**Каракозова Регина, 2 ФИТ 7/2**

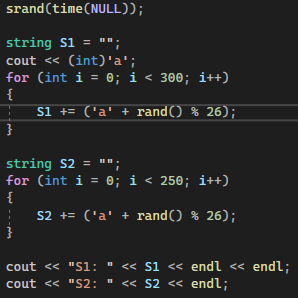
**Лабораторная работа №4. Динамическое программирование**

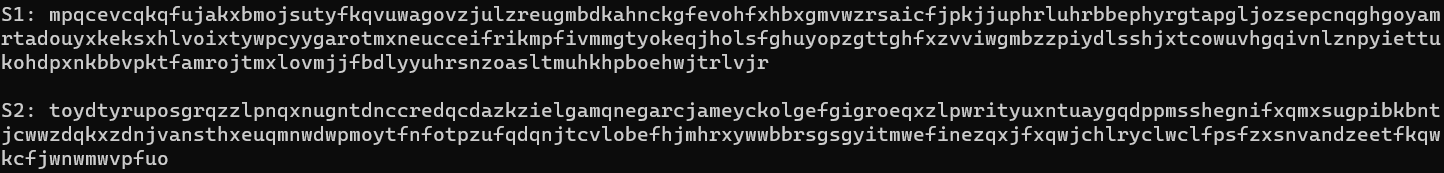
**Цель работы:** освоить общие принципы решения задач методом динамического программирования, сравнить полученные решения задач с рекурсивным методом.

**Задание 1.**

*На языке С++ сгенерировать случайным образом строку букв латинского алфавита S1 длиной 300 символов и S2 длиной 200.*

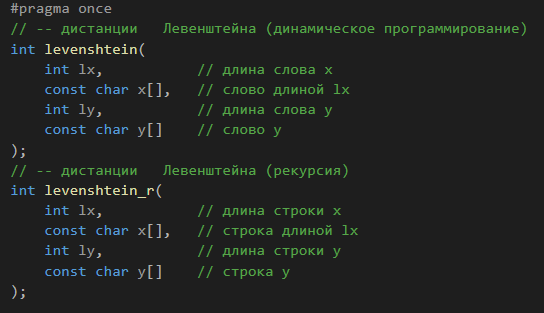
**Выполнение:** код на C++, генерирующий случайным образом букву латинского алфавита, путём добавления к первой букве ‘a’ числового значения, полученного путём остатка от деления числа, сгенерированного функцией rand() на 26 (в латинском алфавите 26 букв), и автоматического неявного преобразования его в уже другую букву латинского алфавита в соответствии с таблицей кодировки ASCII.



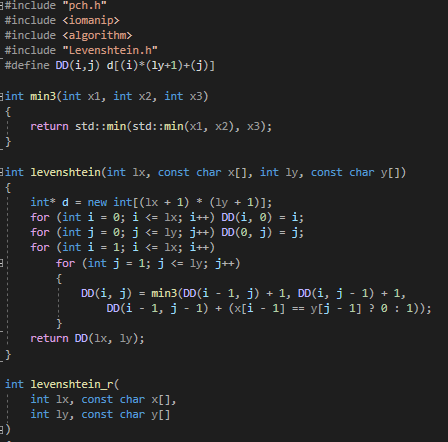


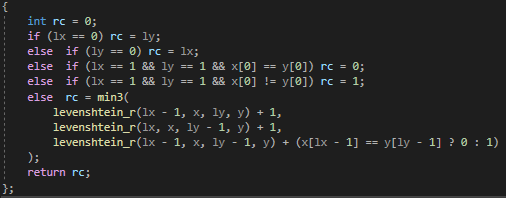
**Задание 2.**

*Вычислить двумя способами (рекурсивно и с помощью динамического программирования)*  *– дистанцию Левенштейна для*  *, где*  *- длина строки* *,*  *- строка состоящая из первых*  *символов строки*  *. (копии экрана и код вставить в отчет).*

