🟢 Простой уровень

> Цель – отработать синтаксис функций, аргументы, `return`, циклы `for` и `while`.

1. **Сумма чисел от 1 до N**
   Напиши функцию `sumTo(n)`, которая возвращает сумму всех чисел от 1 до `n`.
   ➡️ Пример: `sumTo(5)` → `15`

2. **Степень числа**
   Напиши функцию `pow(base, exponent)`, которая возводит `base` в степень `exponent` с помощью цикла.
   ➡️ Пример: `pow(2, 3)` → `8`

3. **Подсчёт чётных чисел**
   Функция `countEven(n)` должна вернуть количество чётных чисел от 1 до `n`.
   ➡️ Пример: `countEven(10)` → `5`

4. **Факториал числа**
   Реализуй `factorial(n)` с помощью цикла.
   ➡️ Пример: `factorial(5)` → `120`

5. **Повтор строки**
   Напиши функцию `repeatText(text, count)`, которая возвращает строку, повторённую `count` раз подряд (без `String.repeat`).
   ➡️ Пример: `repeatText('Hi', 3)` → `'HiHiHi'`

🟡 Средний уровень

> Здесь проверяем комбинирование условий, вложенные циклы, параметры функций.

6. **Числа в диапазоне**
   Функция `printRange(start, end)` выводит все числа от `start` до `end` (включительно).
   ➡️ Пример: `printRange(3, 7)` → `3 4 5 6 7`

7. **Подсчёт цифр**
   Функция `countDigits(num)` возвращает, сколько цифр в числе `num`.
   ➡️ Пример: `countDigits(12345)` → `5`

8. **Сумма цифр числа**

   Функция `sumDigits(num)` находит сумму всех цифр числа (используя `%` и `Math.floor`).

   ➡️ Пример: `sumDigits(123)` → `6`

9. **Таблица умножения**

   Функция `printTable(n)` выводит таблицу умножения для числа `n` (от 1 до 10).

   ➡️ Пример:

```
3 × 1 = 3
3 × 2 = 6
...
3 × 10 = 30
```

10. **Обратный отсчёт**

    Функция `countdown(from)` выводит все числа от `from` до 1.

    ➡️ Пример: `countdown(5)` → `5 4 3 2 1`

🔴 Сложный уровень

> Используем комбинации циклов, условий и функций. Всё ещё без массивов/объектов.

11. **Числа Фибоначчи**

    Функция `fibonacci(n)` возвращает `n`-е число Фибоначчи.

    ➡️ Пример: `fibonacci(6)` → `8`

12. **Реверс числа**

    Функция `reverseNumber(num)` возвращает число в обратном порядке цифр.

    ➡️ Пример: `reverseNumber(1234)` → `4321`

13. **Проверка на палиндром (число)**

    Функция `isPalindrome(num)` возвращает `true`, если число читается одинаково в обе стороны.

    ➡️ Пример: `isPalindrome(121)` → `true`

14. **Нахождение делителей**

    Функция `divisors(n)` выводит все числа, на которые делится `n` без остатка.

    ➡️ Пример: `divisors(12)` → `1 2 3 4 6 12`

15. **Простое число**

    Функция `isPrime(n)` возвращает `true`, если число простое.

➡️ Пример: `isPrime(7)` → `true`, `isPrime(9)` → `false`

🌟 Задачи со звёздочкой (для продвинутых)

16. **Сумма всех нечётных чисел от 1 до 99**
    ➡️ Используй цикл `for` и условие `if`.

17. **Подсчёт количества нулей в числе**
    ➡️ Пример: `countZeros(10020)` → `3`

18. **Найти первую цифру числа**
    ➡️ Пример: `firstDigit(9876)` → `9`

19. **Сколько раз число делится на 2, пока не станет нечётным**
    ➡️ Пример: `countDivisions(40)` → `3` (40 → 20 → 10 → 5)

⭐ 20. Замыкание: счётчик вызовов

> Напиши функцию, которая «запоминает» количество своих вызовов.

```
function createCounter() {
  let count = 0;
  return function() {
    count++;
    console.log(`Вызов №${count}`);
  };
}

const counter = createCounter();
counter(); // Вызов №1
counter(); // Вызов №2
counter(); // Вызов №3
```

💡 Усложнение:

- Сделай так, чтобы `counter('reset')` сбрасывал счётчик.
- Или чтобы `counter('get')` возвращал текущее значение.