Nama: Asehpryanto Rari Parinding

Nim: H071211065

Kelas: Struktur Data B

**Stack And Queue**

**\*Stack**

Stack didefinisikan sebagai struktur data yang digunakan untuk menyimpan kumpulan objek. Objek tersebut dapat terdiri dari item individu yang ditambahkan dan disimpan dalam tumpukan menggunakan operasi push. Sebaliknya, item dapat dihapus dari Stack dengan operasi pop. Ketika sebuah objek ditambahkan ke dalam Stack, maka keberadaan objek tersebut ditempatkan di atas semua item yang dimasukkan sebelumnya. Saat sebuah item dihapus, item tersebut dapat dihapus dari atas atau bagian bawah dari Stack. Stack di mana item dihapus lewat bagian atas disebut sebagai LIFO (Last In, First Out). Sedangkan, Stack sebaliknya adalah FIFO (First In, First Out) yang menghapus item dari bagian bawah.

Konsep Stack sebagai pengorganisasian data umumnya dikenal dalam dua bentuk, yaitu wadah akses acak (array) dan berurutan (linked list). Array merupakan wadah akses dimana elemen apa saja dapat diakses secara instan. Sedangkan linked list merupakan wadah akses yang hanya bisa diakses secara berurutan.

**Karakteristik Stack**

• Stack digunakan pada banyak algoritma yang berbeda seperti Tower of Hanoi, Tree traversal, rekursi dll.

• Stack diimplementasikan dengan struktur data array atau linked list.

• Mengikuti prinsip operasi Last In First Out, yaitu elemen yang dimasukkan pertama akan muncul terakhir dan sebaliknya.

• Penyisipan dan penghapusan terjadi di satu ujung yaitu dari atas tumpukan.

• Apabila ruang memori yang dialokasikan untuk struktur data stack sudah penuh namun masih dilakukan operasi penyisipan elemen maka akan terjadi stack overflow.

• Apabila struktur data tidak memiliki elemen data atau kosong, namu

**Fungsi dan Kegunaan Stack**

• Struktur data stack digunakan dalam evaluasi dan konversi ekspresi aritmatika. Proses ini banyak dipakai untuk program kompiler.

• Stack digunakan dalam pemrograman rekursi.

• Digunakan untuk pemeriksaan tanda kurung.

• Stack digunakan dalam manajemen memori.

• Dipakai untuk memproses pemanggilan sebuah fungsi.

**Jenis-jenis Stack** Berdasarkan kemampuan menyimpan data, struktur data stack dapat dibagi menjadi 2 jenis, yaitu register stack dan memory stack.

1. Register stack Register stack merupakan stack yang hanya mampu menampung data dalam jumlah yang kecil. Kedalaman maksimum pada register stack cenderung dibatasi karena ukuran unit memorinya sangat kecil dibandingkan dengan memory stack.

2. Memory stack Pada stack jenis ini, kedalaman dari stack cukup fleksibel dan mampu menangani dalam dalam skala yang lebih besar dibandingkan jenis sebelumnya.

**\*Queue**

Antrian (Queue) merupakan kumpulan data yang mana penambahan elemen hanya bias dilakukan pada suatu ujung yaitu rear/tail/belakang, dan penghapusan dilakukan melalui ujung yang lainnya yaitu front/head/depan. Antrian disebut FIFO (First In First Out) yaitu elemen yang lebih dulu disisipkan merupakan elemen yang akan lebih dulu diambil.

**Karakteristik Queue**

• Queue adalah struktur FIFO (First In First Out).

• Untuk menghapus elemen terakhir dari Queue, semua elemen yang dimasukkan sebelum elemen tersebut harus dihilangkan atau dihapus.

• Queue adalah daftar berurutan dari elemen-elemen dengan tipe data yang serupa.

**Operasi-operasi Dasar pada Queue**

• Enqueue: Menambahkan elemen ke akhir antrian

• Dequeue: Menghapus elemen dari depan antrian

• IsEmpty: Memeriksa apakah antrian kosong

• IsFull: Memeriksa apakah antrian sudah penuh

• Peek: Mendapatkan nilai bagian depan antrian tanpa menghapusnya

• Initialize: Membuat antrian baru tanpa elemen data (kosong)

**Fungsi dan Kegunaan Queue**

• Queue banyak digunakan untuk menangani lalu lintas (traffic) situs web.

• Membantu untuk mempertahankan playlist yang ada pada aplikasi media player

• Queue digunakan dalam sistem operasi untuk menangani interupsi.

• Membantu dalam melayani permintaan pada satu sumber daya bersama, seperti printer, penjadwalan tugas CPU, dll.

• Digunakan dalam transfer data asinkronus misal pipeline, IO file, dan socket.