Generating braids X45181_en

Write a program that, given triplets of positive natural numbers `lines, width, segment`, generates braids as shown in the examples.

Input

An arbitrary number of cases, each consisting of three positive natural numbers in one line.

Output

For each case, the corresponding braid, followed by a blank line.

Sample input 1

```
15 1 2
12 2 3
4 2 3
16 4 1
7 3 5
2 3 3
11 5 4
7 1 2
13 2 2
9 3 5
3 1 3
4 3 1
10 3 3
19 5 3
14 2 2
3 5 4
2 5 5
18 4 5
16 1 4
7 2 1
17 4 3
11 2 2
```

Sample output 1

```
*
 *
  *
 *
*
 *
  *
 *
*
 *
  *
 *
*
 *
  *

**
 **
  **
   **
  **
 **
**
 **
  **
   **
  **
 **

**
 **
  **
   **

****
 ****
****
 ****
****
 ****
```

```
****
 ****
****
 ****
****
 ****
****
 ****
****
 ****

***
 ***
  ***
   ***
    ***
     ***
   ***

***
 ***

*****
 *****
  *****
   *****
    *****
   *****
  *****
 *****
*****
 *****
  *****

*
 *
  *
 *
*
 *
  *

**
 **
  **
 **
**
 **
  **
 **
**
 **
  **
 **
**

***
 ***
  ***
   ***
    ***
```

```
     ***
    ***
   ***
  ***

*
 *
  *

***
 ***
***
 ***

***
 ***
  ***
   ***
  ***
 ***
***
 ***
  ***
   ***

*****
 *****
  *****
   *****
  *****
 *****
*****
 *****
  *****
   *****
  *****
 *****
*****

**
 **
  **
 **
**
 **
  **
 **
**
 **

*****
```

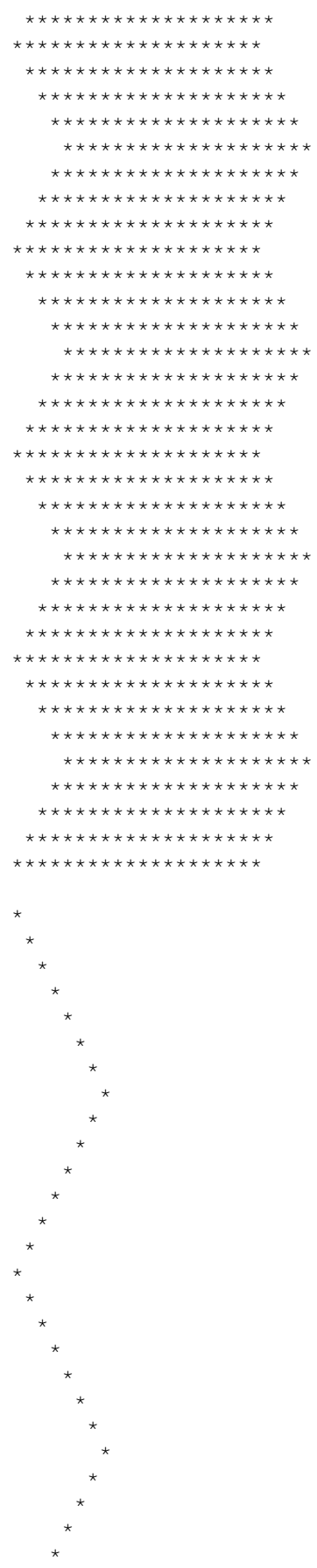
Sample input 2

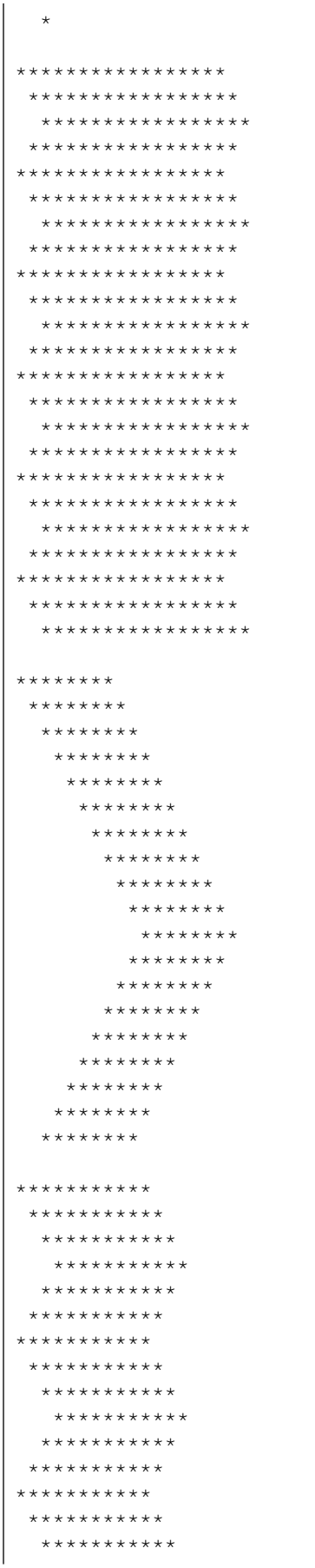
```
34 7 8
16 14 6
37 13 10
22 3 8
41 20 4
27 1 7
23 17 2
19 8 10
33 11 3
24 8 6
30 3 3
9 10 8
44 17 2
43 10 4
22 20 5
38 19 5
16 11 4
27 12 1
```

