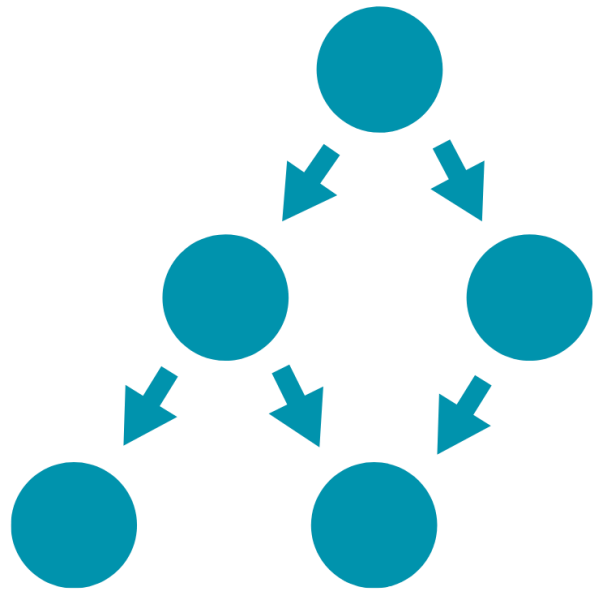


VERİ YAPILARILARI VE ALGORİTMALAR

Queue

Giriş

1. Kuyruk Kavramı
2. Kuyruk nasıl kullanılır?
3. Kuyruk soyut veri türü
4. Kuyruk uygulamaları
5. Performans ve Sınırlar
6. Uygulama
    1. Dizi ile kuyruk tasarımı
    2. Bağlı liste ile kuyruk tasarımı

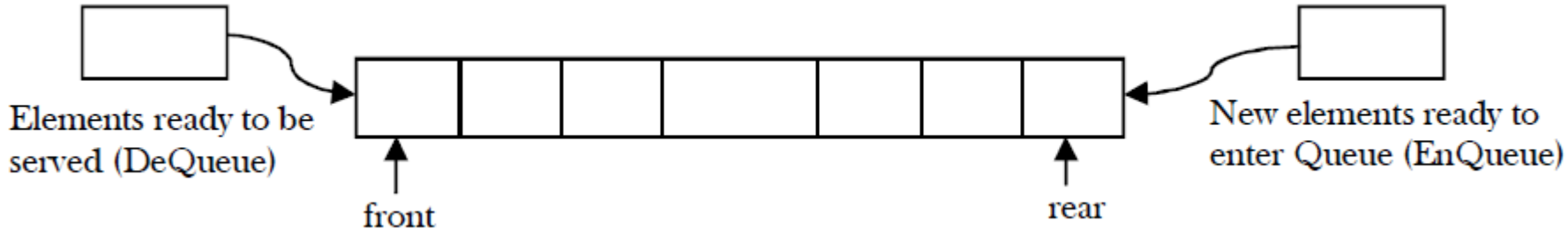
Queue

A queue is a linear data structure that follows an order in which the elements can be accessed. It is very similar to stacks, but the only difference is that a queue is ***open on both ends****.*

Queue

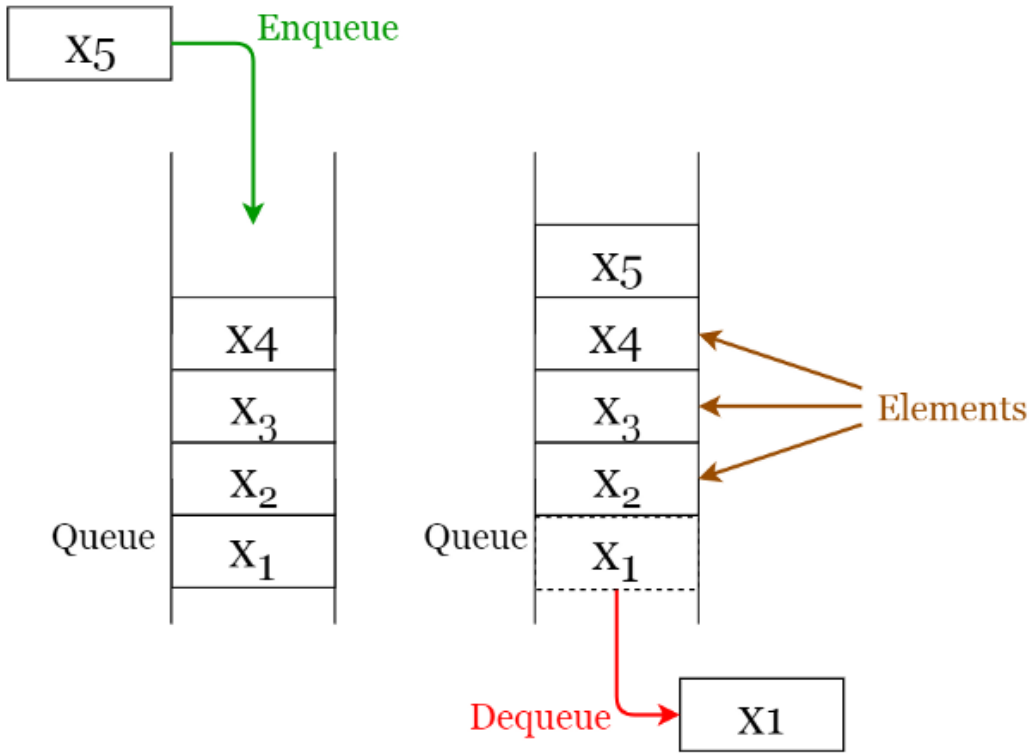
- Bilgisayar bilimleri açısından liste tabanlı veri yapıları arasında yaygın bir şekilde kullanılan veri yapılarından biri de kuyruklar.

