1. **Функция Fibonacci**.

Алгоритмическая сложность: O(2^n), где n — входное число.

Оптимизация: Использование мемоизации для сохранения уже вычисленных значений.

Сложность после оптимизации: O(n).

2. **Функция Ackermann**.

Алгоритмическая сложность: O(A(m, n)), где A — сама функция Аккермана.

Оптимизация: Хвостовая рекурсия для уменьшения глубины вызовов и помогает избежать переполнение стеков.

Сложность после оптимизации: O(A(m, n)).

3. **Функция FindNumber**.

Алгоритмическая сложность: O(m \* n), где m — количество строк, n — количество столбцов в матрице.

Оптимизация: Использование двоичного поиска, если массив отсортирован.

Сложность после оптимизации: O(n \* log m).