Öncelikli Kuyruklar (Priority Queues)

1) Bir çeşit kuyruklar
2) Önceliği en yüksek olan kuyruktan ilk çıkar.
3) Her nesnenin karşılaştırılabilir bir öncelik değeri (key) vardır.
   - Küçük değer yüksek öncelikli dir veya tersi.

Örnek Uygulamalar
printer: Az sayfaya sahip belge önce yazılır.
işletim sistemi: En kısa zamanlı işlem önce çalışır.

ADT (Abstract data type) Soyut veri tipi

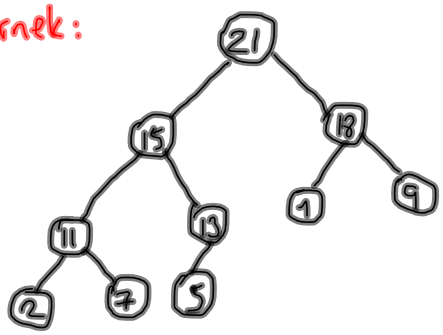
Binary Heap (İkili Yığın)

Öncelikli kuyrukları gerçekleştirmek için kullanılır.
Bir yığın son satırı (level) hariç her satırı dolu olan bir ikili ağaçtır. Son satır soldan sağa doğru doldurulur.

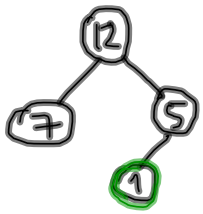
Heap-Order Property (Yığın Özelliği)

Max Heap için; her düğümün değeri çocuklarının değerlerinden büyüktür veya eşittir ($\geq$).
Min Heap için; her düğümün değeri çocuklarının değerlerinden küçüktür veya eşittir ($\leq$).

Örnek:



Heap BT'dir ama BST değildir.



→ heap değildir

Yığınlar genelde dizilerde tutulur.

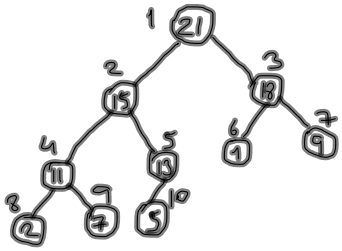
Mapping (eşleme):

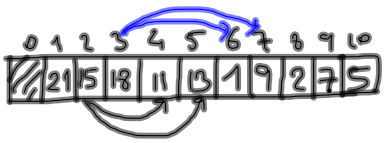
Agacın kökü : dizideki ilk eleman, i=1

parent(i) = $\lfloor i/2 \rfloor$; i düğümünün ebeveyninin indisi

left(i) = 2i ; i " sol çocuğunun "

right(i) = 2i+1 ; i " sağ " "



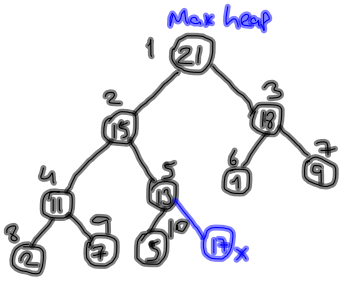


Heap işlemleri

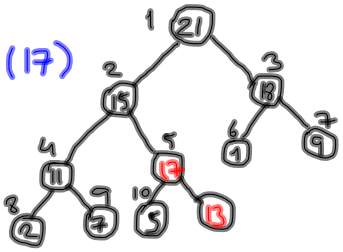
1) Insert (Ekleme)

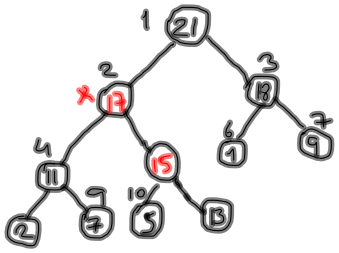
Eklenecek x değeri heap'in en alt satırında ilk boşluğa yerleştirilir.

Tekrarla:
- x değerini parent'ın değeri ile karşılaştır.
- x büyük ise swap yap.

