LAPORAN PRAKTIKUM
STRUKTUR DATA DAN ALGORITME

MODUL 6

STACK

Disusun Oleh :

Nama : Fatkhurrohman Purnomo

NIM : 21102125

Dosen Pengampu

Ipam Fuaddina Adam, S.T., M.Kom.

PROGRAM STUDI TEKNIK INFORMATIKA

FAKULTAS INFORMATIKA

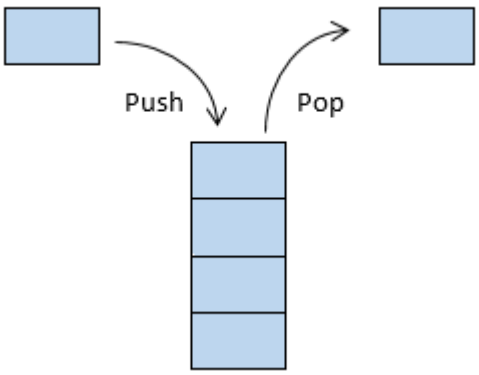
INSTITUT TEKNOLOGI TELKOM PURWOKERTO

PURWOKERTO

2022

A. Dasar Teori

Stack (Tumpukan) merupakan sebuah kumpulan data yang diletakkan atau disusun di atas data lainnya. Stack menerapkan prinsip LIFO (Last In First Out) yakni elemen yang terakhir disimpan (masuk) dalam stack, akan menjadi elemen yang pertama diambil (keluar). Untuk meletakkan sebuah elemen pada bagian atas (top) dari stack, maka dilakukan operasi Push. Sedangkan untuk memindahkan sebuah elemen teratas dalam stack, maka dilakukan operasi Pop.



Representasi Stack dapat dilakukan menggunakan Array atau Linked List. Kedua representasi mempunyai keunggulan dan kelemahan. Pada bab ini Stack direpresentasikan dengan kedua cara tersebut. Representasi Stack dengan Array dapat dilakukan dengan asumsi bahwa elemen maksimal suatu Stack tidak boleh melebihi maksimum banyak elemen atau ukuran Array. Untuk menjaga hal ini terjadi maka harus ada penunjuk atau suatu nama yang mencatat posisi puncak elemen Stack. Dengan Array, Stack juga dapat disajikan dengan Single Stack dan Double Stack.

Ref:

Modul 6 : STACK

Stack pada C++ - nblognlife

Stack pada C++ (8/10) - pintarkom

B. Guided

1. Program stack 1

```
// Nama : Fatkhurrohman Purnomo
// NIM : 21102125

#include <iostream>
#define MAX 5 //menetapkan nilai konstanta MAX = 5
(ukuran stack)
using namespace std;

///PROGRAM STACK DENGAN STRUCTURE
//Deklarasi Stack
struct Stack{
    int top;
    int data[MAX]; //array dengan ukuran MAX (5)
}myStack;

void initStack(){//inisialisasi stack
    myStack.top = -1;
}

// mengecek apakah tumpukan kosong
int isEmpty(){
    if (myStack.top == -1)
    return 1;
    else
    return 0;
}

// mengecek apakah tumpukan penuh
int isFull(){
    if (myStack.top == MAX - 1){
        return 1;
    }
    else {
        return 0;
    }
}

//Insert Data (Push)
void push(int data){
    if (isEmpty() == 1){
        myStack.top++;
        myStack.data[myStack.top] = data;
```

```
        cout << " Data " << data << " telah dimasukkan"
<< endl;
    }
    else if (isFull() == 0){
        myStack.top++;
        myStack.data[myStack.top] = data;
        cout << " Data " << data << " telah dimasukkan"
<< endl;
    }
    else{
        cout << " Stack penuh!" << endl;
    }
}

//Hapus Data (Pop)
void pop(){
    if (isEmpty() == 0){
        myStack.top--;
        cout << " Data teratas terambil" << endl;
    }
    else{
        cout << " Stack masih kosong!" << endl;
    }
}

//Tampilkan Top
void top(){
    if (isEmpty() == 0){
        cout << " " << myStack.data[myStack.top] << endl;
    }
    else{
        cout << " Stack masih kosong!" << endl;
    }
}

//Tampilkan Stack
void display(){
    if (isEmpty() == 0){
        for (int i = myStack.top; i>=0; i--){
            cout << " " << myStack.data[i] << endl;
        }
    }
    else{
        cout << " Stack masih kosong!" << endl;
    }
```

