GEBZE TEKNİK ÜNİVERSİTESİ

2016 – 2017

BAHAR DÖNEMİ

CSE 222 – VERİ YAPILARI VE ALGORİTMALAR

HW7 – RAPOR

HAKKI ERDEM DUMAN

151044005

TEST CASE

SORU 1:

Objemiz “turkey” olsun. Metotların ne döndürdükleri ve özel durumlarda ne döndürmesi gerektikleri aşağıda belirtilmiştir.

turkey.put(K key, V value);

Eklenen entry’nin value’sunu döndürür.

turkey.entrySet()

Bir “Entry” kümesi döndürür.

turkey.firstEntry()

Senaryo 1: Key değeri en küçük olan entry’yi döndürür.

Senaryo 2: Ağaç boş ise null döndürür.

turkey.lastEntry()

Senaryo 1: Key değeri en büyük olan entry’yi döndürür.

Senaryo 2: Ağaç boş ise null döndürür.

turkey.lowerEntry(K key)

Senaryo 1: Key değeri, verilen parametreden küçük olmak şartı ile, en büyük olan entry’yi döndürür.

Senaryo 2: Girilen key ağaçta yok ise null döndürür.

Senaryo 3: Girilen key null ise NullPointerException fırlatır.

turkey.lowerKey(K key)

Senaryo 1: Key değeri, verilen parametreden küçük olmak şartı ile, en büyük olan key'i döndürür.

Senaryo 2: Girilen key ağaçta yok ise null döndürür.

Senaryo 3: Girilen key null ise NullPointerException fırlatır.

turkey.floorEntry(K key)

Senaryo 1: Key değeri, verilen parametreden küçük veya eşit olmak şartı ile, en büyük olan entry'i döndürür.

Senaryo 2: Girilen key ağaçta yok ise null döndürür.

Senaryo 3: Girilen key null ise NullPointerException fırlatır.

turkey.floorKey(K key)

Senaryo 1: Key değeri, verilen parametreden küçük veya eşit olmak şartı ile, en büyük olan key'i döndürür.

Senaryo 2: Girilen key ağaçta yok ise null döndürür.

Senaryo 3: Girilen key null ise NullPointerException fırlatır.

turkey.ceilingEntry(K key)

Senaryo 1: Key değeri, verilen parametreden büyük veya eşit olmak şartı ile, en küçük olan entry'i döndürür.

Senaryo 2: Girilen key ağaçta yok ise null döndürür.

Senaryo 3: Girilen key null ise NullPointerException fırlatır.

turkey.ceilingKey(K key)

Senaryo 1: Key değeri, verilen parametreden büyük veya eşit olmak şartı ile, en küçük olan key'i döndürür.

Senaryo 2: Girilen key ağaçta yok ise null döndürür.

Senaryo 3: Girilen key null ise NullPointerException fırlatır.

turkey.higherEntry(K key)

Senaryo 1: Key değeri, verilen parametreden büyük olmak şartı ile, en küçük olan entry'i döndürür.

Senaryo 2: Girilen key ağaçta yok ise null döndürür.

Senaryo 3: Girilen key null ise NullPointerException fırlatır.

turkey.higherKey(K key)

Senaryo 1: Key değeri, verilen parametreden büyük olmak şartı ile, en küçük olan key'i döndürür.

Senaryo 2: Girilen key ağaçta yok ise null döndürür.

Senaryo 3: Girilen key null ise NullPointerException fırlatır.

turkey.subMap(K fromKey, boolean fromInclusive, K toKey, boolean toInclusive)

Senaryo 1: fromInclusive değerinin true oluşu fromKey değerini ve toInclusive değerinin true oluşu toKey değerini dahil etmekle beraber bu iki key arasındaki entryleri döndürür.

Senaryo 2: fromKey değeri veya toKey değeri ağaç içerisinde yok ise IllegalArgumentException fırlatır.

Senaryo 3: fromKey değeri, toKey değerinden büyük ise IllegalArgumentException fırlatır.

Senaryo 4: fromKey değeri veya toKey değeri null ise NullPointerException fırlatır.

turkey.subMap(K fromKey, K toKey)

Senaryo 1: fromKey ve toKey değerleri dahil olmakla beraber bu iki key arasındaki entryleri döndürür.

Senaryo 2: fromKey değeri veya toKey değeri ağaç içerisinde yok ise IllegalArgumentException fırlatır.

Senaryo 3: fromKey değeri, toKey değerinden büyük ise IllegalArgumentException fırlatır.

Senaryo 4: fromKey değeri veya toKey değeri null ise NullPointerException fırlatır.

turkey.headMap(K toKey, K inclusive)

Senaryo 1: İlk eleman ile toKey dahil olmak üzere (inclusive true olmak şartıyla) toKey arasındaki entryleri döndürür.