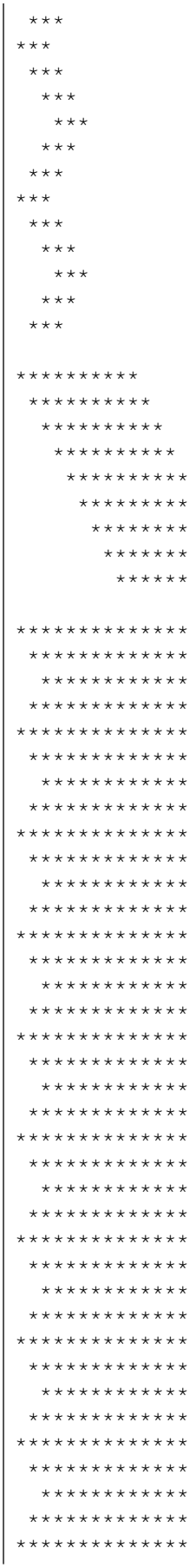
```
7 14 3
21 17 2
```

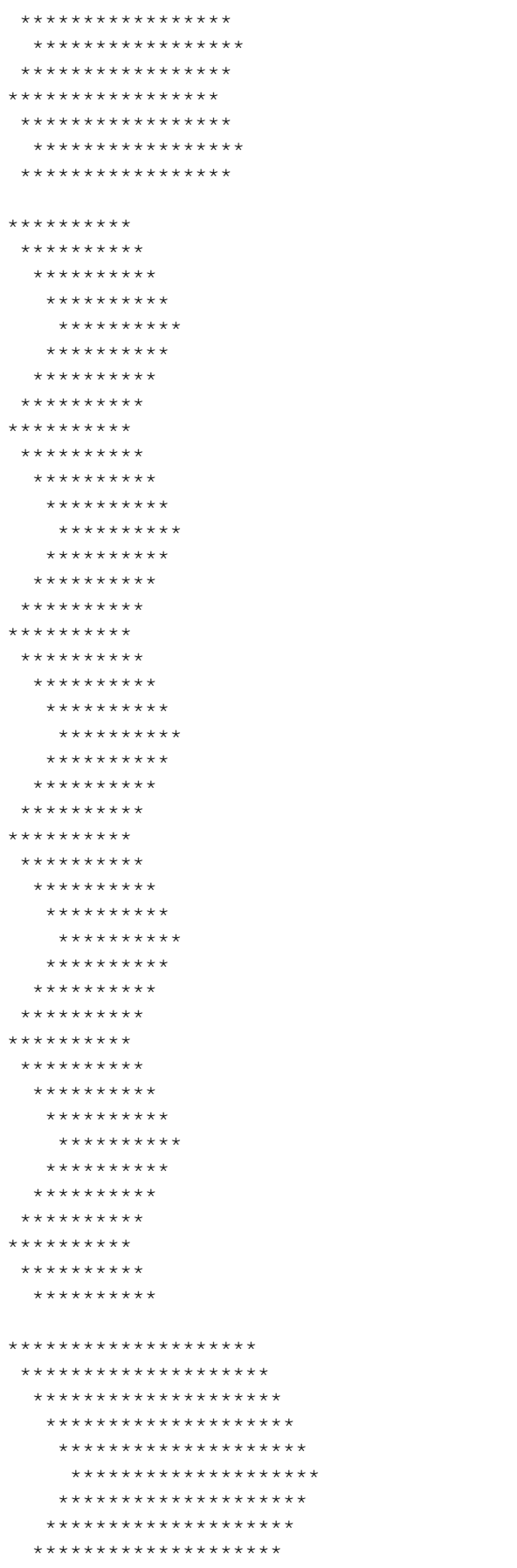
Sample output 2

```
*******
 *******
  *******
   *******
    *******
     *******
      *******
       *******
        *******
       *******
      *******
     *******
    *******
   *******
  *******
 *******
*******
 *******
  *******
   *******
    *******
     *******
      *******
       *******
        *******
       *******
      *******
     *******
    *******
   *******
  *******
 *******
*******
 *******

***************
 ***************
  ***************
   ***************
    ***************
     ***************
      ***************
       ***************
      ***************
     ***************
    ***************
   ***************
  ***************
 ***************
***************
 ***************
  ***************
   ***************

**************
 **************
  **************
   **************
    **************
     **************
      **************
       **************
      **************
     **************
    **************
   **************
  **************
 **************
**************
 **************
  **************
   **************
    **************
     **************
      **************
       **************
      **************
     **************
    **************
   **************
  **************
 **************
**************
 **************

***
 ***
  ***
   ***
    ***
     ***
      ***
       ***
        ***
       ***
      ***
     ***
    ***
   ***
  ***
 ***
***
 ***
  ***
   ***
    ***
     ***

********************
 ********************
  ********************
   ********************
    ********************
     ********************
      ********************
       ********************
      ********************
     ********************
    ********************
   ********************
  ********************
 ********************
********************
 ********************
  ********************
   ********************
    ********************
     ********************
      ********************
       ********************
      ********************
     ********************
    ********************
   ********************
  ********************
 ********************
********************
 ********************
```









```
***********
 ***********
  ***********
   ***********
    ***********
   ***********
  ***********
 ***********

************
 ************
************
 ************
************
 ************
************
 ************
************
 ************
************
 ************
************
 ************
************
 ************
************
 ************
************
```

```
 ************
************

**************
 **************
  **************
   **************
  **************
 **************
**************

******************
 ******************
  ******************
 ******************
******************
 ******************
  ******************
 ******************
******************
 ******************
  ******************
 ******************
******************
 ******************
  ******************
 ******************
******************
 ******************
  ******************
 ******************
******************
```

Observation

You can write the output character by character or by previously building strings, as you wish.
Grading up to 10 points:

- Slow solution: 5 points.
- Fast solution: 10 points.

We understand as a fast solution one which is correct, with linear cost and which passes the public and private tests. We understand as slow solution one which is not fast, but it is correct and passes the public tests.

Problem information

Author: PRO1

Generation: 2026-01-25T21:31:46.161Z