemudian membalikkan kalimat tersebut dan memeriksa apakah kalimat tersebut adalah palindrom atau bukan. Hasilnya akan dicetak ke layar. Program akan mengulang proses ini dua kali, masing-masing untuk dua input kalimat yang berbeda.

## KESIMPULAN

Stack adalah struktur data yang mengikuti prinsip Last-In-First-Out (LIFO), artinya elemen terakhir yang dimasukkan ke dalam stack akan menjadi elemen pertama yang dihapus atau diakses. Stack dapat diimplementasikan menggunakan array atau linked list. Dalam contoh program C++ yang telah diberikan sebelumnya, stack diimplementasikan menggunakan array. Operasi dasar pada stack meliputi push (menambahkan elemen ke dalam stack), pop (menghapus elemen teratas dari stack), dan peek (melihat elemen teratas tanpa menghapusnya). Pengecekan kondisi stack kosong atau penuh (isEmpty dan isFull) juga merupakan operasi penting pada stack. Stack dapat digunakan dalam berbagai aplikasi, seperti pengelolaan tumpukan data, pengelolaan pemanggilan fungsi (call stack), evaluasi ekspresi matematika, dan sebagainya. Memahami konsep dan implementasi stack pada program C++ dapat membantu dalam pemecahan masalah yang melibatkan tumpukan data atau pemanggilan fungsi. Penting untuk memperhatikan batasan ukuran stack agar tidak terjadi overflow (stack penuh) atau underflow (stack kosong) yang dapat menyebabkan kesalahan atau ketidakseimbangan pada program. Selain array, dalam C++ juga terdapat struktur data stack yang sudah terdefinisi dalam library STL (Standard Template Library), yaitu `std::stack`. Penggunaan `std::stack` dapat memudahkan pengelolaan stack dalam program C++.