BAB 4

QUEUE (ANTREAN)

ANTREAN (Queue)
Suatu bentuk khusus dari linear list, dengan operasi ***penyisipan*** (insertion) hanya diperbolehkan pada salah satu sisi, yang disebut ***REAR***, dan operasi ***penghapusan*** (deletion) hanya diperbolehkan pada sisi yang lainnya, yang disebut ***FRONT*** dari list.

Antrean Q = [Q_1, Q_2, ... , Q_N]
Front(Q) = Q_1 ⟶ bagian depan antrean
Rear(Q) = Q_N ⟶ bagian belakang antrean
Noel(Q) = N ⟶ jumlah elemen dalam antrean

Operasi Antrean : ***FIFO*** (***F***irst ***I***n ***F***irst ***O***ut)
Elemen yang pertama masuk merupakan elemen yang pertama keluar.

Operator : Penyisipan : Insert
Penghapusan : Remove

Empat operasi dasar antrean, yaitu :
1. CREATE
2. ISEMPTY
3. INSERT
4. REMOVE

- **CREATE (Q)**
  Operator yang menunjukkan suatu antrean hampa Q.
  Berarti : Noel (Q) = 0
  Front (Q) & Rear (Q) = tidak terdefinisi

- **ISEMPTY (Q)**
  Operator yang menunjukkan apakah antrean Q hampa.
  Operand : tipe data antrean
  Hasil : tipe data boolean

  ISEMPTY (CREATE (Q)) = True

- **INSERT (E, Q)**
  Operator yang menginsert elemen E ke dalam antrean Q.
  E ditempatkan di bagian belakang antrean.
  Hasil : antrean yang lebih besar.

  REAR (INSERT (E, Q)) = E
  ISEMPTY (INSERT (E, Q)) = False

- **REMOVE (Q)**
  Operator yang menghapus elemen bagian depan dari antrean Q.
  Hasil : antrean yang lebih pendek.
  Pada setiap operasi, Noel (Q) berkurang 1 dan elemen ke-2 menjadi elemen terdepan.
  Jika Noel (Q) = 0 maka Q = hampa
  Remove (Q) = kondisi error (underflow condition)
  Remove (Create (Q)) = kondisi error (underflow condition)

PENYAJIAN DARI ANTREAN

1. One Way List (Linear Linked List)
2. Array

Array Queue

Kalau tidak disebutkan lain, maka Antrean disajikan dalam Array Queue, dilengkapi 2 variabel penunjuk :
FRONT (elemen depan antrean)
REAR (elemen belakang antrean)

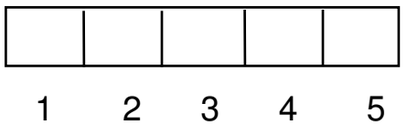
Contoh :

Antrean dalam array queue dengan 5 lokasi memori

1. Pada awal antrean hampa

F = 0
R = 0

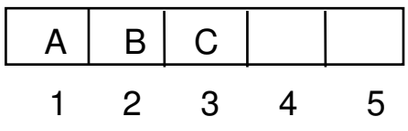
Queue



2. A, B dan C dimasukkan

F = 1
R = 3

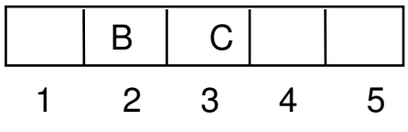
Queue



3. Hapus 1 elemen : A dihapus

F = 2
R = 3

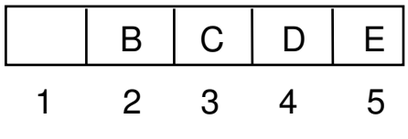
Queue



4. D dan E dimasukkan

F = 2
R = 5

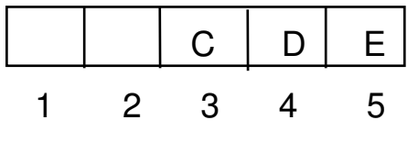
Queue



5. Hapus 1 elemen : B dihapus

F = 3
R = 5

Queue

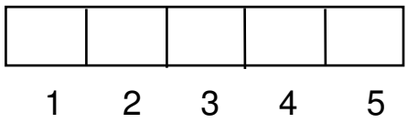


Untuk setiap pemasukan elemen, nilai Rear + 1
penghapusan elemen, nilai Front + 1