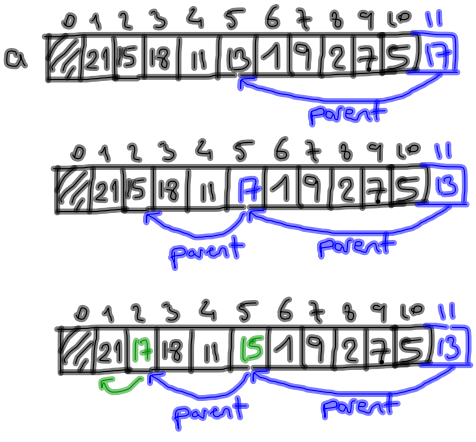


insert (17)





percolate up

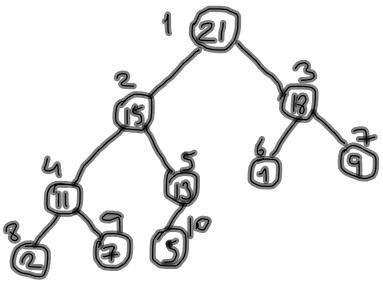



```
a[last]= x;
while(a[last]>a[last/2]){
  swap(a[last],a[last/2]);
  last = last/2;
}
```

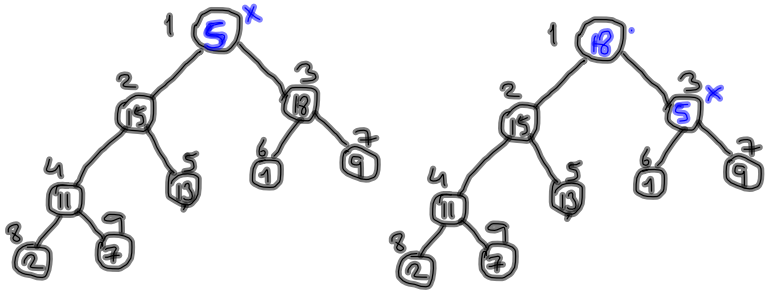
2. deleteMax (Max elemanı Silme)

Ağacın kökündeki değer silinir. Yerine ağacın en alt satırındaki en sağdaki düğüm (x) konulur.

Tekrarla: Eğer x çocuklarının değerlerinden küçük ise en büyüğü ile swap edilir.

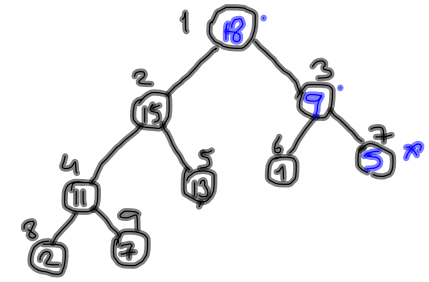


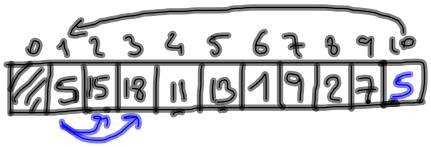
deleteMax 21 değeri geri döndürür ve heapten siler

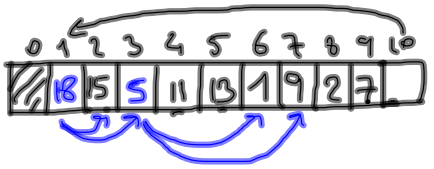


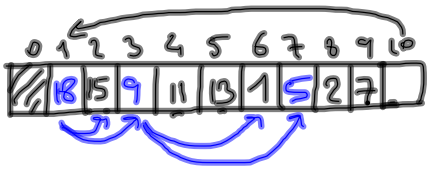
Soru: n düğüme sahip bir heap'in yüksekliği nedir?

Cevap: lgn









deleteMax

```
max= a[1];
a[1]= a[last];
i=1;
while ( a[i]<a[2i]||a[i]<a[2i+1])
  {
    if(a[2i]> a[2i+1]){
       swap(a[2i],a[i])
       i=2i;
    }else{
       swap(a[2i+1], a[i])
       i=2i+1;
    }
  }
return max;
```