Algoritma dan Struktur Data 2

Modul 7
Queue (Antrian)

Disusun oleh:

Dwi Intan Af'idah, S.T., M.Kom

PROGRAM STUDI TEKNIK INFORMATIKA
POLITEKNIK HARAPAN BERSAMA
TAHUN AJARAN 2020/2021

Daftar Isi

1 Queue dan Priority Queue

1.1 Target Pembelajaran

1. Memahami konsep Queue dan operasi-operasi pada Queue.
2. Mengimplementasikan Queue menggunakan Array.
3. Memahami konsep Priority Queue.
4. Mengimplementasikan Priority Queue menggunakan Array.

1.2 Dasar Teori

1.2.1 Antrian (Queue)

Antrian (queue) adalah sekumpulan elemen, jika ada elemen baru yang ditambahkan, maka elemen tersebut akan berada di bagian belakang antrian. Jika ada elemen yang harus dihapus atau keluar dari antrian, maka elemen yang keluar adalah elemen yang berada di sisi depan antrian. Atau konsep ini sering juga disebut dengan konsep FIFO (First In First Out).

Queue dalam kehidupan sehari-hari seperti antrian pada penjualan tiket kereta api, dimana orang yang pertama datang adalah orang yang pertama kali dilayani untuk membeli tiket. Jika ada orang baru yang datang akan membali tiket, maka posisinya berada pada urutan paling belakang dalam antrian tersebut.Orang yang berada pada posisi terakhir dalam antrian adalah yang terakhir kali dapat dilayani dan memperoleh tiket kereta api (kalau kurang beruntung, maka akan kehabisan tiket). Contoh lain adalah nasabah yang antri di teller bank, paket data yang menunggu untuk ditransmisikan lewat internet, antrian printer dimana terdapat antrian print job yang menunggu giliran untuk menggunakan printer, dan sebagainya.

Array pada Queue adalah suatu array yang dibuat seakan-akan merupakan suatu garis lurus dengan satu pintu masuk dan satu pintu keluar. Pada gambar di bawah merupakan implementasi queue menggunakan array. **qfront** untuk menandai elemen yang pertama, sedangkan qback untuk menambahkan elemen baru pada queue.

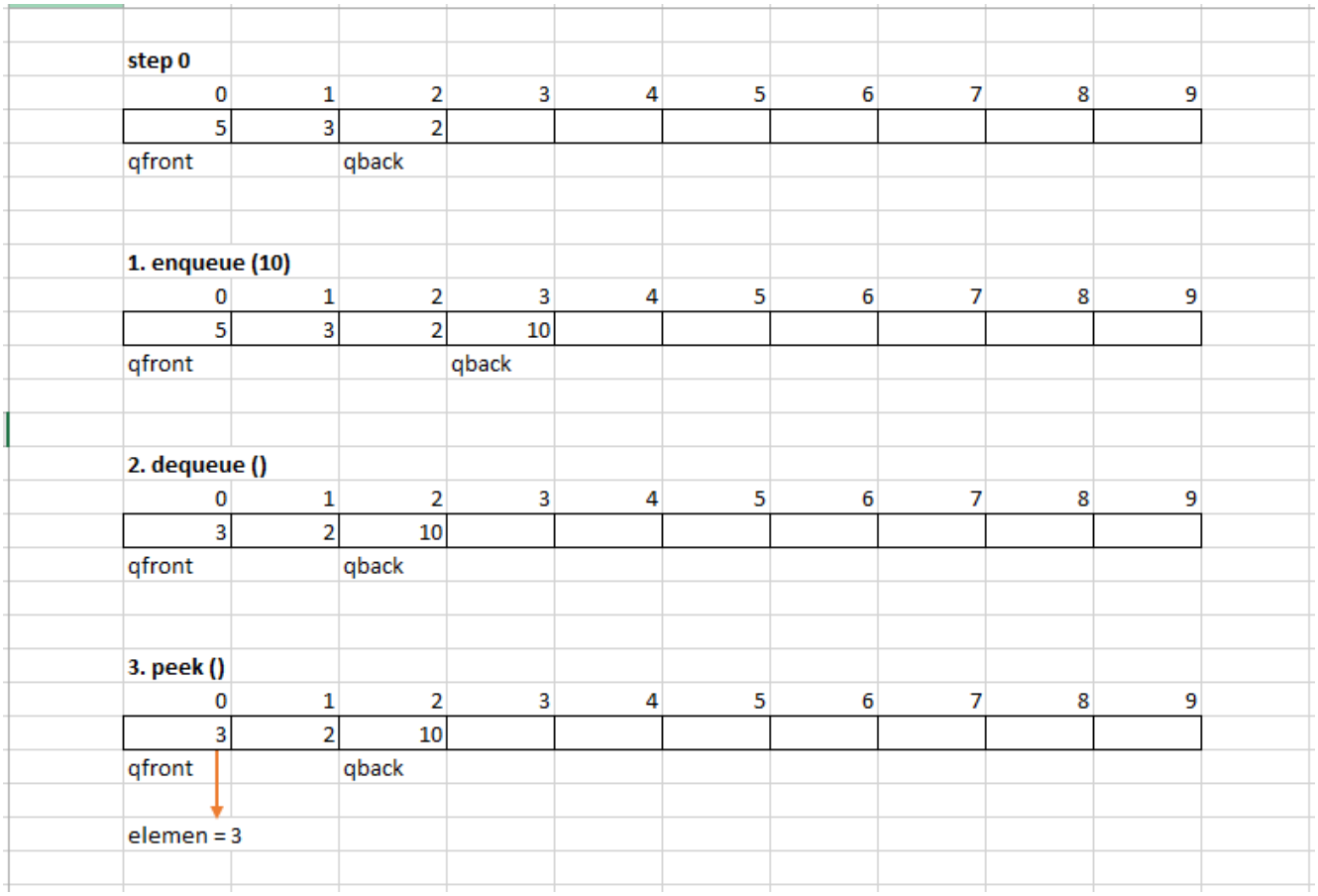
5	3	2		

qfront qback

Operasi yang terdapat pada queue adalah:

