Paint By Numbers

Paint By Numbers is a well-known puzzle game. We consider a simple one-dimensional version of this puzzle. In this puzzle, the player is given a row of n cells. The cells are numbered 0 through $n-1$ from the left to the right. The player has to paint each cell black or white. We use 'X' to denote black cells and '_' to denote white cells.

The player is given a sequence $c = [c_0, \ldots, c_{k-1}]$ of k positive integers: the *clues*. He has to paint the cells in a way such that the black cells in the row form exactly k blocks of consecutive cells. Moreover, the number of black cells in the i-th block (0-based) from the left should be equal to c_i. For example, if the clues are $c = [3, 4]$, the solved puzzle must have exactly two blocks of consecutive black cells: one of length 3 and then another of length 4. Hence, if $n = 10$ and $c = [3, 4]$, one solution satisfying the clues is "_XXX__XXXX". Note that "XXXX_XXX__" does not satisfy the clues because the blocks of black cells are not in the correct order. Also, "__XXXXXXX_" does not satisfy the clues because there is a single block of black cells, not two separate blocks.

You are given a partially solved Paint By Numbers puzzle. That is, you know n and c, and additionally you know that some cells must be black and some cells must be white. Your task is to deduce additional information about the cells.

Specifically, a *valid solution* is one that satisfies the clues, and also agrees with the colors of the known cells. Your program should find cells that are painted black in every valid solution, and cells that are painted white in every valid solution.

You may assume that the input is such that there is at least one valid solution.

Implementation details

You should implement the following function (method):

- `string solve_puzzle(string s, int[] c)`.
  - `s`: string of length n. For each i ($0 \le i \le n-1$) character i is:
    - 'X', if cell i must be black,
    - '_', if cell i must be white,
    - '.', if there is no information about cell i.
  - `c`: array of length k containing clues, as defined above,
  - the function should return a string of length n. For each i ($0 \le i \le n-1$) character i of the output string should be:
    - 'X', if cell i is black in every valid solution,
    - '_', if cell i is white in every valid solution,

    - '?', otherwise (i.e., if there exist two valid solutions such that cell i is black in one of them and white in the other one).

In the C language the function signature is a bit different:

- `void solve_puzzle(int n, char* s, int k, int* c, char* result)`
    - `n`: length of the string `s` (number of cells),
    - `k`: length of the array `c` (number of clues),
    - the other parameters are the same as above,
    - instead of returning a string of n characters, the function should write the answer to the string `result`.

The ASCII codes of characters used in this problem are:

- 'X': 88,
- '_': 95,
- '.': 46,
- '?': 63.

Please use the provided template files for details of implementation in your programming language.

Examples

Example 1

```
solve_puzzle("..........", [3, 4])
```

These are all possible valid solutions of the puzzle:

- "XXX_XXXX__",
- "XXX__XXXX_",
- "XXX___XXXX",
- "_XXX_XXXX_",
- "_XXX__XXXX",
- "__XXX_XXXX".

One can observe that the cells with (0-based) indices 2, 6, and 7 are black in each valid solution. Each of the other cells can be, but does not have to be black. Hence, the correct answer is "??X???XX??".

Example 2

```
solve_puzzle("........", [3, 4])
```

In this example the entire solution is uniquely determined and the correct answer is "XXX_XXXX".

Example 3

```
solve_puzzle("..._._....", [3])
```

In this example we can deduce that cell 4 must be white as well — there is no way to fit three consecutive black cells between the white cells at indices 3 and 5. Hence, the correct answer is "???___????".

Example 4

```
solve_puzzle(".X........", [3])
```

There are only two valid solutions that match the above description:

- "XXX_______",
- "_XXX______".

Thus, the correct answer is "?XX?______".

Subtasks

In all subtasks $1 \leq k \leq n$, and $1 \leq c_i \leq n$ for each $0 \leq i \leq k-1$.

1. (7 points) $n \leq 20$, $k = 1$, s contains only '.' (empty puzzle),
2. (3 points) $n \leq 20$, s contains only '.',
3. (22 points) $n \leq 100$, s contains only '.',
4. (27 points) $n \leq 100$, s contains only '.' and '_' (information only about white cells),
5. (21 points) $n \leq 100$,
6. (10 points) $n \leq 5\,000$, $k \leq 100$,
7. (10 points) $n \leq 200\,000$, $k \leq 100$.

Sample grader

The sample grader reads the input in the following format:

- line 1: string s,
- line 2: integer k followed by k integers $c_0, \ldots, c_{k-1}$.