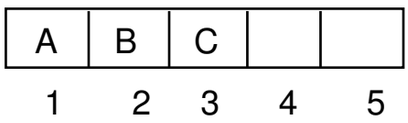
Akibatnya, setelah pemasukan elemen ke-5 maka lokasi Queue (5) telah diduduki mungkin saja tidak sebanyak 5 elemen ada dalam antrean, karena sudah dilakukan beberapa penghapusan. Untuk pemasukan elemen berikutnya, yakni memasukkan elemen ITEM, gunakan lokasi QUEUE (1), dan seterusnya. Array Sirkular yaitu elemen Queue (1) datang sesudah Queue (N) di dalam array, maka Rear = 1. Jika Front = N, dilakukan penghapusan maka Front = 1, bukan N + 1.

Contoh :
Array Sirkular dengan 5 lokasi memori

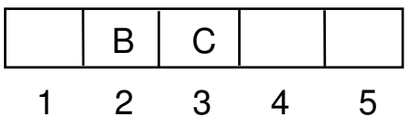
1. Pada awal antrean hampa
   F = 0
   R = 0



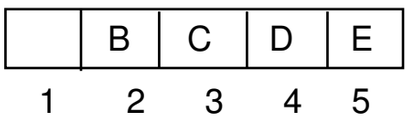
2. A, B dan C dimasukkan
   F = 1
   R = 3



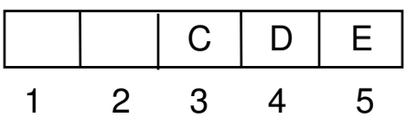
3. Hapus 1 elemen : A dihapus
   F = 2
   R = 3



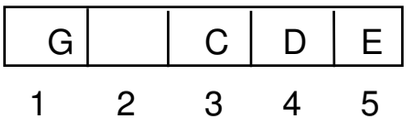
4. D dan E dimasukkan
   F = 2
   R = 5



5. Hapus 1 elemen : B dihapus
   F = 3
   R = 5



6. G dimasukkan
   F = 3
   R = 1



ALGORITMA

1. QINSERT (Memasukkan data ke dalam suatu antrean)
   Memeriksa kemungkinan terjadi overflow, yakni dengan melihat apakah antrean tersebut terisi penuh.
2. QDELETE (Menghapus elemen depan dari antrean)
   Memeriksa kemungkinan terjadi underflow, yakni dengan melihat apakah antrean tersebut kosong.

QINSERT (QUEUE, N, FRONT, DATA)

1. {Apakah antrean penuh}
   Jika Front = 1 dan Rear = N, atau
   Jika Front = Rear + 1 , maka write overflow, return
2. Jika Front = Null, maka Front := 1
   Rear := 1
   Dalam hal lain
   Jika Rear = N, maka Rear := 1
   Dalam hal lain
   Rear := Rear + 1
3. Queue (Rear) := Data (masukkan elemen baru)
4. Return.

QDELETE (QUEUE, N, FRONT, REAR, DATA)

1. {Apakah antrean kosong}
   Jika Front := Null, maka write underflow, return
2. Data := Queue (Front)
3. (Front mendapat nilai baru)
   Jika Front := Rear, maka
   Front := Null,
   Dalam hal lain
   Jika Front = N, maka Front := 1
   Dalam hal lain
   Front := Front + 1
4. Return.

DEQUE (Queue Ganda atau Double Queue)

Suatu linear list, yang penambahan dan penghapusan elemen dapat dilakukan pada kedua sisi ujung list, tetapi tidak dapat dilakukan ditengah-tengah list.

Deque (menggunakan array sirkular)
Menggunakan 2 pointer/penunjuk :
1. LEFT : sisi kiri dari deque
2. RIGHT : sisi kanan dari deque

Asumsi : elemen deque berurut dari kiri ke kanan.