Senaryo 2: toKey değeri ağaç içerisinde yok ise IllegalArgumentException fırlatır.

Senaryo 3: toKey değeri null ise NullPointerException fırlatır.

turkey.headMap(K toKey)

Senaryo 1: İlk eleman ile toKey dahil olmak üzere toKey arasındaki entryleri döndürür.

Senaryo 2: toKey değeri ağaç içerisinde yok ise IllegalArgumentException fırlatır.

Senaryo 3: toKey değeri null ise NullPointerException fırlatır.

turkey.tailMap(K fromKey, K inclusive)

Senaryo 1: fromKey dahil olmak üzere (inclusive true olmak şartıyla) fromKey ile son eleman arasındaki entryleri döndürür.

Senaryo 2: fromKey değeri ağaç içerisinde yok ise IllegalArgumentException fırlatır.

Senaryo 3: fromKey değeri null ise NullPointerException fırlatır.

turkey.tailMap(K fromKey)

Senaryo 1: fromKey dahil olmak üzere fromKey ile son eleman arasındaki entryleri döndürür.

Senaryo 2: fromKey değeri ağaç içerisinde yok ise IllegalArgumentException fırlatır.

Senaryo 3: fromKey değeri null ise NullPointerException fırlatır.

turkey.firstKey()

Senaryo 1: Key değeri en küçük olan key'i döndürür.

Senaryo 2: Tree boş ise NoSuchElementException fırlatır.

turkey.lastKey()

Senaryo 1: Key değeri en büyük olan key'i döndürür.

Senaryo 2: Tree boş ise NoSuchElementException fırlatır.

turkey.navigableKeySet()

Sadece keylerin bulunduğu bir küme döndürür.

turkey.pollFirstEntry()

Senaryo 1: Key değeri en küçük olan elemanı siler ve döndürür.

Senaryo 2: Tree boş ise null döndürür.

turkey.pollLastEntry()

Senaryo 1: Key değeri en büyük olan elemanı siler ve döndürür.

Senaryo 2: Tree boş ise null döndürür.

SORU 2:

Objemiz “turkey” olsun. Metotların ne döndürdükleri ve özel durumlarda ne döndürmesi gerektikleri aşağıda belirtilmiştir.

turkey.put(K key, V value)

Senaryo 1: Eklenen entry’nin value değerini döndürür.

Senaryo 2: Gönderilen key hash table içerisinde varsa, hash table içerisindeki key’in value değerini, parametre olarak gönderilen value değeri ile günceller.

turkey.size()

Hash table içerisindeki key sayısını döndürür.

turkey.get(K key)

Senaryo 1: Parametre olarak verilen key’in karşılık geldiği value’yu döndürür.

Senaryo 2: Gönderilen key, hash table içerisinde yok ise null döndürür.

turkey.remove(K key)

Senaryo 1: Parametre olarak gönderilen key’i içeren entry’yi siler.

Senaryo 2: Gönderilen key, hash table içerisinde yok ise null döndürür.

turkey.isEmpty()

Hash table boş ise true, dolu ise false döndürür.

PROBLEM SOLUTION APPROACH

SORU 1:

- Yazdığım kodun üzerinde daha fazla hakimiyet kurmak için, “Entry” interface’ini implement eden “Entries” inner class’ını yazdım.

- Gerekli classlarda, Generic tipin Comparable class’ını extend etmesini sağlayarak, kendi compareTo metodumu (keyleri kıyaslıyor) implement ettim ve Comparator implement’ine gerek kalmadı.

- isThereAnyKey metodunu yazarak, parametre olarak verilen key’in tree içerisinde olup olmadığını tespit etmeye çalıştım.

- Binary Search Tree içerisinde in-order bir şekilde traverse edip bu elemanları, bir ArrayList içerisine traverse sırasında attım. In-order traverse’i kullandım çünkü Binary Search Tree’yi küçükten büyüğe dolaşmış oluyor. ArrayList’i kullanmamın sebebi ise bütün işleri array üzerinde yapmanın çok daha kolay olması.

SORU 2:

- HashTableOpen adlı inner class içerisine ArrayList tipinde bir data field oluşturdum ve bu field’a ulaşmak için 3 adet metot yazdım. Bunlardan bir tanesi array’i döndürürken diğeri verilen indexteki değeri döndürüyor ve bir diğeri ise verilen index’e verilen değeri atıyor.