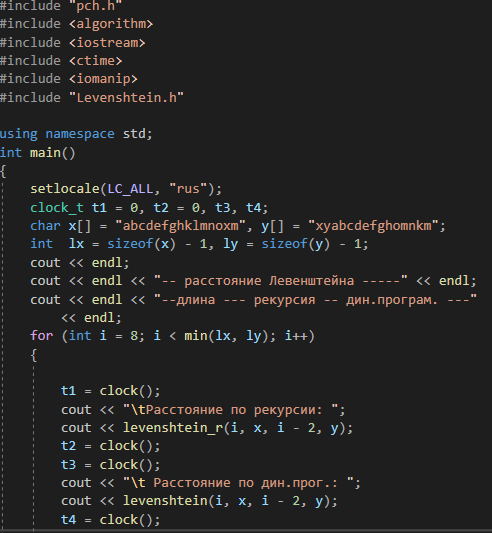


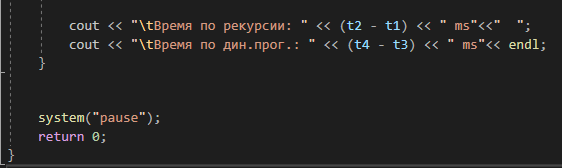
Исходный код файла Levenshtein.h



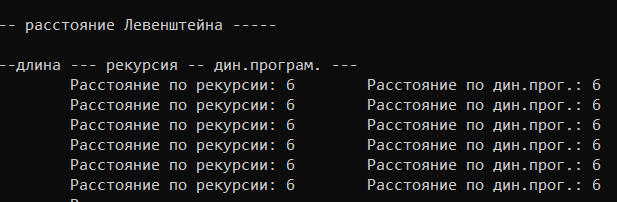


Исходный код файла Levenshtein.cpp





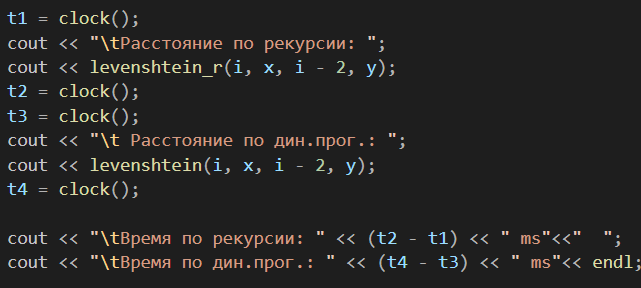
Исходный код файла Main.cpp



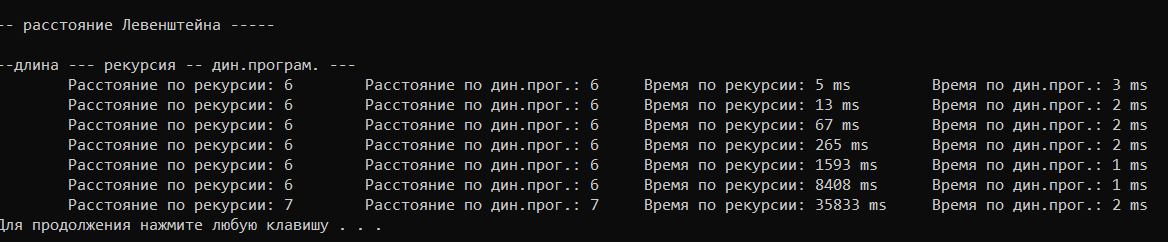
Результат

**Задание 3.**

*Выполнить сравнительный анализ времени затраченного на вычисление дистанции Левенштейна для двух методов решения. Построить графики зависимости времени вычисления от*  *. (копии экрана и график вставить в отчет).*



Часть исходного кода файла Main.cpp



Результат времени по данным методам

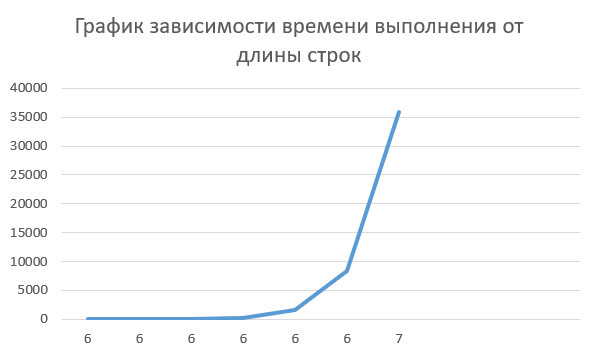


График для рекурсивного метода



График для метода динам. программ.

