ALGORITMA ANALIZI

4.HAFTA ÖDEVİ

HEAP VERİ YAPISI VE KARMAŞIKLIK ANALİZİ BEDİRHAN ÇELİK - 16040142

1. Heap yapısının diğer isimleri nelerdir?

Heap yapısının diğer ismi "Priority Queues(Öncelik Kuyrukları)" dır.

2. Heap veri yapısının iki özelliği nedir, nasıl avantaj sağlar?

Heap veri yapısının iki özelliği; Yapı Özelliği (Structure Property) ve Yığın Sırası Özelliği (Heap-Order Property) dir.

• Yapı Özelliği (Structure Property):

Bir yığın, yapı olarak " tam ikili ağaç (complete binary tree)" tır. Ağacın en alt düzeyi hariç, tüm düzeylerinin düğümlerle dolu olması gerekir. En son düzeyde olabilecek boşluklar ise ağacın tamamen sağ tarafında olmalı yani ağaç doldurulmaya hep sol taraftan başlanılmalıdır.

• Yığın Sırası Özelliği (Heap-Order Property):

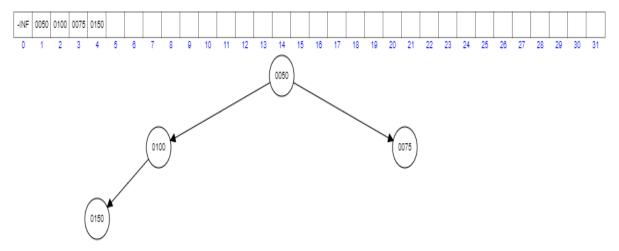
Bir düğüm her zaman çocuklarından küçük bir değere sahip olmalıdır(minHeap için). Böylece kök düğümün ağaçtaki en küçük düğüm olduğunu garanti altına almış oluruz.

3. MaxHeap yapısı ile MinHeap yapısı arasında ne fark vardır?

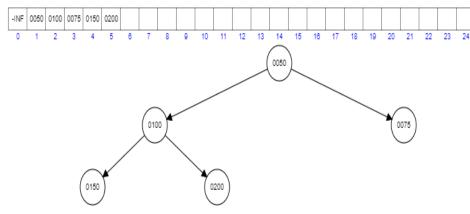
MaxHeap ve MinHeap arasındaki fark, MinHeap'in küçükten büyüğe, MaxHeap'in ise büyükten küçüğe sıralamasıdır. Yani eşitlik kontrolü.

4. Video derste geçen linkler ile bir heap üzerinde insert ve remove işlemlerini uygulayıp karmaşıklığı yorumlayınız.

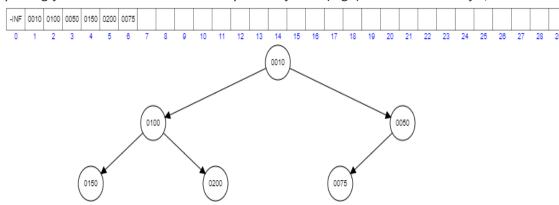
MinHeap için ekleme (insert) ve çıkarma (remove,extract) işlemleri uygularsak. Aşağıdaki heap yapısı için **EKLEME**;



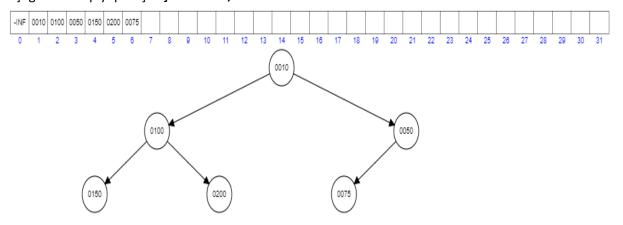
 Best Case: Eğer eklediğimiz elemanın parent'ı kendisinden küçükse yer değişikliği yapılmaz. Bu nedenden dolayı karmaşılık O(1) olur. 200 elemanı için;



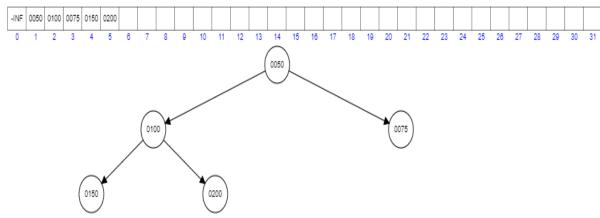
 Worst Case: Eğer eklediğimiz eleman heap'te en küçük değere sahipse maksimum yer değiştirme olur. Bu nedenden dolayı karmaşıklık O(logn) olur. 10 elemanı için;



Aşağıdaki heap yapısı için ÇIKARTMA;



 Average and Worst Case: Silme işleminde en küçük eleman yani root silinir ve listedeki en büyük eleman root'un yerine eklenir. Sonra en büyüğün sıralaması yapılır. Bu yüzden ortalama ve en kötü senaryo O(logn) dir. 10 elemanı için;



5. Heap yapısı ile amortized cost arasında nasıl bir ilişki vardır?

Heap yapısında her bir operasyonun karmaşıklığına bakılırsa (amortized cost) ekleme işleminde sabit bir zaman harcandığı görülür. Her eleman yapının en sonuna eklendiğinden O(1) zaman harcanır. Ancak bazı durumlarda eklenen elemanın parent'ı kendisinden büyükse yer değiştirme uygulanır. Böylelikle karmaşıklık O(logn) olur veya yaklaşır. Çıkarma işleminde ise neredeyse her zaman O(logn) karmaşıklığı görülür. Çünkü silinen elemanın yerine yapıdaki en büyük eleman yerleştirilir ve onun tekrar sıralanması gerekir. Yine bazı durumlarda mesela yapıda 2 eleman varsa silme işlemi O(1) olur.