ZAYID MUSIAFA

BUKU AJAR

STRUKTUR DATA DAN IMPLEMENTASI ALGORITMA

(SDIA)

Bahasa Pemrograman
Python Java C C++

BUKU AJAR
STRUKTUR DATA DAN IMPLEMENTASI ALGORITMA

(SDIA)

Bahasa Pemrograman Python Java C C++

ZAYID MUSIAFA, S.KOM., M.KOM.

Buku Ajar
STRUKTUR DATA DAN IMPLEMENTASI ALGORITMA

(SDIA)

Bahasa Pemrograman Python Java C C++

ISBN: -
Ukuran Standar unesco 15,5 cm x 23 cm
Cetakan I, _____ 2021

Penulis:
Zayid Musiafa, S.Kom., M.Kom.

Editor:
-
Proofreader
-

Lay out
Zayid Musiafa, S.Kom., M.Kom.

Desain sampul:
Zayid Musiafa, S.Kom., M.Kom.

Penerbit:
Diterbitkan Pertama Kali Oleh:
- Indonesia

Instrumen Buku ini

Pembelajar (The Learner); pembelajar /pem·bel·a·jar/ *n* orang yg mempelajari;
Kamus Besar Bahasa Indonesia, Edisi Ketiga, Cet.2, Jakarta : Balai Pustaka., 2002.

Student Centered Learning (SCL) dan Kaidah KKO Taksonomi Bloom

Student Centered Learning (SCL) proses paradigma konsep transformasi pembelajaran dari yang berpusat pada pengajar kepada berpusat pada pembelajar. Pembelajaran tidak tergantung pada pengajar, karena akses informasi (knowledge) lebih luas dan lengkap, sehingga pembelajar dapat belajar kapan saja dan dimana saja.
Peran pembelajar menjadi lebih aktif mempelajari materi pembelajaran, memperoleh ilmu pengetahuan atau informasi secara mandiri tidak mengandalkan pemberian dari pengajar, disesuaikan pula dengan keinginan dan minatnya terhadap materi pembelajaran.
Penulis menawarkan KKO dalam; Tujuan Pembelajaran; Capaian Pembelajaran; beserta contoh implementasi bab pembelajaran ke RPS;
Bagi pengajar dan pembelajar dapat menggunakan materi pembelajaran yang ruang lingkup (scope) dan urutan (sekuensnya) sudah sistematis terjadwal (RPS), sehingga bagi pengajar bisa menilai seberapa jauh materi pembelajaran tersebut disajikan, dan bagi pembelajar dapat menilai seberapa jauh materi pembelajar tersebut dapat dipelajari dan dikuasainya.

Materi – Praktek – Contoh Latihan.

Harapan penulis kepada pembelajar dapat mengulang atau mempelajari kembali materi pembelajaran yang telah dipelajarinya setiap saat dan dimana saja sesuai dengan keperluannya. Pembelajar dapat menilai materi pembelajaran mana yang telah dikuasainya dan terus dilanjutkan, atau materi pembelajaran mana yang belum

dikuasainya sehingga perlu dipelajari ulang (direview) sampai dikuasainya (kompeten) atau dikonsultasikan melakukan komunikasi dengan mudah dan cepat, baik kepada pembelajar dengan pembelajar, pembelajar dengan pengajar atau tutor sehingga capaian pembelajaran lebih optimal.

Dedikasi

Pembelajar-pengajar bahasa Struktur Data dan Algoritma. Tanpa minat dan dedikasi dari Anda buku ini menjadi sebuah keniscayaan. Semoga dengan menggunakan buku ini Anda memperoleh ilmu, manfaat..

Ijtihad ini sebagai `Cicilan sedekah' untuk Bangsa Indonesia demi kemajuan, pengembangan pedidikan dan ilmu pengetahuan dan untuk anak^{2}ku-cucu^{2}ku-penerus^{2}ku.

PRAKATA

Alhamdulillah Puji dan syukur kepada Tuhan Yang Maha Esa, karena atas berkat dan rahmat-Nya sehingga ijtihad ini dapat diselesaikan dengan baik. Buku ini membawa Pembelajar memecahkan masalah pada kasus melalui pendekatan sederhana dan praktis saat belajar Struktur Data beserta implementasinya menggunakan empat bahasa pemrograman Python, Java, C, dan C++.

Penulis berharap dari hadirnya buku ini Pembelajar dapat memperoleh manfaat, pula sebagai penunjang Program Pendidikan Medeka Balajar-Kampus Merdeka, berfaedah bagi perkembangan ilmu pengetahuan dalam mempelajari-menerapkan Struktur Data dan Algoritma di Indonesia.

Last but not least, tentu saja buku ini masih jauh dari baik dan perlu disempurnakan karena keterbatasan dimensi pengetahuan, ruang dan kapasitas waktu penulis. Kepada pembelajar, penulis berharap untuk terus mendalami (upgrade keilmuan) dari narasumber yang lebih kompeten (pakar). Untuk para pengajar pengampu mata kuliah Struktur Data maka dengan hadirnya buku ini dapat dijadikan sebagai referensi ataupun sebagai Buku Ajar.

Akhirnya, dengan diterbitkannya buku ini, sebagaimana kata pepatah, "tak ada gading yang tak retak", karena itu, penulis mengharapkan kritik, koreksi, dan masukan dari pembaca, untuk perbaikan dan menambah faedah-manfaat buku ini di masa mendatang. Penulis mengucapkan terima kasih atas setiap kritik, koreksi, dan masukan dari pembaca. Terima kasih juga penulis ucapkan kepada Penerbit yang bersedia menerbit-sebarkan buku ini.

Banjarmasin, 08 April 2022
Zayid Musiafa, S.Kom., M.Kom.

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB I
Pengantar Algoritma dan Struktur Data

Ulasan ringkas yang akan Pembelajar dapatkan dalam Bab ini;
Tujuan Pembelajaran
[C1] Pembelajar mempunyai pengetahuan dasar (*basic science*) tentang definisi Struktur Data, Algoritma.
[C2] Mengintrepretasikan bahasa pemrograman, Tingkat Bahasa Pemrograman, Paradigma Pemrograman.

Outcome Pembelajaran bagi Pembelajar
[C2] mempunyai kemampuan untuk menjelaskan definisi Struktur Data dan Algoritma.
[C4] Mampu memahami dan menggunakan tipe data.
[C2,C6] Mengerti tentang tipe data dan perbedaan kegunaanya. Menguraikan dan menyimpulkan dalam bentuk resuman, projek sederhana.

Perlunya Skill bahasa pemrograman bagi mahasiswa computing
Berikut ilustrasi dari laman yang di unggah dalam web milik Prof. Romi Satria Wahono tokoh IT terkemuka di Indonesia, semoga menjadi 'trigger' positif dalam pembelajaran IT; halaman lengkap dapat Pembelajar lihat di source berikut:

https://romisatriawahono.net/2009/04/13/wajibnya-skill-coding-bagi-mahasiswa-computing/

> *Mas Romi, saya mahasiswa jurusan teknik informatika, semester akhir dengan peminatan software engineering. Karena saya lemah di coding, kira-kira nanti kesulitan ga ya untuk mengerjakan tugas akhir? (Taufik, Universitas Swasta di Jakarta)*
>
> *Inna lillahi wa inna ilaihi rajiun*, segera lakukan taubat dan perbanyak istighfar 🙂 Jurusan teknik informatika semester akhir, peminatan software engineering pula, ga bisa coding? Selama ini kemana aja om? 😕

Apa itu Algoritma?