**Задание 4.**

*Реализовать вручную пример вычисления дистанции Левенштейна при помощи рекурсивного алгоритма (в соответствии с вариантом) (каждый шаг алгоритма по примеру из лекции вставить в отчет).*

Слово 1: Вар; Слово 2: Баран;



1. L(“вар”, “баран”) = min  
  
2. L("ва", "баран") = min  
3. L(“вар”, “бара ”) = min  
  
4. L(“ва”, “бара ”) = min

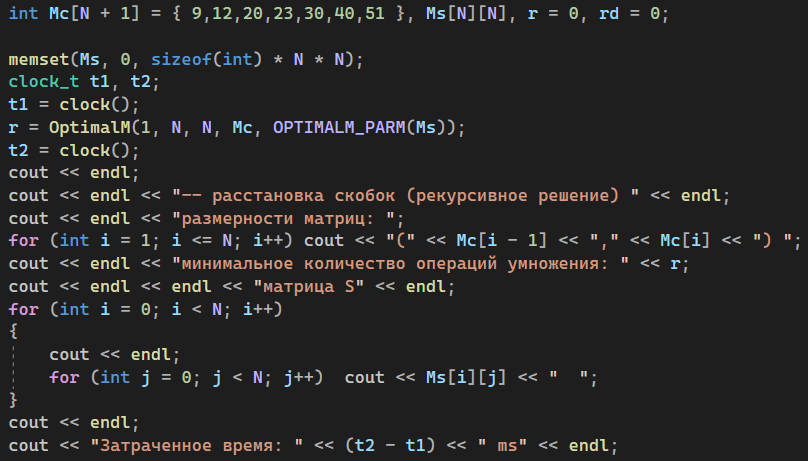
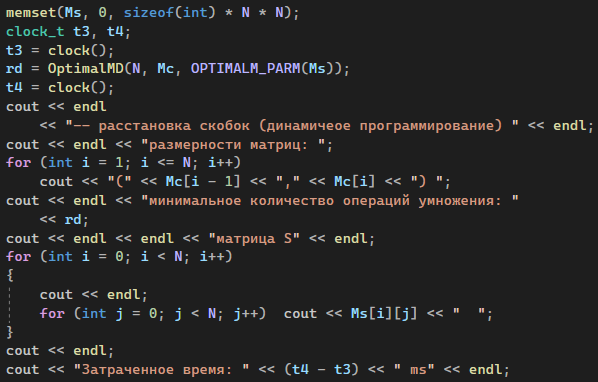
5. L(“в”, “баран ”) = min  
  
