Pamukkale Üniversitesi
Bilgisayar Mühendisliği
Veri Yapıları Arasınav
Soruları **(14.11.2018)**

Öğrenci Numarası : _____ Adı Soyadı : _____

Soru	1	2	3	4	5	Toplam
Puan	20	20	20	20	20	100
Not						

1. (20P) Kuyruk özelliklerini gerçekleştirmek için `Kuyruk` arayüzü verilmiştir. Parametre olarak gönderilen kuyruğun yapsını ve elemanların sırasını bozmadan istenilen değeri silip sonuç kuyruğunu döndüren `kuyruktanSil` metodunu yazın.

```
interface Kuyruk<T> {
    public void ekle(T deger);//Sona ekle
    public T sil();//Baştan sil
    public Kuyruk<T> klonla();//Kopya oluştur
    public boolean bos();//kuyruğun boş olup olmadığını döndürür
    public Kuyruk<T> bosKuyrukOlustur();//T tipinde boş kuyruk oluşturur
}
```

```
public static<T> Kuyruk<T> kuyruktanSil(T silinecek, Kuyruk<T> kuyruk){









    return
}
```

2. Boş bir ikili arama ağacına:
   (a) (8P) 76, 60, 56, 57, 55, 97, 66, 70 değerlerini ekleyin

(b) (9P) Önce değer(preorder), ortada değer(inorder) ve sonra değer(postorder) gezinmelerini yazın

(c) (3P) 76 ve 70 değerlerini önceli(predecessor) kullanarak silin

3. (20P) Aşağıda ikili yığın yapısını gerçekleştirmek için BinaryHeap sınıfı verilmiştir. BinaryHeap sınıfı içindeki kok, solCocuk ve sagCocuk metodlarının yazılı olduğu varsayılacaktır. `kok` metodu ağacın kök değerini verir. **solCocuk** ve **sagCocuk** metodları verilen parametre değerine göre çocuğun değerini verir, eğer çocuğu yoksa **null** döndürür. Bu metodları kullanarak parametre olarak verilen elemanın yığında olup olmadığını döndüren `varmi` metodunu yazın(BinaryHeap sınıfındaki BTNode sınıfı veya dizi kullanılmamaktadır. Metodların iç detayları önemli değildir, sadece ne iş yaptıklarına odaklanmanız gerekmetedir.)

```
public class BinaryHeap<T> {
public T kok(){ /*kökün değerini verir*/ }
public T solCocuk(T ebeveyn){ /* sol çocuğu verir. sol çocuk yoksa null
↪  döndürür */}
public T sagCocuk(T ebeveyn){ /* sağ çocuğu verir. sağ çocuk yoksa null
↪  döndürür */}
    public boolean varmi(T aranan){
        return varmi(kok(), aranan);
    }
    public boolean varmi(T mevcut, T aranan){









    }
}
```

4. Boş bir **AVL** ağacına:

   (a) (15P) 27, 44, 33, 91, 6, 2, 28, 60, 15 değerlerini döndürme işlemlerini göstererek ekleyin

   (b) (5P) 2 değerini silin ve gerek olursa döndürme işlemlerini yapın

5. Boş bir **2-3** ağacına:

   (a) (15P) 24, 27, 29, 26, 44, 12, 14, 13 değerlerini ekleyin. Parçalama işlemlerini yeni ağaç çizerek gösterin.

   (b) (5P) 24 değerini öncele(predecessor) göre silin