2/6

1. Создайте функцию, которая повторяет каждый символ в строке n раз.

Пример:

```
repeat("mice", 5) ➞ "mmmmmiiiiiccccceeeee"

repeat("hello", 3) ➞ "hhheeellllllooo"

repeat("stop", 1) ➞ "stop"
```

2. Создайте функцию, которая принимает массив и возвращает разницу между самыми большими и самыми маленькими числами.

Пример:

```
differenceMaxMin([10, 4, 1, 4, -10, -50, 32, 21]) ➞ 82
// Smallest number is -50, biggest is 32.

differenceMaxMin([44, 32, 86, 19]) ➞ 67
// Smallest number is 19, biggest is 86.
```

3. Создайте функцию, которая принимает массив в качестве аргумента и возвращает true или false в зависимости от того, является ли среднее значение всех элементов массива целым числом или нет.

Пример:

```
isAvgWhole([1, 3]) ➞ true

isAvgWhole([1, 2, 3, 4]) ➞ false

isAvgWhole([1, 5, 6]) ➞ true

isAvgWhole([1, 1, 1]) ➞ true

isAvgWhole([9, 2, 2, 5]) ➞ false
```

4. Создайте метод, который берет массив целых чисел и возвращает массив, в котором каждое целое число является суммой самого себя + всех предыдущих чисел в массиве.

Пример:

```
cumulativeSum([1, 2, 3]) ➞ [1, 3, 6]

cumulativeSum([1, -2, 3]) ➞ [1, -1, 2]

cumulativeSum([3, 3, -2, 408, 3, 3]) ➞ [3, 6, 4, 412, 415, 418]
```

5. Создайте функцию, которая возвращает число десятичных знаков, которое имеет число (заданное в виде строки). Любые нули после десятичной точки отсчитываются в сторону количества десятичных знаков.

Пример:

```
getDecimalPlaces("43.20") ➞ 2

getDecimalPlaces("400") ➞ 0
```

```
getDecimalPlaces("3.1") ➞ 1
```

6. Создайте функцию, которая при заданном числе возвращает соответствующее число Фибоначчи.

Пример:

```
Fibonacci(3) ➞ 3

Fibonacci(7) ➞ 21

Fibonacci(12) ➞ 233
```

7. Почтовые индексы состоят из 5 последовательных цифр. Учитывая строку, напишите функцию, чтобы определить, является ли вход действительным почтовым индексом. Действительный почтовый индекс выглядит следующим образом:

– Должно содержать только цифры (не допускается использование нецифровых цифр).
– Не должно содержать никаких пробелов.
– Длина не должна превышать 5 цифр.

Пример:

```
isValid("59001") ➞ true

isValid("853a7") ➞ false

isValid("732 32") ➞ false

isValid("393939") ➞ false
```

8. Пара строк образует странную пару, если оба из следующих условий истинны:

– Первая буква 1-й строки = последняя буква 2-й строки.
– Последняя буква 1-й строки = первая буква 2-й строки.
– Создайте функцию, которая возвращает true, если пара строк представляет собой странную пару, и false в противном случае.

Пример:

```
isStrangePair("ratio", "orator") ➞ true
// "ratio" ends with "o" and "orator" starts with "o".
// "ratio" starts with "r" and "orator" ends with "r".

isStrangePair("sparkling", "groups") ➞ true

isStrangePair("bush", "hubris") ➞ false

isStrangePair("", "") ➞ true
```

9. Создайте две функции: isPrefix(word, prefix-) и isSuffix (word, -suffix).

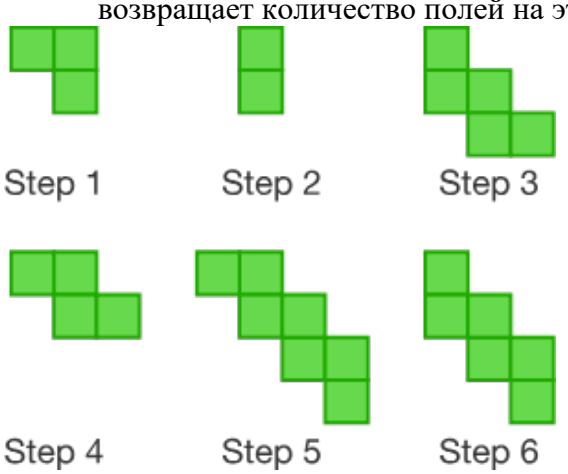
– isPrefix должен возвращать true, если он начинается с префиксного аргумента.
– isSuffix должен возвращать true, если он заканчивается аргументом суффикса.
– В противном случае верните false.

Пример:

```
isPrefix("automation", "auto-") → true

isSuffix("arachnophobia", "-phobia") → true

isPrefix("retrospect", "sub-") → false

isSuffix("vocation", "-logy") → false
```

Примечание:
Аргументы префикса и суффикса имеют тире - в них.

10. Создайте функцию, которая принимает число (шаг) в качестве аргумента и возвращает количество полей на этом шаге последовательности.



Шаг 0: начните с 0
Шаг 1: Добавьте 3
Шаг 2: Вычтите 1
Повторите Шаги 1 И 2 ...

Пример:

```
boxSeq(0) → 0

boxSeq(1) → 3

boxSeq(2) → 2
```