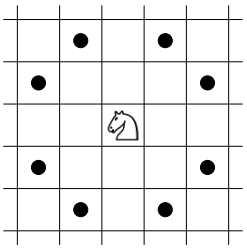
C. Skoczki

Dostępna pamięć: 256 MB

Postawiony na polu szachownicy skoczek atakuje pola zaznaczone na poniższym rysunku kropką. Jeśli na atakowanym polu stoi inny skoczek, to mówimy, że para skoczków się (wzajemnie) atakuje.



Dana jest szachownica o rozmiarach $3 \times n$, gdzie n jest dodatnią liczbą całkowitą. Część pól szachownicy jest *zniszczona* i nie nadaje się do użytku (a tym bardziej do stawiania na nich skoczków). Twoim zadaniem jest określenie, jaka jest największa liczba skoczków, które można ustawić na szachownicy, tak żeby żadna para skoczków się nie atakowała.

Specyfikacja danych wejściowych

W pierwszym wierszu danych wejściowych znajduje się liczba naturalna $1 \leqslant n \leqslant 2 \cdot 10^6$, będąca długością szachownicy. W każdym z kolejnych n wierszy znajduje się ciąg trzech znaków opisujący kolejny wiersz szachownicy. Każdy z tych znaków to „`x`” (zniszczone pole) albo „`.`” (niezniszczone pole). Znaki nie są oddzielone spacjami.

Specyfikacja danych wyjściowych

W pierwszym wierszu wyjścia Twój program powinien wypisać jedną liczbę naturalną będącą maksymalną liczbą nieatakujących się skoczków, które można umieścić na zadanej szachownicy, tak żeby żaden skoczek nie stał na zniszczonym polu.

W kolejnych n wierszach należy wypisać ustawienie tych skoczków. W każdym z tych wierszy powinny znaleźć się trzy znaki ze zbioru {`x`, `.`, `S`} opisujące kolejne pola danego wiersza szachownicy: `S` jest polem na którym stoi skoczek, `x` jest zniszczonym polem szachownicy, zaś `.` jest niezniszczonym polem szachownicy bez skoczka.

Znaki nie powinny być oddzielone spacjami. Jeśli jest więcej niż jedno ustawienie o maksymalnej liczbie skoczków, Twój program może wypisać dowolne z nich.

Przykład A

Wejście:	**Wyjście:**
2 ..x x..	3 .Sx xSS

Przykład B

Wejście:	**Wyjście:**
3	5 S.S .S. S.S

Przykład C

Wejście:	**Wyjście:**
3 xxx .x. .x.	2 xxx .x. SxS