Цель работы

Написать функцию, которая возвращает кортеж из двух индексов элементов списка `lst`, таких что сумма элементов по этим индексам равна переменной `target`.
Элемент по индексу может быть выбран лишь единожды, значения в списке могут повторяться.
Если в списке встречается больше чем два индекса, подходящих под условие - вернуть наименьшие из всех. Элементы находятся в списке в произвольном порядке.

Комментарии по выполнению

Стартовый борд в repl.it
Для входного набора

```
lst = [1, 2, 3, 4, 5, 6, 7, 8, 9]
target = 8
```

программа должна вернуть

```
(0, 6)
```

Для начала решим эту задачу "в лоб" с помощью циклов

```
def two_sum(lst, target):
    pass
```

Усложнение 1

Решим задачу другим способом, чтобы вычислительная сложность решения была не O(n^2), а ниже (например, O(n)).

Решение оформить в виде отдельной функции:

```
def two_sum_hashed(lst, target):
    pass
```

Усложнение 2
Усложним задачу и вернем все наборы индексов, удовлетворяющих условию.

Для входного набора:

```
lst = [1, 2, 3, 4, 5, 6, 7, 8, 9]
target = 8
```

программа должна вернуть:

```
[(0,6), (1,5), (2,4)]
```

Опубликуйте решение с помощью сервиса repl.it и предоставьте в конце занятия ссылку на собственное решение в repl.it