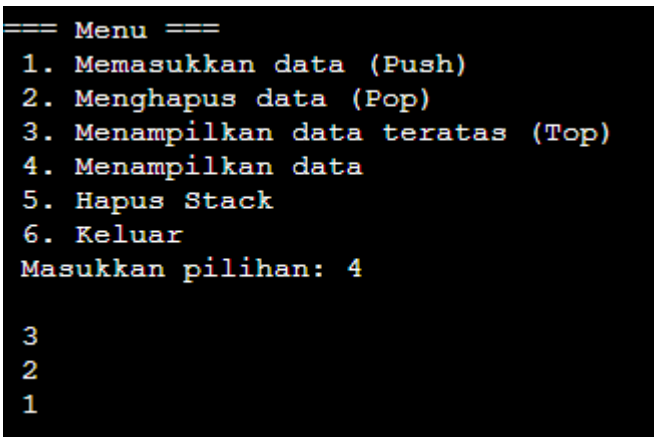
```
}

//Hapus Stack
void clear(){
    myStack.top = -1;
    cout << " Stack berhasil dihapus" << endl;
}

int main(){
    int pilihan, nilai;
    initStack();

    // menu
    do{
        system("cls");
        cout << "=== Menu ===" << endl;
        cout << " 1. Memasukkan data (Push)" << endl;
        cout << " 2. Menghapus data (Pop)" << endl;
        cout << " 3. Menampilkan data teratas (Top)" <<
endl;
        cout << " 4. Menampilkan data" << endl;
        cout << " 5. Hapus Stack" << endl;
        cout << " 6. Keluar" << endl;
        cout << " Masukkan pilihan: ";
        cin >> pilihan;
        cout << endl;

        // untuk input data
        if (pilihan==1){
            cout << " Masukkan data: ";
            cin >> nilai;
            push(nilai);
        }
        else if (pilihan==2){
            pop();
        }
        else if (pilihan==3){
            top();
        }
        else if (pilihan==4){
            display();
        }
        else if (pilihan==5){
            clear();
        }
```

```
        else{
           return 0;
        }

        cout << endl;
        system("pause");
    }while (pilihan != 6);
    return 0;
}
```

Deskripsi:

Yang pertama adalah membuat Deklarasi dan inisialisasi dari stack itu sendiri. Kemudian membuat fungsi untuk mengecek apakah stack sedang kosong atau penuh. Dilanjut membuat fungsi push and pop untuk hapus dan menambah data. Fungsi top, menampilkan data paling atas, fungsi menampilkan stack, dan yang terakhir fungsi untuk menghapus seluruh stack. Dilanjut membuat menu pilihan bagi user/pengguna.

Output:



Input diatas menggunakan 1, 2, 3. Kemudian saat ditampilkan maka yang pertama adalah yang terakhir di inputkan.

2. Program Stack Dengan Array

```
// Nama : Fatkhurrohman Purnomo
// NIM : 21102125

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

```
// PROGRAM STACK DENGAN ARRAY
// Deklarasi Global
const int n = 5; //konstanta ukuran array
int stack[n];
int top = -1;

// Mengecek apakah tumpukan kosong
int isEmpty(){
    if (top == -1)
        return 1;
    else
        return 0;
}

// Mengecek apakah tumpukan penuh
int isFull(){
    if (top == n-1)
    return 1;
    else
    return 0;
}

