LAPORAN PRAKTIKUM STRUKTUR DATA

MODUL 7 (VII)
STACK

Disusun Oleh :
NAMA : Gien Darrel Adli
NIM : 10312430008

Dosen
FAHRUDIN MUKTI WIBOWO

PROGRAM STUDI STRUKTUR DATA
FAKULTAS INFORMATIKA
TELKOM UNIVERSITY PURWOKERTO
2025

A. Dasar Teori

Algoritma merupakan fondasi dalam pembuatan program komputer. Secara sederhana, algoritma adalah serangkaian langkah logis dan sistematis yang disusun untuk menyelesaikan suatu masalah. C++, berfungsi sebagai alat untuk mengimplementasikan algoritma tersebut agar dapat dimengerti dan dieksekusi oleh komputer. C++ sering digunakan sebagai bahasa pengantar untuk mempelajari konsep pemrograman dasar karena strukturnya yang terorganisir dan kemampuannya untuk menangani operasi tingkat rendah.

Untuk membangun logika dalam program sesuai dengan algoritma yang dirancang, C++ menyediakan struktur kontrol, yang terbagi menjadi dua jenis utama:

Struktur Percabangan (Conditional): Digunakan untuk pengambilan keputusan, di mana program akan menjalankan blok kode tertentu jika suatu kondisi terpenuhi. Struktur ini mencakup if-else untuk mengevaluasi kondisi boolean dan switch-case untuk memilih blok kode berdasarkan nilai dari sebuah variabel.

Struktur Perulangan (Looping): Digunakan untuk mengeksekusi blok kode yang sama secara berulang kali selama kondisi tertentu masih terpenuhi. C++ menyediakan tiga jenis perulangan utama: for, while, dan do-while, yang masing-masing memiliki karakteristik penggunaan yang spesifik dalam implementasi algoritma.

B. Guided (berisi screenshot source code & output program disertai penjelasannya)

Guided 1 (stack.cpp)

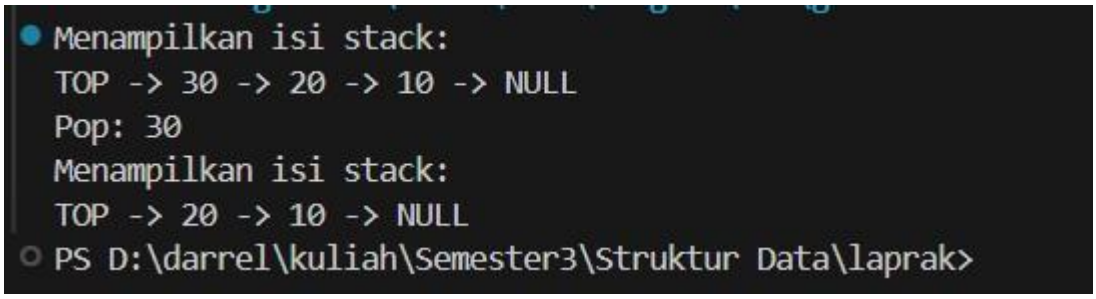
```
#include <iostream>
using namespace
std; struct Node {
int data;
   Node *next;
};

bool isEmpty(Node *top) {
return top == nullptr;
}

void push(Node *&top, int data) {
Node *newNode = new Node();
newNode->data = data;
newNode->next = top;     top =
newNode;
}
```

```
int pop(Node *&top) {     if (isEmpty(top)) {
cout << "Stack kosong. tidak bisa pop!" << endl;
return 0;
    }
    int poppedData = top-
>data;     Node *temp = top;
top = top->next;

    delete temp;
return poppedData;
}

void show(Node *top) {     if
(isEmpty(top)) {         cout << "Stack
kosong." << endl;         return;
    }
    cout << "TOP -> ";
    Node *temp = top;

    while (temp != nullptr) {
cout << temp->data << " -> ";
temp = temp->next;
    }
    cout << "NULL" << endl;
}

int main() {
    Node *stack = nullptr;

    push(stack, 10);
push(stack, 20);
push(stack, 30);

    cout << "Menampilkan isi stack:" << endl;
show(stack);

    cout << "Pop: " << pop(stack) << endl;

    cout << "Menampilkan isi stack: "<< endl;
show(stack);

    return 0;
       }
```

