Modul 4. Stack

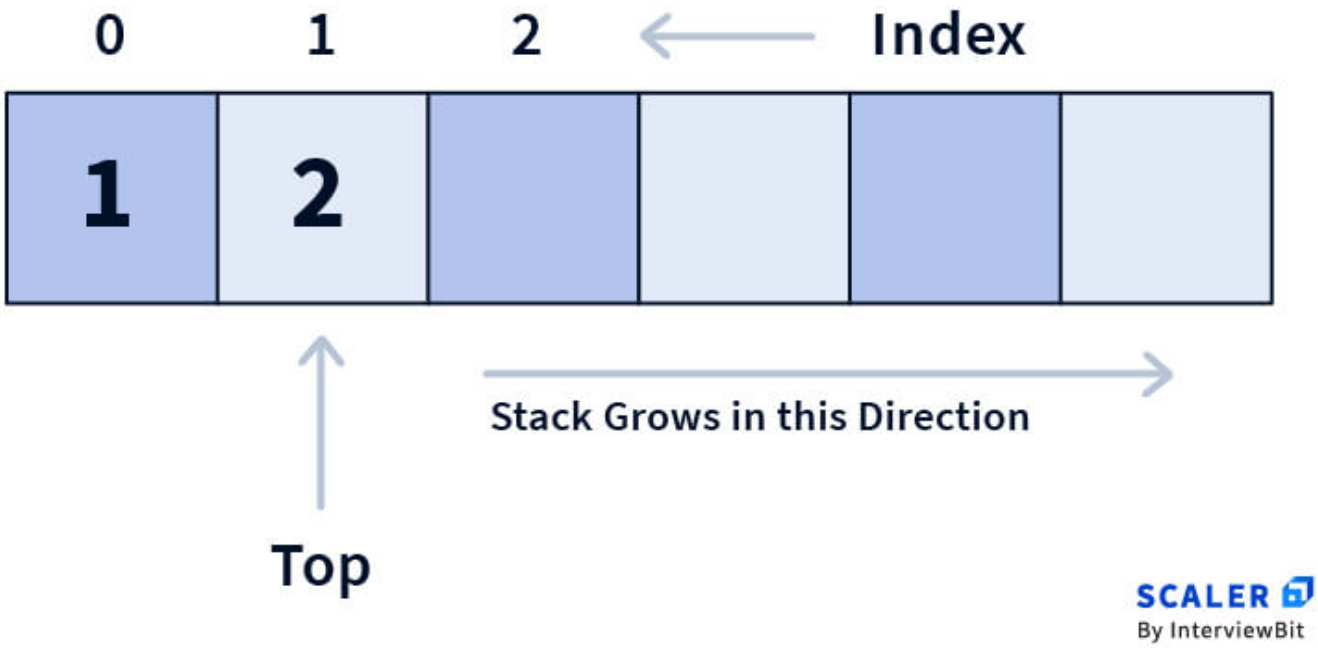
Jump to bottom

berliana-amelia edited this page 3 weeks ago · 17 revisions

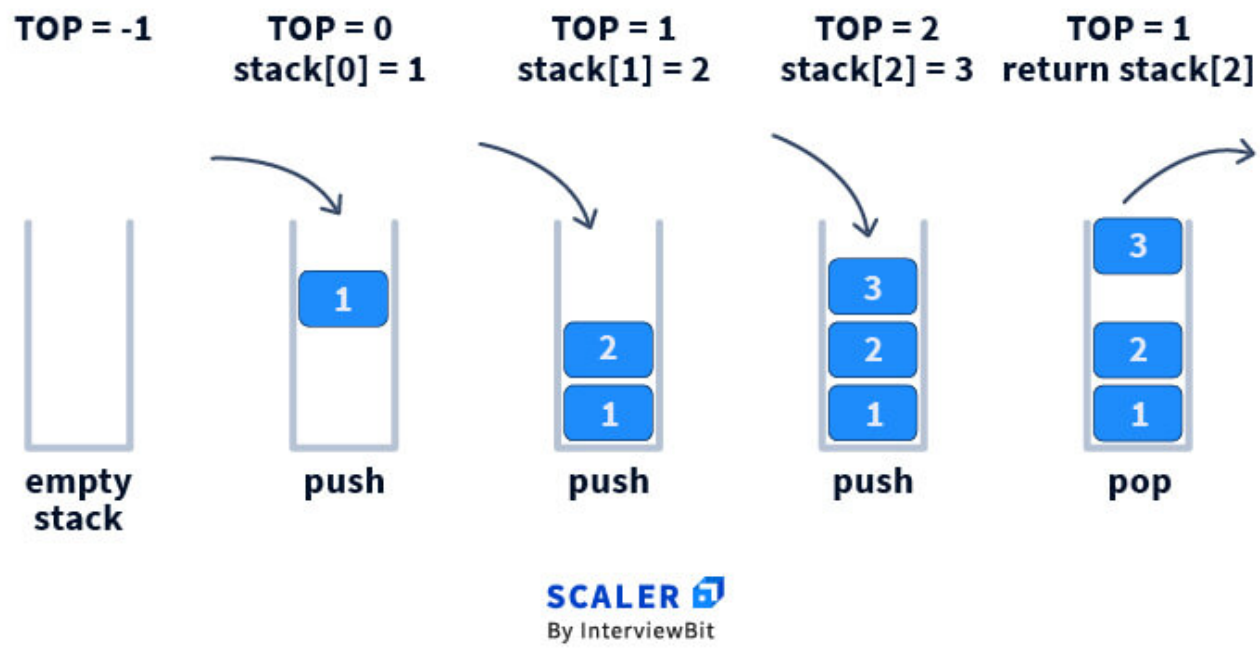
Teori Singkat

Stack adalah struktur data yang menyimpan kumpulan elemen secara linear. **Stack mengikuti prinsip Last-In-First-Out (LIFO)**, yang berarti elemen terakhir yang ditambahkan ke stack akan menjadi elemen pertama yang dihapus. Stack memiliki dua operasi utama: push, yang menambahkan elemen ke bagian atas stack, dan pop, yang menghapus elemen teratas dari stack.

Dalam ilmu komputer, stack banyak digunakan untuk menyimpan frame data pemanggilan fungsi di dalam memori. Ketika sebuah fungsi dipanggil, sebuah frame baru ditambahkan ke stack, yang berisi variabel lokal dan data terkait fungsi tersebut. Ketika fungsi selesai, frame-nya dihapus dari stack dan kontrol dikembalikan ke fungsi yang memanggilnya.



Di dalam bahasa C, stack dapat diimplementasikan menggunakan **array atau linked list**. Implementasi berbasis array menggunakan array berukuran tetap untuk menyimpan elemen-elemen stack, sementara implementasi berbasis linked list menggunakan linked list yang dialokasikan secara dinamis untuk menyimpan elemen-elemen tersebut.



Implementasi stack dalam bahasa C melibatkan definisi struct yang merepresentasikan stack, dan penulisan fungsi-fungsi yang melakukan operasi push, pop, peek, dan operasi stack lainnya pada struct tersebut.

Praktikum Stack dengan Array

Apabila Gambar bermasalah bisa akses di sini

1. Membuat Stack dengan Struct

```c
#include <stdio.h>
#include <stdlib.h>

#define MAX_STACK_SIZE 5

// Mendefinisikan Stack dengan Struct
typedef struct Stack {
    int data[MAX_STACK_SIZE];
    int top;
} Stack;
```

2. Menginsisialisasi Stack Kosong

```c
// membuat stack kosong
void createEmptyStack(Stack *stack) {
    stack->top = -1;
}
```

3. Membuat Fungsi isEmpty, isFull, dan Peek

```c
// Menegecek apakah Stack Kosong atau Tidak
int isEmpty(Stack *stack) {
    return stack->top == -1;
}

// Mengecek apakah stack penuh atau tidak
int isFull(Stack *stack) {
    return stack->top == MAX_STACK_SIZE - 1;
}
// Mengecek Elemen Top dalam Stack tanpa Menghapus elemen tersebut dari stack
