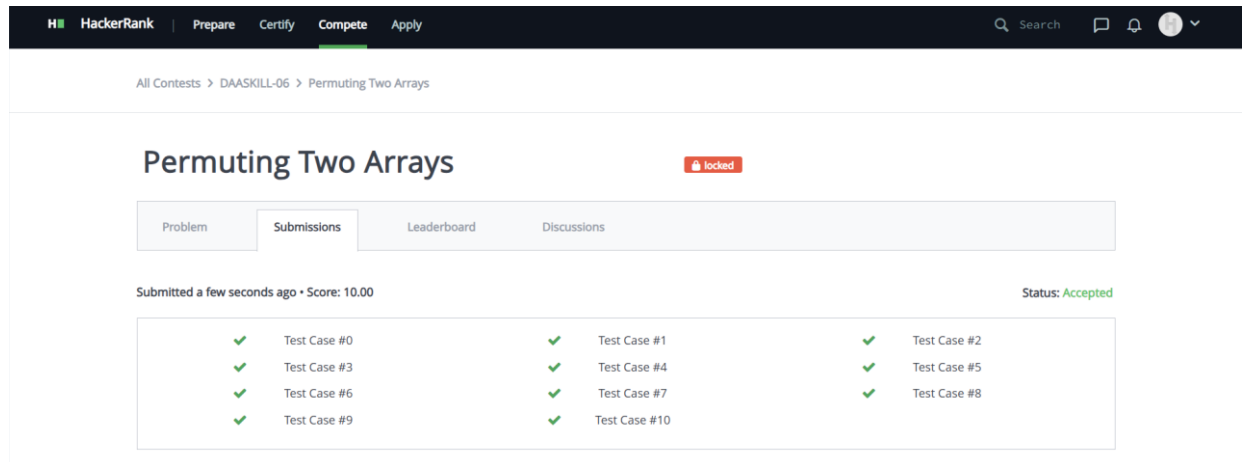


ID NO : 2300031672

NAME : K.Imran

LAB SKILL-WEEK 6

1) Permuting Two Arrays:



The screenshot shows the HackerRank interface for the 'Permuting Two Arrays' problem. The top navigation bar includes 'HackerRank', 'Prepare', 'Certify', 'Compete', and 'Apply'. The main header shows 'All Contests > DAASKILL-06 > Permuting Two Arrays'. The problem title 'Permuting Two Arrays' is displayed with a 'locked' status. Below the title are tabs for 'Problem', 'Submissions', 'Leaderboard', and 'Discussions'. The 'Submissions' tab is active, showing 'Submitted a few seconds ago' and 'Score: 10.00'. The status is 'Accepted'. A table of test cases is shown, all with green checkmarks indicating they passed.

Test Case #	Status
Test Case #0	✓
Test Case #1	✓
Test Case #2	✓
Test Case #3	✓
Test Case #4	✓
Test Case #5	✓
Test Case #6	✓
Test Case #7	✓
Test Case #8	✓
Test Case #9	✓
Test Case #10	✓

```
import java.io.*;
import java.util.*;
```

```
class Result {
```

```
    public static String twoArrays(int k, List<Integer> A, List<Integer> B) {
        Collections.sort(A);
        Collections.sort(B, Collections.reverseOrder());
        int len = A.size();
        for (int i = 0; i < len; i++) {
            if (A.get(i) + B.get(i) < k) {
                return "NO";
            }
        }
        return "YES";
    }
}
```

```
public class Solution {
```

```
    public static void main(String[] args) throws IOException {
```

```
        BufferedReader bufferedReader = new BufferedReader(new InputStreamReader(System.in));
        BufferedWriter bufferedWriter = new BufferedWriter(new
        FileWriter(System.getenv("OUTPUT_PATH")));
```

```
        int q = Integer.parseInt(bufferedReader.readLine().trim());
```

```
        for (int qItr = 0; qItr < q; qItr++) {
            String[] firstMultipleInput = bufferedReader.readLine().replaceAll("\\s+$", "").split(" ");
```

```

int n = Integer.parseInt(firstMultipleInput[0]);
int k = Integer.parseInt(firstMultipleInput[1]);

String[] ATemp = bufferedReader.readLine().replaceAll("\\s+$", "").split(" ");
List<Integer> A = new ArrayList<>();
for (int i = 0; i < n; i++) {
    int AItem = Integer.parseInt(ATemp[i]);
    A.add(AItem);
}

String[] BTemp = bufferedReader.readLine().replaceAll("\\s+$", "").split(" ");
List<Integer> B = new ArrayList<>();
for (int i = 0; i < n; i++) {
    int BItem = Integer.parseInt(BTemp[i]);
    B.add(BItem);
}

String result = Result.twoArrays(k, A, B);

bufferedWriter.write(result);
bufferedWriter.newLine();
}

bufferedReader.close();

bufferedWriter.close();
}
}

