

Задачи по теории алгоритмов и структурам данных

Содержание

1 Задача two-sums (Easy)	1
2 Задача Longest Common Prefix (Easy)	2
Список литературы	2

1. Задача two-sums (Easy)

<https://leetcode.com/problems/two-sum/description/> Дана коллекция целых чисел `nums` и целое число `target`. Требуется вернуть коллекцию индексов, сумма элементов которых совпадает со значением `target`.

То есть, если `nums = [2, 7, 11, 15]` и `target=9`, то возвращается `[0, 1]`, так как `nums[0] + nums[1] = 9`.

Мое решение с использованием средств Python

```
import itertools

def two_sum(self, nums: List[int], target: int) -> List[int]:
    for num_left, num_right in itertools.combinations(nums, 2):
        if num_left + num_right == target:
            if num_left != num_right:
                return [nums.index(num_left), nums.index(num_right)]
            else:
                left_index = nums.index(num_left)
                right_index = nums[left_index + 1:].index(num_left) + left_index + 1
                return [left_index, right_index]
```

Это решение с точки зрения потребления памяти оказалось лучше 76,89% предложенных решений (ничего удивительного, так как используются генераторы), но с точки зрения временных издержек лучше только 36.37% предложенных решений.

Было еще вот такое решение

```
def two_sum(self, nums: List[int], target: int) -> List[int]:
    elem_to_idx = {}
    for idx, elem in enumerate(nums):
        diff = target - elem
        if diff in elem_to_idx:
            return [elem_to_idx[diff], idx]

    elem_to_idx[elem] = idx
```

Оно лучше по времени (42,88%), но значительно хуже с точки зрения потребления памяти (5,76%). Кроме того, на мой взгляд код гораздо труднее для понимания.

2. Задача Longest Common Prefix (Easy)

<https://leetcode.com/problems/longest-common-prefix/description/> Требуется написать функцию, которая возвращает наибольший общий префикс для коллекции строк. Если префикс пустой, то следует вернуть пустую строку.

Решение

```
import typing as t

def longest_common_prefix(strs: t.List[str]) -> str:
    chars: t.List[str] = []
    shortest_word = min(strs, key=len)
    for idx, char in enumerate(shortest_word):
        _chars = set(word[idx] for word in strs)
        if len(_chars) == 1:
            chars.append(char)
        else:
            break

    return "".join(chars)
```

Список литературы

1. Бизли Д. Python. Подробный справочник. – СПб.: Символ-Плюс, 2010. – 864 с.
2. Бизли Д. Python. Книга рецептов. – М.: ДМК Пресс., 2019. – 648 с.
3. Маккинли У. Python и анализ данных, 2015. – 482 с.
4. Рамальо Л. Python – к вершинам мастерства: Лаконичное и эффективное программирование. – М.: МК Пресс, 2022. – 898 с.
5. Хейдт М., Груздев А. Изучаем pandas. – М.: ДМК Пресс, 2019. – 682 с.
6. Хостманн К. Scala для нетерпеливых. – М.: ДМК Пресс, 2013. – 408 с.