// Insert Data (Push)
void push(int data){
    if (isEmpty() == 1){
        top++;
        stack[top] = data;
        cout << " Data " << data << " telah dimasukkan"
<< endl;
    }
    else if (isFull() == 0){
        top++;
        stack[top] = data;
        cout << " Data " << data << " telah dimasukkan"
<< endl;
    }
    else{
        cout << " Stack penuh!" << endl;
    }
}

// Hapus Data (Pop)
void pop(){
    if (isEmpty() == 0){
```

```
        top--;
        cout << " Data teratas terambil" << endl;
    }
    else{
        cout << " Stack masih kosong!" << endl;
    }
}

// Tampilkan Stack
void display(){
    if (isEmpty() == 0){
        for (int i = top; i>=0; i--){
            cout << " " << stack[i] << endl;
        }
    }
    else{
        cout << " Stack masih kosong!" << endl;
    }
}

// Hapus Stack
void clear(){
    top = -1;
    cout << " Stack berhasil dihapus" << endl;
}
int main(){
    int pilihan, nilai;

    // Menu
    do{
        system("cls");
        cout << "=== Menu ===" << endl;
        cout << " 1. Memasukkan data (Push)" << endl;
        cout << " 2. Menghapus data (Pop)" << endl;
        cout << " 3. Menampilkan data" << endl;
        cout << " 4. Hapus Stack" << endl;
        cout << " 5. Keluar" << endl;
        cout << " Masukkan pilihan: ";
        cin >> pilihan;
        cout << endl;

        if (pilihan==1){
            cout << " Masukkan data: ";
            cin >> nilai;
            push(nilai);
```

```
        }
        else if (pilihan==2){
            pop();
        }
        else if (pilihan==3){
            display();
        }
        else if (pilihan==4){
            clear();
        }
        else{
           return 0;
        }

        cout << endl;
        system("pause");
    }while (pilihan != 5);
    return 0;
}
```

Dekripsi:

Yang pertama adalah membuat Deklarasi dan inisialisasi dari stack itu sendiri. Kemudian membuat fungsi untuk mengecek apakah stack sedang kosong atau penuh. Dilanjut dengan membuat fungsi untuk melakukan push and pop, untuk menambah dan menghapus isi stack. Lalu fungsi menampilkan stack, dan yang terakhir fungsi untuk menghapus semua stack. Dilanjut membuat menu pilihan bagi user/pengguna.

Output:

```
=== Menu ===
1. Memasukkan data (Push)
2. Menghapus data (Pop)
3. Menampilkan data teratas (Top)
4. Menampilkan data
5. Hapus Stack
6. Keluar
Masukkan pilihan: 4

3
2
1
```

Input diatas menggunakan 1, 2, 3. Kemudian saat ditampilkan maka yang pertama adalah yang terakhir di inputkan.

3. Program Stack Dengan Linked List

```
// Nama : Fatkhurrohman Purnomo
// NIM : 21102125

#include <iostream>
using namespace std;

// PROGRAM STACK DENGAN SINGLE LINKED LIST
struct Node{
    int data;
    Node *link;
};

Node *top = NULL;

// Mengecek apakah tumpukan kosong
int isEmpty(){
    if(top == NULL)
        return 1;
    else
        return 0;
}

// Insert Data (Push)
void push (int nilai){
    Node *ptr = new Node();
    ptr->data = nilai;
    ptr->link = top;
    top = ptr;
    cout << " Data " << nilai << " telah dimasukkan" <<
endl;
}

// Hapus Data (Pop)
void pop (){
    if (isEmpty() == 1)
        cout << " Stack masih kosong!" << endl;
    else{
        Node *ptr = top;
        top = top->link;
```

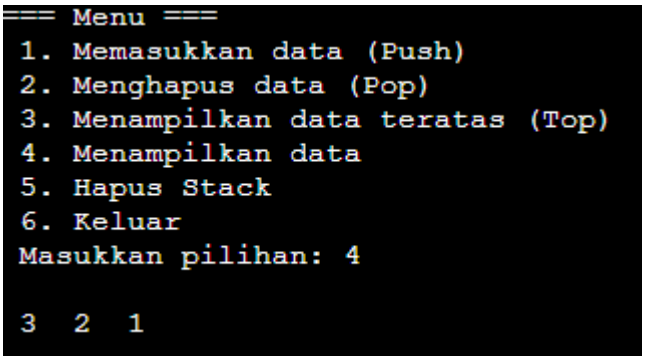
```
        cout << " Data " << ptr->data << " terambil" <<
endl;
        delete(ptr);
    }
}