Dalam istilah pemrograman komputer, algoritma adalah seperangkat instruksi yang terdefinisi dengan baik untuk memecahkan masalah tertentu. Dan, algoritma adalah kumpulan langkah-langkah untuk memecahkan masalah tertentu. Mempelajari struktur dan algoritme data memungkinkan kita menulis program komputer yang efisien dan optimal. Dibutuhkan satu set input dan menghasilkan output yang diinginkan. Sebagai contoh, Algoritma untuk menambahkan dua angka, konsepnya sebagai berikut;

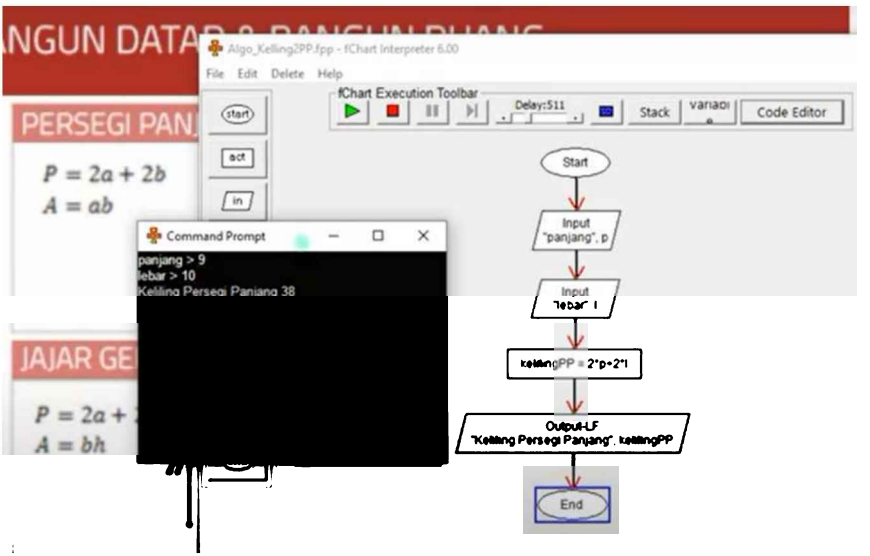
1. Ambil dua input angka
2. Tambahkan angka menggunakan + operator
3. Tampilkan hasilnya

Detail langkah-langkahnya:

Step 1: Start
Step 2: Declare variables num1, num2 and sum.
Step 3: Read values num1 and num2.
Step 4: Add num1 and num2 and assign the result to sum.
sum←num1+num2
Step 5: Display sum
Step 6: Stop

Contoh lainnya; menentukan urutan dari tiga angka terbesar

```
Step 1: Start
Step 2: Declare variables a,b and c.
Step 3: Read variables a,b and c.
Step 4: If a > b
          If a > c
            Display a is the largest number.
          Else
            Display c is the largest number.
        Else
          If b > c
            Display b is the largest number.
          Else
            Display c is the greatest number.
Step 5: Stop
```



Gambar 1 Flowchart Hitung Keliling Persegi Panjang

Cara memvisualkan Algoritma salah satunya dengan menggunakkan Flowchart, Contoh berikut Flowchart Perhitungan Keliling Persegi Panjang. Source : https://youtu.be/2Y606FaFuhk

Tingkat bahasa pemrograman[1]

1. Bahasa Pemrograman Tingkat Rendah (Umumnya, ini mengacu pada kode mesin atau bahasa assembly)
2. Bahasa Pemrograman Tingkat Sedang (C, Pascal, Fortran)
3. Bahasa Pemrograman Tingkat Tinggi (Java, C++, C#).

Paradigma Pemrograman

Sudut pandang dan style pemrograman berhubungan dengan bagaimana sebuah masalah diformulasikan dalam bahasa pemrograman

- Functional Programming: Urutan fungsi secara sekuensial (Scheme, Lisp) [2]

[1] https://id.wikipedia.org/wiki/Bahasa_pemrograman, 2019
[2] https://id.wikipedia.org/wiki/Pemrograman_fungsional, 2019

- Procedural Programming: Pemecahan masalah berdasarkan prosedural kerja yg terkumpul dalam unit pemrograman bernama fungsi (C, Pascal)[3]
- Object-Oriented Programming: berdasarkan konsep "objek", yang dapat berisi data, dalam bentuk field atau dikenal juga sebagai atribut; serta kode, dalam bentuk fungsi/prosedur atau dikenal juga sebagai metode (method). (Java, C#, C++)[4]

Perlunya bahasa pemrograman

- Komputer bekerja seperti switching dan hanya mengenali 0 dan 1
- Manusia tidak (paham) berbicara dengan bahasa 0 dan 1
- Perlu bahasa pemrograman yang dapat menjadi perantara percakapan antara komputer dan manusia
- Bahasa pemrograman diubah ke dalam bahasa yang dipahami oleh komputer dengan menggunakan interpreter atau kompiler

Mengapa Struktur Data?

Pengetahuan tentang struktur data membantu Anda memahami cara kerja setiap struktur data. Dan, berdasarkan itu Anda dapat memilih struktur data yang tepat untuk proyek Anda. Ini membantu Anda menulis kode memori dan waktu yang efisien. Pemrograman adalah semua tentang struktur data dan algoritma. Struktur data digunakan untuk menyimpan data sementara algoritma digunakan untuk memecahkan masalah menggunakan data tersebut. Buku Ajar ini hadir kepada Pembelajar untuk mempelajari berbagai jenis struktur data dan algoritme serta implementasinya dalam Python, C, C++, dan Java.

Memori itu mahal

Memori tidak selalu tersedia dalam jumlah banyak. Saat berurusan dengan kode/sistem yang mengharuskan Anda untuk menyimpan atau menghasilkan banyak data, sangat penting bagi algoritme Anda untuk menghemat penggunaan memori sedapat mungkin. Misalnya: Saat menyimpan data tentang orang-orang, Anda dapat menghemat memori dengan hanya menyimpan tanggal lahir mereka, bukan usia mereka. Anda selalu dapat menghitungnya dengan cepat menggunakan tanggal lahir dan tanggal saat ini.

Data dan Struktur Data

Program komputer adalah kumpulan instruksi untuk melakukan tugas tertentu. Untuk ini, program komputer mungkin perlu menyimpan data, mengambil data, dan melakukan perhitungan pada data.

---

[3] https://id.wikipedia.org/wiki/Pemrograman_Prosedural, 2019

[4] https://id.wikipedia.org/wiki/Pemrograman_berorientasi_objek, 2019

Pengertian Struktur Data

Struktur data adalah lokasi/tempat bernama yang dapat digunakan untuk menyimpan dan mengatur data. Struktu data ini adalah cara mengatur data di komputer sehingga dapat diakses dan diperbarui secara efisien. Tergantung pada kebutuhan dan proyek Anda, penting untuk memilih struktur data yang tepat untuk proyek Anda. Misalnya, jika Anda ingin menyimpan data secara berurutan dalam memori, maka Anda dapat menggunakan struktur data Array.

Jenis Struktur Data

Catatan :

Struktur data dan tipe data sedikit berbeda. Struktur data adalah kumpulan tipe data yang disusun dalam urutan tertentu.