- İşimi kolaylaştırması adına whereIs diye bir metot yazdım. Bu metot şöyle çalışıyor: Eğer gönderilen key’in hashCode’una karşılık gelen index’in içerisi boş ise -1 return ediyor. Eğer bu index’in içerisinde belirli elemanlar varsa fakat bu elemanların hiçbirinin key değeri, parametre olarak gönderilen key değerini tutmuyorsa -2 return ediyor. Bütün bunlar olmadığı takdirde return ettiği değer ise, gönderilen key’in bulunduğu inner index oluyor.

- hashCodeGenerator adlı metot, hash table’ın size’ını dikkate alarak, key için bir hashCode return ediyor.

EKRAN GÖRÜNTÜLERİ

Soru 1: Test

```
public static Boolean Q1Test(){

    try {
        NavigableMap<String, String> turkey = new BinaryNavMap<~>();
        turkey.put("uskudar", "istanbul");
        turkey.put("kadıkoy", "istanbul");
        turkey.put("cekirge", "bursa");
        turkey.put("gebze", "kocaeli");
        turkey.put("niksar", "tokat");
        turkey.put("kecıoren", "ankara");
        turkey.put("aksaray", "istanbul");
        turkey.put("foca", "izmir");
        turkey.put("manavgat", "antalya");
        turkey.put("kahta", "adıyaman");
        turkey.put("biga", "canakkale");
```

```
System.out.println("THE ORIGINAL SET ODDS IS: " + turkey);
System.out.println("\nentrySet() METHOD: " + turkey.entrySet() + "\n");
System.out.println("firstEntry() METHOD: " + turkey.firstEntry() + "\n");
System.out.println("lastEntry() METHOD: " + turkey.lastEntry()+ "\n");
System.out.println("lowerEntry() METHOD: " + turkey.lowerEntry( key: "gebze")+ "\n");
System.out.println("lowerKey() METHOD: " + turkey.lowerKey( key: "cekirge")+ "\n");
System.out.println("floorEntry() METHOD: " + turkey.floorEntry( key: "kadıkoy")+ "\n");
System.out.println("floorKey() METHOD: " + turkey.floorKey( key: "kahta")+ "\n");
System.out.println("ceilingEntry() METHOD: " + turkey.ceilingEntry( key: "kecıoren")+ "\n");
System.out.println("ceilingKey() METHOD: " + turkey.ceilingKey( key: "foca")+ "\n");
System.out.println("higherEntry() METHOD: " + turkey.higherEntry( key: "manavgat")+ "\n");
System.out.println("higherKey() METHOD: " + turkey.higherKey( key: "niksar")+ "\n");
System.out.println("subMap() METHOD: " + turkey.subMap("gebze",true, "uskudar", false)+ "\n");
System.out.println("subMap() METHOD WITH RETURN SortedMap: " + turkey.subMap("gebze", "uskudar")+ "\n");
System.out.println("headMap() METHOD: " + turkey.headMap("gebze",false)+ "\n");
System.out.println("headMap() METHOD WITH RETURN SortedMap: " + turkey.headMap( toKey: "gebze")+ "\n");
System.out.println("tailMap() METHOD: " + turkey.tailMap("gebze",false)+ "\n");
System.out.println("tailMap() METHOD WITH RETURN SortedMap: " + turkey.tailMap( fromKey: "gebze")+ "\n");
System.out.println("firstKey() METHOD: " + turkey.firstKey()+ "\n");
System.out.println("lastKey() METHOD: " + turkey.lastKey()+ "\n");
System.out.println("navigableKeySet() METHOD: " + turkey.navigableKeySet()+ "\n");
System.out.println("pollFirstEntry() METHOD: " + turkey.pollFirstEntry()+ "\n");
System.out.println("pollLastEntry() METHOD: " + turkey.pollLastEntry()+ "\n");
System.out.println("NEW SET AFTER DELETION FIRST AND THE LAST ENTRY: " + turkey);

    }
    catch(IllegalArgumentException e){
        System.out.println("ERROR: there is a problem with arguments that you give.");
    }
    catch(NullPointerException e){
        System.out.println("ERROR: THERE IS A NULL POINTER EXCEPTION!");
    }
    return Boolean.TRUE;
```

