# Paint By Numbers

Paint By Numbers는 유명한 퍼즐게임으로 이 문제에서는 단순한 1차원 버전의 퍼즐을 고려한다. 우선, 플레이어에게는 $n$ 개의 칸을 가진 한 행이 주어진다. 각 칸에는 왼쪽부터 오른쪽으로 0번부터 $n-1$ 까지 번호가 매겨져 있다. 플레이어는 각 칸을 검정색이나 흰색으로 칠해야 한다. 우리는 검정색 칸을 'X'로, 흰색 칸을 '_'로 표현한다.

플레이어에게는 추가로 **단서들**이 주어지는데, 이는 $k$ 개의 양의 정수로 구성된 수열 $c = [c_0, \ldots, c_{k-1}]$ 이다. 플레이어는 다음과 같은 방식으로 칸들을 색칠해야 한다: 검정색 칸들이 정확히 $k$ 개의 블록을 이루며, 이때 각 블록은 연속된 칸들로 구성된다. 또한, 왼쪽부터 ($0$ 번째부터 시작해서) $i$ 번째 블록의 검정색 칸의 개수는 $c_i$ 와 같다. 예를 들어, 만약 단서들이 $c = [3, 4]$ 라면, 퍼즐의 해에는 연속된 검정색 칸들로 구성된 블록이 정확히 두 개 있어야 한다: 한 블록은 길이가 3이고 다른 블록은 길이가 4이다. 따라서, 만약 $n = 10$ 이고 $c = [3, 4]$ 라면, 단서들을 만족하는 해 중 한가지는 "_XXX__XXXX"이다. 참고로, "XXXX_XXX__"는 검정색 칸들의 블록들의 순서가 바르지 않으므로 단서들을 만족하지 않는다. 또한, "__XXXXXXX_"도 두 개의 구분된 검정색 칸들의 블록이 아니라 단 하나의 블록만 있으므로 단서들을 만족하지 않는다.

여러분에게는 일부분이 풀린 퍼즐의 해가 주어진다. 즉, 여러분은 $n$ 과 $c$ 를 알며, 추가로 검정색으로 칠해진 일부 칸들과 흰색으로 칠해진 일부 칸들도 안다. 여러분의 작업은 칸들에 대한 추가적인 정보를 추론하는 것이다.

구체적으로, **유효한 해**는 단서들을 만족하고 또한 색깔이 알려진 칸들을 만족하는 해이다. 여러분의 프로그램은 모든 유효한 해들에서 항상 검정색으로 칠해져야만 하는 칸들을 찾아야 하고, 모든 유효한 해들에서 항상 흰색으로 칠해져야만 하는 칸들을 찾아야 한다.

유효한 해가 적어도 하나는 존재하는 입력이 주어진다고 가정한다.

## Implementation details

다음 함수를 구현하여야 한다:

- `string solve_puzzle(string s, int[] c)`.
  - `s`: 길이 $n$ 인 문자열. 각 $i$ ($0 \le i \le n-1$)에 대해 문자 $i$ 는:
    - 'X', 칸 $i$ 가 검정색인 경우,
    - '_', 칸 $i$ 가 흰색인 경우,
    - '.', 칸 $i$ 에 대한 정보가 없는 경우.
  - `c`: (위에서 정의한)단서들에 대한 길이 $k$ 인 배열.
  - 이 함수는 길이 $n$ 인 하나의 문자열을 리턴해야 한다. 각 $i$ ($0 \le i \le n-1$)에 대해 리턴 문자열의 문자 $i$ 는:
    - 'X', 칸 $i$ 가 모든 유효한 해에서 검정색이어야 하는 경우,
    - '_', 칸 $i$ 가 모든 유효한 해에서 흰색이어야 하는 경우,
    - '?', 나머지 경우(즉, 칸 $i$ 가 검정색인 유효한 해도 존재하고 칸 $i$ 가 흰색인 유효한 해도 존재하는 경우).

C 언어를 사용한다면 아래와 같이 작성한다:

- `void solve_puzzle(int n, char* s, int k, int* c, char* result)`
  - `n`: 문자열 `s`의 길이(칸의 개수),
  - `k`: 배열 `c`의 길이(단서의 개수),
  - 다른 파라미터들은 위와 동일하다.
  - $n$ 개의 문자들로 구성된 문자열을 리턴하는 대신, 문자열 `result`에 답을 저장한다.

문제에서 사용되는 문자들에 대한 ASCII 코드는 다음과 같다:

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

사용하는 언어 별로 제공되는 template 파일을 참고하여 구현의 디테일을 확인하시오.

## Examples

### Example 1

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

가능한 모든 유효한 해들은 다음과 같다:

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

(0부터 시작해서) 인덱스 2, 6, 7인 칸은 모든 유효한 해에서 검정색이다. 다른 칸들은 검정색인 경우도 있지만 항상 검정색이어야 하는 것은 아니다. 따라서, 바른 답은 "`??X???XX??`"이다.

### Example 2

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

이 예에 대해서는 유일한 해만 존재하므로, 바른 답은 "`XXX_XXXX`"이다.

### Example 3

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

이 예에서는 인덱스 4인 칸이 흰색이어야만 한다고 추론할 수 있다 — 흰색의 인덱스 3인 칸과 인덱스 5인 칸 사이에 3개의 연속된 검정색 칸을 넣을 수 없으므로. 따라서, 바른 답은 "`???___????`"이다.

### Example 4

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

입력에 해당하는 유효한 해는 두 가지 밖에 없다:

- "`XXX_______`",
- "`_XXX______`".

따라서, 바른 답은 "?XX?_____"이다.

## Subtasks

모든 subtask에서, $1 \le k \le n$, 그리고 각 $0 \le i \le k-1$ 에 대해 $1 \le c_i \le n$ 임.

1. (7 points) $n \le 20$, $k = 1$, $s$는 '.'만 가짐(빈 퍼즐),
2. (3 points) $n \le 20$, $s$는 '.'만 가짐,
3. (22 points) $n \le 100$, $s$는 '.'만 가짐,
4. (27 points) $n \le 100$, $s$는 '.'과 '_'만 가짐(오직 흰색 칸에 대한 정보),
5. (21 points) $n \le 100$,
6. (10 points) $n \le 5\,000$, $k \le 100$,
7. (10 points) $n \le 200\,000$, $k \le 100$.

## Sample grader

Sample grader는 다음의 형식으로 입력을 읽어들인다:

- line 1: 문자열 $s$,
- line 2: 정수 $k$가 나온 다음 이어서 $k$개의 정수 $c_0, \ldots, c_{k-1}$.