Sorting, finding max and min of List in java:

**import** java.util.ArrayList;

**import** java.util.Collections;

**import** java.util.List;

**public** **class** LinkedList {

**public** **static** **void** main(String[] args) {

List<String> freak\_list = **new** ArrayList<String>();

String[] str = { "Jenish", "Shakya", "is", "so", "freaking", "nice!!!" };

**for** (String s : str)

freak\_list.add(s);

Collections.*sort*(freak\_list);

**for** (String string : freak\_list)

System.*out*.println(string);

**int** index = Collections.*binarySearch*(freak\_list, "freaking");

System.*out*.println("the index of freakin is " + index);

Object objMin = Collections.*min*(freak\_list);

Object objMax = Collections.*max*(freak\_list);

System.*out*.println("The max is '" + objMax + "' and the min is '"

+ objMin + "'");

}

}

**Simple Sorting Tutorial: (Sorts list items alphabetically)**

**import** java.util.ArrayList;

**import** java.util.Collections;

**import** java.util.Comparator;

**public** **class** Sorting {

**public** **static** **void** main(String[] args) {

Sorting x = **new** Sorting();

Person p = x.**new** Person("Bruce", "Willis");

Person p1 = x.**new** Person("Tom", "Hanks");

Person p2 = x.**new** Person("Nicolas", "Cage");

Person p3 = x.**new** Person("John", "Travolta");

ArrayList<Person> list = **new** ArrayList<Person>();

list.add(p);

list.add(p1);

list.add(p2);

list.add(p3);

System.*out*.println("Before sorting...");

**for** (Person person : list) {

System.*out*.println(person.firstname);

}

System.*out*.println("\n");

Collections.*sort*(list, **new** Comparator() {

**public** **int** compare(Object o1, Object o2) {

Person p1 = (Person) o1;

Person p2 = (Person) o2;

**return** p1.firstname.compareToIgnoreCase(p2.firstname);

// return p1.secondname.compareToIgnoreCase(p2.secondname);

}

});

System.*out*.println("After sorting...");

**for** (Person person : list) {

System.*out*.println(person.firstname);

}

System.*out*.println("\n");

**for** (Person person : list) {

System.*out*.println(person.secondname);

}

}

**public** **class** Person {

String firstname;

String secondname;

**public** Person(String firstname, String secondname) {

**this**.firstname = firstname;

**this**.secondname = secondname;

}

}

}

O//p

Before sorting...

Bruce

Tom

Nicolas

John

After sorting...

Bruce

John

Nicolas

Tom

Willis

Travolta

Cage

Hanks

Sorting HashMap using values

import java.util.Collections;

import java.util.Comparator;

import java.util.HashMap;

import java.util.LinkedHashMap;

import java.util.LinkedList;

import java.util.List;

import java.util.Map;

import java.util.Map.Entry;

public class SortMapByValue

{

public static boolean ASC = true;

public static boolean DESC = false;

public static void main(String[] args)

{

// Creating dummy unsorted map

Map<String, Integer> unsortMap = new HashMap<String, Integer>();

unsortMap.put("B", 55);

unsortMap.put("A", 80);

unsortMap.put("D", 20);

unsortMap.put("C", 70);

System.out.println("Before sorting......");

printMap(unsortMap);

System.out.println("After sorting ascending order......");

Map<String, Integer> sortedMapAsc = sortByComparator(unsortMap, ASC);

printMap(sortedMapAsc);

System.out.println("After sorting descindeng order......");

Map<String, Integer> sortedMapDesc = sortByComparator(unsortMap, DESC);

printMap(sortedMapDesc);

}

private static Map<String, Integer> sortByComparator(Map<String, Integer> unsortMap, final boolean order)

{

List<Entry<String, Integer>> list = new LinkedList<Entry<String, Integer>>(unsortMap.entrySet());

// Sorting the list based on values

Collections.sort(list, new Comparator<Entry<String, Integer>>()

{

public int compare(Entry<String, Integer> o1,

Entry<String, Integer> o2)

{

if (order)

{

return o1.getValue().compareTo(o2.getValue());

}

else

{

return o2.getValue().compareTo(o1.getValue());

}

}

});

// Maintaining insertion order with the help of LinkedList

Map<String, Integer> sortedMap = new LinkedHashMap<String, Integer>();

for (Entry<String, Integer> entry : list)

{

sortedMap.put(entry.getKey(), entry.getValue());

}

return sortedMap;

}

public static void printMap(Map<String, Integer> map)

{

for (Entry<String, Integer> entry : map.entrySet())

{

System.out.println("Key : " + entry.getKey() + " Value : "+ entry.getValue());

}

}

}