Pada dasarnya, struktur data dibagi menjadi dua kategori:

1. **Struktur data linier**; Dalam struktur data linier, elemen-elemen disusun secara berurutan satu demi satu. Karena elemen disusun dalam urutan tertentu, mereka mudah diimplementasikan. Namun, ketika kompleksitas program meningkat, struktur data linier mungkin bukan pilihan terbaik karena kompleksitas operasional.
   a. Array (Larik);
      Dalam sebuah array, elemen-elemen dalam memori diatur dalam memori berkelanjutan. Semua elemen array memiliki tipe yang sama. Dan, jenis elemen yang dapat disimpan dalam bentuk array ditentukan oleh bahasa pemrograman.

Gambar 2 Grid (Rak) Buku

b. Stack (Tumpukan);
Dalam struktur data tumpukan, elemen disimpan dalam prinsip LIFO (Last in, first out). Artinya, elemen terakhir yang disimpan dalam tumpukan akan dihapus terlebih dahulu.
Ini bekerja seperti tumpukan keping pada menara Hanoi di mana keping terakhir yang disimpan di tumpukan akan dilepas terlebih dahulu.

Gambar 3 Menara Hanoi

c. Queue (Antrian/Urutan);
Tidak seperti stack, struktur data antrian bekerja dengan prinsip FIFO dimana elemen pertama yang disimpan dalam antrian akan dihilangkan terlebih dahulu.

Gambar 4 Antrian

Ini bekerja seperti antrian orang di loket tiket di mana orang pertama dalam antrian akan mendapatkan tiket terlebih dahulu.

d. Linked List (daftar tertaut);
Dalam struktur data daftar tertaut, elemen data dihubungkan melalui serangkaian node. Dan, setiap node berisi item data dan alamat ke node berikutnya.

Gambar 5 Lokomotif dan gerong tertaut

Ini mirip serangkaian lokomotif (head) dan gerbong kereta yang tertaut dengan gerbong (node) lainnya.

2. **Struktur data non-linier**; Tidak seperti struktur data linier, elemen dalam struktur data non-linier tidak berada dalam urutan apa pun. Sebaliknya mereka disusun secara hierarkis di mana satu elemen akan terhubung ke satu atau lebih elemen. Struktur data non-linier dibagi lagi menjadi grafik (graph) dan struktur data berbasis pohon (tree).
Beberapa graph yang populer ialah:
   a. Spanning Tree and Minimum Spanning Tree
   b. Strongly Connected Components
   c. Adjacency Matrix
   d. Adjacency List

   Beberapa tree yang populer ialah:
   a. Binary Tree
   b. Binary Search Tree
   c. AVL Tree
   d. B-Tree
   e. B+ Tree
   f. Red-Black Tree

Struktur Data Linier Vs Non-linier

Sekarang setelah kita mengetahui tentang struktur data linier dan non-linier, mari kita lihat perbedaan utama di antara keduanya.

Tabel 1.1 Data Linier vs Non linier

Struktur Data Linier	Struktur Data Non Linier
Item data disusun secara berurutan, satu demi satu.	Item data disusun dalam urutan yang tidak berurutan (secara hierarkis).
Semua item hadir di satu lapisan.	Item data hadir pada lapisan yang berbeda.
Itu dapat dilalui dalam sekali jalan. Artinya, jika kita mulai dari elemen pertama, kita dapat melintasi semua elemen secara berurutan dalam satu lintasan.	Ini membutuhkan banyak putaran. Artinya, jika kita mulai dari elemen pertama, tidak mungkin melintasi semua elemen dalam satu lintasan.
Pemanfaatan memori tidak efisien.	Struktur yang berbeda memanfaatkan memori dengan cara efisien yang berbeda tergantung pada kebutuhan.
Kompleksitas waktu meningkat dengan ukuran data.	Kompleksitas waktu tetap sama.
Contoh: Array, Stack, Antrian	Contoh: Tree, Graph, Map

Pre Test

1. Tuliskan kembali dengan ringkas definisi Struktur Data;
2. Ada berapa tingkat bahasa pemrogrman, jelaskan dan tulis contoh;
3. Jelaskan kembali apa itu Paradigma Pemrograman;
4. Mengapa menggunakan Struktur Data pada penyimpanan dan pengelolaan data dalam komputer?
5. Apa saja jenis Struktur Data (Berikan 3 contohnya masing-masing dari jenis yang populer);

BAB II
ARRAY

Ulasan ringkas yang akan Pembelajar dapatkan dalam Bab ini;
Tujuan Pembelajaran
[C1] Pembelajar mempunyai pengetahuan dasar (*basic science*) tentang definisi array.
[C2] Mengintrepretasikan bahasa pemrograman dan membuat array.

Outcome Pembelajaran bagi Pembelajar
[C2] mempunyai kemampuan untuk menjelaskan definisi, konsep, kegunaan array.
[C4] Mampu memahami dan menggunakan array diprogram sederhana.
[C2,C6] Mengerti tentang array dan perbedaan kegunaanya. Menguraikan dan menyimpulkan dalam bentuk resuman, projek sederhana.

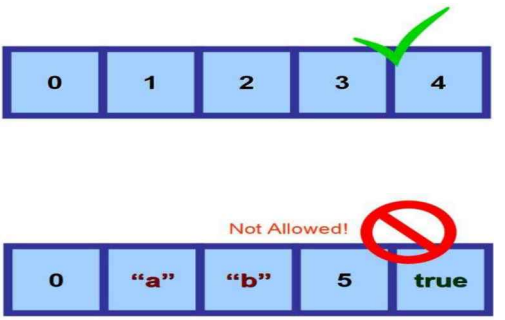
Gambar 6 Grid Rak Buku

Pernahkah anda mengunjungi disebuah toko buku yang bersisi ragam multi disiplin ilmu-genre? Terlihat jelas bukan deretan display ragam buku nampak apik tertata di rak buku tersebut?
Pernahkah Anda melihat rak buku tersebut diisikan selain buku, misalnya selembar pakaian, sepatu, perkakas rumah tangga, dll.? Tidak. Demikian analoginya sebuah ruang dalam memori komputer menampung data bertipe data sama (homogen) yang merupakan klasifikasi dari array pada ilmu komputer.

Untuk memahami cara kerjanya, sangat membantu untuk memvisualisasikan memori komputer Anda sebagai kotak (rak) buku, seperti di atas ini. Setiap bagian informasi disimpan di salah satu elemen kecil (kotak) yang membentuk kisi.

Tabel 2.1 Representasi Array

Konsep Array



Gambar 7 Klasifikasi Array

Array adalah objek yang menyimpan beberapa variabel dengan tipe yang sama (homogen) bertipe yang sama yang disimpan di lokasi memori yang berdekatan Untuk menjamin efisiensi ekstrim untuk menemukan nilai-nilai tersebut. Misalnya jika sebuah array bertipe "int", array hanya dapat menyimpan elemen integer dan tidak dapat mengizinkan elemen bertipe lain seperti double, float, char, dll. Namun, array itu sendiri adalah objek

pada heap. Dan, jenis elemen yang dapat disimpan dalam bentuk array ditentukan oleh bahasa pemrograman.

Struktur data array, yang menyimpan koleksi sekuensial berukuran tetap dari jenis yang sama. Sebuah array digunakan untuk menyimpan koleksi data, namun seringkali lebih berguna untuk memikirkan sebuah array sebagai kumpulan variabel dengan tipe yang sama.

