Paint By Numbers

Colorear por números es un juego de rompecabezas muy conocido. Sin embargo, consideraremos una version simple de este juego de unicamente una dimensión para el rompecabezas. En este rompecabezas, se le da al jugador una fila de n celdas. Las celdas se encuentran enumeradas desde 0 a $n-1$ de izquierda a derecha. El jugador tiene que pintar cada celda ya sea con blanco o negro. Para ello, utilizaremos la letra 'X' para denotar las celdas pintadas de negro y '_' para denotar las celdas pintadas de blanco.

Al jugador se le da una secuencia de k enteros positivos $c = [c_0, \ldots, c_{k-1}]$: que son consideradas las *pistas*. El jugador debe lograr pintar las celdas de tal manera que las celdas negras en la fila, formen exactamente un bloque consecutivo de k celdas. Sin embargo, el numero de celdas negras en el bloque i-th (0 -considerando como la posicion inicial) tomandolo desde la izquierda debe ser igual a c_i . Por ejemplo, si las pistas son $c = [3, 4]$, el rompecabezas ya resuelto deberia tener exactamente dos bloques de celdas negras consecutivas: un bloque de 3 celdas de longitud y otro de 4 celdas. Por lo tanto, si $n = 10$ y $c = [3, 4]$, una posible solución valida sería "_XXX__XXXX". Ten en cuenta que "XXXX_XXX__" no es una solución valida ya que los bloques de celdas negras no estan en el orden correcto. Así tambien "__XXXXXXX_" no es una solución valida ya que existe un solo bloque de celdas negras y no se encuentra separado.

A tí te daran una solución casi ya terminada del rompecabeza, de esa manera solamente necesitaras n y c , pero adiconalmente sabrás tambien que algunas de las celdas ya estarán predefinidas como celdas blancas o negras. Tu trabajo es deducir la información restante. Especificamente, tu debes encontrar las celdas que deberan ser pintadas con negro o blanco para cada posible solución valida. Debes asumir que la entrada producirá al menos una solución valida.

Detalles de implementación

Debes implementar la siguiente función (method):

- `string solve_puzzle(string s, int[] c)`.
  - `s`: cadena de longitud n . Para cada i ($0 \leq i \leq n-1$) el caracter i puede ser:
    - 'X', si las celdas i deben ser negras,
    - '_', si las celdas i deben ser blancas,
    - '.', si no hay información acerca de la celda i .
  - `c`: vector de longitud k que contiene las pistas, como está definido arriba,
  - la función debe retornar una cadena de longitud n . Para cada i ($0 \leq i \leq n-1$) el caracter i de la cadena de salida debe ser:

    - 'X', si la celda i es negra para cada solución valida,
    - '_', si la celda i es blanca para cada solución valida,
    - '?', para el resto (Es decir, si existen dos soluciones validas de tal manera que la celda i es negra en una de ellas y blanca en las otras).

En el lenguaje C la el formato de la función es un tanto diferente:

- `void solve_puzzle(int n, char* s, int k, int* c, char* result)`
    - `n`: longitud de la cadena `s` (número de celdas),
    - `k`: longitud del vector `c` (número de pistas),
    - los otros parametros son los mismos explicados anteriormente,
    - En vez de devolver una cadena de n caracteres, la función debe retornar la respuesta en la cadena `result`.

El código ASCII utilizado para los caracteres de este problema son:

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

Porfavor usa los ejemplos de archivos plantillas provistos para los detalles de implementación en tu lenguaje de programación

Ejemplos

Ejemplo 1

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

Estas son todas las posibles soluciones para el rompecabeza:

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

Se puede observar que la celda con los indices 2, 6 y 7 son negras para cada solución valida(Teniendo en cuenta como indice 0 el primer caracter). Cada una de las otras celdas tambien podrian serlas, pero no deben ser negras, es aí, que la respuesta correcta es "??X???XX??".

Ejemplo 2

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

En este ejemplo la solución entera esta determinada unicamente y la respuesta correcta es "XXX_XXXX".

Ejemplo 3

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

En este ejemplo podemos deducir que la celda 4 debe ser blanca, de esa manera no existe forma de acomodar tres celdbas negras de forma consecutiva entre las celdas blancas en los indices 3 y 5. Es así, que las respuesta correcta es "`???___????`".

Ejemplo 4

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

Solo existe dos soluciones validas que concuerdes con las descripción provista:

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

De esa manera, la resuesta correcta es "`?XX?______`".

Subtasks

En todas las subtareas $1 \leq k \leq n$, y $1 \leq c_i \leq n$ para cada $0 \leq i \leq k-1$.

1. (7 puntos) $n \leq 20$, $k = 1$, donde s solo contiene '`.`' (un rompecabezas vacio),
2. (3 puntos) $n \leq 20$, donde s contiene solo '`.`',
3. (22 puntos) $n \leq 100$, donde s contiene solo '`.`',
4. (27 puntos) $n \leq 100$, donde s contiene solo '`.`' y '`_`' (información unicamente de las celdas blancas),
5. (21 puntos) $n \leq 100$,
6. (10 puntos) $n \leq 5\,000$, $k \leq 100$,
7. (10 puntos) $n \leq 200\,000$, $k \leq 100$.

Ejemplo de "Grader"

El Grader de muestra lee las entradas con el siguiente formato:

- línea 1: cadena s,
- línea 2: entero k seguido de k enteros $c_0, \ldots, c_{k-1}$.