Sort a HashMap by Key or Value using Java Stream

**Sort by Key**

**Simplest way**

Map<String, Integer> map = **Map.*of***("Ram", 10, "Shyam",  
 20,"Sophie", 9, "Hari", 23, "Radheshyam", 13);  
map.forEach( (k,v) -> System.*out*.println("Key: "+k+"<===>"+"Value: "+v));

*//* ***Sort a Map by Key***Map<String, Integer> lm = **new** LinkedHashMap<>();  
map.entrySet().stream()**.sorted(Map.Entry.*comparingByKey*())** **.forEach(a -> lm.put(a.getKey(), a.getValue()));**  
  
System.***out***.println(**"LM: "** + lm);

**Complex way**  
  
*//* ***Sort a Map by key***LinkedHashMap<String, Integer> lm1 = map.entrySet().stream()  
 **.sorted(Map.Entry.*comparingByKey*())  
 .collect(Collectors.*toMap*(e -> e.getKey(), e -> e.getValue(),  
 (e1, e2) -> e1, LinkedHashMap::new));**  
System.***out***.println(**"LM 1: "** + lm1);  
  
*// Sort a Map by key*LinkedHashMap<String, Integer> lm2 = map.entrySet().stream()  
 **.sorted(Comparator.*comparing*(a -> a.getKey()))** .collect(Collectors.*toMap*(e -> e.getKey(), e -> e.getValue(),  
 (e1, e2) -> e1, LinkedHashMap::**new**));  
  
System.***out***.println(**"LM 2: "** + lm2);

**Sort by Value**

**Simplest way**

*// Sort a Map by value*Map<String, Integer> lm = **new** LinkedHashMap<>();  
map.entrySet().stream()  
 .sorted(Map.Entry.*comparingByValue*())  
 .**forEach(a -> lm.put(a.getKey(), a.getValue()));**  
  
System.***out***.println(**"LM: "** + lm);  
  
*// Sort a Map by value*LinkedHashMap<String, Integer> lm1 = map.entrySet().stream()  
 .sorted(Map.Entry.*comparingByValue*())  
 .collect(Collectors.*toMap*(e -> e.getKey(), e -> e.getValue(),  
 (e1, e2) -> e1, LinkedHashMap::**new**));  
System.***out***.println(**"LM 1: "** + lm1);  
  
*// Sort a Map by key*LinkedHashMap<String, Integer> lm2 = map.entrySet().stream()  
 .sorted(Comparator.*comparing*(a -> a.getValue()))  
 .collect(Collectors.*toMap*(e -> e.getKey(), e -> e.getValue(),  
 (e1, e2) -> e1, LinkedHashMap::**new**));  
  
System.***out***.println(**"LM 2: "** + lm2);

**Sort a Map by Key as Object**

**public class** Employee {  
  
 **private** String **name**;  
 **private int age**;  
 **private** String **city**;  
 **private** String **gender**;  
  
 **public** Employee(String name, **int** age, String city, String gender) {  
 **this**.**name** = name;  
 **this**.**age** = age;  
 **this**.**city** = city;  
 **this**.**gender** = gender;  
 }

//get()/set() Methods

}

Map<Employee, Integer> map = **new** HashMap<>();  
 map.put(**new** Employee(**"Ram"**,11, **"Hyderabad"**, **"Male"**), 9);  
 map.put(**new** Employee(**"Shyam"**,12, **"Bangalore"**, **"Male"**), 8);  
 map.put(**new** Employee(**"Sophie"**,13, **"Cuttack"**, **"FeMale"**), 13);  
 map.put(**new** Employee(**"Hari"**,14, **"Chennai"**, **"Male"**), 50);  
 map.put(**new** Employee(**"Radheshyam"**,15, **"Bhubaneswar"**, **"Male"**), 41);  
 map.put(**new** Employee(**"Chitra"**,16, **"Coimbatore"**, **"FeMale"**), 23);  
  
 *// Sort Map by Key by City* **Comparator<Employee> byCity = (Employee e1, Employee e2) -> e1.getCity().compareTo(e2.getCity());**  
 LinkedHashMap lm = map.entrySet().stream()  
 **.sorted(Map.Entry.*comparingByKey*(byCity))** .collect(Collectors.*toMap*(e->e.getKey(), e -> e.getValue(),  
 (e1,e2) -> 1, LinkedHashMap::**new**));  
 System.***out***.println(lm);  
  