Contoh : Menggambarkan 2 buah deque, masing-masing berisi 4 elemen, yang ditempatkan di dalam sebuah Array dengan 8 lokasi memori.

DEQUE

LEFT : 4				A	B	C	D	
RIGHT : 7	1	2	3	4	5	6	7	8

(a)

DEQUE

LEFT : 7	Y	Z					W	X
RIGHT : 2	1	2	3	4	5	6	7	8

(b)

2 variasi deque, yaitu :

1. Deque Input Terbatas
   Pemasukan elemen pada satu ujung list, penghapusan elemen pada kedua ujung list.
2. Deque Output Terbatas
   Pemasukan elemen pada kedua ujung list, penghapusan elemen pada salah satu ujung list.

ANTREAN BERPRIORITAS

Himpunan elemen, yang setiap elemennya telah diberikan sebuah prioritas, dan urutan proses penghapusan elemen adalah berdasarkan aturan berikut :

1. Elemen yang prioritasnya lebih tinggi, diproses lebih dahulu dibandingkan dengan elemen yang prioritasnya lebih rendah.
2. Dua elemen dengan prioritas yang sama, diproses sesuai dengan urutannya sewaktu dimasukkan ke dalam antrean berprioritas.

PENYAJIAN ONE WAY LIST DARI ANTREAN BERPRIORITAS

Ketentuan :

1. Setiap simpul dalam list berisi 3 buah data/field yaitu :
   (**Info**, **Prn**, **Link**)
2. Simpul X mendahului simpul Y di dalam list, bila :
   a. X mempunyai prioritas lebih tinggi dari Y.
   b. Mempunyai prioritas yang sama, tetapi X dimasukkan ke dalam queue terlebih dahulu sebelum Y.

Ket : Simpul dengan **Prn** rendah, mendapat prioritas tertinggi.

Contoh : Gambar di bawah memperlihatkan diagram skematik dari Antrean Berprioritas dengan 7 elemen dan penyajian dalam memori.

Diagram tidak dapat menceritakan kepada kita apakah B dimasukkan ke dalam list sebelum atau sesudah D. Di lain pihak, diagram dapat memperlihatkan kepada kita, bahwa B dimasukkan sebelum C, karena B dan C mempunyai Nomor Prioritas yang sama, dan B berada sebelum C di dalam list.

START

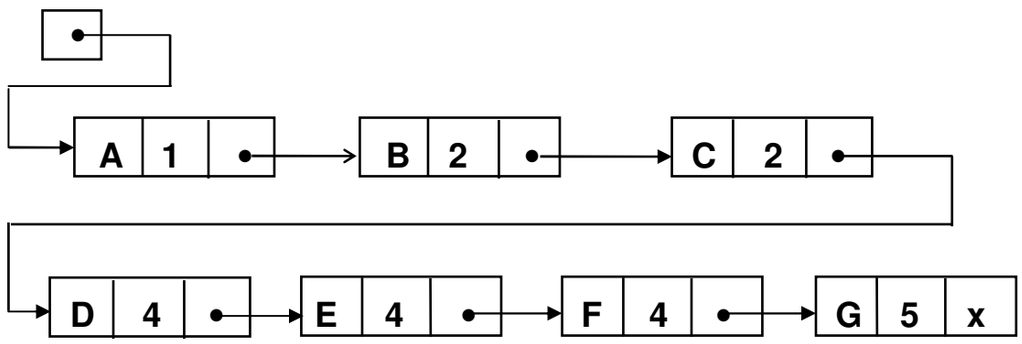
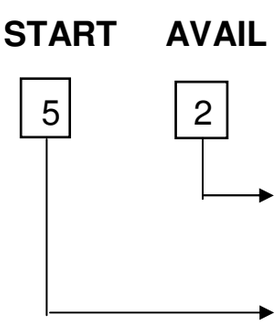


Diagram Skematik

Penyajian Dalam Memori



	INFO	PRN	LINK
1	B	2	6
2			7
3	D	4	4
4	E	4	9
5	A	1	1
6	C	2	3
7			10
8	G	5	0
9	F	4	8
10			11
11			12
12			0

Sifat utama dari penyajian One-way list dari sebuah Antrean Berprioritas adalah bahwa elemen dalam Antrean yang seharusnya diproses pertama kali selalu muncul pada bagian permulaan One-way list. Oleh karena itu, adalah sangat sederhana untuk menghilangkan dan memproses sebuah elemen Antrean Prioritas tersebut.

