

**LAPORAN PRAKTIKUM**

**MODUL 6**

**STACK**



**Disusun oleh:**

**ALDI RANSI**

**NIM : 2311102328**

**Dosen Pengampu:**

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

**PROGRAM STUDI 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. *Praktikum Struktur Data dan Algoritma 2* 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;
}
```

## SCREENSHOOT PROGRAM

```
Struktur Data dan Algoritme\Code\Modul6\"tempCodeRunnerFile
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
```

### DESKRIPSI PROGRAM

Fungsi Hash: Fungsi hash adalah fungsi yang digunakan untuk menghasilkan nilai unik untuk setiap elemen dalam tabel hash. Dalam contoh ini, kita menggunakan fungsi hash untuk menghasilkan nilai unik untuk setiap elemen dalam tabel hash.

engembalikan sisa pembagian kunci dengan

### DESKRIPSI PROGRAM

pushArrayBuku(string data): Fungsi ini digunakan pada program untuk menambahkan elemen baru ke dalam stack. Jika stack sudah penuh, fungsi ini akan mencetak pesan "Data telah penuh". Sebaliknya jika belum penuh maka elemen baru akan ditambahkan ke dalam stack. popArrayBuku(): Fungsi ini digunakan untuk menghapus elemen teratas dari stack. Jika stack kosong, fungsi ini akan mencetak pesan "Tidak ada data yang dihapus". Jika stack tidak kosong, elemen teratas akan dihapus dari stack. peekArrayBuku(int posisi): Fungsi ini digunakan untuk melihat elemen pada posisi tertentu dalam stack. Jika stack kosong, fungsi ini akan mencetak pesan "Tidak ada data yang bisa dilihat". Jika stack tidak kosong, fungsi ini akan mencetak elemen pada posisi yang ditentukan.



## UNGUIDED

### 1. 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

### SOURCE CODE

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

using namespace std;

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

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

    for (i = mid + length % 2; i < length; i++) {
```

```

        if (charStack.top() != str[i]) {
            return false;
        }
        charStack.pop();
    }

    return true;
}

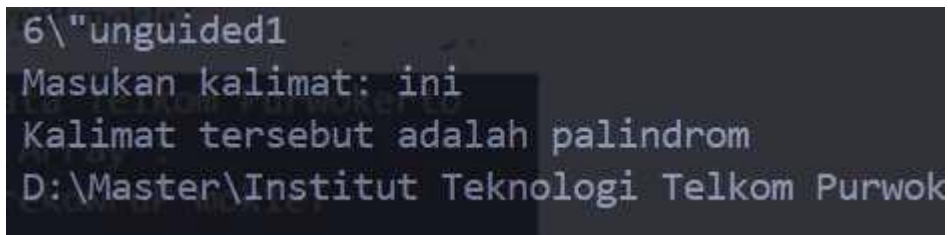
int main() {
    string input;
    cout << "Masukan kalimat: ";
    cin >> input;

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

    return 0;
}

```

## SCREENSHOOT PROGRAM



```

6\"unguided1
Masukan kalimat: ini
Kalimat tersebut adalah palindrom
D:\Master\Institut Teknologi Telkom Purwok

```

## DESKRIPSI PROGRAM

program berfungsi untuk memeriksa apakah sebuah string merupakan palindrom atau tidak. Sebuah string dikatakan palindrom jika string tersebut dapat dibaca

sama dari depan maupun dari belakang. Dalam implementasi ini, digunakan stack untuk membandingkan karakter dari string.

## 2. 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?

Contoh

Kalimat : Telkom Purwokerto

Hasil : otrekowruP mokleT . . .

### SOURCE CODE

```
#include <iostream>
#include <stack>
#include <sstream>
#include <string>
using namespace std;

string reverseWords(string sentence) {
    stack<char> charStack;
    stringstream ss(sentence);
    string word;
    string reversedSentence = "";

    while (ss >> word) {
        for (char c : word) {
            charStack.push(c);
        }

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

```
        charStack.pop();
    }

    reversedSentence += ' ';
}
if (!reversedSentence.empty()) {
    reversedSentence.pop_back();
}

return reversedSentence;
}
int main() {
    string sentence;

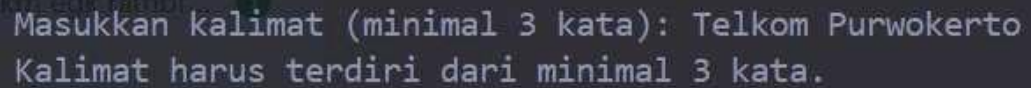
    cout << "Masukkan kalimat (minimal 3 kata): ";
    getline(cin, sentence);

    int wordCount = 0;
    stringstream ss(sentence);
    string word;
    while (ss >> word) {
        wordCount++;
    }

    if (wordCount < 3) {
        cout << "Kalimat harus terdiri dari minimal 3 kata." <<
endl;
    } else {
        string result = reverseWords(sentence);
        cout << "Data: " << result << endl;
    }

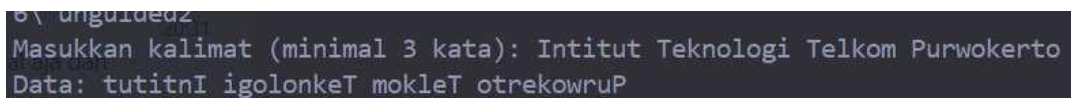
    return 0;
}
```

## SCREENSHOOT PROGRAM



```
Masukkan kalimat (minimal 3 kata): Telkom Purwokerto
Kalimat harus terdiri dari minimal 3 kata.
```

Program pada saat ketika memvalidasi minimal 3 kata



```
8\ unguided2
Masukkan kalimat (minimal 3 kata): Intitut Teknologi Telkom Purwokerto
Data: tutitnI igolonkeT mokleT otrekowruP
```

## DESKRIPSI PROGRAM

Stack: Digunakan untuk menyimpan karakter-karakter dari setiap kata sebelum dibalikkan. Stringstream: Digunakan untuk memproses string kata per kata. Fungsi reverseWords: Fungsi utama yang membalikkan huruf-huruf dalam setiap kata.

## **BAB IV**

### **KESIMPULAN**

Setelah melakukan pembelajaran mengenai Hash Table di Bahasa Pemrograman C++ berikut poin utama yang telah dipelajari :

- a) Materi stack dalam C++ mengajarkan konsep dasar struktur data LIFO dan implementasi operasi-operasi fundamental yang berkaitan dengan stack.
- b) Penerapan program stack banyak digunakan dalam berbagai aplikasi, seperti penelusuran pohon, evaluasi ekspresi, dan manajemen memori.
- c) Implementasi menggunakan array seperti pada contoh di atas memberikan gambaran dasar yang bisa dikembangkan lebih lanjut, termasuk dengan menggunakan struktur data lain seperti linked list untuk mengatasi keterbatasan kapasitas tetap.

## **DAFTAR PUSTAKA**

Karumanchi, N. (2016). Data Structures and algorithms made easy: Concepts, problems, Interview Questions. CareerMonk Publications.