 *// Sort Map by Key by Name* **Comparator<Employee> byName = (Employee e1, Employee e2) -> e1.getName().compareTo(e2.getName());** LinkedHashMap lm1 = map.entrySet().stream()  
 **.sorted(Map.Entry.*comparingByKey*(byName))** .collect(Collectors.*toMap*(e -> e.getKey(), e -> e.getValue(),  
 (e1,e2) -> e1, LinkedHashMap::**new**));  
 System.***out***.println(lm1);  
  
 *// Sort Map by key by Age* **Comparator<Employee> byAge = (Employee e1, Employee e2) ->  
 Integer.*valueOf*(e1.getAge()).compareTo(Integer.*valueOf*(e2.getAge()));** LinkedHashMap lm2 = map.entrySet().stream()  
 **.sorted(Map.Entry.*comparingByKey*(byAge))** .collect(Collectors.*toMap*(e -> e.getKey(), e -> e.getValue(),  
 (e1,e2) -> e1, LinkedHashMap::**new**));  
 System.***out***.println(lm2);  
  
}

**Sum of Integer in an array or list**

**int**[] a = **new int**[]{1,2,3,4,5};  
**int** sum = **Arrays.*stream*(a).reduce((x,y) -> x+y).getAsInt();**  
System.***out***.println(sum);  
  
**int** sumVal = **Arrays.*stream*(a).sum();**  
System.***out***.println(**"Sum Val : "**+sumVal);  
  
List<Integer> list = Arrays.*asList*(1,2,3,4,5);  
**int** sum1 = **list.stream().reduce((x,y) -> x+y).get();**  
System.***out***.println(sum1);  
  
**sum = (int) list.stream().collect(Collectors.*summingInt*( e -> e.intValue()));**System.***out***.println(sum);

**Max Min using Java Stream**

**int**[] a = **new int**[]{55, 45, 23, 12, 34, 78};  
**int** max = Arrays.*stream*(a).max().getAsInt();  
System.***out***.println(**"Max Value: "** + max);  
  
List<Integer> list = Arrays.*asList*(55, 45, 23, 12, 34, 78);  
  
**int** maxVal = **list.stream().max(Comparator.*naturalOrder*()).get();**  
System.***out***.println(**"Max: "** + maxVal);  
  
maxVal = **list.stream().max(Comparator.*comparing*(Integer::*valueOf*)).get();**  
System.***out***.println(**"Max: "** + maxVal);  
  
maxVal = **list.stream().max(Comparator.*comparing*(e -> e)).get();**System.***out***.println(**"Max: "** + maxVal);

*// For List of Strings*List<String> slist = Arrays.*asList*(**"a"**, **"b"**, **"c"**, **"d"**);  
String maxStr = **slist.stream().max(Comparator.*naturalOrder*()).get();**  
System.***out***.println(maxStr);  
maxStr = **slist.stream().max(Comparator.*comparing*(e -> e)).get();**  
System.***out***.println(maxStr);  
maxStr = **slist.stream().max(Comparator.*comparing*(String::*valueOf*)).get();**  
System.***out***.println(maxStr);  
  
List<Person> plist = Arrays.*asList*(**new** Person(**"Ram"**, 23, 1000),  
 **new** Person(**"Hari"**, 63, 3000),  
 **new** Person(**"Shyam"**, 23, 7000));  
  
*// Who is drawing highest salary***Comparator<Person> bySal = Comparator.*comparing*(p -> p.getSal());**  
Person maxSalPerson = **plist.stream().max(bySal).get();**  
System.***out***.println(maxSalPerson);  
  
**Comparator<Person> bySal1 = (Person p1, Person p2) ->  
 Integer.*valueOf*(p1.getSal()).compareTo(Integer.*valueOf*(p2.getSal()));**  
Person maxSalPerson1 = **plist.stream().max(bySal1).get();**  
System.***out***.println(maxSalPerson1);  
  
*// Who is seniorMost***Comparator<Person> byAge = Comparator.*comparing*(e -> e.getAge());**  
Person senior1 = **plist.stream().max(byAge).get();**  
System.***out***.println(senior1);