Octo-Octothorpes

Filename: `octothorpe`

There are several names for the symbol that appears when you hold the shift key and press 3 on a regular keyboard. Mathematicians commonly call it a number sign. German students sometimes call it a bicrux (meaning two crosses, one in the top left, one in the bottom right). Teenagers usually call it a hashtag. Telephone salesmen often call it a pound sign. Musical theorists and Microsoft developers call it a sharp sign. We prefer to call it an octothorpe.

Let's define a k-octothorpe. A 1-Octothorpe is simply 25 #'s and .'s in the shape of a # character as shown below:

```
.#.#.
#####
.#.#.
#####
.#.#.
```

A 2-octothorpe is the same symbol, but with all the periods replaced with a 5x5 grid of periods, and all #'s replaced by the above 5x5 1-octothorpe. Similarly, for any k > 1, a k-octothorpe is a (k-1)-octothorpe with all periods and #'s replaced as described above. For example, an 8-octothorpe (or octo-octothorpe, if you will) is created by taking a 7-octothorpe and replacing all #'s with the 5x5 1-octothorpe and all periods with a 5x5 grid of periods. A 2-octothorpe is shown below the sample output.

Please find the *n*th character of a *k*-octothorpe, if the characters are numbered starting at 1 in row-major order (that is, first numbered left-to-right on the top row, then left-to-right on the second row, et cetera).

The Problem:

Given *n* and *k*, find the *n*th character of a *k*-octothorpe.

The Input:

The first line of the input file begins with a single, positive integer, *t*, representing the number of test cases. Each test case begins starts with two integers $1 \leq n$, $k \leq 10^{18}$, representing the 1-based index of the requested character and the type of octothorpe respectively. For all test cases, it is guaranteed that a *k*-octothorpe contains at least *n* characters.

The Output:

For each test case, output a single line of the form "`Request #i: c`" without the quotes, where `i` is the number of the test case, and `c` is either "#" or ".".

Sample Input:

```
4
14 1
100 2
31 2
463 2
```

Sample Output:

```
Request #1: #
Request #2: .
Request #3: #
Request #4: #
```

Note: The requested locations are shown below in red.

```
.#.#.
#####
.#.#.   ← Request 1
#####
.#.#.
```

```
......#.#.......#.#......
.....#####.....#####.....  ← Request 3
......#.#.......#.#......
.....#####.....#####.....  ← Request 2
......#.#.......#.#......
.#.#..#.#..#.#..#.#..#.#.
###########################
.#.#..#.#..#.#..#.#..#.#.
###########################
.#.#..#.#..#.#..#.#..#.#.
......#.#.......#.#......
.....#####.....#####.....
......#.#.......#.#......
.....#####.....#####.....
......#.#.......#.#......
.#.#..#.#..#.#..#.#..#.#.
###########################
.#.#..#.#..#.#..#.#..#.#.
##########################  ← Request 4
.#.#..#.#..#.#..#.#..#.#.
......#.#.......#.#......
.....#####.....#####.....
......#.#.......#.#......
.....#####.....#####.....
......#.#.......#.#......
```