L(“”, “баран”) = 5, L(“”, “бара”) = 4  
  
6. L(“в”, “бара”) = min  
  
L(“”, “бара ”) = 4, L(“”, “бар”) = 3  
  
7. L(“вар”, “бар”) = min  
  
8. L(“ва”, “бар”) = min  
  
9. L(“вар”, “ба”) = min  
  
10. L(“вар”, “б”) = min  
  
L(“вар”, “”) = 3, L(“ва”, “”) = 2  
  
11. L(“в”, “бар ”) = min  
  
L(“”, “бар”) = 3, L(“”, “ба”) = 2  
  
12. L(“ва”, “ба”) = min  
  
L(“в”, “б”) = 1  
  
13. L(“в”, “ба”) = min  
  
L(“”, “ба”) = 2, L(“в”, “б”) = 1, L(“”, “б”) = 1  
  
14. L(“ва”, “б”) = min  
  
L(“в”, “б”) = 1, L(“ва”, “”) = 2, L(“в”, “”) = 1  
  
15. L(“ва”, “б”) = min(2, 3, 2) = 2  
  
16. L(“в”, “ба”) = min(3, 2, 2) = 2  
  
17. L(“ва”, “ба”) = min(3, 3, 1) = 1  
  
18. L(“в”, “бар”) = min(4, 3, 3) = 3  
  
19. L(“вар”, “б”) = min(3, 4, 3) = 3  
  
20. L(“вар”, “ба ”) = min(2, 4, 3) = 2  
  
21. L(“ва”, “бар ”) = min(4, 2, 3) = 2  
  
22. L(“вар”, “бар ”) = min(3, 3, 1) = 1  
  
23. L(“в”, “бара ”) = min(5, 4, 4) = 4  
  
24. L(“в”, “баран ”) = min(6, 5, 5) = 5  
  
25. L(“ва”, “бара ”) = min(5, 3, 4) = 3  
  
26. L(“вар”, “бара ”) = min(4, 2, 3) = 2  
  
27. L(“ва”, “баран ”) = min(6, 4, 5) = 4  
  
28. L(“вар”, “баран ”) = min(5, 3, 4) = 3

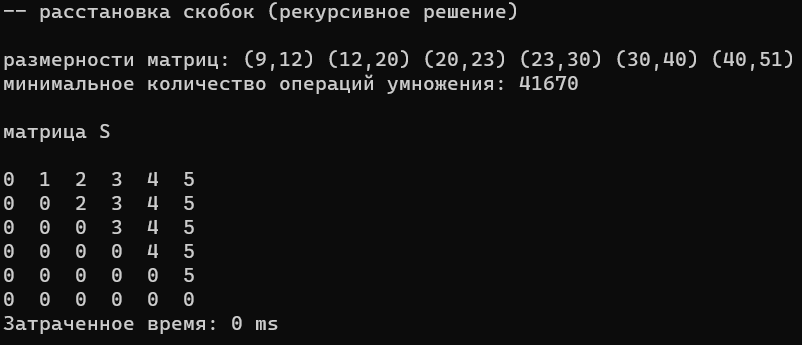
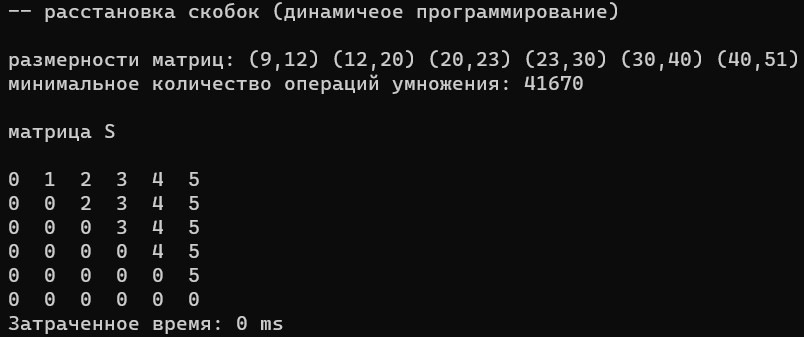
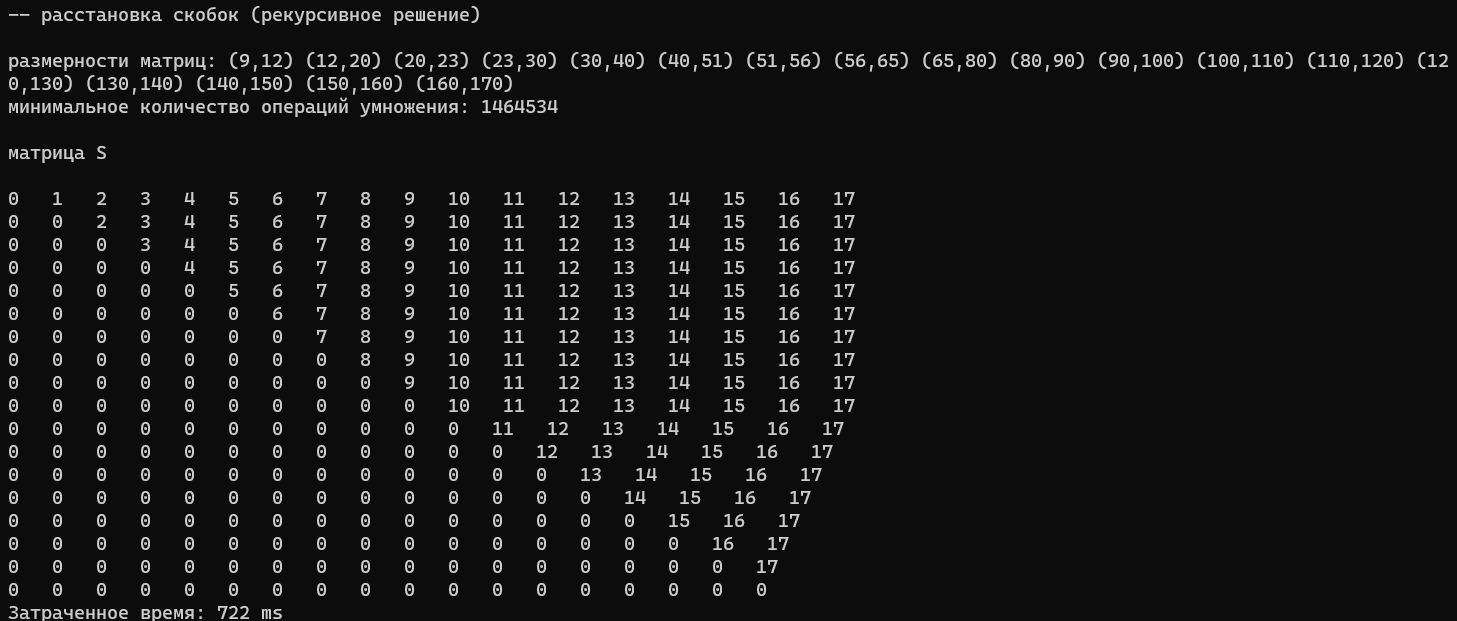
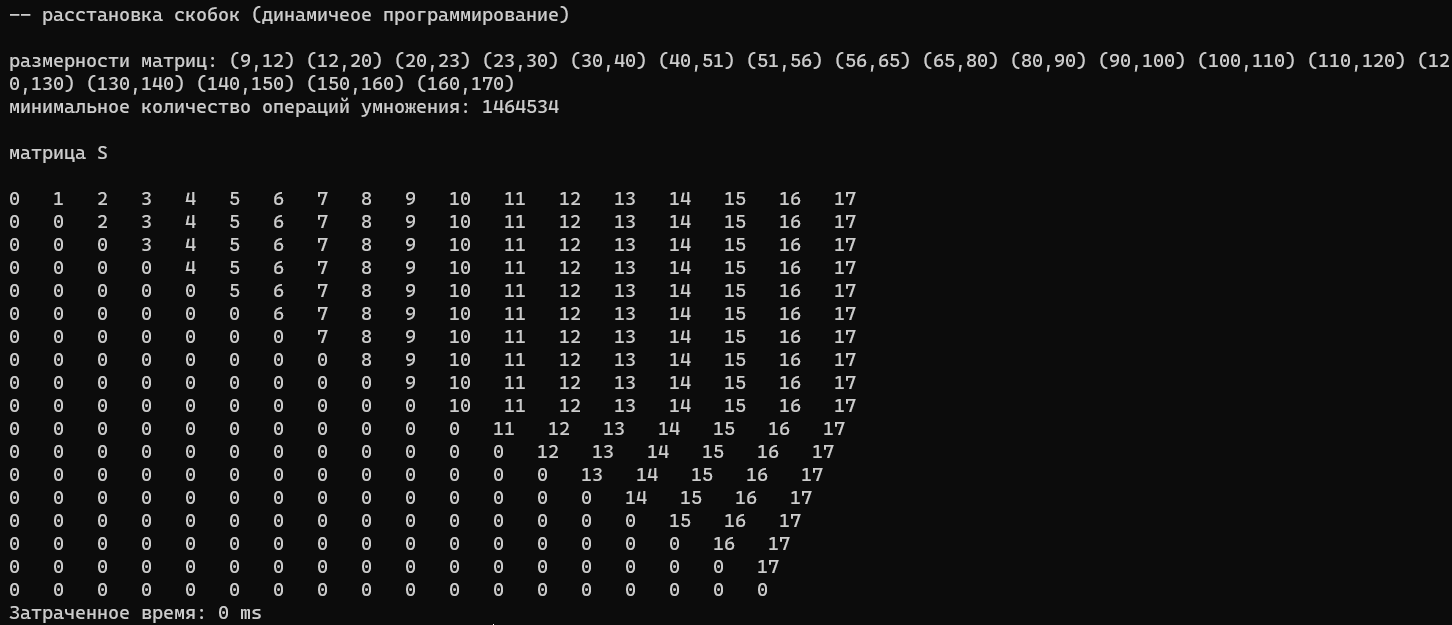
Ответ: 3.

