**Difference between two collections**

13393% of 7941 of335[jteodoro](http://www.codewars.com/users/jteodoro)

Java

* [TRAIN AGAIN](http://www.codewars.com/kata/difference-between-two-collections/train/java)
* [NEXT KATA](http://www.codewars.com/trainer/java)

Details

[Solutions](http://www.codewars.com/kata/difference-between-two-collections/solutions/java)

[Discourse (8)](http://www.codewars.com/kata/difference-between-two-collections/discuss/java)

* Add to Collection
* |
* Share this kata:

Find the difference between two collections. The difference means that either the character is present in one collection or it is present in other, but not in both. Return a sorted set with difference.

The collections can contain any character and can contain duplicates.

For instance:

A = [a,a,t,e,f,i,j]

B = [t,g,g,i,k,f]

difference = [a,e,g,j,k]

<http://www.codewars.com/kata/difference-between-two-collections/java>

public static List<Character> diff(Collection<Character> a, Collection<Character> b) {

// Your code here!

HashSet<Character> hs = new HashSet<Character>();

for(char ch : a) {

if(!b.contains(ch)){

hs.add(ch);

}

}

for(char ch : b) {

if(!a.contains(ch)) {

hs.add(ch);

}

}

ArrayList<Character> res = new ArrayList<Character>(hs);

Collections.sort(res);

return res;

}

-----------------------

public static List<Character> diff(Collection<Character> a, Collection<Character> b) {

// Your code here!

Set<Character> hsa = new HashSet<Character>();

for (Iterator iterator = a.iterator(); iterator.hasNext();) {

//type type = (type) iterator.next();

hsa.add((Character)iterator.next());

}

Set<Character> hsb = new HashSet<Character>();

for (Iterator iterator = b.iterator(); iterator.hasNext();) {

//type type = (type) iterator.next();

hsb.add((Character)iterator.next());

}

Character[] arr\_a = hsa.toArray(new Character[hsa.size()]);

Character[] arr\_b = hsb.toArray(new Character[hsb.size()]);

Arrays.sort(arr\_a);

Arrays.sort(arr\_b);

List<Character> dif = new ArrayList<Character>();

for(int i =0; i<arr\_a.length; i++) {

if(Arrays.binarySearch(arr\_b, arr\_a[i]) < 0) {

dif.add(arr\_a[i]);

}

}

for(int i =0; i<arr\_b.length; i++) {

if(Arrays.binarySearch(arr\_a, arr\_b[i]) < 0) {

dif.add(arr\_b[i]);

}

}

Collections.sort(dif);

return dif;

}

-------------otras soluciones--------------

**import java.util.\*;**

**public class Kata {**

**public static List<Character> diff(Collection<Character> a, Collection<Character> b) {**

**Set s1 = new TreeSet(a), s2 = new HashSet(b);**

**s1.removeAll(s2);**

**s2.removeAll(new HashSet(a));**

**s1.addAll(s2);**

**return new ArrayList(s1);**

**}**

**}**

**import java.util.Collection;**

**import java.util.Set;**

**import java.util.HashSet;**

**import java.util.List;**

**import java.util.ArrayList;**

**class Kata {**

**static List<Character> diff(final Collection<Character> aa, final Collection<Character> bb) {**

**Set<Character> a = new HashSet<>(aa);**

**Set<Character> b = new HashSet<>(bb);**

**Set<Character> intersection = new HashSet<>(a);**

**intersection.retainAll(b);**

**Set<Character> union = new HashSet<>(a);**

**union.addAll(b);**

**// union minus intersection equals symmetric-difference**

**Set<Character> symmetricDifference = new HashSet<>(union);**

**symmetricDifference.removeAll(intersection);**

**List<Character> result = new ArrayList<>(symmetricDifference);**

**result.sort(Character::compareTo);**

**return result;**

**}**

**}**

**import java.util.\*;**

**public class Kata {**

**public static List<Character> diff(Collection<Character> a, Collection<Character> b) {**

**TreeSet<Character> ts = new TreeSet<>();**

**for (char c : a)**

**if (!b.contains(c))**

**ts.add(c);**

**for (char c : b)**

**if (!a.contains(c))**

**ts.add(c);**

**return new ArrayList<Character>(ts);**

**}**

**}**