# Задача 1

Дан массив целых чисел nums и целое число target. Необходимо вернуть индексы двух чисел таких, чтобы их сумма равна target.

Имеется ровно одно решение.

Один и тот же элемент нельзя использовать дважды.

Результат можно вернуть в любом порядке.

**Пример 1:**

Вход: nums = [2,7,11,15], target = 9

Выход: [0,1]

Объяснение: Так как nums[0] + nums[1] == 9, возвращаем [0, 1].

**Пример 2:**

Вход: nums = [3,2,4], target = 6

Выход: [1,2]

**Пример 3:**

Вход: nums = [3,3], target = 6

Выход: [0,1]

**Ограничения:**

2 <= nums.length <= 104

-109 <= nums[i] <= 109

-109 <= target <= 109

Необходимо предоставить анализ сложности по времени и памяти в нотации O.

Реализуйте метод solve и сохраните решение в файле **test\_1.py**

def solve(nums: List[int], target: int) -> List[int]:

...

# Задача 2

Вы продукт-менеджер и в настоящее время возглавляете команду по разработке нового продукта. К сожалению, последняя версия вашего продукта не прошла проверку качества. Поскольку каждая версия разрабатывается на основе предыдущей версии, все версии после сломанной версии тоже сломаны.

Предположим, у вас есть n версий [1, 2, ..., n] и вы хотите найти первую сломанную версию, из-за которой все последующие будут сломаны.

Вам предоставляется bool API isBrokenVersion (версия), который возвращает, является ли версия сломанной. Реализуйте функцию для поиска первой сломанной версии. Вы должны свести к минимуму количество обращений к API.

|  |  |
| --- | --- |
| **Пример 1:**  Вход: n = 5, bad = 4  Выход: 4  Объяснение:  вызов isBrokenVersion(3) → false  вызов isBrokenVersion(5) → true  вызов isBrokenVersion(4) → true  4 — первая сломанная версия. | **Пример 2:**  Вход: n = 1, bad = 1  Выход: 1 |

**Ограничения:**

|  |
| --- |
| **1 <= bad <= n <= 230** |

Необходимо предоставить анализ сложности по времени и памяти в О-нотации.

Реализуйте метод **solve** и сохраните решение в файле **test\_2.py**

# isBrokenVersion API уже реализован, его в **test\_2.py** помещать не надо!

# def isBrokenVersion(version: int) -> bool:

# Можно ограничится вот таким импортом:

from is\_broken import isBrokenVersion

def solve(n: int) -> int:

...