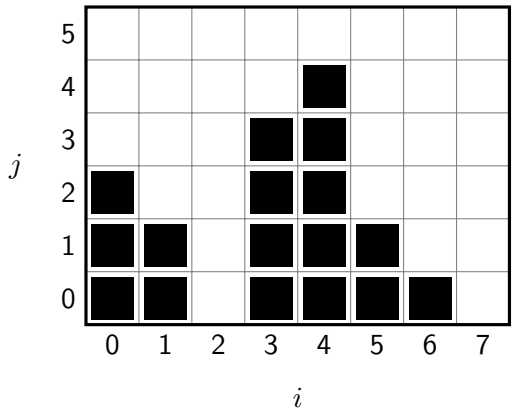
beCP
2020
Task 3: Skyline (skyline)

Author: Prasanna Ramakrishnan Preparation: Victor Lecomte
Time limit: 1 s Memory limit: 512 MB

Note: This task is interactive. Please look at the special instructions to implement, compile and test your program. Don't hesitate to ask the staff for help.

Instead of winning a trip to Singapore, all you got was a cheesy postcard of the city's skyline. You're understandably disappointed, but cherish this symbolic picture nevertheless. In a moment of boredom, you decide to figure out the height of the highest building in the postcard.

The image is n pixels wide and m pixels tall. Each pixel is either black (building) or white (empty). There are no overhangs: no black pixel is directly above a white pixel. You can query pixels one by one using their coordinates. Determine the maximum number of black pixels in a single column, using q queries or less.



in the above postcard, $n = 8$ and $m = 6$

Functions to implement

C++	`long long findHighest(long long n, long long m)`
	Given dimensions n and m of the image, calls `isBlack(i,j)` repeatedly to find the highest building.
return	An integer, the number of black pixels in the column that has the most black pixels.

Functions to call

C++	`bool isBlack(long long i, long long j)`
	Queries the color of the pixel (i, j): the pixel in column i ($0 \leq i < n$) and row j ($0 \leq j < m$). Note the 0-indexing, and note that rows are numbered from bottom to top!
`return`	`true` if pixel (i, j) is black, and `false` if it is white.

Limits

- $1 \leq n \leq 10^5$, the width of the postcard;
- $1 \leq m \leq 10^{18}$, the height of the postcard.

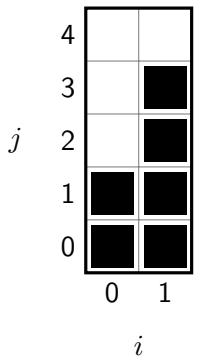
Additional constraints

For each subtask, this table gives an **upper bound** on n, m, and a **lower bound** on q (the maximum number of times you can call `isBlack()`).

Subtask	Points	$n \leq$	$m \leq$	$q \geq$
A	5	20	10^3	10^6
B	10	20	10^{18}	10^6
C	15	10^5	10^5	10^6
D	20	10^5	10^7	10^6
E	30	10^5	10^{18}	10^6
F	20	20	10^{18}	190

Example interaction

Consider the following postcard, with $n = 2$ and $m = 5$.



Your function is called as `findHighest(2,5)`. It makes the following calls to function `isBlack(i,j)`.

Call	Result
`isBlack(0,0)`	**`true`**
`isBlack(0,1)`	**`true`**
`isBlack(0,2)`	**`false`**
`isBlack(1,4)`	**`false`**
`isBlack(1,3)`	**`true`**

The information obtained is sufficient to conclude that the first column has 2 black pixels and the second column has 4 black pixels. The bigger of the two is 4, so `findHighest()` returns `4`. Of course this is only an example; you are free to use the queries as you see fit.

Implementation, compilation and testing

You must implement function `findHighest(n,m)` located in file `skyline.cpp`. To compile then execute, use the following commands:

C++	`g++ -std=c++11 -Wall grader.cpp skyline.cpp` `./a.out < skyline.in`

To test your code on several possible cases, you can modify file `sample.in`. The first line of this file contains n and m. The second line contains n integers, each between 0 and m: the number of black pixels in each column. Here are the inputs corresponding to the two examples in this task description.

sample.in (first example)
```
8 6
3 2 0 4 5 2 1 0
```

sample.in (second example)
```
2 5
2 4
```

Submission and verdict

- You should only submit file `skyline.cpp`.
- A "Wrong Answer" verdict can either mean that you used too many questions of a certain type, or that the array you return is incorrect.
- A "Runtime Error" verdict can (among other things) mean that you asked an invalid query.
- Do not print anything to `stdout` in your program: never use `cout`, `printf()` or `System.out.println()`.