Mendeklarasikan Variabel Array

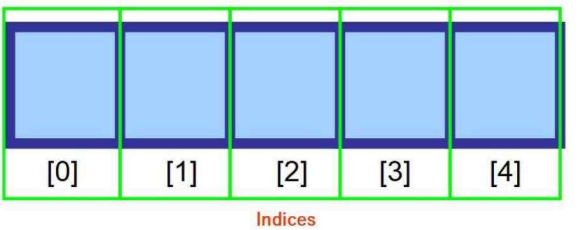
Saat Anda membuat array, Anda:

1. Menetapkannya ke variabel.
2. Tentukan Jenis elemen yang akan disimpannya.
3. Tentukan Ukurannya (jumlah maksimum elemen).

Catatan : Nama yang Anda tetapkan untuk variabel ini sangat penting karena Anda akan menggunakannya nanti dalam kode Anda untuk mengakses nilai dan memodifikasi larik.

Tetapi bagaimana Anda bisa memberi tahu komputer nilai tertentu yang ingin Anda akses? Di sinilah indeks mengambil peran penting!

Index
Catatan : untuk diketahui bahwa pada awalnya terasa aneh untuk mulai menghitung dari 0 bukannya 1, tetapi ini disebut Penomoran Berbasis Nol . Ini sangat umum dalam ilmu komputer.



Gambar 8 Indeks array

Alih-alih mendeklarasikan variabel individual, seperti number0, number1, …, dan number99, Anda mendeklarasikan satu variabel array seperti angka dan nomor penggunaan [0], angka [1], dan …, angka [99] untuk mewakili variabel individu.

Tutorial ini memperkenalkan cara mendeklarasikan variabel array, membuat array, dan array proses menggunakan variabel terindeks.

menggunakan apa yang disebut "indeks" ("indeks" dalam bentuk jamak) untuk mengakses nilai dalam array. Ini adalah nomor yang mengacu pada lokasi di mana nilai disimpan.

Seperti yang dapat Anda lihat pada diagram di bawah, elemen pertama dalam array dirujuk menggunakan indeks 0. Saat Anda bergerak lebih jauh ke kanan, indeks meningkat satu untuk setiap ruang dalam memori.

Untuk menggunakan array dalam sebuah program, Pembelajar harus mendeklarasikan sebuah variabel untuk referensi array, dan Pembelajar harus menentukan jenis array yang dapat referensi referensi. Berikut adalah sintaks untuk mendeklarasikan variabel array -

Sintaksis

```
dataType[] arrayRefVar;   // preferred way.
or
dataType arrayRefVar[];  // works but not preferred way.
```

Catatan - Tipe data gaya **[] arrayRefVar** lebih disukai. Tipe **dataType arrayRefVar []** berasal dari bahasa C / C ++ dan diadopsi di Java untuk mengakomodasi pemrogram C / C ++.

Contoh

Potongan kode berikut adalah contoh sintaks ini -

```
double[] myList;   // preferred way.
or
double myList[];   // works but not preferred way.
```

Membuat array

Pembelajar bisa membuat array dengan menggunakan operator baru dengan sintaks berikut -

Sintaksis

```
arrayRefVar = new dataType[arraySize];
```

Pernyataan di atas melakukan dua hal -

- Ini menciptakan array menggunakan data baruType [arraySize].
- Ini memberikan referensi array yang baru dibuat ke variabel arrayRefVar.

Mendeklarasikan variabel array, membuat sebuah array, dan menetapkan referensi dari array ke variabel dapat digabungkan dalam satu pernyataan, seperti yang ditunjukkan di bawah ini -

```
dataType[] arrayRefVar = new dataType[arraySize];
```

Atau Pembelajar bisa membuat array sebagai berikut -

```
dataType[] arrayRefVar = {value0, value1, ..., valuek};
```

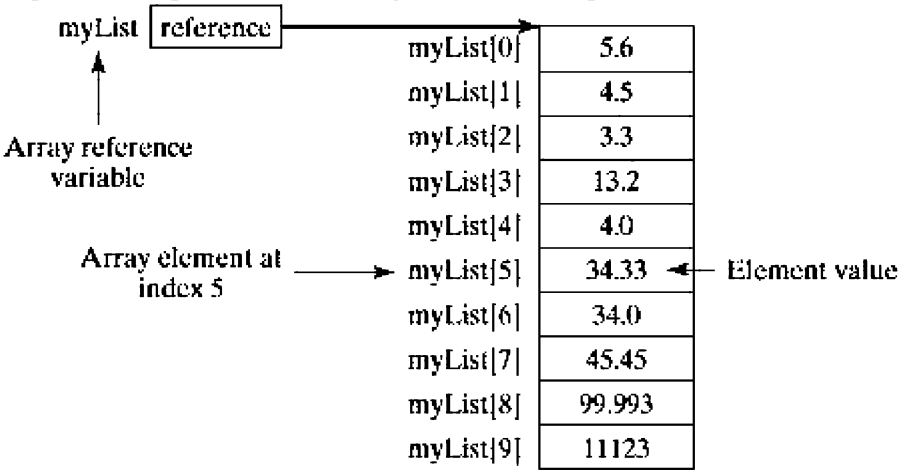
Elemen array diakses melalui **indeks** . Indeks Array berbasis 0; **Artinya** , mereka mulai dari 0 sampai **arrayRefVar.length-1** .

Contoh

Pernyataan berikut mendeklarasikan variabel array, myList, membuat array dari 10 elemen tipe ganda dan memberikan rujukannya kepada myList -

```
double[] myList = new double[10];
```

Berikut gambar mewakili array myList. Di sini, myList memegang sepuluh nilai ganda dan indeksnya adalah 0 sampai 9.



Pengolahan Array

Saat memproses elemen array, kita sering menggunakan baik **untuk** loop atau **foreach** lingkaran karena semua elemen dalam array adalah dari jenis yang sama dan ukuran array dikenal.

Contoh

Berikut adalah contoh lengkap yang menunjukkan bagaimana membuat, menginisialisasi, dan memproses array -

```
public class TestArray {

  public static void main(String[] args) {
    double[] myList = {1.9, 2.9, 3.4, 3.5}; //4

    // Print all the array elements
    for (int i = 0; i < myList.length; i++) {
      System.out.println(myList[i] + " ");
    }

    // Summing all elements
    double total = 0;
    for (int i = 0; i < myList.length; i++) {
      total += myList[i];
    }
    System.out.println("Total is " + total);

    // Finding the largest element
    double max = myList[0];
    for (int i = 1; i < myList.length; i++) {
      if (myList[i] > max) max = myList[i];
    }
    System.out.println("Max is " + max);
```

```
   }
}
```

Ini akan menghasilkan hasil sebagai berikut -

Keluaran

```
1.9
2.9
3.4
3.5
Total is 11.7
Max is 3.5
```

Kode berikut menampilkan semua elemen dalam array myList -

```
public class TestArray {

   public static void main(String[] args) {
      double[] myList = {1.9, 2.9, 3.4, 3.5};

      // Print all the array elements
      for (double element: myList) {
         System.out.println(element);
      }
   }
}
```

Ini akan menghasilkan hasil sebagai berikut -

Keluaran

```
1.9
2.9
3.4
3.5
```

Melewati Array ke Metode

Sama melewati nilai tipe primitif pada metode, juga bisa melewati array ke metode. Sebagai contoh, metode berikut menampilkan elemen dalam array **int** -

Contoh

```
public static void printArray(int[] array) {
   for (int i = 0; i < array.length; i++) {
      System.out.print(array[i] + " ");
   }
}
```

Pembelajar bisa memintanya dengan melewatkan sebuah array. Misalnya, pernyataan berikut memanggil metode printArray untuk menampilkan 3, 1, 2, 6, 4, dan 2 -