```

2) Jim and the Orders:

The screenshot shows the HackerRank interface for the problem 'Jim and the Orders'. The top navigation bar includes 'HackerRank', 'Prepare', 'Certify', 'Compete' (highlighted), and 'Apply'. A search bar and user profile icon are on the right. The breadcrumb trail is 'All Contests > DAASKILL-06 > Jim and the Orders'. The problem title 'Jim and the Orders' is displayed with a 'locked' status icon. Below the title are tabs for 'Problem', 'Submissions', 'Leaderboard', and 'Discussions'. The 'Submissions' tab is active, showing a submission from 'a few seconds ago' with a 'Score: 10.00' and a status of 'Accepted'. A table below lists 11 test cases, all of which are passed, indicated by green checkmarks.

Test Case	Status
Test Case #0	✓
Test Case #1	✓
Test Case #2	✓
Test Case #3	✓
Test Case #4	✓
Test Case #5	✓
Test Case #6	✓
Test Case #7	✓
Test Case #8	✓
Test Case #9	✓
Test Case #10	✓

```

import java.io.*;
import java.math.*;
import java.security.*;

```

```

import java.text.*;
import java.util.*;
import java.util.concurrent.*;
import java.util.regex.*;

class Result {

    public static List<Integer> jimOrders(List<List<Integer>> orders) {
        class Order {
            int index;
            int time;

            Order(int index, int time) {
                this.index = index;
                this.time = time;
            }
        }

        List<Order> orderTimes = new ArrayList<Order>();

        for (int i = 0; i < orders.size(); i++) {
            int time = orders.get(i).get(0) + orders.get(i).get(1);
            orderTimes.add(new Order(i + 1, time));
        }

        Collections.sort(orderTimes, new Comparator<Order>() {
            @Override
            public int compare(Order order1, Order order2) {
                if (order1.time == order2.time) {
                    return order1.index - order2.index;
                } else {
                    return order1.time - order2.time;
                }
            }
        });

        List<Integer> result = new ArrayList<Integer>();
        for (Order order : orderTimes) {
            result.add(order.index);
        }

        return result;
    }
}

public class Solution {
    public static void main(String[] args) throws IOException {
        BufferedReader bufferedReader = new BufferedReader(new InputStreamReader(System.in));
        BufferedWriter bufferedWriter = new BufferedWriter(new
        FileWriter(System.getenv("OUTPUT_PATH")));

        int n = Integer.parseInt(bufferedReader.readLine().trim());
    }
}

