Name : Iffat Jahan
ID : IT-22014

① Write a program to find the k^{th} smallest element in an Array list.

```
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
                                element in :" + result);
  }
}
```

IT-22014

(2) Create a Treemap to store the mappings of words to their frequencies in a given text.

```
import java.util.*;
public class WordFrequencyMap {
public static void main (String args[]) {
String text = "Hello world Hello java";
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 : frequency
     map.entrySet () {

    }
  }
  }
```

3]

```
public clann Queuestackpo {
public Static void main (String[] angs) {
priority Queue<Integen>queue = new priority Queue<> ();
        queue.add (3);
        queue.add (1);
        queue.add(2);
System.out.println (Queue (ascending onden): ");
    While (!queue.inempty 0){
 System.out.print ( queue.poll 0 +" ) ;
                priority Queue (Integen> Stack = new Prionityqueue
                                                    <>
                                                (collection.neuenne Onden());
      Stack.add(1);
      Stack.add(2);
   System.out.println (" |n Stack (deceending onden):"
     While (!Stack.empty ()) {
   System.out.println ( Stack.poll +" ");
     }
     }
     }
```

4. TreeMap to store student's IDs and their details.

```
import java.util.*;
// Create a student class to hold student details
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> students = new TreeMap<>();
Students.put(101. new student("Adam", 20);
Student.put(102. new student("Aniw", 21);
```

IT 22014 Iffat Jahan

```
// Print each student ID and details
    for (Map.Entry<Integer, Student> entry :
            students.entrySet()){
System.out.println("ID: " + entry.getKey() + ",
        Details: " + entry.getValue());
    }
  }
}
```

5. check if two linkedlists are equal.

```
import.java.util.*;
Public class LinkedListEqual {
Public static Void main (string[] args) {
// create two linked lists with the same values
LinkedList<Integer> list1 = newLinkedlist<>
                    (Arrays.asList(1,2,3));
Linkedlist <Integer> list2 = new linkedlist<>
                    (Arrays.aslist(1,2,3));
// use equal method to compare
    boolean isEqual = list1.equals(list2);
System.out.println("Are the lists equal
                    2" + isEqual);
}
}
```

IT 22014 Iffat Jahan

6. HashMap for employee ID to depantment

```
import.java.util.*;
public class EmployeeDept {
public static void main(String args[]){
// create a hashMap to store employee ID & Dept
HashMap<Integer, String> empDept = new HashMaps
                                   ();

empDept.put(1001, "HR");
empDept.put(1006' "CSE");
empDept.put(1005, "Accounting");

// Print
for(Map.Entry<integer, string> entry: empDept.
                    entrySet()){

System.out.println("EmployeeID: "+ entry
                    ()+", Depantment:
                    "+entry()+");

}
}
}
```