Contoh

```
printArray(new int[]{3, 1, 2, 6, 4, 2});
```

Mengembalikan Array dari sebuah Metode
Metode juga bisa mengembalikan array. Sebagai contoh, metode berikut mengembalikan sebuah array yang merupakan pembalikan array lain - Contoh

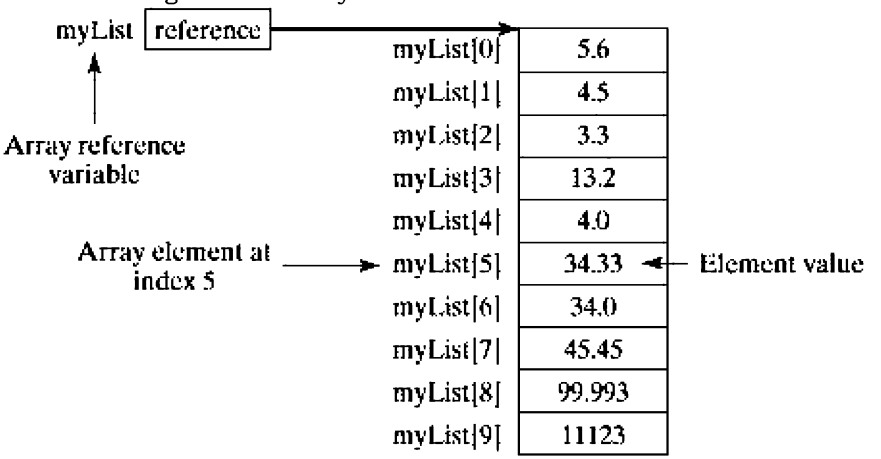
```
public static int[] reverse(int[] list) {
  int[] result = new int[list.length];

  for (int i = 0, j = result.length - 1; i < list length; i++, j--) {
    result[j] = list[i];
  }
  return result;
}
```

Klasifikasi Array

1. **Array Dimensi Satu**

Telah dijelaskan pada bagian diatas bahwa array satu dimensi dianggap sebagai daftar variabel tipe data homogen, dan tiap variabel diakses secara jelas dengan menentukan indeksna dalam tanda kurung kotak yang didahului dengan nama array tersebut.



2. **Array Dimensi Dua**

Array multidimensi memiliki elemen-elemen yang disusun kedalam baris dan kolom membentuk tabel data (bentuk matriks baris-kolom). Berikut ini ialah contoh dari Array dimensi dua (bentuk umum dan pemetaan ke storage).
Untuk mendeklarasikan variabel array dua dimensi, harus menentukan nama array diikuti oleh dua tanda kurung dimana indeks kedua adalah set kedua tanda kurung kotak. Indeks pertama menunjukan baris, dan indeks kedua menunjukan kolom. Setiap elemen dari array multidimensi adalah array itu sendiri. Sebagai contoh,

int[][] a = new int[3][4];

Di sini, telah membuat array multidimensi bernama `a`. Ini adalah array 2 dimensi, yang dapat menampung maksimal 12 elemen.

	Column 1	Column 2	Column 3	Column 4
Row 1	a[0][0]	a[0][1]	a[0][2]	a[0][3]
Row 2	a[1][0]	a[1][1]	a[1][2]	a[1][3]
Row 3	a[2][0]	a[2][1]	a[2][2]	a[2][3]

Ingat, menggunakan pengindeksan berbasis nol, yaitu, pengindeksan array dimulai dengan 0 dan bukan 1. Mari kita ambil contoh lain dari array multidimensi. Kali ini kita akan membuat array 3 dimensi. Sebagai contoh,

String[][][] data = new String[3][4][2];

larik 3d yang dapat menampung maksimum 24 (3*4*2) elemen bertipe String. Perhatikan tabel matrik berikut ini;

	Column 1	Column 2	Column 3	Column 4
Row 1	1 a[0][0]	2 a[0][1]	3 a[0][2]	
Row 2	4 a[1][0]	5 a[1][1]	6 a[1][2]	9 a[1][3]
Row 3	7 a[2][0]			

```
public class MultidimensionalArray {
   public static void main(String[] args) {

      // buat `a` 2d array
      int[][] a = {
         {1, 2, 3},  // berisi 3 data
         {4, 5, 6, 9}, // berisi 4 data
```

```
        {7}, // berisi 1 data
    };

    // hitung panjang data dalam baris
    System.out.println("Panjang isi baris 1 : " + a[0].length);
    System.out.println("Panjang isi baris 2 : " + a[1].length);
    System.out.println("Panjang isi baris 3 : " + a[2].length);
  }
}
```

Menampilkan semua elemen array `a`;

```
public class MultidimensionalArray {
  public static void main(String[] args) {

    int[][] a = {
        {1, 2, 3},
        {4, 5, 6, 9},
        {7},
    };

    for (int i = 0; i < a.length; ++i) {
        for(int j = 0; j < a[i].length; ++j) {
            System.out.println(a[i][j]);
        }
    }
  }
}
```

Pre Test

1. Definisi array ialah;
2. Klasifikasi array berupa apa saja;
3. Perbedaan dan pesamaan array tunggal dengan array multidimensi;
4. Bagaimana tahapan membuat array tunggal, dan tahapan membuat array multidimensi;
5. Buat array multidimensi `Satwa` menggunakan tipe data string berikut;
   "Ayam", "Burung", "Bebek", "Kuda", "Sapi", "Domba"
   "Kaki2", "Kaki4"
   "Bertelur","Beranak"
   Buat output dari elemen array Satwa dengan membentuk pernyataan berdasarkan hubungan dari baris 0 ke baris 1 dan baris 3, berikut; contoh - Ayam, Kaki2, Bertelur

```
public class SatwaMultidimensionalArray {
   public static void main(String[] args) {

      // buat Satwa 2d array
      String [][] Satwa = {
         { "...."},
         {"...."},
         {"....."},
      };

      // Buat output dari elemen array Satwa dengan membentuk pernyataan berikut; contoh [Ayam, Kaki2, Bertelur]
      System.out.println(.....); //indeks
   }
}
```

6. Buatkan tabel dengan isi kolom berupa data kelas anda; Lalu buat ke bentuk array multidimensi berdasarkan (f) frequently/kemunculan elemen(*Kota Provinsi PNS/ nonPNS/ lainnya);

NIM	Nama	Kota	Nilai	Provinsi	PNS/nonPNS/lainnya

BAB III
STACK

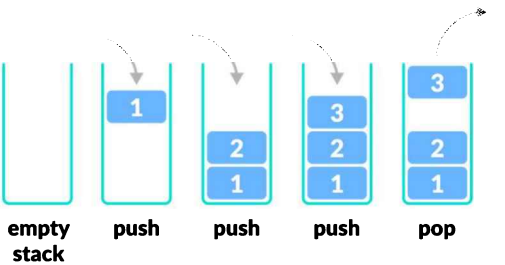
Ulasan ringkas yang akan Pembelajar dapatkan dalam Bab ini;
Tujuan Pembelajaran
[C1] Pembelajar mempunyai pengetahuan dasar (*basic science*) tentang definisi stack.
[C2] Mengintrepretasikan bahasa pemrograman dan membuat stack.

Outcome Pembelajaran bagi Pembelajar
[C2] mempunyai kemampuan untuk menjelaskan definisi, konsep, kegunaan stack.
[C4] Mampu memahami dan menggunakan stack diprogram sederhana.
[C2,C6] Mengerti tentang stack dan perbedaan kegunaanya. Menguraikan dan menyimpulkan dalam bentuk resuman, projek sederhana.

