

Paint By Numbers

Paint By Numbers е известна игра от тип пъзъл. Ние разглеждаме простата едномерна версия, която се състои от един ред с n клетки. Клетките са номерирани от 0 до $n - 1$ отляво-надясно. Играчът трябва да оцвети всяка клетка в черно или в бяло. Означаваме с 'X' черна клетка, а с '_' бяла клетка.

Играчът получава редица $c = [c_0, \dots, c_{k-1}]$ от k положителни цели числа: *ключове*. Той трябва да оцвети клетките така, че черните клетки в реда да образуват точно k блока от последователни клетки. Освен това, броят на черните клетки в i -тия блок (броенето започваме от 0) от ляво трябва да е равен на c_i . Например, ако ключовете са $c = [3, 4]$, нареденият пъзел трябва да има точно 2 блока от последователни черни клетки: единият с дължина 3 и след това друг блок с дължина 4. Следователно, ако $n = 10$ и $c = [3, 4]$, едно валидно решение е "_XXX_XXXX". Забележете, че "XXXX_XXX_" не е валидно решение: блоковете от черни клетки не са в правилна наредба. Освен това, "_XXXXXXXX_" не е валидно решение: има един блок от черни клетки, а не 2 отделни блока.

Даден ви е частично решен пъзел Paint By Numbers. Т.е. вие знаете n и c , и допълнително знаете, че някои клетки трябва да са черни и някои трябва да са бели.

Валидно решение е такова, което удовлетворява ключовете и цветовете на изветните клетки.

Вашата задача е да извлечете допълнителна информация. По-специално, вие трябва да намерите клетки, които са оцветени черно във всяко валидно решение и клетки, които са оцветени бяло във всяко валидно решение. Може да считате, че входът е такъв, че има поне едно валидно решение.

Детайли по реализацията

Трябва да реализирате следната функция:

- `string solve_puzzle(string s, int[] c)`.
 - s : стринг с дължина n . За всяко i ($0 \leq i \leq n - 1$) знакът i е:
 - 'X', ако клетката i трябва да е черна,
 - '_', ако клетката i трябва да е бяла,
 - '.', ако няма информация за цвета на клетката i .
 - c : масив с дължина k , съдържащ ключовете, както е дефинирано по-горе,
 - функцията трябва да върне стринг с дължина n . За всяко i ,

$0 \leq i \leq n - 1$, знакът i от изходния стринг трябва да е:

- 'X', ако клетката i е черна във всяко валидно решение,
- '_', ако клетката i е бяла във всяко валидно решение,
- '?', в притивен случай (т.е. ако съществуват две валидни решения, такива че клетката i е черна в едното решение и бяла в другото решение).

В езика C функцията е малко по-различна:

- `void solve_puzzle(int n, char* s, int k, int* c, char* result)`
 - n : дължина на стринг s (брой на клетките),
 - k : дължина на стринг c (брой на клетките),
 - останалите параметри са същите, както по-горе,
 - вместо да върне стринг от n знака, функцията трябва да запише отговора в стринга `result`.

ASCII кодовете на знаците са:

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

Примери

Пример 1

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

Всички възможни валидни решения на пъзела са:

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

Забелязваме, че клетките с индекси 2, 6 и 7 (индексите броим от 0) са черни във всяко валидно решение. Всяка от останалите клетки може, но не е задължена да е черна. Следователно правилният отговор е "??X???XX??".

Пример 2

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

В този пример решението е еднозначно определено и отговорът е "XXX_XXXX".

Пример 3

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

За този пример заключаваме, че клетка 4 трябва да е бяла и освен това, не може да има 3 последователни черни клетки между бели клетки с индекси 3 и 5.

Верният отговор е "???"__"???"

Пример 4

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

Има само две валидни решения на пъзела, които са:

- "XXX_____",
- "_XXX_____".

Следвателно, верният отговор е "?XX?_____".

Подзадачи

Във всички подзадачи $1 \leq k \leq n$, и $1 \leq c_i \leq n$ за всяко $0 \leq i \leq k - 1$.

1. (7 points) $n \leq 20$, $k = 1$, s съдържат само '.' (празен пъзел),
2. (3 points) $n \leq 20$, s съдържат само '.',
3. (22 points) $n \leq 100$, s съдържат само '.',
4. (27 points) $n \leq 100$, s съдържат само '.' и '_' (информация само за бели клетки),
5. (21 points) $n \leq 100$,
6. (10 points) $n \leq 5\,000$, $k \leq 100$,
7. (10 points) $n \leq 200\,000$, $k \leq 100$.

Примерен грейдер

Примерният грейдер чете входа във следния формат:

- ред 1: стринг s ,
- ред 2: цяло число k , следвано от k цели числа c_0, \dots, c_{k-1} .