И така, в задачата се търси броят на n-цифрените двоични числа без водещи нули, които се делят на 3. Като извършим елементарни изчисления, получаваме, че търсеният брой е $C(n) = \begin{cases} \frac{2^{n-1}-1}{3}, \text{при нечетно } n \\ \frac{2^{n-1}+1}{3}, \text{при четно } n \end{cases}$.

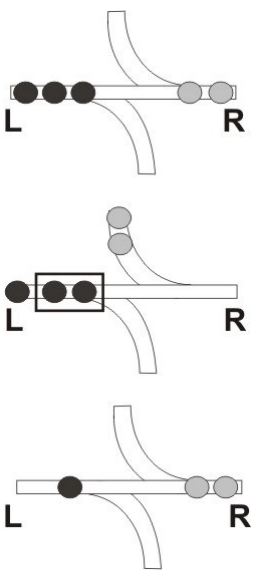
Тъй като ограничението за n е 60, то 64-битово цяло число ще е достатъчно да обхване този брой. По-долу е даден вариант на решението, написан на C/C++:

```
#include <iostream>
using namespace std;
int main(void)
{  int n;
   long long c;
   cin>>n;
   c=(long long)1<<(n-1);
   if (n&1) c--;
   else c++;
   cout<<c/3<<endl;
}
```

Група D

Задача D1. ЧАЙКА, Галина Момчева

На една от улиците в курортен комплекс „Чайка" е паднал пътният знак, който показвал, че улицата е еднопосочна. Автомобили, идващи от двете посоки трябва да се разминат. Но как?



Случайно минаващият от там пешеходец (и начинаещ програмист) Станчо измислил алгоритъм за разминаване. За щастие има достатъчно дълги отбивки в дясно за всяка от колоните. Правилата, които Станчо измислил (според него справедливи) са следните:

1. Дава знак на колоната с по-малко коли да влезе в отбивката си. Ако колите от двете страни са по равен брой, тогава в отбивката влизат колите отляво.
2. Пуска да минат през кръстовището толкова коли от по-дългата колона, колкото е имало в по-късата.
3. Връща колите от отбивката обратно на улицата.

След изпълнението на стъпки 1, 2 и 3 следва нова оценка на ситуацията, която се регулира по същите правила. Не е изключено, обаче, по време на изпълнението на стъпки 1, 2 и 3 да се появят нови коли и от двете страни. Напишете програма **CHAI**, която да помогне на Станчо в регулирането на движението.

Вход

На първия ред на стандартния вход са зададени две цели числа: общият брой N на пристигналите отляво и отдясно коли и броят C на повторенията на стъпките 1, 2 и 3, $0 < N \leq 100$, $0 < C \leq 20$. Следва последователност от $N + C$ реда, на всеки от които има по една от латинските букви `L`, `R` и `P`, като броят на редовете с букви `P` е точно C. Буквата `L` означава, че е пристигнала кола отляво, а буквата `R` – отдясно. Буквата `P` означава, че е извършено регулиране. Входът започва с буква, различна от `P` и завършва с буква `P`. При всяко регулиране, във всяка от двете посоки ще има поне по една кола.

Изход

На първите C реда на стандартния изход, за всяко изпълняване на стъпките 1, 2 и 3, се извежда колко коли са минали през кръстовището и от коя страна на кръстовището са дошли. На последните два реда се извеждат броя на колите останали след последното регулиране в кръстовището, съответно от лявата и дясната част на пътя. Ако няма такива се извежда 0.

ПРИМЕР

Вход	**Изход**
7 2 L R L R L P R R P	2 L 1 R 1 3

Решение

Броят на колите, пристигнали в кръстовището отляво, ще съраняваме в променливата `l`, а тези пристигнали отдясно – в променливата `r`, които в началото имат стойности нула. След това `n+c` пъти се извършва четене на поредният ред и увеличаване на брояча за левите (ако буквата е `L`), увеличаване на брояча за десните (ако буквата е `R`) или извършване на регулиране (ако буквата е `P`). Регулирането е определяне на по-малкото от `l` и `r`. Ако `r` е по-малко от `l`, то колите от дясно влизат в отбивката и същият брой коли от ляво преминават кръстовището, т.е. трябва да се изведе броят на колите отясно и буквата `L` и да се отчете, че колите в ляво са намалели с `r`. Аналогично, ако `l` е по-малко или равно на `r`.

```
#include <iostream>
#include <string>
using namespace std;
int main()
{
   int n, c;
   cin >> n >> c;
   n += c;
```