

STAIRCASE PROBLEM

```
1  import java.io.*;
2  import java.math.*;
3  import java.security.*;
4  import java.text.*;
5  import java.util.*;
6  import java.util.concurrent.*;
7  import java.util.function.*;
8  import java.util.regex.*;
9  import java.util.stream.*;
10 import static java.util.stream.Collectors.joining;
11 import static java.util.stream.Collectors.toList;
12
13 class Result {
14
15     /*
16      * Complete the 'staircase' function below.
17      *
18      * The function accepts INTEGER n as parameter.
19      */
20
21     public static void staircase(int n) {
22
23         int space = n - 1;
24         int numhash = 1;
25
26         String ret = "";
27
28         for(int i = 0; i < n; i++){
29
30             ret += (" ".repeat(space)) + ("#".repeat(numhash));
31
32             if(i != (n - 1)){
33                 ret += "\n";
34             }
35
36             space = space - 1;
37             numhash = numhash + 1;
38
39         }
40
41         System.out.println(ret);
42
43     }
44
45 }
46
47 }
```

Congratulations

You solved this challenge. Would you like to challenge your friends?



Next Challenge

✓ Test case 0

✓ Test case 1

✓ Test case 2

✓ Test case 3

✓ Test case 4

✓ Test case 5

✓ Test case 6

Compiler Message

Success

Hidden Test Case

Unlock this testcase for 5 hackos.

Unlock

DIAGONAL DIFFERENCE

```
1  import java.io.*;
2  import java.math.*;
3  import java.security.*;
4  import java.text.*;
5  import java.util.*;
6  import java.util.concurrent.*;
7  import java.util.function.*;
8  import java.util.regex.*;
9  import java.util.stream.*;
10 import static java.util.stream.Collectors.joining;
11 import static java.util.stream.Collectors.toList;
12
13 class Result {
14
15     /*
16      * Complete the 'diagonalDifference' function below.
17      *
18      * The function is expected to return an INTEGER.
19      * The function accepts 2D_INTEGER_ARRAY arr as parameter.
20      */
21
22     public static int diagonalDifference(List<List<Integer>> arr) {
23         // Check if the matrix is square
24         int numRows = arr.size();
25         int numCols = arr.get(0).size();
26
27         if (numRows != numCols) {
28             throw new IllegalArgumentException("Input matrix must be square.");
29         }
30
31         int sum1 = 0;
32         int sum2 = 0;
33
34         int c1 = 0;
35         int c2 = numRows - 1;
36
37         for (int r = 0; r < numRows; r++) {
38             for (int c = 0; c < numCols; c++) {
39                 if (c == c1) {
40                     sum1 += arr.get(r).get(c1);
41                 }
42
43                 if (c == c2) {
44                     sum2 += arr.get(r).get(c2);
45                 }
46             }
47         }
48     }
49 }
```

```

43
44 ✓
45         if (c == c2) {
46             sum2 += arr.get(r).get(c2);
47         }
48
49         c1++;
50         c2--;
51     }
52
53     return Math.abs(sum1 - sum2);
54 }
55 }

```

Congratulations

You solved this challenge. Would you like to challenge your friends?


[Next Challenge](#)

✓ Test case 0

✓ Test case 1

✓ Test case 2

✓ Test case 3

✓ Test case 4

✓ Test case 5

✓ Test case 6

Compiler Message

Success

Input (stdin)

[Download](#)

```

1 3
2 11 2 4
3 4 5 6
4 10 8 -12

```

Expected Output

[Download](#)

```

1 15

```