BAB 6
QUEUE (ANTRIAN)

TUJUAN PRAKTIKUM

1. Praktikan mengenal salah satu lagi tipe khusus dari link list yaitu queue/ antrian beserta seluruh operasi yang ada padanya.
2. Praktikan diharapkan dapat menerapkan teori mengenai single link list di dalam pembuatan program queue.
3. Praktikan dapat menjelaskan jenis-jenis queue.

TEORI PENUNJANG

6.1. Pengertian *Queue* (Antrian)

Queue* / Antrian** adalah suatu kumpulan data yang mana penambahan elemen hanya bisa dilakukan pada satu ujung (disebut dengan sisi belakang atau ***rear) dan penghapusan atau pengambilan elemen dilakukan lewat ujung lain (disebut dengan sisi depan atau ***front***).

Antrian menggunakan prinsip Pertama Masuk Pertama Keluar – ***First In First Out*** (**FIFO**). Dengan kata lain urutan masuk sama dengan urutan keluar. Antrian banyak dijumpai dalam kehidupan sehari-hari. Mobil-mobil yang mengantri digerbang tol untuk membeli karcis tol; orang-orang yang mengantri di loket untuk membeli karcis film juga membentuk antrian.

Pada antrian kita tidak menentukan batasan seberapa banyak antrian itu akan berakhir tapi jika kita menggunakan array untuk mengimplementasikan *queue/* tumpukan kita harus membatasi jumlah antrian yang dapat masuk. Ini dikarenakan array memiliki batasan (*upperbound*) yang menjadi penghambat jika kita menggunakan antrian. Oleh sebab itu kita akan mengimplementasikan antrian ini dengan menggunakan *link list*.

Dengan menggunakan *link list* tepatnya *Single Link List* maka elemen dapat dimasukkan secara tidak terbatas. Kita menggunakan *Header Single Link List*

yang seperti *Stack* pada posisi *Header* dapat kita pergunakan untuk menyimpan informasi mengenai banyaknya elemen dalam *Queue*.

6.2. Notasi Pada *Queue*

Notasi yang dapat digunakan didalam *Queue* Q adalah :

1. FRONT(Q) menunjukkan posisi terdepan dari suatu antrian. Contoh jika kita mempunyai antrian Q = [A,B,C,D,E] maka FRONT(Q) = A.
2. REAR(Q) menunjukkan posisi terakhir dari suatu antrian. Contoh jika kita mempunyai antrian Q = [A,B,C,D,E] maka REAR(Q) = E.
3. NOEL(Q) menunjukkan jumlah elemen di dalam Antrean Q. Contoh jika kita mempunyai antrian Q = [A,B,C,D,E] maka NOEL(Q) = 5.

6.3. Deklarasi *Queue* Dalam Link List

Pendeklarasian *Queue* di dalam link list sama seperti kita mendeklarasikan *link list*. Menggunakan pointer sebagai variabel yang menunjuk ke elemen antrian selanjutnya.

Deklarasi *Queue* menggunakan *Linked List* di dalam Pascal :

```
Type
        Queue   =   ^Simpul
        Simpul  =   Record
                        Info : Char;
                        Next : Queue;
                    End;
Var
        Head, Tail  : Queue;
        Max         : Byte;
```

Link list yang kita gunakan ialah jenis *Header Single Link List*. Kita menggunakan jenis *link list* ini karena pada bagian header dapat kita manfaatkan untuk menyimpan informasi mengenai banyaknya elemen dalam Antrian (NOEL(Q)).

