**Отчет по лабораторной работе №5**

**Оценка сложности рекурсивного подхода.**

Ограничения рекурсивных функций можно выявить на основе тестирования программы. Выяснено, что для рекурсивной функции recursive\_1 n не должен превышать значение 2048, а для рекурсивной функции recursive\_2 n не превышает 1023. Значения различаются, поскольку две рекурсии вычисляют значения двух разных функций.

Рассмотрим рекурсивную функцию recursive\_1. Если n=8, общее количество вызовов функции составляет 4; при n=10: 5 вызовов; при n=12: 6 вызовов и т.д. Временная сложность алгоритма составляет O(n/2). Аналогично, пространственная сложность составляет O(n/2), потому что требуется примерно n блоков в стеке вызовов. Такие же рассуждения применимы к рекурсивной функции recursive\_2. Для нее пространственная сложность O(n-1) и временная сложность O(n-1). Отсюда становится ясным различие в ограничениях для обеих функций. В общей сложности, можно оценить сложность рекурсии как O(n), так как константы принято опускать.

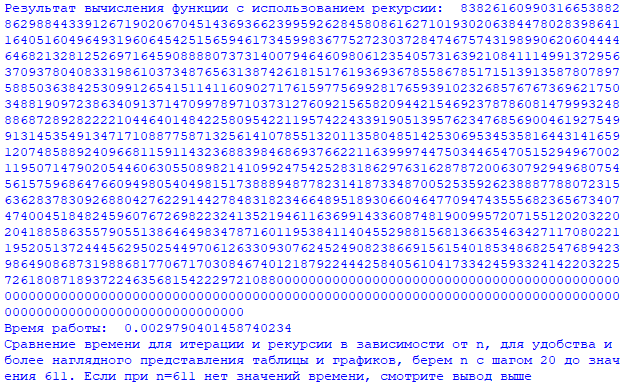
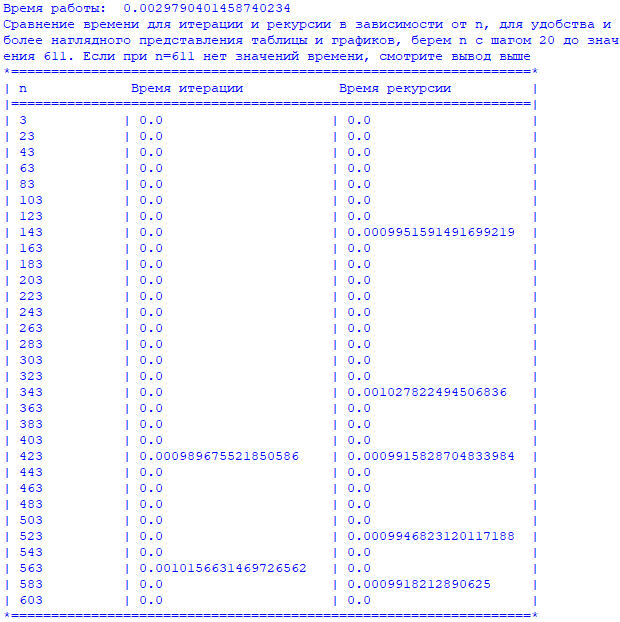
**Оценка сложности итерационного подхода.**

В функции iter\_1 код выполняется (n-4)/2+1 раз. Временная сложность составляет O((n-4)/2+1), но так как принято не учитывать константы, то принимаем временную сложность здесь также равной O(n).

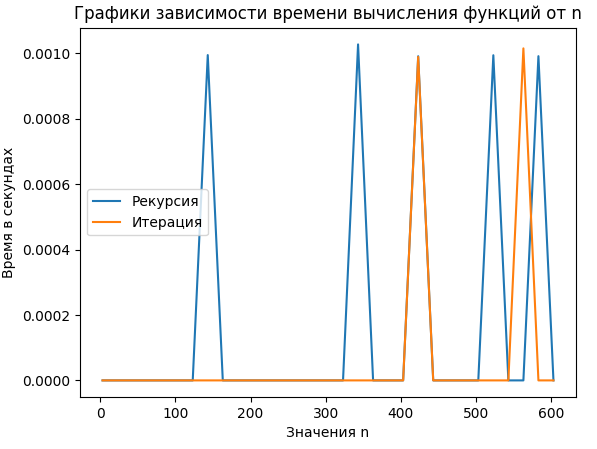
Функция iter\_2 имеет временную сложность O(n): количество итераций зависит от n, то есть от зависит от входных данных.

Пространственная сложность составляет O(1) для итерационного подхода, поскольку используется одна переменная для вычислений и дополнительная память не требуется.

Итерация работает при очень больших значениях n, значительно превышающих те, с которыми может функционировать в условиях данной задачи рекурсия. Это связано с тем, что вызовы рекурсивных функций хранятся в стеке, размер которого строго ограничен.

При вычислении функции с помощью итерационного подхода n может быть очень большим числом, однако c увеличением n больше 1000000 программа работает очень медленно (данный небольшой участок кода выполняется около 275 секунд, при этом не учитывается время работы остального кода), и очевидно, что по этой причине вычислять значение функции от подобных n не является рациональным. Ниже приведен пример работы кода. Для наглядности лучше брать значения n по возможности больше.

Ниже приведены два графика, иллюстрирующие зависимость времени работы итерационного и рекурсивного подхода от величины n.

Графики получаются скачкообразными, поскольку одновременно с выполнением кода происходит выполнение и других операций в компьютере и время работы функций может сильно различаться в разные моменты времени. Однако последний график для итерационного подхода в целом показывает, что с увеличением значения n растет время выполнения соответствующих вычислений.