Nama : Stefany Frans Sarira

NIM : H071211049

Kelas : Struktur Data A

Stack dan Queue

Struktur data merupakan aspek yang cukup penting diketahui oleh setiap orang yang memulai bidang pemrograman dan komputer. Struktur data didefinisikan sebagai cara menyimpan dan mengatur data secara terstruktur pada sistem komputer atau database sehingga lebih mudah diakses. Secara teknis, data dalam bentuk angka, huruf, simbol, dan lainnya ini diletakkan dalam kolom-kolom dan susunan tertentu. Struktur data Stack dan Queue adalah contohnya. Berikut adalah penjelasan dari Stack dan Queue.

1. **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.

   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.

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, namun tetap dilakukan operasi penghapusan maka akan terjadi stack underflow

Operasi-operasi Dasar pada Stack

- Push: Menyisipkan elemen ke bagian atas stack
- Pop: Menghapus elemen atas dari stack
- IsEmpty: Memeriksa apakah stack kosong
- IsFull: Memerika apakah stack sudah penuh
- Peek: Mendapatkan nilai elemen teratas tanpa menghapusnya

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.

Kelebihan Menggunakan Stack

- Manajemen data yang efisien: Stack membantu mengelola data berdasarkan prinsip operasi LIFO yang tidak bisa dilakukan dengan linked list dan array.
- Manajemen fungsi yang efisien: Ketika suatu fungsi dipanggil, variabel lokal disimpan dalam stack, dan secara otomatis dihancurkan setelah dikembalikan.
- Kontrol atas memori: Stack memungkinkan kita untuk mengontrol bagaimana memori dialokasikan dan tidak dialokasikan.
- Manajemen memori cerdas: Stack secara otomatis membersihkan objek.
- Tidak mudah rusak: Stack tidak mudah rusak, oleh karena itu stack cenderung lebih aman dan dapat diandalkan.
- Tidak mengizinkan pengubahan ukuran variabel: Variabel pada stack tidak dapat diubah ukurannya.

Kekurangan Menggunakan Stack

- Ukuran memori terbatas: Memori pada stack cukup terbatas.
- Kemungkinan stack overflow: Terlalu banyak membuat objek di stack dapat meningkatkan risiko stack overflow.
- Akses acak tidak dimungkinkan: Dalam stack, akses data secara acak tidak bisa dilakukan. Data yang dapat diakses adalah data yang berada pada elemen atas.
- Dapat menyebabkan fungsi tidak tedefinisi: Ketika penyimpanan variabel akan ditimpa, kadang-kadang akan menyebabkan perilaku fungsi atau program yang tidak terdefinisi.
- Penghentian yang tidak diinginkan: Jika stack berada di luar memori maka dapat menyebabkan penghentian yang tidak normal.

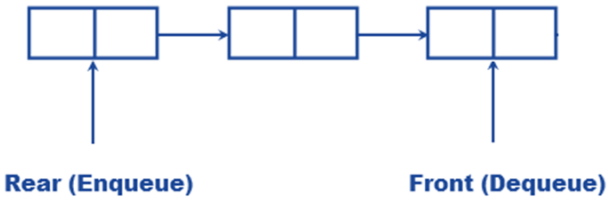
2. QUEUE

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.

Secara umum ada 4 jenis struktur data queue, meliputi:

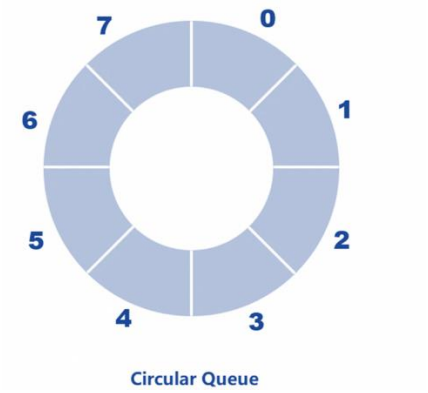
1. Simple Queue

Simple queue adalah struktur data queue paling dasar di mana penyisipan item dilakukan di simpul belakang (**rear** atau **tail**) dan penghapusan terjadi di simpul depan (**front** atau **head**).



2. Circular Queue

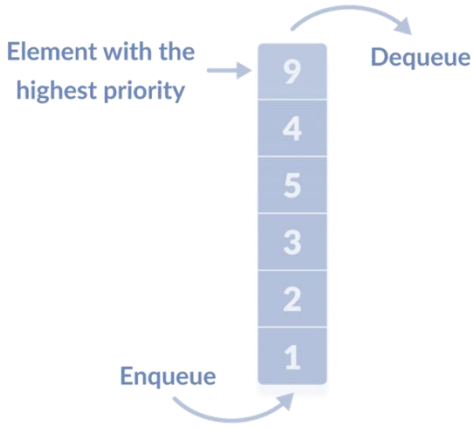
Pada circular queue, simpul terakhir terhubung ke simpul pertama. Queue jenis ini juga dikenal sebagai Ring Buffer karena semua ujungnya terhubung ke ujung yang lain. Penyisipan terjadi di akhir antrian dan penghapusan di depan antrian.



Circular Queue

3. Priority Queue

Priority Queue adalah strruktur data queue dimana simpul akan memiliki beberapa prioritas yang telah ditentukan. Simpul dengan prioritas terbesar akan menjadi yang pertama dihapus dari antrian. Sedangkan penyisipan item terjadi sesuai urutan kedatangannya.



Aplikasi priority queue antara lain algoritma jalur terpendek Dijkstra, algoritma prim, dan teknik kompresi data seperti kode Huffman.

4. Double-Ended Queue (Dequeue)

Dalam double-ended queue (dequeue), operasi penyisipan dan penghapusan dapat terjadi di ujung depan dan belakang dari queue.

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.

Kelebihan Queue

- Data dalam jumlah besar dapat dikelola secara efisien.
- Operasi seperti penyisipan dan penghapusan dapat dilakukan dengan mudah karena mengikuti aturan masuk pertama keluar pertama.
- Queue berguna ketika layanan tertentu digunakan oleh banyak konsumen.
- Queue cepat untuk komunikasi antar-proses data.
- Queue dapat digunakan dalam implementasi struktur data lainnya.

Kekurangan Queue

- Operasi seperti penyisipan dan penghapusan elemen dari tengah cenderung banyak memakan waktu.
- Dalam queue konvensional, elemen baru hanya dapat dimasukkan ketika elemen yang ada dihapus dari antrian.
- Ukuran maksimum antrian harus ditentukan sebelumnya.

Referensi:

- https://repository.unikom.ac.id/41990/1/pertemuan%203.pdf
- https://www.trivusi.web.id/2022/07/struktur-data-stack.html
- https://www.trivusi.web.id/2022/07/struktur-data-queue.html
- https://www.techiedelight.com/queue-implementation-in-java/
- https://www.techiedelight.com/stack-implementation-in-java/