1. Опишем преобразование SX(m) двоичной записи числа m. Каждая единица заменяется на SX, а ноль на S, после чего вычёркивается первую слева S. Так $\text{bin}(5) = 101 \rightarrow$ $\rightarrow XSSX$. Алгоритм вычисления функции F от положительных целых чисел (x, m) задан псевдокодом:

```
Function F(x, m) :
    a = SX(m);
    y = 1;
    for i = 1 to a.size do
        if a[i] == X then
            y = y × x
        else
            y = y × y
        end
    end
    return y
end
```

1. Вычислите F$(3, 5)$.

2. Какую математическую функцию реализует данный алгоритм?

3. Докажите корректность данного алгоритма (нужно доказать, что алгоритм действительно реализует отгаданную функцию).

4. Оцените время работы алгоритма, считая, что арифметические операции стоят $O(1)$.

2. На прямой задано n отрезков, причем известно, что они образуют систему строго вложенных отрезков (их можно упорядочить так, чтобы каждый строго содержался в следующем). Отрезки заданы координатами концов $[l_i, r_i]$ (и могут быть даны в неупорядоченном виде). Предложите асимптотически эффективный алгоритм (с точки зрения количества арифметических операций), который находит все точки прямой, которые покрыты ровно $2n/3$ отрезками.

3. Рассмотрим детерминированный алгоритм поиска порядковой статистики за линейное время из параграфа 9.3 Кормена. Какая асимптотика будет у алгоритма, если делить элементы массива на группы по семь, а не по пять?

4. Дан массив длины n, состоящий только из нулей и единиц. Предложите линейный алгоритм сортировки данного массива.

5. Предложите полиномиальный от длины входа алгоритм решения сравнения

$$a \cdot x + b \equiv 0 \pmod{M}$$

(На вход дают целые числа a, b, M в двоичной системе исчисления).

6. 1. Оцените глубину стека (рекурсивных вызовов) при работе быстрой сортировки в худшем случае.

2. Измените алгоритм быстрой сортировки так, чтобы глубина стека в худшем случае была $\Theta(\log n)$.

7*. Запишите рекурентное соотношение для сложности и докажите по индукции, что трудоемкость алгоритма Randomized-Qsort (см. следующую страницу) равна $O(n \log n)$.

```
функция RandomQsort(A[1..n])
if n > 1
    x = A[RandomInteger(1, n)]
    (x - случайный элемент из A)
    Partition(A, x) → B[1..k − 1] x C[1..n − k]
    (разбили A на массивы меньше x, и больше x)
    RandomQsort(B[1..k − 1])
    RandomQsort(C[1..n − k])
    return B[1..k − 1] x C[1..n − k]
endif
```

8*. Запишите рекурентное соотношение для сложности и докажите по индукции, что трудоемкость алгоритма Randomized-Selection поиска k-й порядковой статистики равна $O(n)$.

```
функция RandomSelection(A[1..n], k)
    if n == 1
        return A[1]
    else
        x = A[RandomInteger(1, n)]
        (x - случайный элемент из A)
        Partition(A, x) → B[1..l − 1] x C[1..n − l]
        (разбили A на массивы меньше x, и больше x)
        if l == k
            return x
        if k < l
            return RandomSelection(B[1..l − 1], k)
        if k > l
            return RandomSelection(C[1..n − l], k − l)
```