- İlk-giren ilk-çıkar (*first-in first-out, FIFO*) çalışma ilkesine göre kuyruk işletilir.



Queue

- İlk-giren ilk-çıkar (first-in first-out, FIFO) çalışma ilkesine göre kuyruk işletilir.



Kuyruk Soyut Veri Türü

Queue Abstract Data Type

- **Ana işlevler**
  - void enQueue
  - T deQueue

- **Yardımcı işlevler**
  - front()
  - rear()
  - size()
  - isEmpty()

Kuyruk Uygulamaları

Queue Applications

- **Doğrudan Uygulamalar**
  - İşletim sistemler iş planlaması (Yazıcı kuyrukları)
  - Gerçek hayattaki kuyruk uygulamalarının modellenmesi
  - Çoklu programlama
  - Asenkron veri transferi

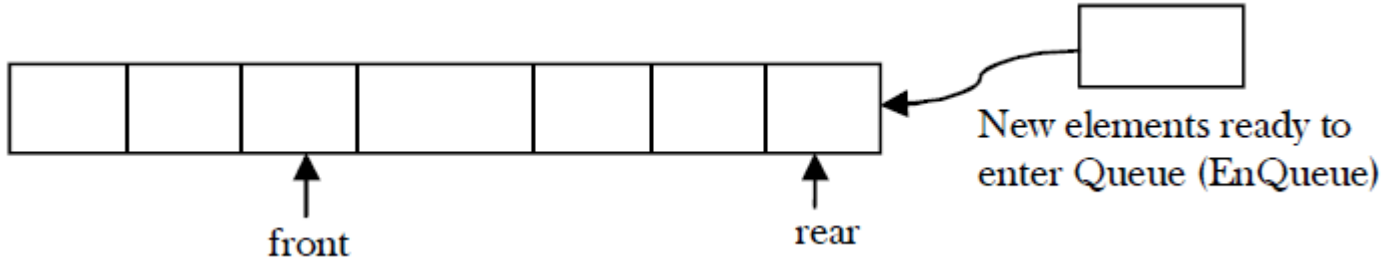
- **Yardımcı işlevler**
  - Algoritmalar için yardımcı veri türü
  - Diğer veri yapılarının bileşenleri

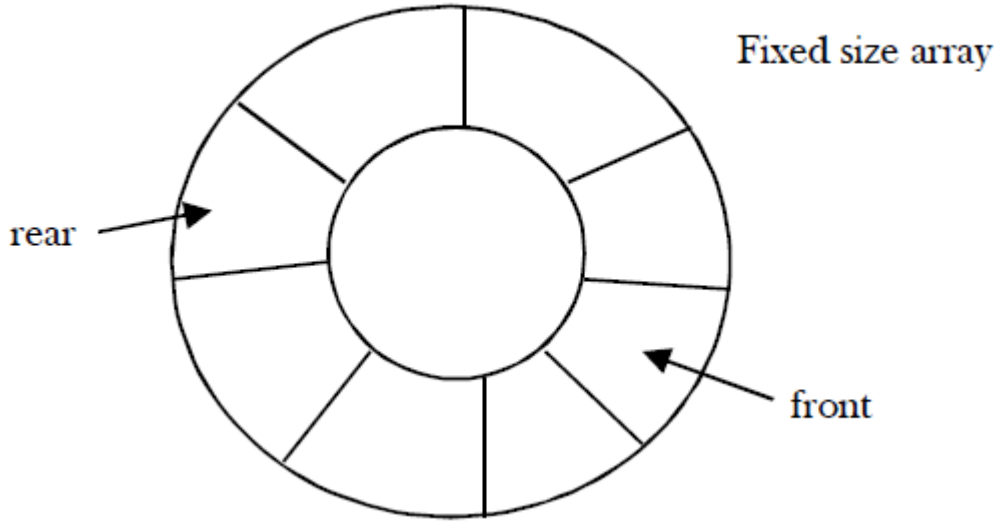
Kuyruk Uygulama

Queue Implementation

- Basit dairesel dizi tabanlı uygulama
- Dinamik dairesel dizi tabanlı uygulama
- Bağlı liste tabanlı uygulama

Çevrimsel kuyruk





Performans ve sınırlamalar

- N boyutlu bir kuyruk için:

Space Complexity (for n enQueue operations)	O(n)
Time Complexity of enQueue()	O(1)
Time Complexity of deQueue()	O(1)
Time Complexity of isEmpty()	O(1)
Time Complexity of isFull ()	O(1)
Time Complexity of size()	O(1)

isFull Kabakod

Pseudecode - isFull

```
begin procedure isfull

   if rear equals to MAXSIZE
      return true
   else
      return false
   endif

end procedure
```

Enqueue Kabakod

Pseudecode - Enqueue

```
int enqueue(int data)
   if(isfull())
      return 0;

   rear = rear + 1;
   queue[rear] = data;

   return 1;
end procedure
```

Dequeue Kabakod

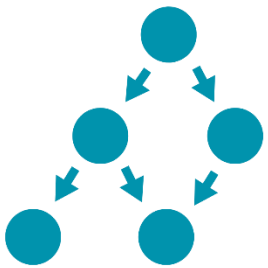
Pseudecode - Dequeue

```
procedure dequeue

   if queue is empty
      return underflow
   end if

   data = queue[front]
   front ← front + 1
   return true

end procedure
```



Veri Yapıları ve Algoritmalar

ZAFER CÖMERT
Öğretim Üyesi