

Gebze Technical University
Computer Engineering

CSE 222
2017 Spring

HOMEWORK 7 REPORT

AYDIN ÇALIKOĞLU
141044078

Course Assistant:
Nur Banu ALBAYRAK

1. Problem Solutions Approach

Q1 :

BinaryNavMap Class;

BinaryNavMap class 'ı içerisinde BST Yapısını kullanarak innerClass Node sınıfını güncelleyerek içerisine Parent Node ekledim ve AbstractMap sınıfından SimpleEntry Clasını Node içerisine extend ederek, NavMap sınıfımı BST ye uygun Hale getirdim. Order Bir şekilde kullabilmek için Inorder dolaşan bir iterator method oluşturdum.

Put : BST içerisindeki recursive add methodunu güncelleyerek. Entry eklenebilir bir add methodu gerçekledim. $T(N)=O(N)$

Get : find methodu ile aradığım key değerini ağaç içersindeki keyleri karşılaştırarak en kısa sürede buldum. $T(N)=O(N)$

Remove : BST içerisindeki recursive delete methodu güncelleyerek ile gerçekleştirdim. $T(N)=O(N)$

entrySet() : EntrySet isimli bir inner class oluşturarak. Absract Set yapısını ekledik, bu sayede innerClass içerisine BST içerisinde Inorder doloşan bir bir fonksiyon yardımı ile EnrySet classını en kısa süreli gerçekleştirdim. $T(N)=O(1)$

lowerEntry: Recursive bir method ile Node içerisinde key değerlerini karşılaştırarak, verilen key değerinden küçük en büyük Entry ya da null olarak gerçekleştirdim. $T(N)=O(N)$

lowerKey: lowerEntry içerisindeki key değeri $T(N)=O(N)$

floorEntry: Recursive bir method ile Node içerisinde key değerlerini karşılaştırarak, verilen key değerinden küçük en büyük Entry ya da null olarak gerçekleştirdim. $T(N)=O(N)$

floorKey : floorEntry methodu içerisindeki key çağırdım. $T(N)=O(N)$

ceilingEntry : verilen key değerinden büyük veya eşit olan en küçük Entry yapısını recursive bir method yarımıyla buldurdum. $T(N)=O(N)$

ceilingKey: ceilingEntry methodunu kullanarak key değerini elde ettim. $T(N)=O(N)$

higherEntry : verilen key değerinden büyük olan , en küçük Entry recursive method ile BST içerisindeki key değerlerini karşılaştırarak elde ettim. $T(N)=O(N)$

higherKey : HigherEntry methodundan key değerini elde ettim. $T(N)=O(N)$

firstEntry: Inorder Iterator ile elde ettiğim ilk değer gönderdim.

LastEntry: recursive bir method ile root node üzerinden son rightNode kadar giderek birkaç işlem ile lastEntry elde ettim.

pollFirstEntry : recursive bir method ile ek değeri bularak. BST üzerinden çıkardım.

pollLastEntry : recursive bir method ile eb değeri bularak BST üzerinden çıkardım.

DescendingMap, NavigableSet, DescendingKeySet, SubMap, headMap, tailMap,subMap yapacak vaktim olmadığından(sınav var) ve siz kolay dediğinizden treeMap kullanarak gerçekleştirdim. Bu methodlar bile çok fazlaydı. Hiçbir ödev bu kadar zor değildi bir de sınav öncesi.

Q2 :

HashTableChaining Class;

HashTableOpen class ı oluşturarak openAdressing methodunu uygun bir hashtable yapısını oluşturdum, oluştururken aynı key değerinin value extra eklenebilmesini sağladım. Böylece bir ilçe bir çok il' de olabildiğini sağladım.

HashTableChaining içerisinde HashTableOpen array olarak oluşturarak, her ilçe key değerine uygun bir il listesi oluşturan ve value değerlerine göre ekleme yapabilen bir hashTable oluşturdum. Böylece hem ilçeler hem de iller için hashTable yapısı için çalışan chaining methodu oluşturdum. toString methodunu override ederek, oluşturulan tüm verileri görüntüleyebildim.