```

```

List<List<Integer>> orders = new ArrayList<>();

for (int i = 0; i < n; i++) {
    String[] ordersRowTempItems = bufferedReader.readLine().replaceAll("\\s+$", "").split(" ");

    List<Integer> ordersRowItems = new ArrayList<>();

    for (int j = 0; j < 2; j++) {
        int ordersItem = Integer.parseInt(ordersRowTempItems[j]);
        ordersRowItems.add(ordersItem);
    }

    orders.add(ordersRowItems);
}

List<Integer> result = Result.jimOrders(orders);

for (int i = 0; i < result.size(); i++) {
    bufferedWriter.write(String.valueOf(result.get(i)));

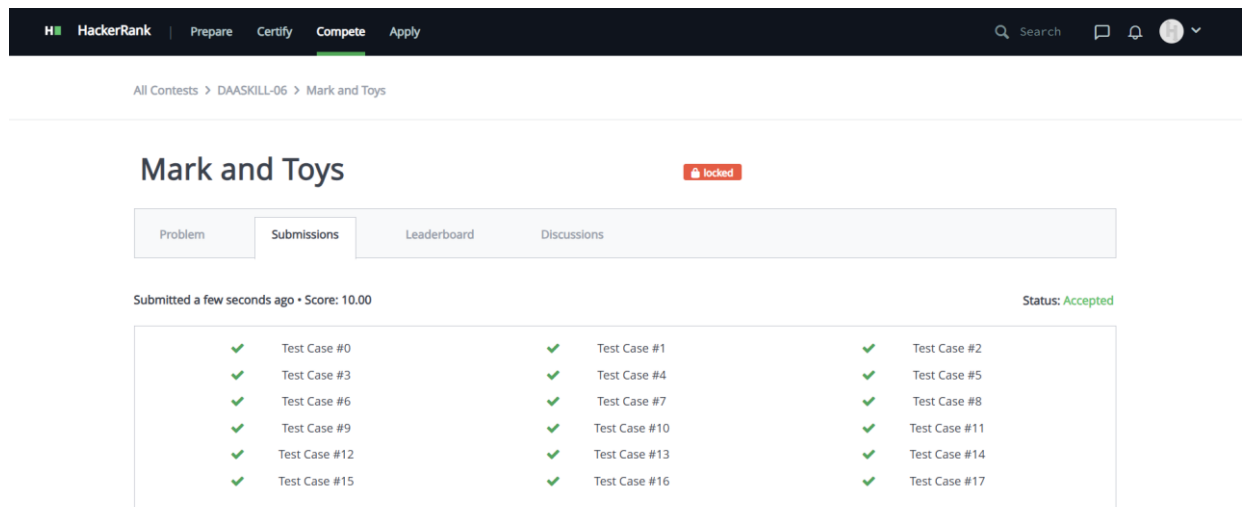
    if (i != result.size() - 1) {
        bufferedWriter.write(" ");
    }
}

bufferedWriter.newLine();

bufferedReader.close();
bufferedWriter.close();
}
}

```

3) Mark and Toys:



