LAPORAN PRAKTIKUM STRUKTUR DATA DAN PEMROGRAMAN

MODUL VI

STUCK

Disusun Oleh :

Ahmadan Syaridin

2311102002

IF-11-A

Dosen Pengampu :

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

PROGRAM STUDI TEKNIK INFORMATIKA

FAKULTAS INFORMATIKA

INSTITUT TEKNOLOGI TELKOM

PURWOKERTO

2024

BAB I

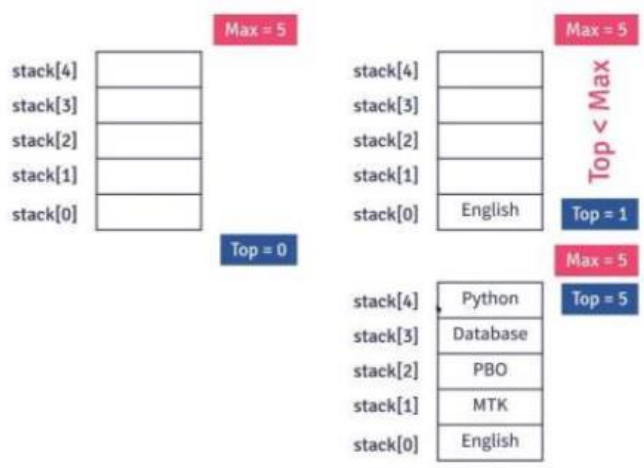
A. TUJUAN

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

B. 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.
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

C. GUIDED

```
#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;
}
```

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
```

DESKRIPSI PROGRAM

Program tersebut menggunakan array untuk mengimplementasikan stack yang menyimpan lima buku. Fungsi `pushArrayBuku` menambahkan buku ke dalam stack jika belum penuh, sedangkan `popArrayBuku` menghapus buku dari stack jika tidak kosong. Fungsi `peekArrayBuku` menampilkan buku pada posisi tertentu dari atas stack, `countStack` mengembalikan jumlah buku dalam stack, dan `changeArrayBuku` mengganti buku pada posisi tertentu. Fungsi `destroyArraybuku` menghapus semua buku dari stack, sementara `cetakArrayBuku` mencetak semua buku dari atas ke bawah. Program ini juga memeriksa apakah stack dalam keadaan penuh atau kosong, serta menampilkan berbagai operasi yang dilakukan pada stack buku tersebut.

BAB IV

D. UNGUIDED

UNGUIDED 1

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

bool isPalindrome(string str) {
    stack<char> charStack;
    string cleanedStr;

    // Menghapus spasi dan tanda baca dari kalimat, serta
mengubah ke huruf kecil
    for (char c : str) {
        if (isalnum(c)) {
            cleanedStr += tolower(c);
        }
    }

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

    // Membandingkan karakter pada stack dengan karakter asli
dari belakang
    for (char c : cleanedStr) {
        if (c != charStack.top()) {
            return false; // Tidak palindrom
        }
        charStack.pop();
    }

    return true; // Palindrom
}

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

    if (isPalindrome(kalimat)) {
        cout << "Kalimat tersebut merupakan palindrom." <<
endl;
    } else {
        cout << "Kalimat tersebut bukan palindrom." << endl;
    }

    return 0;
}
```

OUTPUT

```
 .\unguided1modul6 }
Masukkan kalimat: TELKOM
Kalimat tersebut bukan palindrom.
```

```
 .\unguided1modul6 }
Masukkan kalimat: ini
Kalimat tersebut merupakan palindrom.
```

DESKRIPSI PROGRAM

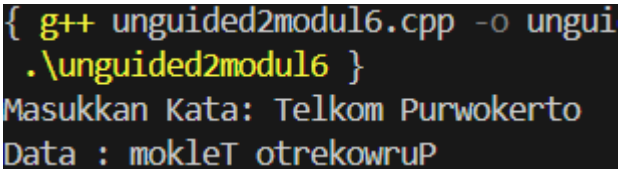
Program tersebut mengecek apakah sebuah kalimat merupakan palindrom dengan mengabaikan spasi, tanda baca, dan perbedaan huruf besar-kecil. Pertama, kalimat dibersihkan dari karakter non-alfanumerik dan diubah ke huruf kecil. Karakter-karakter yang tersisa dimasukkan ke dalam stack. Program kemudian membandingkan karakter dari awal kalimat yang sudah dibersihkan dengan karakter yang diambil dari stack untuk memeriksa kesesuaian simetris. Jika semua karakter cocok, kalimat tersebut merupakan palindrom; jika tidak, kalimat bukan palindrom. Hasilnya ditampilkan kepada pengguna.

UNGUIDED 2

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

void reverseSentence(string sentence) {
    stack<char> charStack;
    string word = "";

    cout << "Data : ";

    for (char c : sentence) {
        if (c == ' ') {
            // Reverse and print the word in the stack
            while (!charStack.empty()) {
                cout << charStack.top();
                charStack.pop();
            }
            cout << " "; // Add space after each word
        } else {
            charStack.push(c);
        }
    }

    // Process the last word after the loop
    while (!charStack.empty()) {
        cout << charStack.top();
        charStack.pop();
    }
    cout << endl;
}
```

```
int main() {
    string kalimat;
    cout << "Masukkan Kata: ";
    getline(cin, kalimat);
    reverseSentence(kalimat);
    return 0;
}
```

OUTPUT



DESKRIPSI PROGRAM

Program ini meminta pengguna untuk memasukkan sebuah kalimat. Kemudian, kalimat tersebut diolah karakter demi karakter. Ketika program menemukan spasi, kata yang telah terkumpul sejauh ini dalam stack akan dibalikkan dan dicetak. Proses ini diulang sampai seluruh kalimat selesai diproses. Setelah semua kata di dalam kalimat diproses, kata terakhir yang masih tersimpan dalam stack akan dibalikkan dan dicetak. Akibatnya, kalimat yang dicetak adalah kalimat asli yang urutan kata-katanya telah dibalik, tetapi urutan karakter dalam setiap kata tetap tidak berubah. Program ini menunjukkan penggunaan stack untuk membalikkan urutan kata dalam kalimat dengan memanfaatkan sifat Last In First Out (LIFO) dari stack untuk mencapai tujuan ini.

BAB V

E. KESIMPULAN

Stack adalah sebuah struktur data yang memiliki prinsip Last In First Out (LIFO), artinya elemen terakhir yang dimasukkan akan menjadi yang pertama kali dihapus. Dalam pemrograman, stack sangat penting karena menyediakan operasi dasar seperti Push untuk menambahkan elemen, Pop untuk menghapus elemen, dan Top untuk melihat elemen teratas tanpa menghapusnya. Hal ini memungkinkan pengelolaan data secara efisien. Selain itu, stack juga menyediakan operasi lain seperti IsEmpty, IsFull, Size, Peek, Clear, dan Search yang memberikan fleksibilitas tambahan dalam penggunaannya. Memiliki pemahaman yang baik tentang stack dan operasi-operasinya sangat penting dalam pemrograman untuk mengimplementasikan algoritma dengan efektif dan menyelesaikan berbagai masalah dengan cepat dan efisien.

BAB VI

F. DAFTAR PUSTAKA

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