Soru 1: Sonuç

```
QUESTION 1:


SO, HERE ARE THE TESTS:

THE ORIGINAL SET ODDS IS: {aksaray=istanbul, biga=canakkale, cekirge=bursa, foca=izmir, gebze=kocaeli, kadıkoy=istanbul, kahta=adıyaman, kecıoren=ankara, manavgat=antalya, niksar=tokat, uskudar=istanbul}

entrySet() METHOD: [{aksaray=istanbul}, {biga=canakkale}, {cekirge=bursa}, {foca=izmir}, {gebze=kocaeli}, {kadıkoy=istanbul}, {kahta=adıyaman}, {kecıoren=ankara}, {manavgat=antalya}, {niksar=tokat}, {usku

firstEntry() METHOD: {aksaray=istanbul}

lastEntry() METHOD: {uskudar=istanbul}

lowerEntry() METHOD: {foca=izmir}

lowerKey() METHOD: biga

floorEntry() METHOD: {kadıkoy=istanbul}

floorKey() METHOD: kahta

ceilingEntry() METHOD: {kecıoren=ankara}

ceilingKey() METHOD: foca

higherEntry() METHOD: {niksar=tokat}

higherKey() METHOD: uskudar

subMap() METHOD: {gebze=kocaeli, kadıkoy=istanbul, kahta=adıyaman, kecıoren=ankara, manavgat=antalya, niksar=tokat}

subMap() METHOD WITH RETURN SortedMap: {gebze=kocaeli, kadıkoy=istanbul, kahta=adıyaman, kecıoren=ankara, manavgat=antalya, niksar=tokat, uskudar=istanbul}

headMap() METHOD: {aksaray=istanbul, biga=canakkale, cekirge=bursa, foca=izmir}

headMap() METHOD WITH RETURN SortedMap: {aksaray=istanbul, biga=canakkale, cekirge=bursa, foca=izmir, gebze=kocaeli}

tailMap() METHOD: {kadıkoy=istanbul, kahta=adıyaman, kecıoren=ankara, manavgat=antalya, niksar=tokat, uskudar=istanbul}

tailMap() METHOD WITH RETURN SortedMap: {gebze=kocaeli, kadıkoy=istanbul, kahta=adıyaman, kecıoren=ankara, manavgat=antalya, niksar=tokat, uskudar=istanbul}

firstKey() METHOD: aksaray

lastKey() METHOD: uskudar

navigableKeySet() METHOD: [aksaray, biga, cekirge, foca, gebze, kadıkoy, kahta, kecıoren, manavgat, niksar, uskudar]

pollFirstEntry() METHOD: {aksaray=istanbul}

pollLastEntry() METHOD: {uskudar=istanbul}

NEW SET AFTER DELETION FIRST AND THE LAST ENTRY: {biga=canakkale, cekirge=bursa, foca=izmir, gebze=kocaeli, kadıkoy=istanbul, kahta=adıyaman, kecıoren=ankara, manavgat=antalya, niksar=tokat}
```

Soru 2: Test ve Sonuç

```java
public static Boolean Q2Test(){
    HashMap<String,String> turkey=new HashTableChaining<~>();
    turkey.put("edremit","balikesir");
    turkey.put("edremit","van");
    turkey.put("kemalpasa","bursa");
    turkey.put("kemalpasa","izmir");
    turkey.put("ortakoy","istanbul");//we assume a district
    turkey.put("ortakoy","aksaray");
    turkey.put("ortakoy","corum");
    turkey.put("kecıoren","ankara");
    turkey.put("pinarbasi","kastamonu");
    turkey.put("pinarbasi","kayseri");
    turkey.put("eregli","konya");
    turkey.put("eregli","tekirdag");
    turkey.put("eregli","zonguldak");
    turkey.put("golbasi","adıyaman");
    turkey.put("golbasi","ankara");
    turkey.put("biga","canakkale");

    System.out.println(turkey);
    System.out.println("\nSize: " + turkey.size());
    System.out.println("Get: " + turkey.get("pinarbasi"));
    System.out.println("Deleted: " + turkey.remove( key: "golbasi"));
    System.out.println(turkey);
```

```
QUESTION 2:


Table[1] ---  0.Element ---  Key: pinarbasi, Value: kayseri
Table[3] ---  0.Element ---  Key: ortakoy, Value: corum
Table[3] ---  1.Element ---  Key: biga, Value: canakkale
Table[4] ---  0.Element ---  Key: edremit, Value: van
Table[5] ---  0.Element ---  Key: kemalpasa, Value: izmir
Table[5] ---  1.Element ---  Key: golbasi, Value: ankara
Table[6] ---  0.Element ---  Key: kecıoren, Value: ankara
Table[6] ---  1.Element ---  Key: eregli, Value: zonguldak

Size: 8
Get: kayseri
Deleted: ankara

Table[1] ---  0.Element ---  Key: pinarbasi, Value: kayseri
Table[3] ---  0.Element ---  Key: ortakoy, Value: corum
Table[3] ---  1.Element ---  Key: biga, Value: canakkale
Table[4] ---  0.Element ---  Key: edremit, Value: van
Table[5] ---  0.Element ---  Key: kemalpasa, Value: izmir
Table[6] ---  0.Element ---  Key: kecıoren, Value: ankara
Table[6] ---  1.Element ---  Key: eregli, Value: zonguldak

Your tests is done. Make sure that you test all methods of class!!

Process finished with exit code 0
```