1. **Enqueue** atau push adalah proses untuk memasukkan elemen artinya menambah data baru.
2. **Dequeue** atau pop adalah proses untuk mengeluarkan elemen artinya menghapus data.
3. **Peek** adalah proses untuk mengetahui elemen yang paling depan.
4. **Clear** adalah operasi untuk mengkosongkan antrian (menghapus seluruh elemen dalam antrian).
5. **Tampil** adalah operasi untuk menampilkan elemen dari antrian satu per satu.
6. **IsEmpty** adalah operasi untuk mengecek apakah antrian dalam kondisi kosong atau tidak.
7. **IsFull** adalah operasi untuk mengecek apakah antrian dalam kondisi penuh atau tidak.

Berikut merupkan ilustrasi dari operasi pada struktur data Queue dengan ukuran (10).



1.2.2 Priority Queue

Dalam antrian yang telah dibahas di atas, semua elemen yang masuk dalam antrian dianggap mempunyai prioritas yang sama, sehingga elemen yang masuk lebih dahulu akan diproses lebih dahulu. Dalam praktek, elemen-elemen yang akan masuk dalam suatu antrian ada yang dikatakan mempunyai prioritas yang lebih tinggi dibanding yang lain. Antrian yang demikian ini disebut dengan antrian berprioritas (priority queue). Dalam antrian berprioritas, setiap elemenn yang akan masuk dalam antrian sudah ditentukan lebih dahulu prioritasnya. Dalam hal ini berlaku dua ketentuan, yaitu:

1. Elemen-elemen yang mempunyai prioritas lebih tinggi akan diproses lebih dahulu.
2. Dua elemen yang mempunyai prioritas sama akan dikerjakan sesuai dengan urutan pada saat kedua elemen ini masuk dalam antrian.

Dengan memperhatikan kedua ketentuan di atas, akan berlaku ketentuan bahwa elemen yang mempunyai prioritas lebih tinggi akan dikerjakan lebih dahulu dibanding elemen yang mempunyai prioritas lebih rendah, meskipun elemen yang berprioritas tinggi masuknya sesudah elemen yang berprioritas rendah. Salah satu contoh antrian berprioritas ini adalah pada sistem berbagi waktu (time-sharing system) dimana program yang mempunyai prioritas tinggi akan dikerjakan lebih dahulu dan program-program yang berprioritas sama akan membentuk antrian biasa.

1.3 Latihan

1.3.1 Source Code Queue

1. Membuat class Antrian

```
1   package Queue;
2
3   public class Antrian {
4           private int ukuran;
5           private long[] antrian;
6           private int belakang;
7           private int jumItem;
8           private int depan;
9
10          public Antrian (int s){
11                  ukuran = s;
12                  antrian = new long[ukuran];
13                  depan = 0;
14                  belakang = -1;
15                  jumItem = 0;
16          }
17
18          public void enqueue (long j){
19                  if (!isFull()){
20                          antrian[++belakang] = j;
21                          jumItem++;
22                  }
23          }
24
```

```
25          public long dequeue (){
26          long temp = antrian[0];
27          if (!isEmpty()){
28                  for (int i=0; i<jumItem; i++)
29                          antrian [i] = antrian[i+1];
                            jumItem--;
31                          belakang--;
32                  }
33                  return temp;
34          }
35
36          public long peek(){
37                  return antrian[depan];
38          }
39
40          public boolean isEmpty(){
41                  return (jumItem==0);
42          }
43
44          public boolean isFull(){
45                  return (belakang==ukuran-1);
46          }
47
48          public int ukuran(){
49                  return jumItem;
50          }
```

```
51
52          public void display(){
53                  for (int i=0; i<jumItem; i++)
54                  System.out.print(antrian[i]+" ");
                    System.out.println("");
56          }
57      }
58
```

2. Membuat class AntriaApp

```
4   public class AntrianApp {
5       public static void main(String[] args){
6               Antrian antrian = new Antrian(10);
7               antrian.enqueue(73);
8               antrian.display();
9               antrian.enqueue(45);
10              antrian.display();
11              antrian.enqueue(80);
12              antrian.display();
13              System.out.println("yang diambil dari antrian = " + antrian.dequeue());
14              antrian.display();
15              System.out.println("yang diambil dari antrian = " + antrian.dequeue());
16              antrian.display();
17
18      }
19  }
```

1.3.2 Ilustrasi

Ilustrasi dari stuktur data tumpukan dengan operasi pada sub bab 1.3.1 adalah sebagai berikut sebagai berikut:

1. enqueue (73)

0	1	2	3	4	5	6	7	8	9
73									
qfront									

2. enqueue (45)

0	1	2	3	4	5	6	7	8	9
73	45								
qfront	qback								

3. enqueue (80)

0	1	2	3	4	5	6	7	8	9
73	45	80							
qfront		qback							

4. dequeue ()

0	1	2	3	4	5	6	7	8	9
45	80								
qfront	qback								

5. dequeue ()

0	1	2	3	4	5	6	7	8	9
80									
qfront									

1.4 Tugas Praktikum 7

- Buatlah
  1. **kode program**
  2. **Ilustrasi**

  sehingga menghasilkan output seperti di bawah ini: