2.

public class StBuffer{

public static void main(String args[]){

String str="java";

StringBuffer sb=new StringBuffer("object");

StringBuilder s=new StringBuilder("programming");

System.out.println(str.concat(" "+"book"));

System.out.println(sb.append(" "+"oriented"));

System.out.println(s.append(" "+"language"));

}

}

***Output:*** java book

Object oriented

Programming language

7.

import java.util.HashSet;

import java.util.TreeSet;

import java.util.Set;

class ConvertHashSettoTreeSet{

public static void main(String[] args) {

HashSet<String> hset = new HashSet<String>();

hset.add("Element1");

hset.add("Element2");

hset.add("Element3");

hset.add("Element4");

System.out.println("HashSet contains: "+ hset);

Set<String> tset = new TreeSet<String>(hset);

System.out.println("TreeSet contains: ");

for(String temp : tset){

System.out.println(temp);

}

}

}

***Output:*** HashSet contains: [Element4, Element3, Element2, Element1]

TreeSet contains:

Element1

Element2

Element3

Element4

8.

import java.util.Comparator;

import java.util.Map;

import java.util.TreeMap;

public class MyReverseOrderMap {

public static void main(String a[]){

TreeMap<String, String> hm = new TreeMap<String, String>(new MyCopr());

hm.put("java", "language");

hm.put("computer", "machine");

hm.put("india","country");

hm.put("mango","fruit");

hm.put("game","cricket");

System.out.println("TreeMap Entries:");

System.out.println(hm);

Map<String, String> rm = hm.descendingMap();

System.out.println("Reverse Map Content: ");

System.out.println(rm);

}

}

class MyCopr implements Comparator<String>{

public int compare(String str1, String str2) {

return str1.compareTo(str2);

}

}

***Output:*** TreeMap Entries: {computer=machine, game=cricket, india=country, java=language, mango=fruit}

Reverse Map Content: {mango=fruit, java=language, india=country, game=cricket, computer=machine}

9.

import java.util.\*;

class TreeMapDemo {

public static <K, V extends Comparable<V>> Map<K, V>

sortByValues(final Map<K, V> map) {

Comparator<K> valueComparator = new Comparator<K>() {

public int compare(K k1, K k2) {

int compare = map.get(k1).compareTo(map.get(k2));

if (compare == 0)

return 1;

else

return compare;

}

};

Map<K, V> sortedByValues = new TreeMap<K, V>(valueComparator);

sortedByValues.putAll(map);

return sortedByValues;

}

public static void main(String args[]) {

TreeMap<String, String> treemap = new TreeMap<String, String>();

treemap.put("Key1", "Jack");

treemap.put("Key2", "Rick");

treemap.put("Key3", "Kate");

treemap.put("Key4", "Tom");

treemap.put("Key5", "Steve");

Map sortedMap = sortByValues(treemap);

Set set = sortedMap.entrySet();

Iterator i = set.iterator();

while(i.hasNext()) {

Map.Entry me = (Map.Entry)i.next();

System.out.print(me.getKey() + ": ");

System.out.println(me.getValue());

}

}

}

***Output:*** Key1: Jack

Key3: Kate

Key2: Rick

Key5: Steve

Key4: Tom

10.

import java.util.HashMap;

public class RemoveAllExample {

public static void main(String[] args) {

HashMap<Integer, String> hashmap = new HashMap<Integer, String>();

hashmap.put(11,"Value1");

hashmap.put(22,"Value2");

hashmap.put(33,"Value3");

hashmap.put(44,"Value4");

hashmap.put(55,"Value5");

System.out.println("HashMap Elements: " + hashmap);

hashmap.clear();

System.out.println("After calling clear():");

System.out.println("---------------------");

System.out.println("HashMap Elements: " + hashmap);

}

}

***Output:*** HashMap Elements: {33=Value3, 22=Value2, 55=Value5, 11=Value1, 44=Value4}

After calling clear(): ---------------------

HashMap Elements: {}