Gambar 9 Representasi Stack Menara Hanoi

Dalam struktur data tumpukan, elemen disimpan dalam prinsip LIFO (Last in, first out). Artinya, elemen terakhir yang disimpan dalam tumpukan akan dihapus terlebih dahulu. Ini bekerja seperti tumpukan keping pada menara Hanoi di mana keping terakhir yang disimpan di tumpukan akan dilepas terlebih dahulu.

Prinsif Stack LIFO

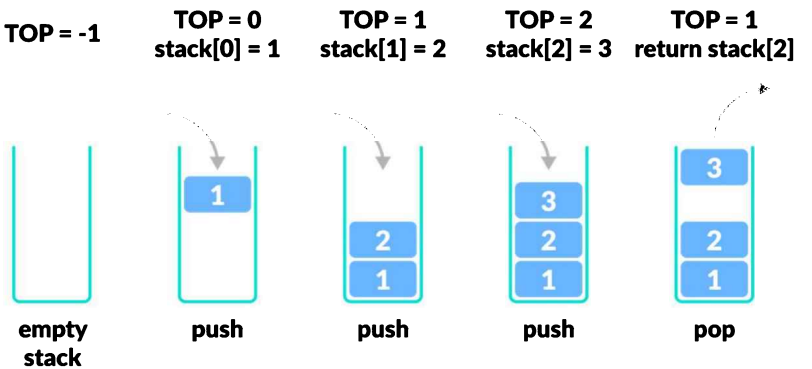


Dalam istilah pemrograman, meletakkan item di atas tumpukan disebut push dan menghapus item disebut pop. Pada gambar di atas, meskipun item 3 disimpan terakhir, dihapus terlebih dahulu. Inilah tepatnya bagaimana Prinsip LIFO (Last In First Out) bekerja.
Stack merupakan metode dalam menyimpan atau mengambil data ke dan dari memori. Stack dapat dibratkan sebuah tumpukan barang dalam sebuah tempat yang hanya memiliki satu pintu diatasnya (memasukan dan mengambil barang hanya dapat dilakukan melalui pintu itu). Ukuran barang tersebut pas dengan pintunya, sehingga barang yang akan dikeluarkan pertama kali adalah barang yang terkhir kali dimasukan.

Operasi dasar Stack LIFO

Ada beberapa operasi dasar yang memungkinkan kita untuk melakukan tindakan yang berbeda pada tumpukan.

Push : Tambahkan elemen ke atas tumpukan
Pop : Menghapus elemen dari atas tumpukan
IsEmpty : Periksa apakah tumpukan kosong
IsFull : Periksa apakah tumpukan sudah penuh
Peek : Dapatkan nilai elemen teratas tanpa menghapusnya



Operasi bekerja sebagai berikut:

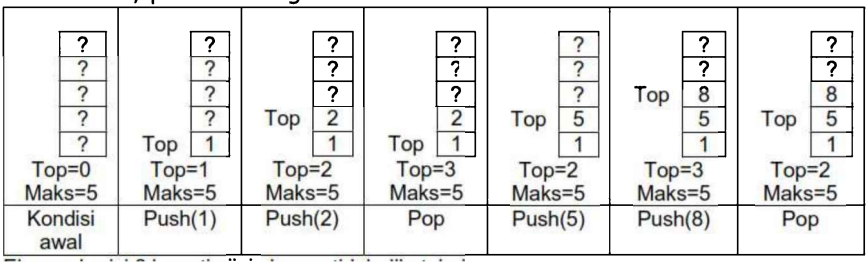
1. Sebuah penunjuk disebut TOP digunakan untuk melacak elemen teratas dalam tumpukan.
2. Saat menginisialisasi tumpukan, menetapkan nilainya ke -1 sehingga dapat memeriksa apakah tumpukan kosong dengan membandingkan TOP == -1.

3. Saat mendorong sebuah elemen, meningkatkan nilai TOP dan tempatkan elemen baru di posisi yang ditunjukkan oleh TOP.
4. Saat memunculkan elemen, mengembalikan elemen yang ditunjukkan oleh TOP dan mengurangi nilainya.
5. Sebelum PUSH, memeriksa apakah tumpukan sudah penuh
6. Sebelum muncul, memeriksa apakah tumpukan sudah kosong

Kompleksitas Waktu Tumpukan

Untuk implementasi stack berbasis array, operasi push dan pop membutuhkan waktu yang konstan, yaitu O(1). Representasi stack dalam pemrograman, dapat dilakukan dengan 2 cara yaitu :
1. Representasi stack dengan array
2. Representasi stack dengan single linked list

Sebagai contoh representasi kedua cara tersebut dengan operasi yang dilakukan adalah push(1), push(2), pop, push(5), push(8), pop. Untuk lebih detail, perhatikan gambar di bawah ini :



Representasi stack dengan menggunakan array dengan maksimal data adalah 5

```
// Stack implementation in Java

public class Stack {
  private int arr[];
  private int top;
  private int capacity;

  // Creating a stack
  Stack(int size) {
    arr = new int[size];
    capacity = size;
    top = -1;
  }

  // Add elements into stack
  public void push(int x) {
    if (isFull()) {
      System.out.println("OverFlow\nProgram Terminated\n");
      System.exit(1);
    }
```

```
    System.out.println("Inserting " + x);
    arr[++top] = x;
  }

  // Remove element from stack
  public int pop() {
    if (isEmpty()) {
      System.out.println("STACK EMPTY");
      System.exit(1);
    }
    return arr[top--];
  }

  // Utility function to return the size of the stack
  public int size() {
    return top + 1;
  }

  // Check if the stack is empty
  public Boolean isEmpty() {
    return top == -1;
  }

  // Check if the stack is full
  public Boolean isFull() {
    return top == capacity - 1;
  }

  public void printStack() {
    for (int i = 0; i <= top; i++) {
      System.out.println(arr[i]);
    }
  }

  public static void main(String[] args) {
    Stack stack = new Stack(5);
    stack.push(1);
    stack.push(2);
    stack.pop();
    stack.push(5);
    stack.push(8);
    stack.pop();
    System.out.println("\nAfter popping out");
    stack.printStack();
  }
}
```

Aplikasi Struktur Data Tumpukan

Meskipun stack adalah struktur data yang sederhana untuk diterapkan, ini sangat kuat. Penggunaan stack yang paling umum adalah:

1. Untuk membalik kata - Letakkan semua huruf dalam tumpukan dan keluarkan. Karena urutan tumpukan LIFO, Anda akan mendapatkan huruf dalam urutan terbalik.
2. Dalam kompiler - Kompilator menggunakan tumpukan untuk menghitung nilai ekspresi seperti 2 + 4 / 5 * (7 - 9)dengan mengubah ekspresi menjadi bentuk awalan atau postfix.
3. Di browser - Tombol kembali di browser menyimpan semua URL yang telah Anda kunjungi sebelumnya dalam tumpukan. Setiap kali Anda mengunjungi halaman baru, halaman itu ditambahkan di atas tumpukan. Saat Anda menekan tombol kembali, URL saat ini dihapus dari tumpukan, dan URL sebelumnya diakses.
4. Konversi Bilangan Desimal ke Bilangan Biner. Contoh nyata implementasi stack adalah proses pengkonversian dari bilangan decimal (basis 10) ke bilangan biner (basis 2).