Deskripsi:

Program tersebut mengimplementasikan struktur data stack menggunakan linked list satu arah. Setiap elemen stack disimpan dalam node yang berisi data dan pointer ke node berikutnya. Operasi push menambahkan elemen baru ke bagian atas stack dengan membuat node baru dan menunjukannya sebagai top, sedangkan operasi pop menghapus dan mengembalikan elemen paling atas; jika stack kosong, program menampilkan pesan kesalahan. Fungsi show digunakan untuk menampilkan seluruh isi stack dari elemen paling atas hingga bawah. Pada fungsi main, stack awalnya kosong lalu tiga data dimasukkan, isi stack ditampilkan, satu elemen di-pop, kemudian isi stack ditampilkan kembali.

Output Guided :



C. Unguided/Tugas (berisi screenshot source code & output program disertai penjelasannya)

stack.h

```
#ifndef STACK_H

#define STACK_H


typedef int infotype;


struct Stack

{

   infotype info[20];     int top;

};


void CreateStack(Stack &S);
```

```
void Push(Stack &S, infotype x);

infotype Pop(Stack &S); void

printInfo(Stack S); void

balikStack(Stack &S);


void pushAscending(Stack &S, infotype x);


#endif
```

Deskripsi:

File `stack.h` tersebut mendefinisikan sebuah struktur stack berbasis array dengan kapasitas 20 elemen bertipe `int`. Di dalamnya terdapat deklarasi struct `Stack` yang memiliki array `info` sebagai tempat penyimpanan data dan variabel `top` sebagai penunjuk posisi elemen teratas stack. Selain itu, file ini hanya berisi deklarasi fungsifungsi yang akan diimplementasikan di file lain, yaitu fungsi untuk membuat stack baru, menambahkan elemen, menghapus elemen, mencetak isi stack, membalik urutan stack, serta fungsi `pushAscending` untuk menyisipkan elemen secara terurut naik.

stack.cpp

```
  #include <iostream>
#include "stack.h" using
namespace std;

void CreateStack(Stack &S)
{
    S.top = -1;
}

void Push(Stack &S, infotype x)
{     if (S.top <
19)
    {
        S.top++;
        S.info[S.top] =
x;     }     else
    {
        cout << "Stack penuh!" << endl;
    }
}
```

infotype Pop(Stack &S)

```
{    if (S.top >= 0)
   {
     infotype x = S.info[S.top];        S.top--;        return x;    }    else
   {
      cout << "Stack kosong!" << endl;        return -1;
   }
}

void printInfo(Stack S)
{    if (S.top == -1)
   {
      cout << "Stack kosong!" << endl;        return;
   }
   cout << "[TOP] ";    for (int i = S.top; i >= 0; i--)
   {
      cout << S.info[i] << " ";
   }
   cout << endl;
}

void balikStack(Stack &S)
{
   Stack Temp;
   CreateStack(Temp);

   while (S.top != -1)
   {
      Push(Temp, Pop(S));
   }

   S = Temp;
}

void pushAscending(Stack &S, infotype x)
{    if (S.top == 19)
```

```
    {
        cout << "Stack penuh!" << endl;
return;
    }

    Stack Temp;
    CreateStack(Temp);

    // Pindahkan elemen yang lebih besar dari x ke Temp
while (S.top != -1 && S.info[S.top] > x)
    {
        Push(Temp, Pop(S));
    }

    // Masukkan x
    Push(S, x);

    // Kembalikan elemen dari Temp
while (Temp.top != -1)
    {
        Push(S, Pop(Temp));
    }
}
```

Deskripsi:

File `stack.cpp` mengimplementasikan operasi stack berbasis array. Stack diinisialisasi dengan `CreateStack`, penambahan data dilakukan dengan `Push`, penghapusan dilakukan dengan `Pop`, isi stack bisa dilihat melalui `printInfo`, fungsi `balikStack` membalik urutan elemen stack, dan `pushAscending` menyisipkan elemen baru sambil menjaga agar data tetap terurut menaik.

