Septiembre-2018-Java.pdf

Naxetee_

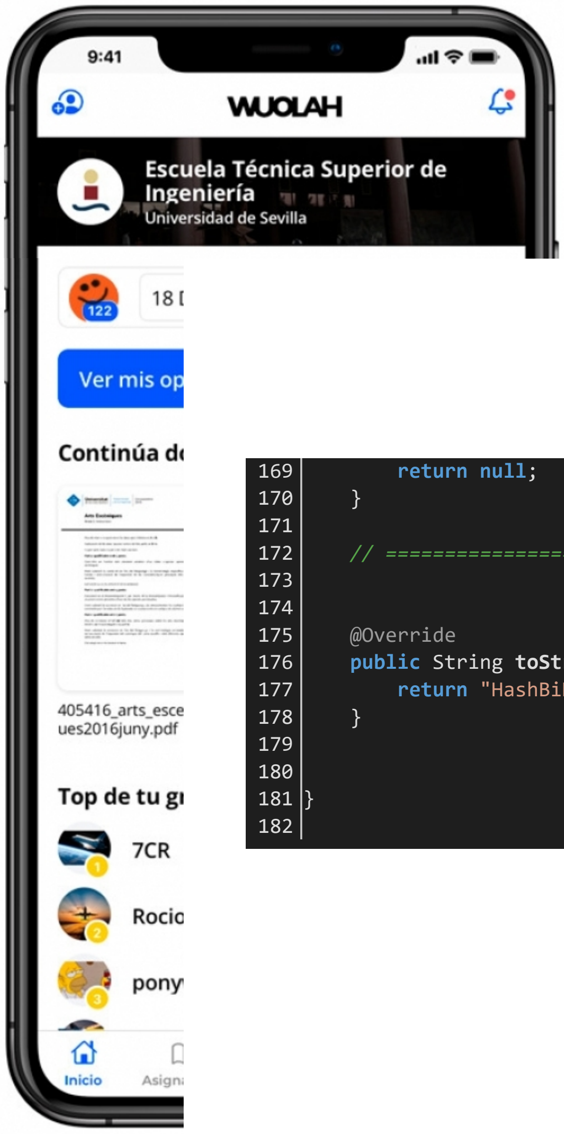
Estructuras de Datos

2º Grado en Ingeniería Informática

Escuela Técnica Superior de Ingeniería Informática
Universidad de Málaga

9:41
WUOLAH
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla
122
18 DES
451 XP
45.55 EUR
Ver mis oportunidades para hoy
Continúa donde lo dejaste.
Ver todos

```java
package dataStructures.dictionary;

import dataStructures.list.List;
import dataStructures.set.AVLSet;
import dataStructures.set.Set;
import dataStructures.tuple.Tuple2;

import java.util.Iterator;
import java.util.NoSuchElementException;

/**
 * Estructuras de Datos. Grados en Informatica. UMA.
 * Examen de septiembre de 2018.
 * <p>
 * Apellidos, Nombre: Avila Reyes, Ignacio
 * Titulacion, Grupo: Doble Grado Matem�ticas + Ingenier�a Inform�tica
 */
public class HashBiDictionary<K, V> implements BiDictionary<K, V> {
    private Dictionary<K, V> bKeys;
    private Dictionary<V, K> bValues;

    public HashBiDictionary() {
        // TODO
        bKeys = new HashDictionary<>();
        bValues = new HashDictionary<>();
    }

    public boolean isEmpty() {
        // TODO
        return bKeys.isEmpty();
    }

    public int size() {
        // TODO
        return bKeys.size();
    }

    public void insert(K k, V v) {
        // TODO
        if (isDefinedKeyAt(k)) {
            bValues.delete(bKeys.valueOf(k));
        }
        if (isDefinedValueAt(v)) {
            bKeys.delete(bValues.valueOf(v));
        }
        bKeys.insert(k, v);
        bValues.insert(v, k);
    }

    public V valueOf(K k) {
        // TODO
        return bKeys.valueOf(k);
    }

    public K keyOf(V v) {
        // TODO
        return bValues.valueOf(v);
    }

    public boolean isDefinedKeyAt(K k) {
        return bKeys.isDefinedAt(k);
    }

    public boolean isDefinedValueAt(V v) {
        return bValues.isDefinedAt(v);
    }

    public void deleteByKey(K k) {
        // TODO
        if (isDefinedKeyAt(k)) {
            bValues.delete(valueOf(k));
            bKeys.delete(k);
        } else {
            throw new NoSuchElementException("deleteByKey");
        }
    }

    public void deleteByValue(V v) {
        // TODO
        if (isDefinedValueAt(v)) {
            bKeys.delete(keyOf(v));
            bValues.delete(v);
        } else {
            throw new NoSuchElementException("deleteByValue");
        }
    }

    public Iterable<K> keys() {
        return bKeys.keys();
    }

    public Iterable<V> values() {
        return bValues.keys();
    }

    public Iterable<Tuple2<K, V>> keysValues() {
        return bKeys.keysValues();
    }


    public static <K, V extends Comparable<? super V>> BiDictionary<K, V> toBiDictionary(Dictionary<K, V> dict) {
        // TODO
        if (isIny(dict.values())) {
            BiDictionary<K, V> biDic = new HashBiDictionary<>();
            for (Tuple2<K, V> a : dict.keysValues()) {
                biDic.insert(a._1(), a._2());
            }
            return biDic;
        } else {
            throw new IllegalArgumentException("No inyectivo");
        }
    }

    private static <V extends Comparable<? super V>> boolean isIny(Iterable<V> values) {
        boolean iny = true;
        V v;
        Set<V> valueSet = new AVLSet<>();
        Iterator<V> it = values.iterator();
        while (it.hasNext() && iny) {
            v = it.next();
            if (valueSet.isElem(v))
                iny = false;
            valueSet.insert(v);
        }
        return iny;
    }

    public <W> BiDictionary<K, W> compose(BiDictionary<V, W> bdic) {
        // TODO
        BiDictionary<K, W> newBiDic = new HashBiDictionary<>();
        for (Tuple2<K, V> tuple : bKeys.keysValues()) {
            if (bdic.isDefinedKeyAt(tuple._2())) {
                newBiDic.insert(tuple._1(), bdic.valueOf(tuple._2()));
            }
        }
        return newBiDic;
    }

    public static <K extends Comparable<? super K>> boolean isPermutation(BiDictionary<K, K>
bd) {
        // TODO
        Set<K> set1 = new AVLSet<>();
        Set<K> set2 = new AVLSet<>();
        boolean isPermutation = true;

        for (Tuple2<K, K> tuple : bd.keysValues()) {
            set1.insert(tuple._1());
            set2.insert(tuple._2());
        }

        Iterator<K> it = set1.iterator();
        while (it.hasNext() && isPermutation) {
            if (!set2.isElem(it.next())) {
                isPermutation = false;
            }
        }
        return isPermutation;
    }

    // Solo alumnos con evaluaci�n por examen final.
    // ====================================

    public static <K extends Comparable<? super K>> List<K> orbitOf(K k, BiDictionary<K, K>
bd) {
        // TODO
        return null;
    }

    public static <K extends Comparable<? super K>> List<List<K>> cyclesOf(BiDictionary<K, K>
bd) {
        // TODO
```



```java
            return null;
        }

        // ====================================


        @Override
        public String toString() {
            return "HashBiDictionary [bKeys=" + bKeys + ", bValues=" + bValues + "]";
        }


}

```