HackerRank | Prepare | Certify | **Compete** | Apply

All Contests > DAASKILL-06 > Mark and Toys

Mark and Toys

locked

Problem | **Submissions** | Leaderboard | Discussions

Submitted a few seconds ago • Score: 10.00

Status: Accepted

✓	Test Case #0	✓	Test Case #1	✓	Test Case #2
✓	Test Case #3	✓	Test Case #4	✓	Test Case #5
✓	Test Case #6	✓	Test Case #7	✓	Test Case #8
✓	Test Case #9	✓	Test Case #10	✓	Test Case #11
✓	Test Case #12	✓	Test Case #13	✓	Test Case #14
✓	Test Case #15	✓	Test Case #16	✓	Test Case #17

```
import java.io.*;
import java.math.*;
import java.security.*;
import java.text.*;
import java.util.*;
import java.util.concurrent.*;
import java.util.regex.*;
```

```
class Result {

    public static int maximumToys(List<Integer> prices, int k) {
        Collections.sort(prices);
        int i = 0;
        while (k > -1){
            k = k - prices.get(i);
            i += 1;
        }
        return i - 1;
    }
}
```

```
public class Solution {
    public static void main(String[] args) throws IOException {
        BufferedReader bufferedReader = new BufferedReader(new
        InputStreamReader(System.in));
        BufferedWriter bufferedWriter = new BufferedWriter(new
        FileWriter(System.getenv("OUTPUT_PATH")));

        String[] firstMultipleInput = bufferedReader.readLine().replaceAll("\\s+$", "").split(" ");

        int n = Integer.parseInt(firstMultipleInput[0]);
```

```

int k = Integer.parseInt(firstMultipleInput[1]);

String[] pricesTemp = bufferedReader.readLine().replaceAll("\\s+$", "").split(" ");

List<Integer> prices = new ArrayList<>();

for (int i = 0; i < n; i++) {
    int pricesItem = Integer.parseInt(pricesTemp[i]);
    prices.add(pricesItem);
}

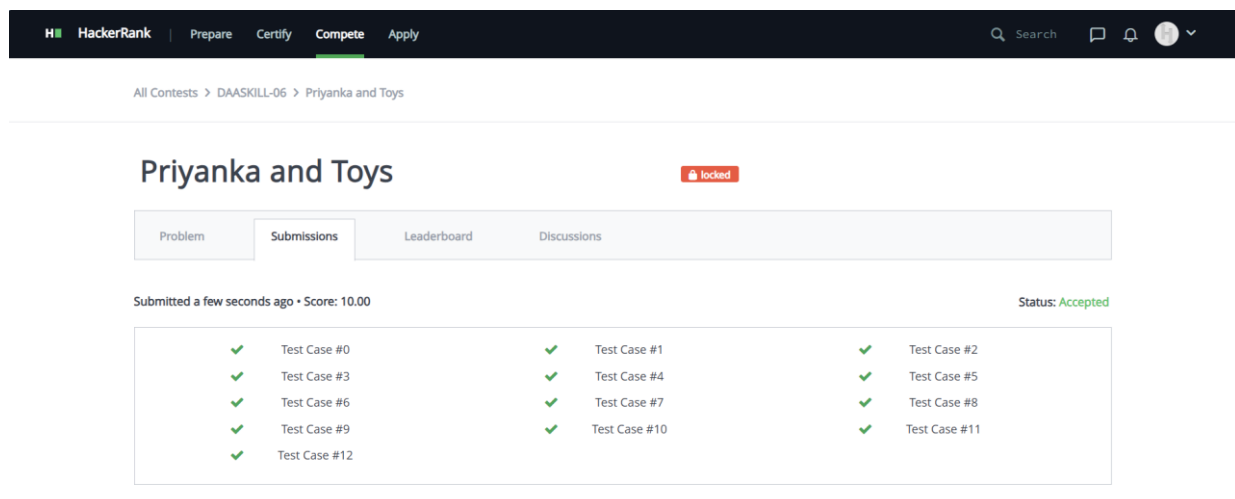
int result = Result.maximumToys(prices, k);

bufferedWriter.write(String.valueOf(result));
bufferedWriter.newLine();

bufferedReader.close();
bufferedWriter.close();
}
}

```

4) Priyanka and Toys:



The screenshot shows the HackerRank interface for the problem 'Priyanka and Toys'. The top navigation bar includes 'HackerRank', 'Prepare', 'Certify', 'Compete' (highlighted), and 'Apply'. A search bar and user profile icon are on the right. The breadcrumb trail is 'All Contests > DAASKILL-06 > Priyanka and Toys'. The problem title 'Priyanka and Toys' is displayed with a 'locked' icon. Below the title are tabs for 'Problem', 'Submissions', 'Leaderboard', and 'Discussions'. The 'Submissions' tab is active, showing 'Submitted a few seconds ago • Score: 10.00' and 'Status: Accepted'. A table of test cases is shown, all with green checkmarks indicating they passed.

Test Case	Status
Test Case #0	✓
Test Case #1	✓
Test Case #2	✓
Test Case #3	✓
Test Case #4	✓
Test Case #5	✓
Test Case #6	✓
Test Case #7	✓
Test Case #8	✓
Test Case #9	✓
Test Case #10	✓
Test Case #11	✓
Test Case #12	✓

```

import java.io.*;
import java.math.*;
import java.security.*;
import java.text.*;
import java.util.*;
import java.util.concurrent.*;
import java.util.regex.*;

class Result {

    public static int toys(List<Integer> w) {

