ЗАДАЧА 1

Есть список `a = [1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89]`.

Выведите все элементы, которые меньше `5`.

Вариант решения
Самый простой вариант, который первым приходит на ум — использовать цикл `for`:

```
for elem in a:
    if elem < 5:
        print(elem)
```

Также можно воспользоваться функцией `filter`, которая фильтрует элементы согласно заданному условию:

```
print(list(filter(lambda elem: elem < 5, a)))
```

И, вероятно, наиболее предпочтительный вариант решения этой задачи — списковое включение:

```
print([elem for elem in a if elem < 5])
```

ЗАДАЧА 2

Даны списки:

`a = [1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89]`;

`b = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13]`.

Нужно вернуть список, который состоит из элементов, общих для этих двух списков.

Вариант решения
Можем воспользоваться функцией `filter`:

```
result = list(filter(lambda elem: elem in b, a))
```

Или списковым включением:

```
result = [elem for elem in a if elem in b]
```

А можно привести оба списка к множествам и найти их пересечение:

```
result = list(set(a) & set(b))
```

Однако в таком случае каждый элемент встретится в результирующем списке лишь один раз, т.к. множество поддерживает уникальность входящих в него элементов. Первые два решения (с фильтрацией) оставят все дубли на своих местах.

ЗАДАЧА 3

Сделайте так, чтобы число секунд отображалось в виде `дни:часы:минуты:секунды`.

Вариант решения

```
def convert(seconds):
    days = seconds // (24 * 3600)
    seconds %= 24 * 3600
```

```
    hours = seconds // 3600
    seconds %= 3600
    minutes = seconds // 60
    seconds %= 60
    print(f'{days}:{hours}:{minutes}:{seconds}')

convert(1234565)
```

ЗАДАЧА 4

Найдите три ключа с самыми высокими значениями в словаре `my_dict = {'a':500, 'b':5874, 'c': 560,'d':400, 'e':5874, 'f': 20}`.

Вариант решения

Можно воспользоваться функцией `sorted`:

```
result = sorted(my_dict, key=my_dict.get, reverse=True)[:3]
```

Аналогичный результат можно получить с помощью функции `nlargest` из модуля `heapq`:

```
from heapq import nlargest
result = nlargest(3, my_dict, key=my_dict.get)
```

ЗАДАЧА 5

Нужно вывести первые n строк треугольника Паскаля. В этом треугольнике на вершине и по бокам стоят единицы, а каждое число внутри равно сумме двух расположенных над ним чисел.

Вариант решения

```
def pascal_triangle(n):
   row = [1]
   y = [0]
   for x in range(max(n, 0)):
      print(row)
      row = [left + right for left, right in zip(row + y, y + row)]

pascal_triangle(6)
```

ЗАДАЧА 6

При заданном целом числе n посчитайте n + nn + nnn.

Вариант решения

```
def solve(n):
    n1 = n
    n2 = int(str(n) * 2)
    n3 = int(str(n) * 3)
    print(n1 + n2 + n3)

solve(5)
```

ЗАДАЧА 7

Напишите программу, которая принимает два списка и выводит все элементы первого, которых нет во втором.

Вариант решения

```
set_1 = set(['White', 'Black', 'Red'])
set_2 = set(['Red', 'Green'])

print(set_1 - set_2)
```

ЗАДАЧА 8

Сложите цифры целого числа.

Вариант решения

```
def sum_digits(num):
    digits = [int(d) for d in str(num)]
    return sum(digits)

print(sum_digits(5245))
```

ЗАДАЧА 9

Нужно проверить, все ли числа в последовательности уникальны.

Вариант решения

```
def all_unique(numbers):
    return len(numbers) == len(set(numbers))
```