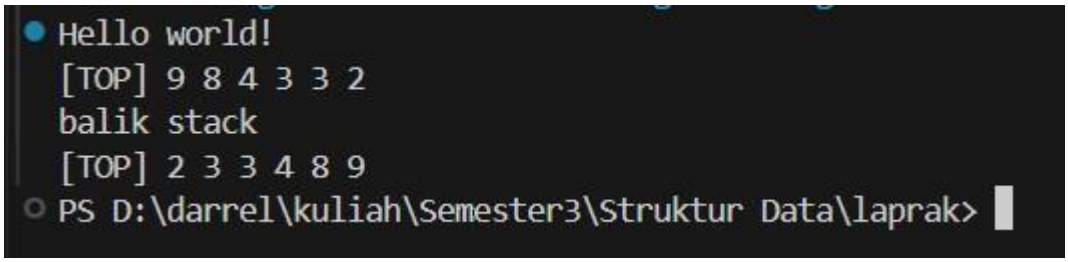
main.cpp

```
#include <iostream> #include
"stack.h"
#include "stack.cpp"

using namespace std;

int main()
{
    cout << "Hello world!" << endl;

    Stack S;
    CreateStack(S);

    pushAscending(S, 3);
pushAscending(S, 4);
pushAscending(S, 8);
pushAscending(S, 2);
pushAscending(S, 3);
pushAscending(S, 9);

    printInfo(S);

    cout << "balik stack" << endl;
balikStack(S);     printInfo(S);

    return 0;
}
```

Deskripsi:

Program `main.cpp` ini berfungsi sebagai pengujian dari implementasi stack. Program membuat sebuah stack, kemudian memasukkan beberapa nilai menggunakan `pushAscending` sehingga elemen-elemen tersimpan dalam urutan menaik. Setelah itu isi stack ditampilkan dengan `printInfo`, lalu stack dibalik menggunakan `balikStack` dan hasilnya ditampilkan kembali.

Screenshots Output Unguided:



D. Kesimpulan

Kesimpulan, keseluruhan program yang terdiri dari `stack.h`, `stack.cpp`, dan `main.cpp` membangun dan mengelola struktur data stack menggunakan array dengan kapasitas tetap. Program tidak hanya menyediakan operasi dasar seperti membuat stack, push, pop, dan menampilkan isi stack, tetapi juga dilengkapi fitur tambahan yaitu membalik isi stack serta memasukkan elemen baru sambil menjaga urutan stack tetap ascending. Melalui fungsi-fungsi ini, program menunjukkan bagaimana konsep stack (LIFO) dapat dimodifikasi dan dikembangkan untuk kebutuhan tertentu sambil tetap mempertahankan struktur dasarnya.

E. Referensi

Kaswar, A. B., & Zain, S. G. (2021). Mudah Belajar Pemrograman Dasar C++. Syiah Kuala University Press.

Hanief, S., Jepriana, I. W., & Kom, S. (2020). Konsep Algoritme dan Aplikasinya dalam Bahasa Pemrograman C++. Penerbit Andi.

Imamuddin, A., & Sobarnas, M. A. (2021). PEMBELAJARAN JARAK JAUH PEMROGRAMAN DASAR MENGGUNAKAN BAHASA C++ UNTUK UMUM: SEBUAH PROGRAM PENGABDIAN KEPADA MASYARAKAT. BEMAS: Jurnal Bermasyarakat, 1(2), 59-67.