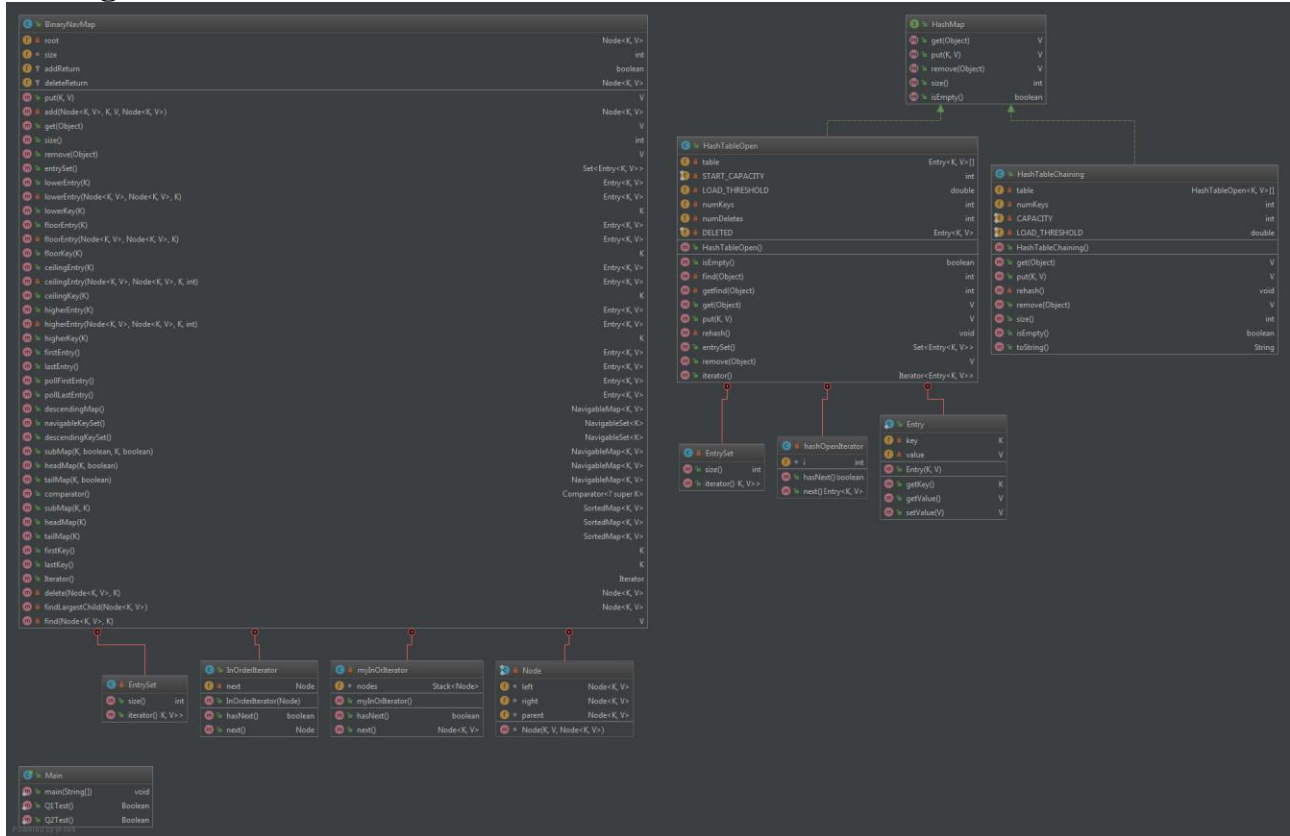
put: key hashcode üzerinden bir index değeri oluşturarak, index değerinin size üzerindeki mutlak modu ile ekleme işlemlerini gerçekleştirdim. $T_{best}(n) = O(1)$ $T_{worst}(N) = O(N)$

get: key hashcode üzerinden bir index değeri oluşturarak, index değerinin size üzerindeki mutlak modu ile get işlemini gerçekleştirdim. $T_{best}(n) = O(1)$ $T_{worst}(N) = O(N)$

remove: oluşan index değeri ile elde ettiğim ilk değeri sildim. $T(N)=O(1)$

toString : HashTable içerisindeki tüm hashTable yapısını dolaşarak il ve ilçeleri listeledim. $T(N)=O(n^2)$

