6/6

1. Число Белла - это количество способов, которыми массив из n элементов может быть разбит на непустые подмножества. Создайте функцию, которая принимает число n и возвращает соответствующее число Белла.

Пример:

```
bell(1) → 1
// sampleArr = [1]
// possiblePartitions = [[[1]]]

bell(2) → 2
// sampleArr = [1, 2]
// possiblePartitions = [[[1, 2]], [[1], [2]]]

bell(3) → 5
// sampleArr = [1, 2, 3]
// possiblePartitions = [[[1, 2, 3]], [[1, 2], [3]], [[1], [2, 3]],
[[1, 3], [2]], [[1], [2], [3]]]
```

2. В «поросячей латыни» (свинский латинский) есть два очень простых правила:
– Если слово начинается с согласного, переместите первую букву (буквы) слова до гласного до конца слова и добавьте «ay» в конец.

have → avehay
cram → amcray
take → aketay
cat → atcay
shrimp → impshray
trebuchet → ebuchettray

– Если слово начинается с гласной, добавьте "yay" в конце слова.

ate → ateyay
apple → appleyay
oaken → oakenyay
eagle → eagleyay

Напишите две функции, чтобы сделать переводчик с английского на свинский латинский. Первая функция translateWord (word) получает слово на английском и возвращает это слово, переведенное на латинский язык. Вторая функция translateSentence (предложение) берет английское предложение и возвращает это предложение, переведенное на латинский язык.

Пример:

```
translateWord("flag") → "agflay"

translateWord("Apple") → "Appleyay"

translateWord("button") → "uttonbay"

translateWord("") → ""

translateSentence("I like to eat honey waffles.") → "Iyay ikelay otay
eatyay oneyhay afflesway."
```

```
translateSentence("Do you think it is going to rain today?") → "Oday
youyay inkthay ityay isyay oinggay otay ainray odaytay?"
```

Примечание:
– Регулярные выражения помогут вам не исказить пунктуацию в предложении.
– Если исходное слово или предложение начинается с заглавной буквы, перевод должен сохранить свой регистр

3. Учитывая параметры RGB (A) CSS, определите, является ли формат принимаемых значений допустимым или нет. Создайте функцию, которая принимает строку (например, " rgb(0, 0, 0)") и возвращает true, если ее формат правильный, в противном случае возвращает false.

Пример:

```
validColor("rgb(0,0,0)") → true

validColor("rgb(0,,0)") → false

validColor("rgb(255,256,255)") → false

validColor("rgba(0,0,0,0.123456789)") → true
```

4. Создайте функцию, которая принимает URL (строку), удаляет дублирующиеся параметры запроса и параметры, указанные во втором аргументе (который будет необязательным массивом).

Пример:

```
stripUrlParams("https://edabit.com?a=1&b=2&a=2") →
"https://edabit.com?a=2&b=2"

stripUrlParams("https://edabit.com?a=1&b=2&a=2", ["b"]) →
"https://edabit.com?a=2"

stripUrlParams("https://edabit.com", ["b"]) → "https://edabit.com"
```

Примечание:
– Второй аргумент paramsToStrip является необязательным.
– paramsToStrip может содержать несколько параметров.
– Если есть повторяющиеся параметры запроса с разными значениями, используйте значение последнего встречающегося параметра (см. Примеры № 1 и № 2 выше).

5. Напишите функцию, которая извлекает три самых длинных слова из заголовка газеты и преобразует их в хэштеги. Если несколько слов одинаковой длины, найдите слово, которое встречается первым.

Пример:

```
getHashTags("How the Avocado Became the Fruit of the Global Trade")
→ ["#avocado", "#became", "#global"]

getHashTags("Why You Will Probably Pay More for Your Christmas Tree
This Year")
→ ["#christmas", "#probably", "#will"]
```

```
getHashTags("Hey Parents, Surprise, Fruit Juice Is Not Fruit")
→ ["#surprise", "#parents", "#fruit"]

getHashTags("Visualizing Science")
→ ["#visualizing", "#science"]
```

Примечание:
– Если заголовок содержит менее 3 слов, просто расположите слова в заголовке по длине в порядке убывания (см. Пример №4).
– Пунктуация не считается с длиной слова.

6. Последовательность Улама начинается с:

```
ulam = [1, 2]
```

Следующее число в последовательности - это наименьшее положительное число, равное сумме двух разных чисел (которые уже есть в последовательности) ровно одним способом. Тривиально, это 3, так как в стартовой последовательности есть только 2 числа.

```
ulam = [1, 2, 3]
```

Следующее число 4, которое является суммой 3 + 1. 4 также равно 2 + 2, но это уравнение не учитывается, так как 2 добавления должны быть различны.

```
ulam = [1, 2, 3, 4]
```

Следующее число не может быть 5, так как 5 = 1 + 4, но также и 5 = 2 + 3. Должен быть только один способ сделать число Улама из 2 различных добавлений, найденных в последовательности. Следующее число 6 (2 + 4). Есть 2 способа сделать 7 (1 + 6 или 3 + 4), поэтому следующий - 8 (2 + 6). И так далее.

```
ulam = [1, 2, 3, 4, 6, 8, 11, 13, 16, 18, 26, 28, 36, 38, 47, 48, 53,
…]
```

Создайте функцию, которая принимает число n и возвращает n-е число в последовательности Улама.

Пример:

```
ulam(4) → 4

ulam(9) → 16

ulam(206) → 1856
```

7. Напишите функцию, которая возвращает самую длинную неповторяющуюся подстроку для строкового ввода.

Пример:

```
longestNonrepeatingSubstring("abcabcbb") → "abc"

longestNonrepeatingSubstring("aaaaaa") → "a"

longestNonrepeatingSubstring("abcde") → "abcde"

longestNonrepeatingSubstring("abcda") → "abcd"
```

Примечание:
– Если несколько подстрок связаны по длине, верните ту, которая возникает первой.
– Бонус: можете ли вы решить эту проблему в линейном времени?

8. Создайте функцию, которая принимает арабское число и преобразует его в римское число.

Пример:

```
convertToRoman(2) → "II"

convertToRoman(12) → "XII"

convertToRoman(16) → "XVI"
```

Примечание:
– Все римские цифры должны быть возвращены в верхнем регистре.
– Самое большое число, которое может быть представлено в этой нотации, - 3,999.

9. Создайте функцию, которая принимает строку и возвращает true или false в зависимости от того, является ли формула правильной или нет.

Пример:

```
formula("6 * 4 = 24") → true

formula("18 / 17 = 2") → false

formula("16 * 10 = 160 = 14 + 120") → false
```

10. Число может не быть палиндромом, но его потомком может быть. Прямой потомок числа создается путем суммирования каждой пары соседних цифр, чтобы создать цифры следующего числа.

Например, 123312 – это не палиндром, а его следующий потомок 363, где: 3 = 1 + 2; 6 = 3 + 3; 3 = 1 + 2.

Создайте функцию, которая возвращает значение true, если само число является палиндромом или любой из его потомков вплоть до 2 цифр (однозначное число - тривиально палиндром).

Пример:

```
palindromedescendant(11211230) → true
// 11211230 → 2333 → 56 → 11

palindromeDescendant(13001120) → true
// 13001120 → 4022 → 44

palindromeDescendant(23336014) → true
// 23336014 → 5665

palindromeDescendant(11) → true
// Number itself is a palindrome.
```

Примечание:
– Числа всегда будут иметь четное число цифр.