```

4. Operasi dalam Stack (Push & Pop)

```c
// Memasukkan Elemen ke dalam Stack
void push(Stack *stack, int value) {
    if (isFull(stack)) {
        printf("Stack Penuh\n");
        return;
    }
    stack->top++;
    stack->data[stack->top] = value;
}

// Menarik / Menghapus elemen paling atas (top)
int pop(Stack *stack) {
    if (isEmpty(stack)) {
        printf("Stack Kosong\n");
        return -1;
    }
    int value = stack->data[stack->top];
    stack->top--;
    return value;
}
```

5. Membuat Fungsi Print Stack

```
// Menampilkan Elemen dalam Stack
void printStack(Stack *stack) {
    if (isEmpty(stack)) {
        printf("\nStack Kosong");
        return;
    }
    printf("\nStack: ");
    for (int i = stack->top; i >= 0; i--) {
        printf("%d ", stack->data[i]);
    }
}
```

6. Membuat Fungsi Main

```
// Main function for testing the stack
int main() {
   Stack varStack;
    createEmptyStack(&varStack);
    int choice, value;
    while (1) {
        printStack(&varStack);
        printf("\nMenu Stack\n");
        printf("1. Push\n");
        printf("2. Pop\n");
        printf("3. Peek\n");
        printf("4. Keluar\n");
        printf("Pilihan anda: ");
        scanf("%d", &choice);
        switch(choice) {
            case 1:
                printf("Masukkan nilai yang akan ditambahkan: ");
                scanf("%d", &value);
                push(&varStack,value);
                break;
            case 2:
                printf("Elemen yang dihapus adalah %d\n", pop(&varStack));
                break;
            case 3:
                printf("Elemen Top adalah %d\n", peek(&varStack));
                break;
            case 4:
                printf("Terima kasih!\n");
                return 0;
            default:
                printf("Error: Pilihan tidak valid!\n");
        }
    }
}
```

Latihan

> Note : Kerjakan di saat praktikum

Buatlah sebuah stack untuk membalikan karakter dalam String. Contoh :

```
Input   : hallo
Output  : ollah
```

Kumpulkan di sini.

Asisten Praktikum Algoritma Struktur Data Informatika 2023

Pages 8

Clone this wiki locally

```
https://github.com/fzl-22/Algoritma-dan-Struktur-Data_Informatika.wiki.git
```