Pre Test

1. Stack merupakan metode dalam menyimpan atau mengambil data ke dan dari memori dengan karaktersitik berupa LIFO, jelaskan dan berikan contoh lainnya;
2. Buatlah aplikasi stack menampilkan 5 peraih pole position (nama pembalap-tim) di MotoGP Mandalika 2022 (representasi stack dengan array). gunakan implementasi bahasa pemrograman yang anda kuasai. Simulasikan dengan skema berikut [push, pop, push, push, push , pop, pop]

BAB IV
QUEUE (ANTRIAN)

Ulasan ringkas yang akan Pembelajar dapatkan dalam Bab ini;
Tujuan Pembelajaran
[C1] Pembelajar mempunyai pengetahuan dasar (*basic science*) tentang definisi antrian.
[C2] Mengintrepretasikan bahasa pemrograman dan membuat antrian.

Outcome Pembelajaran bagi Pembelajar
[C2] mempunyai kemampuan untuk menjelaskan definisi, konsep, kegunaan antrian.
[C4] Mampu memahami dan menggunakan antrian diprogram sederhana.
[C2,C6] Mengerti tentang antrian dan perbedaan kegunaanya. Menguraikan dan menyimpulkan dalam bentuk resuman, projek sederhana.

Tidak seperti stack, struktur data antrian bekerja dengan prinsip FIFO dimana elemen pertama yang disimpan dalam antrian akan dihilangkan terlebih dahulu.

Representasi FIFO dari Antrian

Pada gambar di atas, karena 1 disimpan dalam antrian sebelum 2, itu juga yang pertama dihapus dari antrian. Ini mengikuti aturan FIFO .

Dalam istilah pemrograman, menempatkan item dalam antrian disebut enqueue , dan menghapus item dari antrian disebut dequeue. Berikut ini jenis dari antrian;

- Simple Queue
- Circular Queue
- Priority Queue
- Double Ended Queue

Gambar 10 Representasi antrian dikehidupan

Operasi Dasar Antrian

Antrian adalah objek (struktur data abstrak - ADT) yang memungkinkan operasi berikut:
Enqueue : Tambahkan elemen ke akhir antrian
Dequeue : Hapus elemen dari depan antrian
IsEmpty : Periksa apakah antrian kosong
IsFull : Periksa apakah antrian sudah penuh
Peek : Dapatkan nilai bagian depan antrian tanpa menghapusnya

Pengerjaan Antrian

Operasi antrian bekerja sebagai berikut:

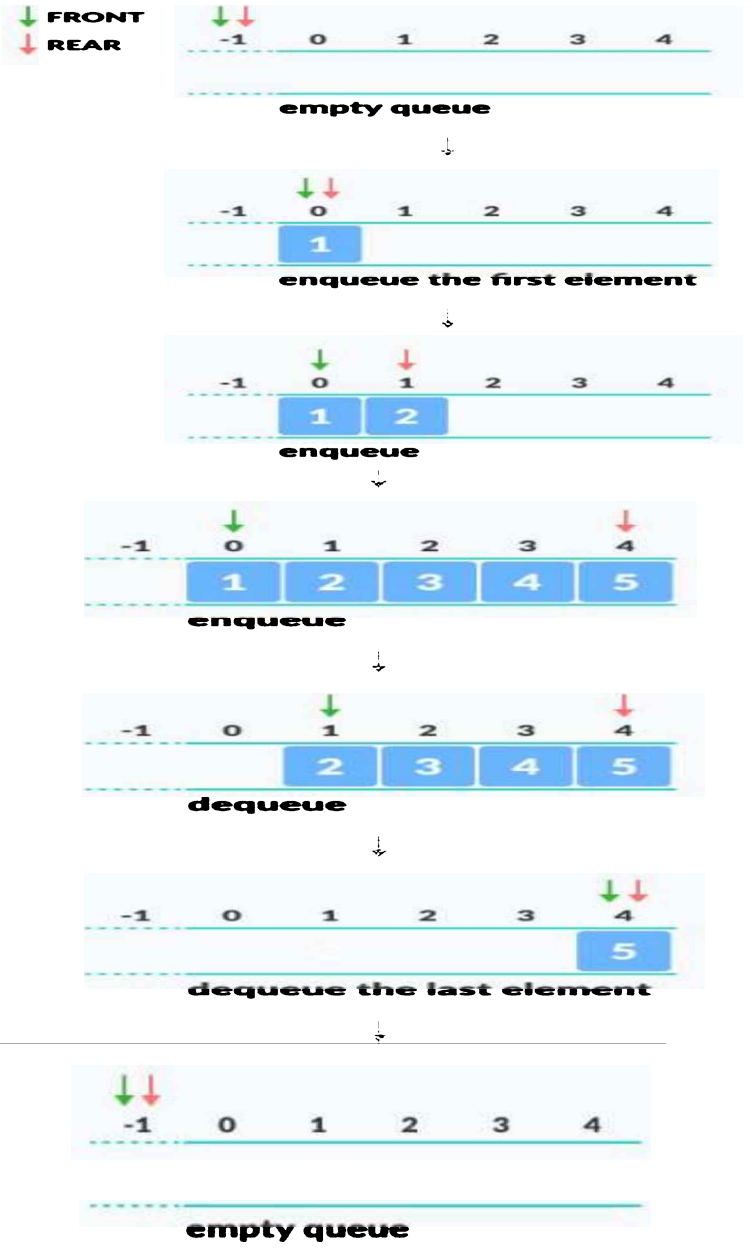
- Dua petunjuk depan dan belakang
- Depan lacak elemen pertama dari antrian
- Belakang lacak elemen terakhir dari antrian
- Awalnya, tetapkan nilaidepan dan belakang ke -1

Operasi Antrian

- Periksa apakah antrian sudah penuh
- Untuk elemen pertama, atur nilai depan ke 0
- Meningkatkan belakang indeks dengan 1
- Tambahkan elemen baru di posisi yang ditunjukkan oleh belakang

Operasi Dequeue

- Periksa apakah antriannya kosong
- Mengembalikan nilai yang ditunjukkan oleh depan
- Meningkatkan depan indeks dengan 1
- Untuk elemen terakhir, atur ulang nilai depan dan belakang ke -1



Operasi Enqueue dan Dequeue

```
// Queue implementation in Java

public class Queue {
  int SIZE = 5;
  int items[] = new int[SIZE];
  int front, rear;

  Queue() {
    front = -1;
    rear = -1;
  }

  boolean isFull() {
    if (front == 0 && rear == SIZE - 1) {
      return true;
    }
    return false;
  }

  boolean isEmpty() {
    if (front == -1)
      return true;
    else
      return false;
  }

  void enQueue(int element) {
    if (isFull()) {
      System.out.println("Queue is full");
    } else {
      if (front == -1)
        front = 0;
      rear++;
      items[rear] = element;
      System.out.println("Inserted " + element);
    }
  }

  int deQueue() {
    int element;
    if (isEmpty()) {
      System.out.println("Queue is empty");
      return (-1);
    } else {
      element = items[front];
      if (front >= rear) {
        front = -1;
        rear = -1;
      } /* Q has only one element, so we reset the queue after deleting
it. */
```

```
    else {
      front++;
    }
    System.out.println("Deleted -> " + element);
    return (element);
  }
}

void display() {
  /* Function to display elements of Queue */
  int i;
  if (isEmpty()) {
    System.out.println("Empty Queue");
  } else {
    System.out.println("\nFront index-> " + front);
    System.out.println("Items -> ");
    for (i = front; i <= rear; i++)
      System.out.print(items[i] + "  ");

    System.out.println("\nRear index-> " + rear);
  }
}

public static void main(String[] args) {
  Queue q = new Queue();

  // deQueue is not possible on empty queue
  q.deQueue();

  // enQueue 5 elements
  q.enQueue(1);
  q.enQueue(2);
  q.enQueue(3);
  q.enQueue(4);
  q.enQueue(5);

  // 6th element can't be added to because the queue is full
  q.enQueue(6);

  q.display();

  // deQueue removes element entered first i.e. 1
  q.deQueue();

  // Now we have just 4 elements
  q.display();
```

```
  }
}
```

