Piastrellature di un bagno 1xn con piastrelle 1x1 e 1x2: rank

Voglio piastrellare un bagno largo 1 e lungo n evitando di sovrapporre le piastrelle ma ricoprendolo perfettamente in ogni punto. Fortunatamente dispongo in quantità illimitata di due tipi di piastrelle:

- le piastrelle 1 per 1;
- le piastrelle 1 per 2.

Ecco le possibili piastrellature di un bagno 1x3 ordinate secondo il nostro criterio di ordinamento:

```
piastrellatura 0:  [-][-][-]
piastrellatura 1:  [-][----]
piastrellatura 2:  [----][-]
```

E' richiesta la competenza di saper individuare la corretta posizione di una piastrellatura di un bagno 1xn entro la lista analogamente ordinata delle piastrellatura di detto bagno:

```
[-][-][-][-]  --> rank -->  0
[-][-][----]  --> rank -->  1
[-][----][-]  --> rank -->  2
[----][-][-]  --> rank -->  3
[----][----]  --> rank -->  4
[-][-][-][-][-]  --> rank -->  0
[-][-][-][-][-][-]  --> rank -->  0
```

Input

La prima riga contiene T, il numero di testcase da risolvere. Seguono T istanze del problema. Ogni istanza è composta da una singola riga contenente una piastrellatura di un qualche bagno 1xn. Ad esempio, la stringa "[—-][—-][-]".

Output

In ogni testase, l'unica riga dell'output deve contenere il rango della piastrellatura assegnata (ad esempio "[—-][—-][-]") tra le piastrellature possibilili per quel bagno 1xn (nell'esempio n=5 e il rango da restituire è 7 perchè il rango della prima piastrellatura è sempre zero mentre quello dell'ultima è pari al numero delle piastrellature possibili ridotto di uno).

Esempio

Input

```
4
[-][-][-][-][-]
```

```
[-][-][----][-]
[----][----][-]
[-][-][-][-][-][-]
```

Output

```
0
2
7
0
```

Assunzioni

Per il subtasking sono previste le seguenti `size`, dove il default è `big` che include tutti i testcase:

- `small`: $n \leq 10$
- `big`: $n \leq 1000$

Il tempo limite per testcase è di 1 secondo.