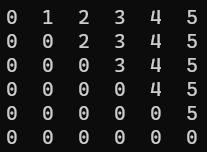
**Задание 5.**

*Четные варианты. Выполнить сравнительный анализ времени затраченного на решение задачи об оптимальной расстановке скобок при умножении нескольких матриц для двух методов решения (рекурсивное решение, динамическое программирование). Размерность матриц взять в соответствии с вариантом. Объяснить в отчете принцип расставления скобок по итоговой матрице + код + копии экрана.*

Условие: 9\*12, 12\*20, 20\*23, 23\*30, 30\*40, 40\*51

**Выполнение:** выполнил сравнительный анализ времени затраченного на решение задачи об оптимальной расстановке скобок при умножении нескольких матриц для двух методов решения(рекурсивного и дин.прогр.)  
  
  
При малом количестве матриц, расчет их оптимальной расстановки выполняется довольно быстро, как для метода динамического программирования, так и для рекурсивного. Однако при большом количестве матриц, рекурсивный метод, естественно, затрачивает гораздо большее время для расчета, в то время как метод динамического программирования высчитывает всё примерно с такой же скоростью:  
  




**Вывод**: в ходе лабораторной работы я освоила общие принципы решения задач методом динамического программирования, сравнила полученные решения задач с рекурсивным методом.

!!!ЭЛЕМЕНТЫ МАТРИЦЫ НАЧИНАЮТСЯ С 1!!!

Исходные матрицы: (9,12); (12,20); (20,23); (23,30); (30,40); (40,51)

Принцип расставления скобок по итоговой матрице:

Скобки расставляются по принципу «сначала внешние – затем внутренние». Элемент матрицы, который имеет индекс (1, 6) – это число 5, оно означает, что точка разрыва скобок находится между 1 и 6 матрицей после 5 матрицы, что позволяет расставить скобки следующим образом:

(A1 x A2 x A3 x A4 x A5) x A6

Точка разрыва между 1 и 5 матрицей находится в элементе матрицы под индексом (1, 5) – это число 4, что означает, что точка разрыва скобок находится между 1 и 5 матрицей после 4 матрицы, что позволяет расставить скобки следующим образом:

((A1 x A2 x A3 x A4) x A5) x A6

Совершая те же действия, получается следующая последовательность перемножений:

((((A1 x A2) x A3) x A4) x A5) x A6

Проверка:

9\*12\*20 + 9\*20\*23 + 9\*23\*30 + 9\*30\*40 + 9\*40\*51 = 41670. Ответ сошёлся.