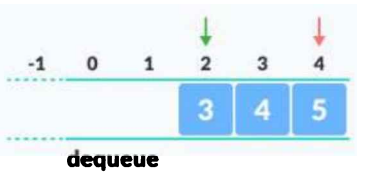
Aplikasi Antrian

- Penjadwalan CPU, Penjadwalan Disk
- Ketika data ditransfer secara tidak sinkron antara dua proses. Antrian digunakan untuk sinkronisasi. Misalnya: IO Buffer, pipa, file IO, dll
- Penanganan interupsi dalam sistem waktu nyata.
- Sistem telepon Call Center menggunakan Antrian untuk menahan orang yang menelepon mereka secara berurutan.

Batasan Antrian

Seperti yang Anda lihat pada gambar di bawah, setelah sedikit enqueuing dan dequeuing, ukuran antrian telah berkurang. Dan kita hanya dapat menambahkan indeks 0 dan 1 hanya ketika antrian direset (ketika semua elemen telah di-dequeued).

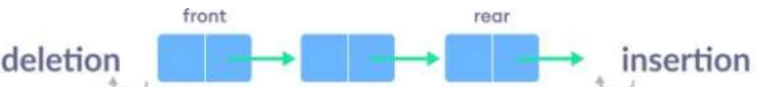
Setelah BELAKANG mencapai indeks terakhir, jika kita dapat menyimpan elemen tambahan di ruang kosong (0 dan 1), kita dapat memanfaatkan ruang kosong. Ini diimplementasikan oleh antrian yang dimodifikasi yang disebut antrian melingkar .



Batasan antrian

Antrian Sederhana/biasa

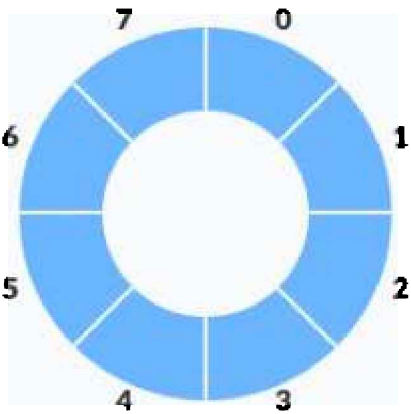
Dalam antrian sederhana, penyisipan dilakukan di bagian belakang dan pelepasan terjadi di bagian depan. Ini secara ketat mengikuti aturan FIFO (First in First out).



Representasi Antrian Sederhana

Antrian Melingkar

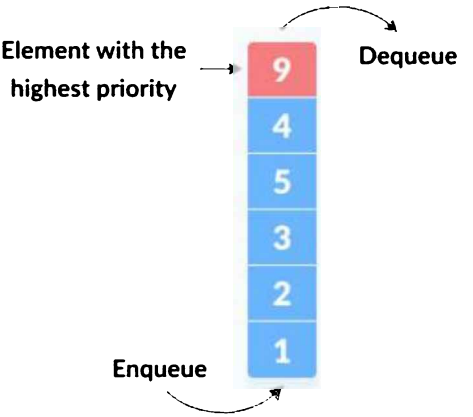
Sebuah antrian melingkar adalah versi diperpanjang dari antrian biasa di mana elemen terakhir terhubung ke elemen pertama. Sehingga membentuk struktur seperti lingkaran. Antrian melingkar memecahkan batasan utama dari antrian normal. Dalam antrian normal, setelah sedikit penyisipan dan penghapusan, akan ada ruang kosong yang tidak dapat digunakan.



Keuntungan utama dari antrian melingkar di atas antrian sederhana adalah pemanfaatan memori yang lebih baik. Jika posisi terakhir penuh dan posisi pertama kosong, kita dapat menyisipkan elemen di posisi pertama. Tindakan ini tidak mungkin dilakukan dalam antrian sederhana.

Antrian Prioritas

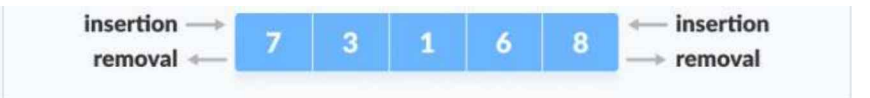
Antrian prioritas adalah jenis antrian khusus di mana setiap elemen dikaitkan dengan prioritas dan dilayani sesuai dengan prioritasnya. Jika elemen dengan prioritas yang sama terjadi, mereka dilayani sesuai dengan urutannya dalam antrian.



Menghapus Elemen Prioritas Tertinggi

Deque (Antrian Berakhir Ganda)

Dalam antrian ujung ganda, penyisipan dan penghapusan elemen dapat dilakukan dari depan atau belakang. Jadi, tidak mengikuti aturan FIFO (First In First Out).



Representasi Deque

Pre Test

1. Tuliskan kembali dengan ringkas definisi Antrian;
2. Jenis-jenis Antrian apa saja, jelakan.

TENTANG PENULIS

Nama
Zayid Musiafa, S.Kom., M.Kom

Tempat/Tgl. Lahir
Semarang, 28 Januari 1990

Jenis Kelamin
Laki-Laki

Web site

Email
elearning.zayidmusiafa@gmail.com

Peminatan scope ilmu mendalami AI, image processing, game development, 2D & 3D design, komputer grafis, animasi dan multimedia secara otodidak.
Pendidikan Akhir:
Menyelesaikan pendidikan program magister (S2 Tahun 2015) - Fakultas Ilmu Komputer Prodi Teknik Infromatika Universitas Dian Nuswantoro Semarang.

Pengalaman Mengajar / Dosen:
- Teknik Informatika Fakultas Teknologi Informasi Universitas Islam Kalimantan Muhammad Arsyad Al Banjari Banjarmasin (2016).
- Sistem Informasi Fakultas Teknologi Informasi Universitas Tangerang Raya (2022).

Aktif sebagai pengajar, penulis buku, pemateri seminar pelatihan & pengabdian kepada Masyarakat dalam Tri Dharma Perguruan Tinggi.

Creator :
Game Poma 3D Adventure (Tahun : 2013),
VR Museum Wasaka (Tahun : 2020),

Penulis Buku ;
Membangun Aplikasi Inventory Multi Store Dengan Visual Basic Dan Mysql Tahun: 2019 ISBN: 978-623-91132-9-2
Studi Kasus Prototype: Membangun Aplikasi Store Management Toko Akhtar Galaxy Menggunakan Bahasa Pemrograman Java Dan Database MYSQL Tahun: 2019 ISBN: 978-623-7373-65-0
Multimedia Immersive Virtual Tour 3d Panorama 360° Tahun: 2020 ISBN: 978-623-7583-16-5
Buku Ajar STRUKTUR DATA DAN IMPLEMENTASI ALGORITMA (SDIA);
Bahasa Pemrograman Python Java C C++ Tahun 2022 ISBN XXX-XXXX-XXX