5/6

1. Пришло время отправлять и получать секретные сообщения.

Создайте две функции, которые принимают строку и массив и возвращают закодированное или декодированное сообщение.
Первая буква строки или первый элемент массива представляет собой символьный код этой буквы. Следующие элементы-это различия между символами: например, A +3 --> C или z -1 --> y.

Пример:

```
encrypt("Hello") ➞ [72, 29, 7, 0, 3]
// H = 72, the difference between the H and e is 29 (upper- and
lowercase).
// The difference between the two l's is obviously 0.

decrypt([ 72, 33, -73, 84, -12, -3, 13, -13, -68 ]) ➞ "Hi there!"

encrypt("Sunshine") ➞ [83, 34, -7, 5, -11, 1, 5, -9]
```

2. Создайте функцию, которая принимает имя шахматной фигуры, ее положение и целевую позицию. Функция должна возвращать true, если фигура может двигаться к цели, и false, если она не может этого сделать.

Возможные входные данные - "пешка", "конь", "слон", "Ладья", "Ферзь"и " король".

Пример:

```
canMove("Rook", "A8", "H8") ➞ true

canMove("Bishop", "A7", "G1") ➞ true

canMove("Queen", "C4", "D6") ➞ false
```

3. Входная строка может быть завершена, если можно добавить дополнительные буквы, и никакие буквы не должны быть удалены, чтобы соответствовать слову. Кроме того, порядок букв во входной строке должен быть таким же, как и порядок букв в последнем слове.

Создайте функцию, которая, учитывая входную строку, определяет, может ли слово быть завершено.

Пример:

```
canComplete("butl", "beautiful") ➞ true
// We can add "ea" between "b" and "u", and "ifu" between "t" and "l".

canComplete("butlz", "beautiful") ➞ false
// "z" does not exist in the word beautiful.

canComplete("tulb", "beautiful") ➞ false
// Although "t", "u", "l" and "b" all exist in "beautiful", they are
incorrectly ordered.

canComplete("bbutl", "beautiful") ➞ false
// Too many "b"s, beautiful has only 1.
```

4. Создайте функцию, которая принимает числа в качестве аргументов, складывает их вместе и возвращает произведение цифр до тех пор, пока ответ не станет длиной всего в 1 цифру.

Пример:

```
sumDigProd(16, 28) → 6

sumDigProd(0) → 0

sumDigProd(1, 2, 3, 4, 5, 6) → 2
```

5. Напишите функцию, которая выбирает все слова, имеющие все те же гласные (в любом порядке и / или количестве), что и первое слово, включая первое слово.

Пример:

```
sameVowelGroup(["toe", "ocelot", "maniac"]) → ["toe", "ocelot"]

sameVowelGroup(["many", "carriage", "emit", "apricot", "animal"]) →
["many"]

sameVowelGroup(["hoops", "chuff", "bot", "bottom"]) → ["hoops", "bot",
"bottom"]
```

6. Создайте функцию, которая принимает число в качестве аргумента и возвращает true, если это число является действительным номером кредитной карты, а в противном случае-false.

Номера кредитных карт должны быть длиной от 14 до 19 цифр и проходить тест Луна, описанный ниже:
– Удалите последнюю цифру (это"контрольная цифра").
– Переверните число.
– Удвойте значение каждой цифры в нечетных позициях. Если удвоенное значение имеет более 1 цифры, сложите цифры вместе (например, 8 x 2 = 16 → 1 + 6 = 7).
– Добавьте все цифры.
– Вычтите последнюю цифру суммы (из шага 4) из 10. Результат должен быть равен контрольной цифре из Шага 1.