```

```

        System.out.println(w);
        Collections.sort(w);
        int containers = 1;
        int tempLimit = w.get(0);
        for (int i = 1; i < w.size(); i++) {
            if (w.get(i) > tempLimit+4) {
                tempLimit = w.get(i);
                containers++;
            }
        }
        return containers;
    }
}

public class Solution {
    public static void main(String[] args) throws IOException {
        BufferedReader bufferedReader = new BufferedReader(new
InputStreamReader(System.in));
        BufferedWriter bufferedWriter = new BufferedWriter(new
FileWriter(System.getenv("OUTPUT_PATH")));

        int n = Integer.parseInt(bufferedReader.readLine().trim());

        String[] wTemp = bufferedReader.readLine().replaceAll("\\s+$", "").split(" ");

        List<Integer> w = new ArrayList<>();

        for (int i = 0; i < n; i++) {
            int wItem = Integer.parseInt(wTemp[i]);
            w.add(wItem);
        }

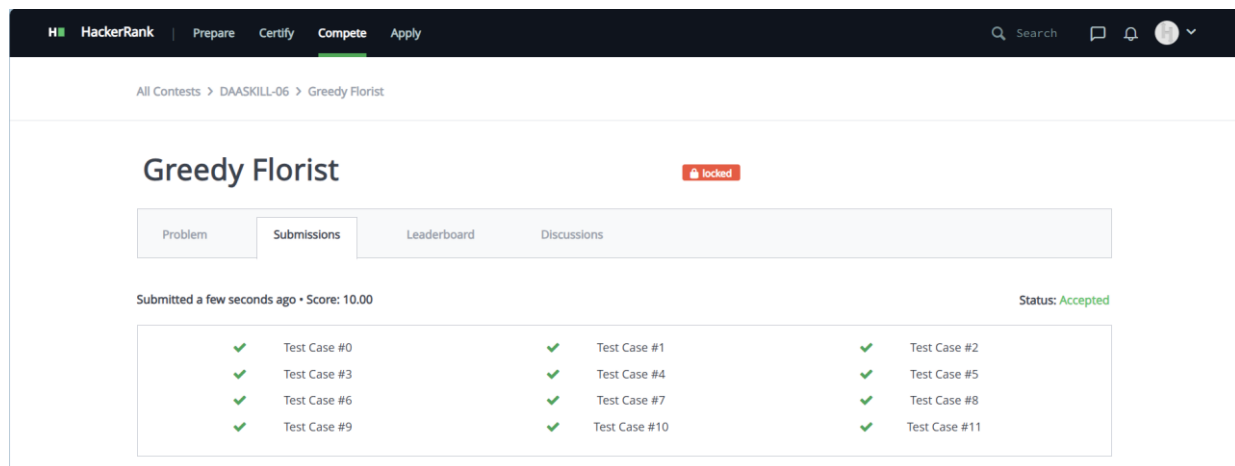
        int result = Result.toys(w);

        bufferedWriter.write(String.valueOf(result));
        bufferedWriter.newLine();

        bufferedReader.close();
        bufferedWriter.close();
    }
}

```

5) Greedy Florist:



Greedy Florist

locked

Problem Submissions Leaderboard Discussions

Submitted a few seconds ago • Score: 10.00 Status: Accepted

✓	Test Case #0	✓	Test Case #1	✓	Test Case #2
✓	Test Case #3	✓	Test Case #4	✓	Test Case #5
✓	Test Case #6	✓	Test Case #7	✓	Test Case #8
✓	Test Case #9	✓	Test Case #10	✓	Test Case #11

```
import java.io.*;
import java.math.*;
import java.security.*;
import java.text.*;
import java.util.*;
import java.util.concurrent.*;
import java.util.regex.*;

public class Solution {

    static int getMinimumCost(int k, int[] c) {
        int price=0,pre=0,j=k-1,n=c.length;
        Arrays.sort(c);
        for(int i=0;i<n;i++)
        {
            price+=c[n-i-1]*(pre+1);
            if(i==j)
            {
                pre++;
                j+=k;
            }
        }
        return price;
    }

    private static final Scanner scanner = new Scanner(System.in);

    public static void main(String[] args) throws IOException {
        BufferedWriter bufferedWriter = new BufferedWriter(new
        FileWriter(System.getenv("OUTPUT_PATH")));

        String[] nk = scanner.nextLine().split(" ");

        int n = Integer.parseInt(nk[0]);

        int k = Integer.parseInt(nk[1]);

        int[] c = new int[n];
```



```
String[] cItems = scanner.nextLine().split(" ");
scanner.skip("\\r\\n|[\\n\\r\\u2028\\u2029\\u0085]?");

for (int i = 0; i < n; i++) {
    int cItem = Integer.parseInt(cItems[i]);
    c[i] = cItem;
}

int minimumCost = getMinimumCost(k, c);

bufferedWriter.write(String.valueOf(minimumCost));
bufferedWriter.newLine();

bufferedWriter.close();

scanner.close();
}
}
```