6.4. Operasi Dasar Pada *Queue*

Ada 4 operasi dasar yang dapat dilakukan pada struktur data antrian, yaitu:

1. CREATE(Q)

   CREATE(Q) adalah suatu operator yang digunakan untuk membentuk dan menunjukkan suatu antrian hampa. Contoh :

   NOEL(CREATE(Q)) = 0

   FRONT(CREATE(Q)) = Tidak Terdefinisi

   REAR(CREATE(Q)) = Tidak Terdefinisi

   Berikut ini merupakan procedure CREATE simpul pada Pascal :

```
Procedure CREATE(Var Head, Tail : Queue);
Begin
        New(Head);
        Head^.Info := 0;
        Head^.Next := Head;
        Tail := Head;
End;
```

2. ISEMPTY(Q)

   ISEMPTY(Q) adalah operator yang menentukan apakah antrian Q hampa atau tidak. ISEMPTY(Q) di terapkan di dalam pascal menjadi sebuah function yang bertipe boolean sehingga hasil dari function ini akan bernilai True jika antrian dalam keadaan kosong / hampa (NOEL(Q) = 0) dan akan bernilai False jika antrian dalam keadaan terisi / tidak kosong (NOEL(Q) > 0). Contoh :

   ISEMPTY(CREATE(Q)) = True

   Berikut ini merupakan procedure ISEMPTY simpul pada Pascal :

```
Function ISEMPTY(Head : Queue);
Begin
ISEMPTY := (Head^.Next = Head);
End;
```

3. INSERT(E,Q)

   INSERT(E,Q) adalah operator yang digunakan untuk memasukkan elemen E pada antrian Q di posisi depan dari antrian. Hasil dari operator ini adalah antrian yang lebih panjang.

   Berikut ini merupakan procedure INSERT simpul pada Pascal :

```
Procedure INSERT(Elemen : Byte; Var Head, Tail : Queue);
Var Temp : Queue;
Begin
New(Temp);
Temp^.Info := Elemen;
Temp^.Next := Head;
Tail := Temp;
        Inc(Head^.Info);
End;
```

4. REMOVE(Q)

   REMOVE(Q) adalah operator yang menghapus elemen bagian depan dari antrian Q. Hasilnya merupakan antrian yang lebih pendek. Pada setiap operasi ini, harga dari NOEL(Q) berkurang satu, dan elemen kedua dari Q menjadi elemen terdepan. Jika NOEL(Q) = 0, maka REMOVE(Q) memberikan suatu kondisi erroe, yakni suatau UNDERFLOW. Contoh :

   REMOVE(CREATE(Q)) = UNDERFLOW.

   Berikut ini merupakan procedure REMOVE simpul pada Pascal :

```
Procedure REMOVE(Var Head : Queue);
Var Temp : Queue;
Begin
If Not (ISEMPTY(Head)) Then
        Begin
            Temp := Head^.Next;
            Head^.Next := Temp^.Next;
            Dispose(Temp);
            Dec(Head^.Info);
        End;
End;
```

Untuk memahami pengertian antrian sekaligus penerapan operator-operator *queue* dan notasi-notasinya perhatikan ilustrasi berikut :

CREATE(Q) = Membuat Antrian Baru

	= Q	FRONT(Q) = Tidak Terdefinisi REAR(Q) = Tidak Terdefinisi NOEL(Q) = 0

ISEMPTY(Q) = Apakah Antrian dalam keadaan kosong ?

TRUE	= Q	FRONT(Q) = Tidak Terdefinisi REAR(Q) = Tidak Terdefinisi NOEL(Q) = 0

REMOVE(Q) = Mengeluarkan elemen terdepan dari dalam antrian.

UNDERFLOW !!!	= Q	FRONT(Q) = Tidak Terdefinisi REAR(Q) = Tidak Terdefinisi NOEL(Q) = 0

INSERT(1,Q) = Memasukkan elemen 1 kedalam antrian.

1	= Q	FRONT(Q) = 1 REAR(Q) = 1 NOEL(Q) = 1

INSERT(2,Q) = Memasukkan elemen 2 kedalam antrian.

1 2	= Q	FRONT(Q) = 1 REAR(Q) = 2 NOEL(Q) = 2

INSERT(3,Q) = Memasukkan elemen 3 kedalam antrian.