2. Diagrams



3. Test Cases

Q1:

```
try
{

    System.out.println("The entrySet is " +turkey.entrySet());
    System.out.println("The lowerEntry is " +turkey.lowerEntry("cekirge"));
    System.out.println("The lowerKey is " +turkey.lowerKey("cekirge"));
    System.out.println();

    System.out.println("The floorEntry is " +turkey.floorEntry("c"));
    System.out.println("The floorKey is " +turkey.floorKey("c"));
    System.out.println();

    /* floor equal test */
    System.out.println("The floorEntry(equal test) is "
+turkey.floorEntry("cekirge"));
    System.out.println("The floorKey(equal test) is "
+turkey.floorKey("cekirge"));
    System.out.println();

    System.out.println("The ceilingEntry is " +turkey.ceilingEntry("d"));
    System.out.println("The ceilingKey is " +turkey.ceilingKey("d"));
    System.out.println();

    System.out.println("The ceilingEntry(equal test) is "
+turkey.ceilingEntry("foca"));
    System.out.println("The ceilingKey(equal test) is "
+turkey.ceilingKey("foca"));
    System.out.println();

    System.out.println("The higherEntry is " +turkey.higherEntry("foca"));
    System.out.println("The higherEntry is " +turkey.higherKey("foca"));
    System.out.println();

    System.out.println("The FirstEntry is "+turkey.firstEntry());
    System.out.println();

    System.out.println("The LastEntry is "+turkey.lastEntry());
    System.out.println();

    System.out.println("The firstKey is "+turkey.firstKey());
    System.out.println();

    System.out.println("The lastKey is "+turkey.lastKey());
    System.out.println();

    System.out.println("The pollFirstEntry is "+turkey.pollFirstEntry());
    System.out.println();

    System.out.println("The pollLastEntry is "+turkey.pollLastEntry());
    System.out.println();

    System.out.println("The desMap is "+turkey.descendingMap());
    System.out.println();

    System.out.println("The navigableKeySet is "+turkey.navigableKeySet());
    System.out.println();
```

```

        System.out.println("The descendingKeySet is "+turkey.descendingKeySet());
        System.out.println();

        System.out.println("The headMap is "+turkey.headMap("uskudar",true));
        System.out.println();

        System.out.println("The tailMap is "+turkey.tailMap("manavgat"));
        System.out.println();

        System.out.println("The original set odds is " + turkey);
        NavigableMap<String,String> m = turkey.subMap("gebze",false,"uskudar",true);
        System.out.println("The ordered set m is " + m);
        System.out.println("The first entry is " +turkey.firstEntry());
    }
    catch (NullPointerException e) {
        System.out.println("NullPointerException : Variable Key can not be null");
    }
    catch (ClassCastException e){
        System.out.println("ClassCast Exceptions : param Key can not Compered key be null");
    }
    catch (NoSuchElementException e){
        System.out.println("NoSuchElementException : param Key can not Compered key be null");
    }
    catch (IllegalArgumentException e){
        System.out.println("IllegalArgumentException : fromKey | toKey");
    }
}

```

Q2:

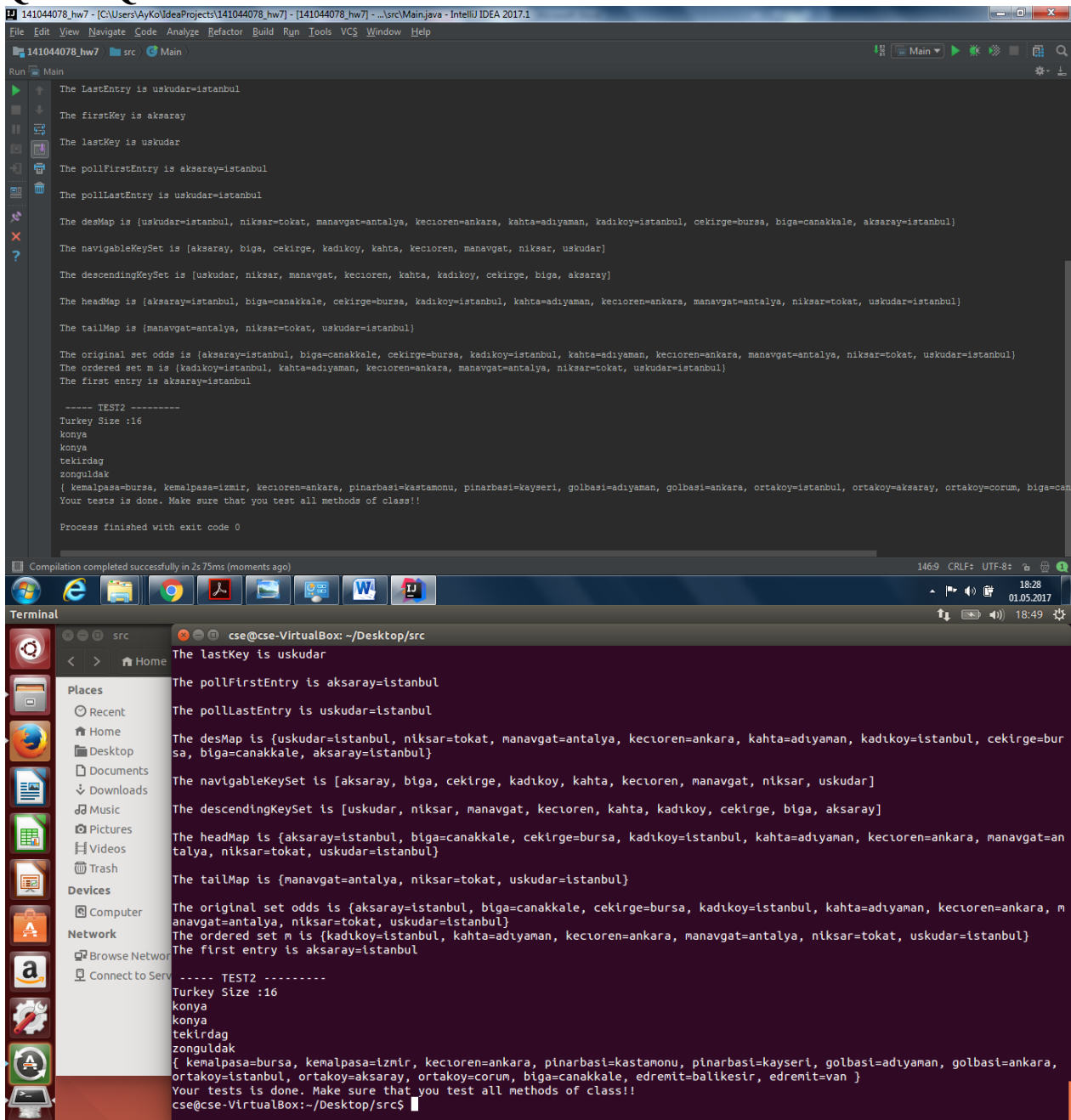
```

try {
    System.out.println(("\\n ----- TEST2 ----- "));
    System.out.println("Turkey Size :"+ turkey.size());
    System.out.println(turkey.get("eregli"));
    System.out.println(turkey.remove("eregli"));
    System.out.println(turkey.remove("eregli"));
    System.out.println(turkey.remove("eregli"));
    System.out.println(turkey.toString());
}
catch (NullPointerException e) {
    System.out.println("NullPointerException : Variable Key can not be null");
}
catch (ClassCastException e){
    System.out.println("ClassCast Exceptions : param Key can not Compered key be null");
}
catch (NoSuchElementException e){
    System.out.println("NoSuchElementException : param Key can not Compered key be null");
}
catch (UnsupportedOperationException e){
    System.out.println("UnsupportedOperationException");
}
}