// Tampilkan data teratas (Top)
void showTop(){
    if (isEmpty() == 1)
        cout << " Stack masih kosong!" << endl;
    else
        cout << " Data teratas (top) : " << top->data <<
endl;
}

// Hapus Stack
void clearStack(){
    Node *bantu, *hapus;
    bantu = top;
    while (bantu != NULL){
        hapus = bantu;
        bantu = bantu->link;
        delete hapus;
    }

    top = NULL;
    cout << "Stack berhasil terhapus!" << endl;
}

// Tampilkan Stack
void displayStack(){
    if (isEmpty() == 1)
        cout << " Stack masih kosong!" << endl;
    else{
        Node *temp = top;
        while(temp != NULL){
            cout << " " << temp->data << " ";
            temp = temp->link;
        }
        cout << endl;
    }
}

int main(){
    int pilihan, nilai;
```

```
    // menu
    do{
        system("cls");
        cout << "=== Menu ===" << endl;
        cout << " 1. Memasukkan data (Push)" << endl;
        cout << " 2. Menghapus data (Pop)" << endl;
        cout << " 3. Menampilkan data teratas (Top)" <<
endl;
        cout << " 4. Menampilkan data" << endl;
        cout << " 5. Hapus Stack" << endl;
        cout << " 6. Keluar" << endl;
        cout << " Masukkan pilihan: ";
        cin >> pilihan;
        cout << endl;

        if (pilihan==1){
            cout << " Masukkan data: ";
            cin >> nilai;
            push(nilai);
        }
        else if (pilihan==2){
            pop();
        }
        else if (pilihan==3){
            showTop();
        }
        else if (pilihan==4){
            displayStack();
        }
        else if (pilihan==5){
            clearStack();
        }
        else{
            return 0;
        }

        cout << endl;
        system("pause");
    }while (pilihan != 6);
    return 0;
}
```

Deskripsi:

Yang pertama adalah membuat struct single linked list, selanjutnya hampir sama seperti yang sebelumnya. Fungsi mengecek stack, fungsi pop and push, fungsi menampilkan data, dan hapus semua stack. Dilanjut membuat menu pilihan bagi user/pengguna.

Output:



Input diatas menggunakan 1, 2, 3. Kemudian saat ditampilkan maka yang pertama adalah yang terakhir di inputkan.

C. Tugas (Unguided)

1. **Buatlah program untuk melakukan pembalikan (reversing) terhadap kalimat dengan menggunakan stack!**

```
// Nama : Fatkhurrohman Purnomo
// NIM : 21102125

#include <iostream>
#include <bits/stdc++.h>
using namespace std;

// stack
class Stack {
    public:
    int top;
    unsigned cap;
    char* arr;
};

// inisilasi stack
Stack* createst(unsigned cap) {
    Stack* st = new Stack();
    st->cap = cap;
```

```
    st->top = -1;
    st->arr = new char[(st->cap * sizeof(char))];
    return st;
}

// cek apakah stack kosong
int isEmpty(Stack* st)
{ return st->top == -1; }

// cek apakah stack penuh
int isFull(Stack* st)
{ return st->top == st->cap - 1; }

// push
void push(Stack* st, char character)
{
    if (isFull(st))
        return;
    st->arr[++st->top] = character;
}

// pop
char pop(Stack* st) {
    if (isEmpty(st))
        return -1;
    return st->arr[st->top--];
}

// reverse stack
void reversestring(char string[])
{
    // Membuat stack
    int strsm = strlen(string);
    Stack* st = createst(strsm);

    // Melakukan push
    for (int i = 0; i < strsm; i++)
        push(st, string[i]);

