Gebze Technical University
Computer Engineering

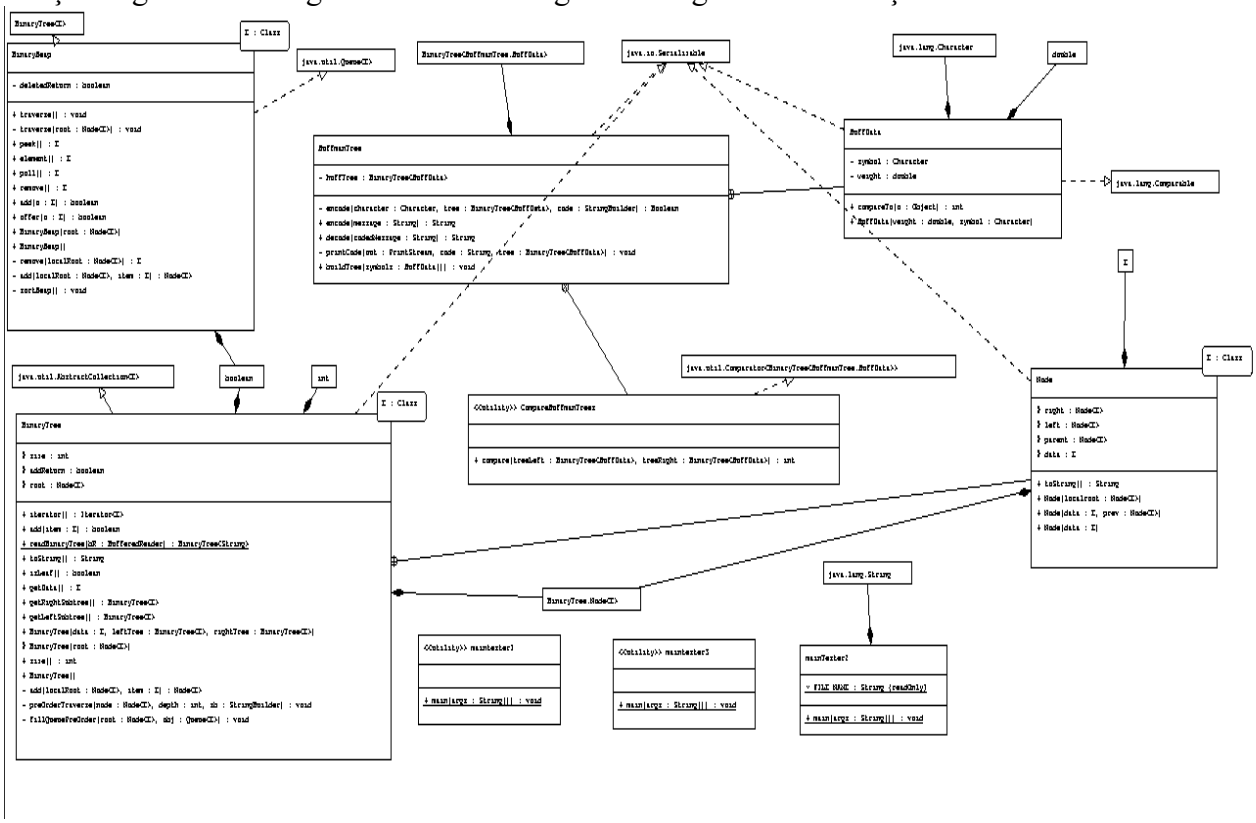
CSE 222
2017 Spring

# HOMEWORK HW06 REPORT

Burak AKTEN
141044045

Course Assistant: Nur Banu Albayrak

# 1. System Requirements
No needs for this homework.
# 2. Use Case Diagrams
No need fort his homework.
# 3. Class Diagrams
Oluşturduğum class diagram 3 sorunun diagramını bağlantılı olarak içermektedir.



# 4. Other Diagrams
No needs for this homework.
# 5. Problem Solutions Approach
- **Question 1**
  - Bu soru için BinaryTree clasını extend eden ve Queue interface'sini implement eden BinaryHeap adında bir class yazılması istenildi.
  - Bunun üzerine kitapta kullanılan BinaryTree class'ına bazı değişiklikler ve eklemeler yaparak bu classı extend ettim.Yaptığım değişiklikler şu şekildedir:
    - Öncelikle Queue interface'sini implement edeceğimizden sıfırdan implement etmemiz gereken Collection interface'sinden gelen metodların implement edilmesi için BinaryTree'yi AbstractCollection classından extend edildi. Bunun üzerine BinaryTree classında implement edilmesi gereken iki method gerekliydi (size() ve itereator()). Bundan dolayı bu iki fonnksiyon imlement edildi(iterator methodu BinaryTree'yi pre-order şekilde gezecek biçimde implement edildi.). Ayrıca karşılaştırma yapmamız gerekeceğinden E tipi Comparable'dan extend edildi.
    - BinaryHeap class'ında yapılacak olan add ve remove işlemlerinde elimizde bulunan Node'un parent'ına ihtiyacımız olunacağı düşünüldüğünden BinaryTree'de bulunan Node adında ki inner class' a 1 adet parent adında Node referansı eklendi. Ayrıca bir adet data ve parent referansı alan bir constructor eklendi.
  - BinaryTree clasın'da gerekli olan bu işlemler yapıldıktan sonra Queue interfacesinden gelen metodlar imlement edildi. Bu metodlar imlement edilirke genel