```

4. Running and Results

Q1 and Q2 Test Results



The screenshot displays the IntelliJ IDEA IDE interface with a Java project named '141044078_hw7'. The 'Run' tab is active, showing the output of the program. The output includes several lines of text describing the state of various data structures and the results of a test.

```
The lastEntry is uskudar=istanbul
The firstKey is aksaray
The lastKey is uskudar
The pollFirstEntry is aksaray=istanbul
The pollLastEntry is uskudar=istanbul

The desMap is {uskudar=istanbul, niksar=tokat, manavgat=antalya, kecioren=ankara, kahta=adiyaman, kadikoy=istanbul, cekirge=bursa, biga=canakkale, aksaray=istanbul}
The navigableKeySet is [aksaray, biga, cekirge, kadikoy, kahta, kecioren, manavgat, niksar, uskudar]
The descendingKeySet is [uskudar, niksar, manavgat, kecioren, kahta, kadikoy, cekirge, biga, aksaray]
The headMap is {aksaray=istanbul, biga=canakkale, cekirge=bursa, kadikoy=istanbul, kahta=adiyaman, kecioren=ankara, manavgat=antalya, niksar=tokat, uskudar=istanbul}
The tailMap is {manavgat=antalya, niksar=tokat, uskudar=istanbul}

The original set odds is {aksaray=istanbul, biga=canakkale, cekirge=bursa, kadikoy=istanbul, kahta=adiyaman, kecioren=ankara, manavgat=antalya, niksar=tokat, uskudar=istanbul}
The ordered set m is {kadikoy=istanbul, kahta=adiyaman, kecioren=ankara, manavgat=antalya, niksar=tokat, uskudar=istanbul}
The first entry is aksaray=istanbul

----- TEST2 -----
Turkey Size :16
konya
konya
teklirdag
zonguldak
{ kemalpasa=bursa, kemalpasa=izmir, kecioren=ankara, pinarbasti=kastamonu, pinarbasti=kayseri, golbasti=adiyaman, golbasti=ankara, ortakoy=istanbul, ortakoy=aksaray, ortakoy=corum, biga=canakkale, edremitt=balikesir, edremitt=van }
Your tests is done. Make sure that you test all methods of class!!

Process finished with exit code 0
```

Below the IDE window, a terminal window is open, showing the same output as the IDE's Run tab. The terminal prompt is 'cse@cse-VirtualBox: ~/Desktop/src\$'. The output is identical to the one shown in the IDE window.