TT22049

1] Write a program to find the kth smallest element in an Arraylist

```
import java.util.*;
public class kthSmallest {
    public static int findkthSmallest (List <Integer>list, int k){
        collection.sort (list);
        return list.get (k-1); }
    public static void main (String[] args) {
        List <Integer> numbers = Array.asList (7, 10, 4, 3, 20)
        int k=3;
        int result = findkthSmallest (numbers, k);
        System.out.println ("The "+k+"rd smallest elemen
                is: "+result); }
}
```

2] Create a TreeMap to store the mappings of words t their frequencies in a given text.

```
import java.util.*;
public class WordFrequencyMap {
    public static void main (String args[]) {
        String text = "hello world";
        String[] words = text.split (" ");
        TreeMap <String, Integer> frequencyMap = new
                TreeMap () for
                (String word: words)
```

```
{ frequencyMap.put(word, frequencyMap.getOrDefault(word,0)+1)
}
System.out.println("Word frequencies: ");
for(Map.Entry<String,Integer>entry:frequencyMap.entry
                                    Set()){
    System.out.println(entry.getkey()":+entry.getvalue());}
}
}
```

3] Queue and stack using priority Queue with a custom compator

```
import java.util.*;
public class QueueStackPQ{
    public static void main(String[] args){
        PriorityQueue<Integer> queue = new PriorityQueue<>();
            queue.add(3);
            queue.add(1);
            queue.add(2);
        System.out.println(Queue(acending order): ");
        while(!queue.isEmpty()){
            System.out.println(queue.poll()+" ");}
        PriorityQueue<Integer> stack= new PriorityQueue<>(
                                        Collection.reverseOrder());
            stack.add(1);
            stack.add(2);
            stack.add(3);
        System.out.println("\n stack (decending order): ");
            while(!stack.empty()){
```

```
            System.out.println (Stack.poll () + " "); }
        }
    }
```

4. TreeMap to store student IDs and their details.

```
import java.util.*;
class Student {
        String name;
        int age;
        Student (String name, int age) {
            this.name = name;
            this.age = age; }
        public String toString () {
            return name + " (" + age + "); }
    }
    public class StudentMap {
        public static void main (String [] args) {
            TreeMap <Integer, student> Student = new
                                        TreeMap <> ();
            Students.put (101, new Student ("Adam", 20);
            Students.put (102, new Student ("Arin", 21);
    for (Map.Entry <Integer, student> entry : students
                                        .entrySet ()) {
        System.out.println ("ID " + entry.getKey () + " Details:
                " + entry.getValue ()); }
    }
}
```

5] Check if two linkedlist are equal

```
import java.util.*;
public class LinkedlistEqual {
    public static void main (String args[]){
        LinkedList <Integer> list1 = new Linkedlist <> (Array.asList
                                                        (1,2,3));
        LinkedList<Integer> list2 = new Linkedlist <> (Array.asList
                                                        (1,2,3));
        boolean isEqual = list1.equals (list2);
        System.out.println ("Are the lists eqaal2" +isEqual); }
}
```

6] HashMap for employee ID to department

```
import java.util
public class EmployeeDept {
    public Static void main (String[] args){
        HashMap <Integer, String> empDept = new HashMap<>();
        empDept.put (1001,"HR");
        empDept.put (1006, "CSE");
        empDept.put (1005, "Accounting");
    for (Map.Entry <Integer, String> entry : empDept.entry
                                                Set()){
        System.out.println ("Employee ID: " + entry() + "Depart-
                    ment:-" + entry.getvalue()); }
    }
}
```