Пример:

```
validateCard(1234567890123456) → false

// Step 1: check digit = 6, num = 123456789012345
// Step 2: num reversed = 543210987654321
// Step 3: digit array after selective doubling: [1, 4, 6, 2, 2, 0, 9,
8, 5, 6, 1, 4, 6, 2, 2]
// Step 4: sum = 58
// Step 5: 10 - 8 = 2 (not equal to 6) → false

validateCard(1234567890123452) → true

// Same as above, but check digit checks out.
```

7. Напишите функцию, которая принимает положительное целое число от 0 до 999 включительно и возвращает строковое представление этого целого числа, написанное на английском языке.

Пример:

```
numToEng(0) → "zero"
```

```
numToEng(18) → "eighteen"

numToEng(126) → "one hundred twenty six"

numToEng(909) → "nine hundred nine"
```

Тоже самое нужно сделать и для русского языка.

8. Хеш-алгоритмы легко сделать одним способом, но по существу невозможно сделать наоборот. Например, если вы хешируете что-то простое, например, password123, это даст вам длинный код, уникальный для этого слова или фразы. В идеале, нет способа сделать это в обратном порядке. Вы не можете взять хеш-код и вернуться к слову или фразе, с которых вы начали.

Создайте функцию, которая возвращает безопасный хеш SHA-256 для данной строки. Хеш должен быть отформатирован в виде шестнадцатеричной цифры.

Пример:

```
getSha256Hash("password123") →
"ef92b778bafe771e89245b89ecbc08a44a4e166c06659911881f383d4473e94f"

getSha256Hash("Fluffy@home") →
"dcc1ac3a7148a2d9f47b7dbe3d733040c335b2a3d8adc7984e0c483c5b2c1665"

getSha256Hash("Hey dude!") →
"14f997f08b8ad032dcb274198684f995d34043f9da00acd904dc72836359ae0f"
```

Примечание:
Бонус, если вы можете сделать это без импорта каких-либо библиотек ;)

9. Напишите функцию, которая принимает строку и возвращает строку с правильным регистром для заголовков символов в серии "Игра престолов".

Слова and, the, of и in должны быть строчными. Все остальные слова должны иметь первый символ в верхнем регистре, а остальные-в Нижнем.

Пример:

```
correctTitle("jOn SnoW, kINg IN thE noRth.")
→ "Jon Snow, King in the North."

correctTitle("sansa stark, lady of winterfell.")
→ "Sansa Stark, Lady of Winterfell."

correctTitle("TYRION LANNISTER, HAND OF THE QUEEN.")
→ "Tyrion Lannister, Hand of the Queen."
```

Примечания:
– Знаки препинания и пробелы должны оставаться в своих первоначальных положениях.
– Дефисные слова считаются отдельными словами.
– Будьте осторожны со словами, которые содержат and, the, of или in.

10. Как указано в онлайн-энциклопедии целочисленных последовательностей: Гексагональная решетка - это привычная двумерная решетка, в которой каждая точка имеет 6 соседей.

Центрированное шестиугольное число - это центрированное фигурное число, представляющее шестиугольник с точкой в центре и всеми другими точками, окружающими центральную точку в шестиугольной решетке.

```
Illustration of initial terms:
.
.                                  o o o o
.                    o o o       o o o o o
.          o o      o o o o      o o o o o o
.    o    o o o    o o o o o    o o o o o o o
.          o o      o o o o      o o o o o o
.                    o o o       o o o o o
.                                  o o o o
.
.    1       7          19             37
.
```

Напишите функцию, которая принимает целое число n и возвращает "недопустимое", если n не является центрированным шестиугольным числом или его иллюстрацией в виде многострочной прямоугольной строки в противном случае.

Пример:

```
hexLattice(1) ➞ " o "
// o

hexLattice(7) ➞ "  o o  \n o o o \n  o o  "
//  o o
// o o o
//  o o

hexLattice(19) ➞ "   o o o   \n  o o o o  \n o o o o o \n  o o o o  \n
o o o   "
//   o o o
//  o o o o
// o o o o o
//  o o o o
//   o o o

hexLattice(21) ➞ "Invalid"
```