Algoritmanya sebagai berikut :

ALGORITMA 1
Untuk menghapus dan memproses elemen pertama dalam sebuah Antrean Berprioritas yang muncul dalam memori sebagai sebuah one-way list.

1. Pasang ITEM := INFO(START). {Langkah ini dimaksudkan untuk menyimpan data dalam simpul pertama}.
2. Hapus simpul pertama dari list.
3. Proses ITEM
4. Exit

ALGORITMA 2
Untuk menambahkan sebuah ITEM dengan Nomor Prioritas N pada suatu Antrean Berprioritas yang disimpan dalam memori sebagai sebuah one-way list.

1. Telusuri one-way list sampai ditemukan suatu simpul X yang nomor prioritasnya melebihi N. Sisipkan ITEM di depan simpul X.
2. Jika tidak ditemukan simpul semacam itu, sisipkan ITEM sebagai elemen terakhir list.

Kesulitan utama dalam Algoritma muncul dari kenyataan bahwa ITEM disisipkan sebelum simpul X. Hal ini berarti bahwa ketika menelusuri List itu, harus tetap memperhatikan alamat simpul yang mendahului simpul yang sedang diakses.

START

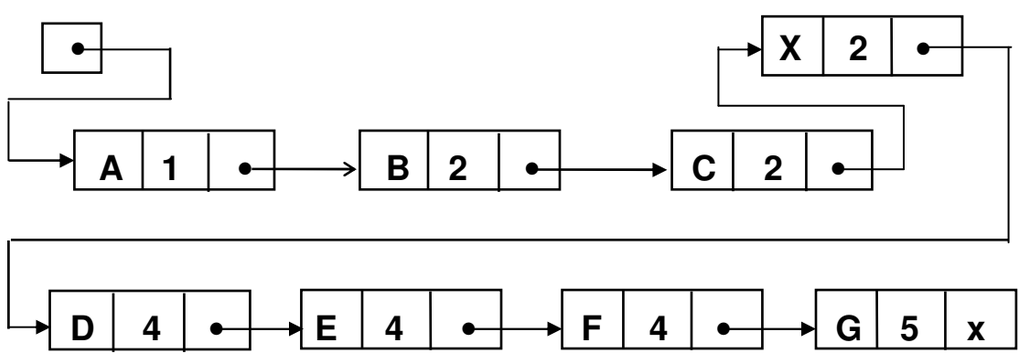
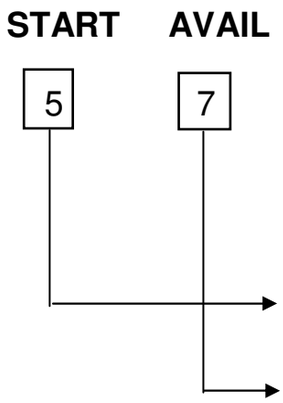


Diagram Skematik

Penyajian Dalam Memori



	INFO	PRN	LINK
1	B	2	6
2	X	2	3
3	D	4	4
4	E	4	9
5	A	1	1
6	C	2	2
7			10
8	G	5	0
9	F	4	8
10			11
11			12
12			0

PENYAJIAN ARRAY DARI ANTREAN BERPRIORITAS

Menggunakan suatu antrean terpisah untuk setiap tingkat prioritas (untuk setiap nomor prioritas). Setiap antrean akan muncul dalam array sirkularnya sendiri dan harus mempunyai sepasang penunjuk yaitu Front dan Rear.

	FRONT	REAR
1	2	2
2	1	3
3	0	0
4	5	1
5	4	4

	1	2	3	4	5	6
1		A				
2	B	C	X			
3						
4	F				D	E
5				G		