

# **LAPORAN PRAKTIKUM**

## **MODUL VI**

### **STACK**



**Disusun oleh:**

**Reli Gita Nurhidayati**

**NIM: 2311102025**

**Dosen Pengampu:**

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

**PROGRAM STUDI TEKNIK INFORMATIKA**

**FAKULTAS INFORMATIKA**

**INSTITUT TEKNOLOGI TELKOM PURWOKERTO**

**PURWOKERTO**

**2024**

## **BAB 1**

### **TUJUAN PRAKTIKUM**

Praktikum ini memiliki tujuan, yaitu:

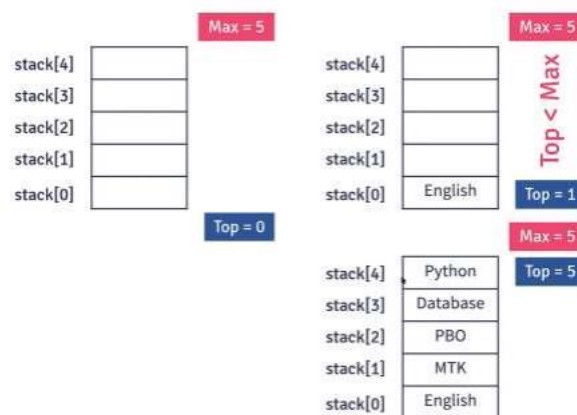
1. Mampu memahami konsep stack pada struktur data dan algoritma
2. Mampu mengimplementasikan operasi yang ada pada stack
3. Mampu memecahkan permasalahan dengan Solusi stack

## 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:

- 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.
- IsFull (Penuh): Memeriksa apakah tumpukan penuh atau tidak (terutama pada implementasi tumpukan dengan kapasitas terbatas).
- Size (Ukuran): Mengembalikan jumlah elemen yang ada dalam tumpukan.
- Peek (Lihat): Melihat nilai atau elemen pada posisi tertentu dalam tumpukan tanpa menghapusnya.
- Clear (Hapus Semua): Mengosongkan atau menghapus semua elemen dari tumpukan.
- Search (Cari): Mencari keberadaan elemen tertentu dalam tumpukan.

## BAB III

### GUIDED

#### 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

```

PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\Users\NITRO 5\Desktop\PRAK STRUKDAT\MODUL 6 PRAKSTRUKDAT> cd "c:\Users\
rFile } ; if ($?) { .\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
PS C:\Users\NITRO 5\Desktop\PRAK STRUKDAT\MODUL 6 PRAKSTRUKDAT>

```

## Deskripsi Program

Program ini mendemonstrasikan implementasi struktur data stack menggunakan array sebagai media penyimpanan datanya. Stack diimplementasikan dengan array `arrayBuku` yang berukuran 5 elemen. Variabel `top` digunakan untuk mencatat indeks elemen teratas pada stack.

Program ini menunjukkan contoh implementasi stack menggunakan array. Program ini menyediakan berbagai fungsi untuk mengelola data dalam stack, seperti menambahkan, menghapus, melihat, dan mengubah data.

## BAB IV

### 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.

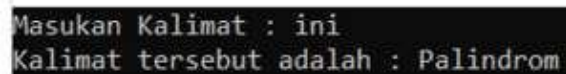
Contoh:

Kalimat : ini

Kalimat tersebut adalah polindrom

Kalimat : telkom

Kalimat tersebut adalah bukan polindrom



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

#### Source Code

```
#include <iostream>
#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 kalimat;
    cout << "Masukkan Kalimat : ";
    getline(cin, kalimat);

    if (isPalindrome(kalimat))
    {
        cout << "Kalimat tersebut adalah : Palindrom";
    }
    else
    {
        cout << "Kalimat tersebut adalah : Bukan Palindrom";
    }

    return 0;
}

```

## Screenshoot Program

The screenshot shows a terminal window with the following content:

```

PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\Users\NITRO 5\OneDrive\Documents\LAPRAK SISTEM DIGITAL\MODUL 6> cd "c:\Users\NITRO 5\OneDrive\
empCodeRunnerFile" ; if ($?) { .\tempCodeRunnerFile }
Masukkan Kalimat : radar
Kalimat tersebut adalah : Bukan Palindrom
PS C:\Users\NITRO 5\OneDrive\Documents\LAPRAK SISTEM DIGITAL\MODUL 6>

```

The screenshot shows a terminal window with the following content:

```

22  .....return false;

PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\Users\NITRO 5\OneDrive\Documents\LAPRAK SISTEM DIGITAL\MODUL 6> cd "c:\Users\
empCodeRunnerFile" ; if ($?) { .\tempCodeRunnerFile }
Masukkan Kalimat : rusak kasur
Kalimat tersebut adalah : Palindrom
PS C:\Users\NITRO 5\OneDrive\Documents\LAPRAK SISTEM DIGITAL\MODUL 6>

```

## Deskripsi Program

Program c++ di atas adalah program untuk mencetak sebuah kalimat yang dimasukkan berupa palindrom atau bukan. Program ini menggunakan konsep stack untuk melakukan pengecekan palindrom.

Program ini mendemonstrasikan implementasi struktur data stack menggunakan array sebagai media penyimpanan datanya. Stack diimplementasikan dengan array arrayBuku yang berukuran 5 elemen. Variabel top digunakan untuk mencatat indeks elemen teratas pada stack.

Program ini menunjukkan contoh implementasi stack menggunakan array. Program ini menyediakan berbagai fungsi untuk mengelola data dalam stack, seperti menambahkan, menghapus, melihat, dan mengubah data.

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 setiap kata dalam kalimat menggunakan stack
string reverseWords(string sentence) {
    stack<char> charStack;
    string reversedSentence = "", word = "";

    // Memproses setiap karakter dalam kalimat
    for (char c : sentence) {
        if (c != ' ') {
            // Jika karakter bukan spasi, masukkan ke dalam stack untuk
            membentuk sebuah kata
            charStack.push(c);
        } else {
            // Jika ditemukan spasi, keluarkan karakter dari stack dan bentuk
            sebuah kata
            while (!charStack.empty()) {
                word += charStack.top();
```

```

        charStack.pop();
    }
    // Tambahkan kata yang sudah dibalikkan ke dalam kalimat terbalik
    reversedSentence += word + " ";
    word = ""; // Reset kata untuk kata berikutnya
}

// Proses kata terakhir dalam kalimat
while (!charStack.empty()) {
    word += charStack.top();
    charStack.pop();
}
reversedSentence += word; // Tambahkan kata terakhir ke dalam kalimat
terbalik

return reversedSentence;
}

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

    string reversedWords = reverseWords(sentence);

    cout << "Kalimat setelah dibalik kata-katanya: " << reversedWords << endl;

    return 0;
}

```

## Screenshoot Program

```

PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\Users\NITRO 5\OneDrive\Documents\LAPRAK SISTEM DIGITAL\MODUL 6> cd "c:\Users\NITRO 5\OneDrive\Documents\LAPRAK SISTEM DIGITAL\MODUL 6"
empCodeRunnerFile } ; if ($?) { .\tempCodeRunnerFile }
Masukkan kalimat (minimal 3 kata): reli gita nurhidayati
Kalimat setelah dibalik kata-katanya: iler atig itayadihrun
PS C:\Users\NITRO 5\OneDrive\Documents\LAPRAK SISTEM DIGITAL\MODUL 6>

```

**Deskripsi Program**

Program ini merupakan program yang akan membalikkan urutan kata dalam sebuah kalimat yang dimasukkan dalam sebuah kalimat yang dimasukkan pengguna. Program ini menggunakan konsep stack untuk melakukan pembalikan kata.

Program ini memanfaatkan struktur data stack untuk membalikkan kata-kata dalam sebuah kalimat secara efektif. Fungsi `reverseWords()` memproses kalimat karakter per karakter dan membangun kata-kata dengan urutan terbalik. Fungsi `main()` mengatur input dan output program. Program ini dapat dimodifikasi untuk menangani berbagai jenis kalimat dan kebutuhan pemrosesan teks lainnya.

## **BAB V**

### **KESIMPULAN**

Stack (tumpukan) merupakan salah satu struktur data linier yang mengikuti prinsip Last In First Out (LIFO) atau yang terakhir masuk adalah yang pertama keluar. Stack dapat diimplementasikan dengan menggunakan array atau linked list. Dalam stack, terdapat dua operasi utama yaitu push dan pop. Operasi push digunakan untuk menambahkan elemen baru ke dalam stack, sedangkan operasi pop digunakan untuk mengeluarkan elemen dari stack.

Selain push dan pop, stack juga memiliki operasi lain seperti peek yang digunakan untuk melihat elemen paling atas tanpa mengeluarkannya dari stack, serta operasi isEmpty dan isFull untuk mengecek apakah stack dalam keadaan kosong atau penuh. Stack memiliki top yang merupakan posisi elemen teratas dalam stack.

Implementasi stack menggunakan array memiliki kelebihan yaitu akses yang cepat, namun memiliki kekurangan yaitu terbatas pada ukuran array yang telah ditentukan sebelumnya. Sedangkan implementasi stack menggunakan linked list memiliki kelebihan yaitu ukurannya dapat berkembang secara dinamis, namun akses datanya agak lebih lambat dibandingkan dengan array.

Stack memiliki banyak aplikasi dalam dunia pemrograman, seperti dalam pengecekan validitas tanda kurung, evaluasi notasi postfix atau prefix dalam ekspresi matematika, pelacakan kembali riwayat navigasi dalam browser web, serta dalam implementasi algoritma tertentu seperti depth-first search (DFS) pada teori graf.

Dalam pengembangan perangkat lunak, stack sering digunakan sebagai bagian dari struktur data lain yang lebih kompleks, seperti dalam implementasi mesin abstrak untuk menjalankan bahasa pemrograman tertentu atau dalam pengembangan kompiler dan interpreter untuk menangani operasi parsing dan eksekusi kode.

Secara keseluruhan, stack merupakan struktur data yang sederhana namun sangat penting dalam dunia pemrograman. Pemahaman yang baik tentang konsep dan operasi stack akan membantu dalam memahami dan mengimplementasikan berbagai algoritma dan struktur data yang lebih kompleks.

## DAFTAR PUSTAKA

- [1] Herawati, E. 2012. Stack. Diakses 20 Mei 2024, dari <https://repository.unikom.ac.id/38844/1/Bab%20VIII%20-%20Stack.pdf>
- [2] Hidayah, AK, 2023. Struktur Data Dengan Phyton. Diakses 20 Mei 2024, dari [https://books.google.co.id/books?hl=id&lr=&id=pnHXEAAAQBAJ&oi=fnd&pg=PA15&dq=stack+dalam+struktur+data&ots=-NI7I9-ET0&sig=FrUBNMZrEdMxvQa0KIQ1ih11gWA&redir\\_esc=y#v=onepage&q=stack%20dalam%20struktur%20data&f=false](https://books.google.co.id/books?hl=id&lr=&id=pnHXEAAAQBAJ&oi=fnd&pg=PA15&dq=stack+dalam+struktur+data&ots=-NI7I9-ET0&sig=FrUBNMZrEdMxvQa0KIQ1ih11gWA&redir_esc=y#v=onepage&q=stack%20dalam%20struktur%20data&f=false)