1 2 3	= Q	FRONT(Q) = 1 REAR(Q) = 3 NOEL(Q) = 3

REMOVE(Q) = Mengeluarkan elemen terdepan dari dalam antrian.

2 3	= Q	FRONT(Q) = 2 REAR(Q) = 3 NOEL(Q) = 2

INSERT(4,Q) = Memasukkan elemen 3 kedalam antrian.

2 3 4	= Q	FRONT(Q) = 2 REAR(Q) = 4 NOEL(Q) = 3

ISEMPTY(Q) = Apakah Antrian dalam keadaan kosong ?

FALSE	= Q	FRONT(Q) = Tidak Terdefinisi REAR(Q) = Tidak Terdefinisi NOEL(Q) = 0

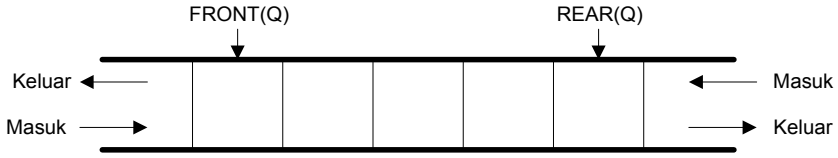
Untuk operasi-operasi selanjutnya menggunakan prinsip yang sama seperti ilustrasi diatas ini.

6.5. Jenis-jenis Antrian

Selain antrian yang telah kita bahas di atas, masih ada dua tipe antrian lagi yang penggunaannya juga banyak di dalam kehidupan sehari hari atau dalam dunia komputer itu sendiri, diantaranya adalah :

1. DEQUE

   DEQUE adalah antrian dimana elemennya bisa masuk dan keluar lewat kedua ujungnya (berbeda dengan *queue* yang hany bisa masuk lewat ujung belakang dan keluar lewat ujung depan). Biasanya DEQUE disajikan dengan menggunakan Double link list yang memiliki dua buah pointer yang menunjuk ke posisi sebelumnya dan sesudahnya. Gambar 5.1 menunjukkan struktur umum dari sebuah DEQUE.



Gambar 6.1 Struktur Umum DEQUE

DEQUE juga mempunyai dua jenis variasi yaitu :

a. Deque input terbatas : suatu deque yang membatasi pemasukkan elemen hanya pada satu ujung dari list, sementara penghapusan elemen boleh dilakukan pada kedua ujung list.
b. Deque output terbatas : merupakan kebalikan dari deque input terbatas yaitu suatu deque yang membatasi penghapusan elemen hanya pada satu ujung dari list, sementara pemasukkan elemen boleh dilakukan pada kedua ujung list.

2. ANTRIAN BERPRIORITAS

   Antrian berprioritas adalah suatu *queue* yang setiap elemennya telah diberikan sebuah prioritas, dan urutan proses penghapusan elemen adalah berdasarkan aturan berikut :

   a. Elemen yang prioritasnya lebih tinggi, diproses lebih dahulu dibandingkan dengan elemen yang prioritas lebih rendah.
   b. Dua elemen dengan prioritas yang sama, diproses sesuai dengan urutan mereka sewaktu dimasukkan ke dalam priority *queue*.

   Salah satu contoh antrian berprioritas ini adalah 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 yang biasa.

LAPORAN PENDAHULUAN

1. Jelaskan apa yang dimaksud dengan:
   a. *Queue*/Tumpukan
   b. DEQUE
   c. Antrian berprioritas.
2. Jelaskan istilah-istilah pada *queue* berikut ini:
   a. Create
   b. Insert
   c. Remove
   d. Front
   e. Rear
   f. IsEmpty
   g. Noel
3. Berikan contoh soal mengenai operasi *queue Create, Insert, Remove* dan *IsEmpty* seperti pada ilustrasi queue diatas.

LAPORAN AKHIR

1. Buatlah Algoritma dari program sebelumnya terutama mengenai ditambah dengan ringkasan tentang operasi Queue.
2. Buat program parkir mobil dengan menggunakan teori antrian yang telah diajarkan menggunakan single link list.