    // Melakukan pop
    for (int i = 0; i < strsm; i++)
        cout << pop(st);
}

int main(){
```

```
    char kalimat[100];
    cout << " Masukkan kalimat: ";
    cin.getline(kalimat, sizeof(kalimat));
    int panjang = strlen(kalimat);

    cout << " Hasil setelah dibalik: ";
    reversestring(kalimat);

    return 0;
}
```

Deskripsi:

Yang pertama adalah melakukan inisialisasi stack, lalu mengecek stack kosong atau stack penuh. Membuat fungsi pop and push untuk menambah dan menghapus data di stack. Lalu melakukan reverse atau membalik input yang dimasukan, kemudian langsung ditampilkan.

Output:

```
Masukkan kalimat: Struktur Data
Hasil setelah dibalik: ataD rutkurtS
```

1. **Buatlah program untuk melakukan konversi notasi infix ke notasi postfix dengan menggunakan stack!**

```
// Nama : Fatkhurrohman Purnomo
// NIM : 21102125

#include <bits/stdc++.h>
using namespace std;

// Fungsi untuk mengembalikan satuan operator
int back(char satuan)
{
    if (satuan == '^')
        return 3;
    else if (satuan == '/' || satuan == '*')
        return 2;
    else if (satuan == '+' || satuan == '-')
        return 1;
    else
        return -1;
```

```
}

// Fungsi utama untuk mengubah ekspresi infiks
// ke ekspresi postfix
void mengubah(string data)
{

    stack<char> st; // Stack untuk menampung operator
    string result; // Hasil akhir

    for (int i = 0; i < data.length(); i++) {
        char c = data[i];

        // Jika karakter yang dipindai adalah
        // sebuah operan, tambahkan ke string output.
        if ((c >= 'a' && c <= 'z') || (c >= 'A' && c <=
'Z')
            || (c >= '0' && c <= '9'))
            result += c;

        // Jika karakter yang dipindai adalah sebuah
kurung buka, maka:
        else if (c == '(')
            st.push('(');

        // Jika karakter yang dipindai adalah sebuah
kurung tutup, maka:
        else if (c == ')') {
            while (st.top() != '(') {
                result += st.top();
                st.pop();
            }
            st.pop();
        }
        else {
            while (!st.empty()
                && back(data[i]) <= back(st.top())) {
                if (c == '^' && st.top() == '^')
                    break;
                else {
                    result += st.top();
                    st.pop();
                }
            }
            st.push(c);
```

```
        }
    }

    // Tambahkan semua karakter yang ada di stack ke
output
    while (!st.empty()) {
        result += st.top();
        st.pop();
    }

    cout << result << endl;
}

// Fungsi utama
int main()
{
    string exp;
    cout << " Infix     : ";
    cin >> exp;

    cout << " Postfix   : ";
    mengubah(exp);
    return 0;
}
```

Deskripsi:

Yang pertama adalah membuat fungsi pembalik operator. Lalu fungsi untuk merubah dari infix ke postfix. Dalam fungsi merubah ini membuat atau Deklarasi stack, lalu membuat pengulangan for yang didalamnya ada percabangan untuk menentukan isi dari inputan, lalu merubahnya menjadi postfix. Jika hasil sudah didapat maka akan ditampilkan di layar. Dalam fungsi utama, melakukan Deklarasi, lalu user memasukkan input, memanggil fungsi mengubah dengan inputan tadi. Dan hasilnya akan tampil di layar.

Output:

```
Infix     : A+B-C
Postfix   : AB+C-
```

```
Infix     : (A+B)*(C-D)
Postfix   : AB+CD-*
```

D. Kesimpulan

1. Bisa membuat stack
2. Belajar lebih dalam stack dan struct
3. Saya lebih mahir dalam menggunakan bahasa C++
4. Saya bisa melakukan problem solving bagi program yang error
5. Lebih paham dalam membuat program
6. Melatih daya pikir, imajinasi, dan langkah-langkah dalam membuat program