[CS 11 25.1] HOPE 1 A2 – Koyuki and Bombs 4

Cheatsheet is available here: https://oj.dcs.upd.edu.ph/cs11cheatsheet/

Submit solution [CS 11 25.1] HOPE 1

Points: 190 (partial)
Time limit: 12.0s
Memory limit: 2G

Problem type

Allowed languages
py3

Problem Statement

Koyuki has invented yet another kind of bomb!

She is still on the same infinite grid of cells. Again, each cell has position (i, j), where the i represents the vertical location and increases as you go down the grid, and the j represents the horizontal location and increases as you go right.

Like her previous bombs, this new bomb's area of effect depends on its *power* p. The area of effect can be described as follows:

- if $p = 0$, the area of effect looks like this:

```
...
.X.
...
```

- if $p > 0$, let `A` be the area of effect of a bomb with power $p - 1$. Then the area of effect looks like this:

```
A
 \
 2^(p-1)
   \
    \X
    X.X
     X\
       \
       2^(p-1)
         \
          A
```

The `2^(p-1)` here indicates that there are 2^{p-1} cells between the center cell and the center of `A`, **excluding** the center cell.

Here is what the area of effect looks like for small values of p:

```
0     1      2           3

X    XX.  XX.....  XX............
     X.X  X.X....  X.X...........
     .XX  .XXX...  .XXX..........
          ..X.X..  ..X.X.........
          ...XXX.  ...XXX........
          ....X.X  ....X.X.......
          .....XX  .....XXX......
                   ......X.X.....
                   .......XXX....
                   ........X.X...
                   .........XXX..
                   ..........X.X.
                   ...........XXX
                   ............X.X
                   .............XX
```

If the bomb is thrown at cell (i, j), and we sort all cells caught in the blast by increasing order of i, breaking ties by increasing order of j, what is the position of the kth cell caught in the blast?

Task Details

Your task is to implement a function named `identify_cell`. This function has three parameters:

- The first parameter is a pair of integers denoting the position of the cell at the center of the blast.
- The second parameter is p, an integer.
- The third parameter is k, an integer.

The function must return a pair of integers denoting the position of the kth cell caught in the blast.

Note that k here is 1-indexed. If there is no such kth cell, return `None`.

Restrictions

- Comprehensions are **disallowed**.
- Your source code must have at most 1400 bytes.

Examples

Example 1 Function Call

```
identify_cell((0, 0), 2, 6)
```

Example 1 Return Value

```
(-1, -1)
```

Example 2 Function Call

```
identify_cell((0, 0), 2, 1000)
```

Example 2 Return Value

```
None
```

Constraints

- The function `identify_cell` will be called at most 1000 times.
- The coordinates of the center of the blast have absolute value at most 10^{12}.
- $0 \le p \le 8888$
- $1 \le k \le 10^{12}$

Scoring

Note: New tests may be added and all submissions may be rejudged at a later time. (All future tests will satisfy the constraints.)

- You get 25 ❤️ points if you solve all test cases where:
  - $p \le 1$
  - The kth cell exists.
- You get 25 ❤️ points if you solve all test cases where:
  - $p \le 1$
- You get 10 🔴 points if you solve all test cases where:
  - $p \le 8$
  - The kth cell exists.
- You get 10 🔴 points if you solve all test cases where:
  - $p \le 8$
- You get 35 🔴 points if you solve all test cases where:
  - $p \le 888$
  - The kth cell exists.
- You get 35 🔴 points if you solve all test cases where:
  - $p \le 888$
- You get 25 💔 points if you solve all test cases where:
  - The kth cell exists.
- You get 25 💔 points if you solve all test cases.

Clarifications

Report an issue

No clarifications have been made at this time.