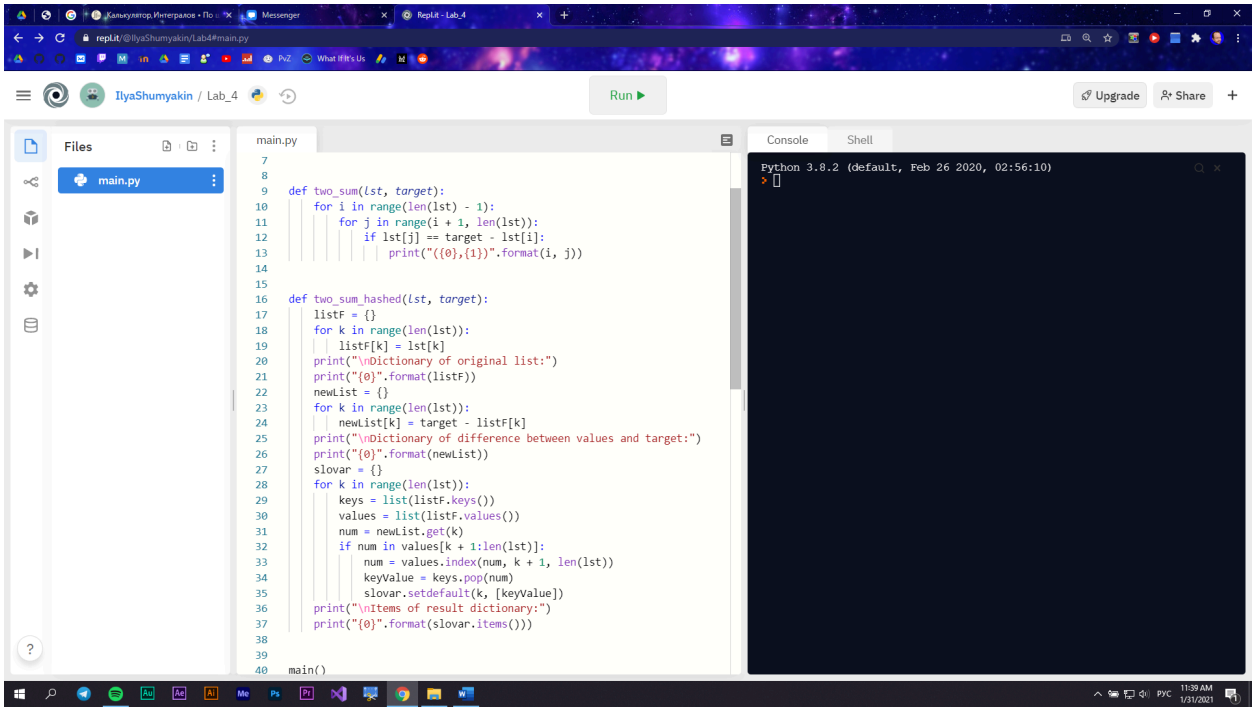
---

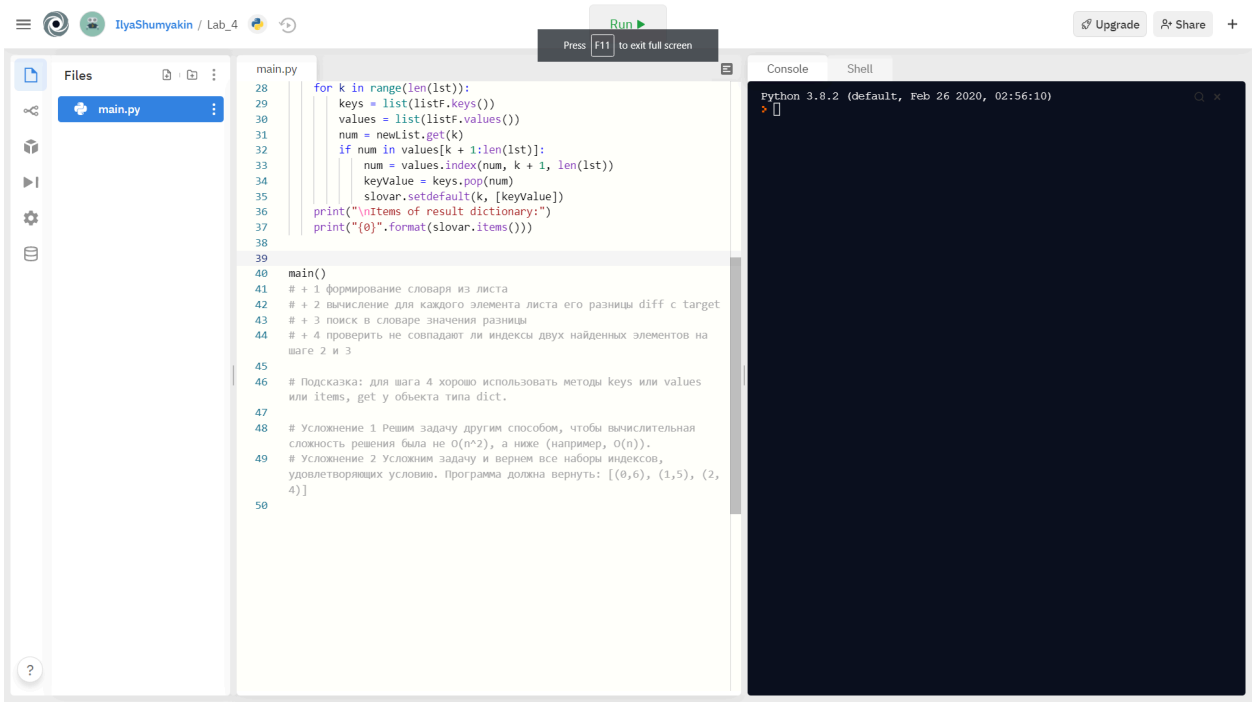
```
def main():
    lst = [1, 2, 3, 4, 5, 6, 7, 8, 9]
    target = 8
    print("First result is:")
    two_sum(lst, target)
    two_sum_hashed(lst, target)


def two_sum(lst, target):
    for i in range(len(lst) - 1):
        for j in range(i + 1, len(lst)):
            if lst[j] == target - lst[i]:
                print("({0},{1})".format(i, j))


def two_sum_hashed(lst, target):
    listF = {}
    for k in range(len(lst)):
        listF[k] = lst[k]
    print("\nDictionary of original list:")
    print("{0}".format(listF))
    newList = {}
    for k in range(len(lst)):
        newList[k] = target - listF[k]
    print("\nDictionary of difference between values and target:")
    print("{0}".format(newList))
    slovar = {}
    for k in range(len(lst)):
        keys = list(listF.keys())
        values = list(listF.values())
        num = newList.get(k)
        if num in values[k + 1:len(lst)]:
            num = values.index(num, k + 1, len(lst))
            keyValue = keys.pop(num)
            slovar.setdefault(k, [keyValue])
    print("\nItems of result dictionary:")
    print("{0}".format(slovar.items()))


main()
# + 1 формирование словаря из листа
# + 2 вычисление для каждого элемента листа его разницы diff с target
# + 3 поиск в словаре значения разницы
# + 4 проверить не совпадают ли индексы двух найденных элементов на шаге 2 и 3
```

Подсказка: для шага 4 хорошо использовать методы keys или values или items, get у объекта типа dict.

Усложнение 1 Решим задачу другим способом, чтобы вычислительная сложность решения была не O(n^2), а ниже (например, O(n)).
Усложнение 2 Усложним задачу и вернем все наборы индексов, удовлетворяющих условию. Программа должна вернуть: [(0,6), (1,5), (2,4)]

https://repl.it/@IlyaShumyakin/Lab4#main.py




IlyaShumyakin / Lab_4
Run
Press F11 to exit full screen
Upgrade
Share
Files
main.py
main.py
for k in range(len(lst)):
keys = list(listF.keys())
values = list(listF.values())
num = newList.get(k)
if num in values[k + 1:len(lst)]:
num = values.index(num, k + 1, len(lst))
keyValue = keys.pop(num)
slovar.setdefault(k, [keyValue])
print("\nItems of result dictionary:")
print("{0}".format(slovar.items()))
main()
# + 1 формирование словаря из листа
# + 2 вычисление для каждого элемента листа его разницы diff с target
# + 3 поиск в словаре значения разницы
# + 4 проверить не совпадают ли индексы двух найденных элементов на шаге 2 и 3
# Подсказка: для шага 4 хорошо использовать методы keys или values или items, get у объекта типа dict.
# Усложнение 1 Решим задачу другим способом, чтобы вычислительная сложность решения была не O(n^2), а ниже (например, O(n)).
# Усложнение 2 Усложним задачу и вернем все наборы индексов, удовлетворяющих условию. Программа должна вернуть: [(0,6), (1,5), (2, 4)]
Console
Shell
Python 3.8.2 (default, Feb 26 2020, 02:56:10)