heap kurallarına uyularak implement edildi. Root olarak en küçük eleman baz alındı. Add işlemini gerçekleştiren metodlar(add ve offer) bu durumu baz alınarak imlement edildi. Remove işlemini yapan metodlar(poll ve remove) işlemi gerçekleştirdikten sonra bu kurallara göre heap'i sıraya koyacak şekilde implement edildi.

- Her bir eleman ekleme ve silme olaylarında size değeri 1 adet artacak veya azalacak şekilde implement edildi. Böylece süper class'dan gelen size metodu ile heapde bulunan node sayısı görülebilir.
- Bu soru için istenilen test yapılmadı ancak herbir fonksiyonu test etmek için gerekli işlemler yapıldı. Bu işlemler code içersinde(main metod içersinde) rahatlıkla görülmektedir.

- **Question 2**
  - Bu soru için kitapda bulunan HuffmanTree classı için bir adet encode metodu yazılması istenildi. Bunun üzerine izlenen yol şu şekildedir.
    - Öncelikle encode edilecek mesajı parametre olarak alan encode adında bir metot oluşturuldu ve Bu metod içersinde parametre olarak alınan string'in her bir indexi için HuffmanTree'yi gezerek her bir indexteki elementi '1' ve '0' lar ile temsil eden bir code ürettildi ve stringin(mesajın) sonuna gelindiğinde bu oluşturuan code'lar bir araya getirilerek return edildi.
    - Her bir indexteki elementin temsil ettiği kodun bulunması için yardımcı bir recursive fonksiyon kullanıldı.
- **Question 3**
  - Bu soru için bir önceki ödevde bulunan bir binary tree içeren bir class için bu ağacı level-order şekilde traverse eden bir metod yazılması istenildi.
  - Bunun üzerine parametre olarak root alan yardımcı bir metod kullanılarak bu işlem gerçekleştirildi. Her bir satır(level) root'un left ve right referansı kullanılarak erişildi.
  - Bir önceki ödevde bazı işlemler tam olarak gerçekleştirilmediğinden dolayı yazmış olduğum bu metodu test etmek için bu metodu ve yardımcı metodu 1. Soruda istenilen BinaryHeap class'ının içersine yerleştirerek test edildi. Yazılan metod herhangi bir binary tree'yi level-order şekilde traverse etmek için yazıldığından bu şekilde test edildi.

## 6. Test Cases

- **Question 1**
  - Add veya offer işlemi gerçekleşmeden remove , poll , peek veya element işlemleri yapıldı beklenen sonuç exception fılatılması. Sonuç exception fılatıldı ve heap'in boş olduğu söylendi.
  - Add işlemi gerçekleştirildi yaklaşıl olarak 10 adet element eklendi. Ve beklenen sonuç yönünde sıralı bir şekilde ekleme olayı gerçekleşti.
  - Sonrasında remove işlemi gerçekleştirildi. Beklenen sonuç En baştakinin silinmesi ve heap'in tekrar sıralı hale gelmesi. Bu sonuç gerçekleşti.
  - Pekk işlemi gerçekleşti ve beklenildiği şekilde gerçeklenerek eleman silinmeden head'de bulunan elemanının datası return edildi.
- **Question 2**
  - Hufftree doldurulmadan encode işlemleri yapıldı. Beklenen sonuç exception fırlatılarak mesaj verilmesi. Beklenen sonuç gerçekleşti ve tree'nin doldurulmadığı söylendi. Bunun üzerine tree herhangi bir dosya oluşturulmadan dosyadan doldurulmaya çalışıldı beklenildiği üzere exception fırlatıldı. Daha sonrasında Dosya oluşturuldu ve dosya okunara tree dolduruldu.
  - Tree dolduruldukten sonra Tree'de olmayan herhangi bir harfi içeren mesaj encode edilmek üzere metoda gönderildi. Bunun üzerine beklenen exception mesaj le

fırlatıldı.

- ❖ Daha sonrasında düzgün bir mesaj göderildi ve düzgün bir şekilde codlanmış olarak return edildi.

- **Question 3**
  - ❖ Traverse edilecek tree'ye (herhangi bir binary tree) herhangi bir ekleme olayı gerçekleşmeden traverse işlemi yapılmak istenildi. Beklenildiği üzere program bir mesaj ile exception fırlattı.
  - ❖ Gerekli ekleme işlemleri gerçekleştirildikten sonra traverse işlemi gerçekleştirilmek üzere traverse metodu çağırıldı ve beklenildiği üzere traverse edilen herbir node'un datası alt alta ekrana basıldı.

## 7. Running and Results

No needs for this homework.