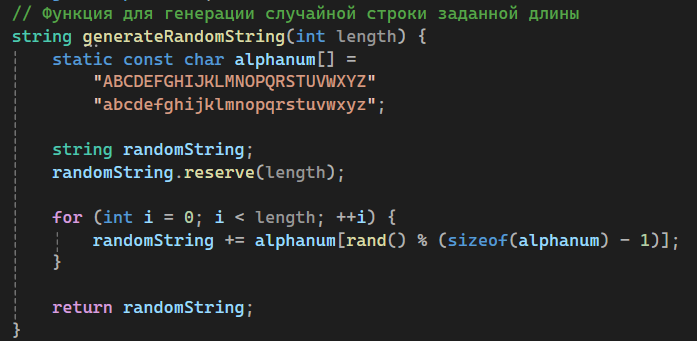
**Лабораторная работа №4. Динамическое программирование**

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

**Ход работы**

## **1. Сгенерировать две строки размером 300 и 250 символов латинского алфавита.**

Код программы:



Листинг 1. Функция генерации случайной строки

Результат:

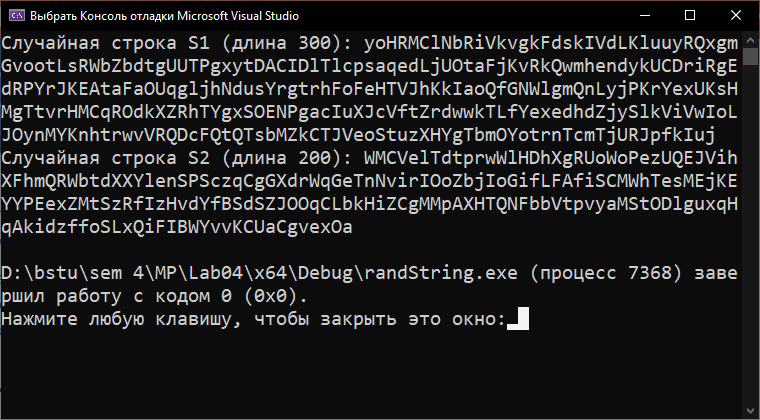
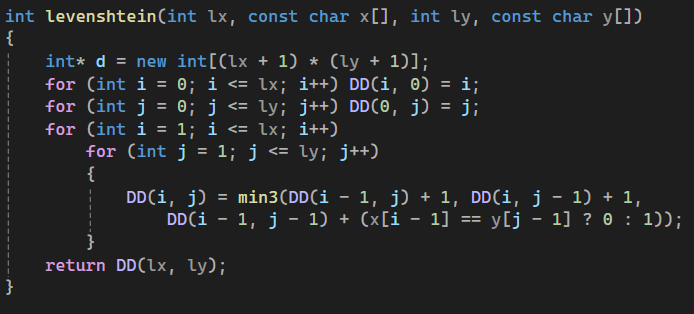


Рисунок 1. Результат генерации строк

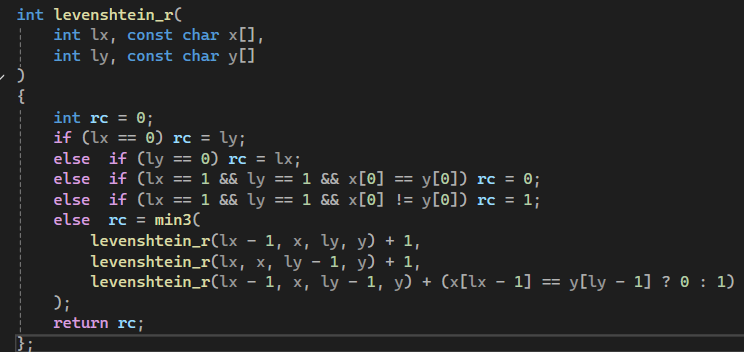
**2 Вычислить двумя способами (рекурсивно и с помощью динамического программирования)  – дистанцию Левенштейна для , где - длина строки ,  - строка состоящая из первых  символов строки . (копии экрана и код вставить в отчет).**

Реализация через динамическое программирование:



Листинг 2.1. Реализация через динамическое программирование

Реализация через рекурсию:



Листинг 2.2. Реализация через рекурсию

Результат выполнения:

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

3:

Изображение выглядит как текст, снимок экрана, линия, число

Автоматически созданное описание

Экспоненциальная зависимость для рекурсивного алгоритма и линейная зависимость для динамического программирования

**4 Реализовать вручную пример вычисления дистанции Левенштейна при помощи рекурсивного алгоритма**

1. L(«Сом», «Домик») = min

2. L(«Со», «Домик») = min

3. L(«Сом», «Доми») = min

4. L(«Со», «Доми») = min

5. L(«С», «Домик») = min

L(«», «Домик») = 5,

L(«», «Доми») = 4

6. L(«С», «Доми») = min

L(«», «Доми») = 4,

L(«», «Дом») = 3

7. L(«Сом», «Дом») = min

7. L(«Сом», «До») = min

7. L(«Сом», «Д») = min

L(«Сом», «») = 3

8. L(«Со», «Дом») = min

9. L(«С», «Дом») = min

L(«», «До») = 2

10. L(«Со», «До») = min

11. L(«С», «До») = min

L(«», «Д») = 1

12. L(«Со», «Д») = min

L(«Со», «») = 2,

L(«С», «») = 1,

13. L(«С», «Д») = min

L(«С», «») = 1

15. L(«С», «Д») = min(2,2,1) = 1

16. L(«Со», «Д») = min (2, 3, 2) = 2

17. L(«С», «До») = min (3, 2, 2) = 2

18. L(«Со», «До») = min (3, 3, 2) = 2

19. L(«С», «Доми») = min (4, 3, 2) = 2

20. L(«Сом», «Д») = min (3, 4, 3) = 3

21. L(«Сом», «До») = min (3, 4, 3) = 3

22. L(«Со», «Дом») = min (3, 3, 3) = 3

23. L(«Сом», «Дом») = min (4, 4, 3) = 3

24. L(«С», «Доми») = min (5, 3, 4) = 4

25. L(«С», «Домик») = min (6, 5, 5) = 5

26. L(«Со», «Доми») = min (5, 4, 2) = 2

27. L(«Сом», «Доми») = min (3, 4, 4) = 3

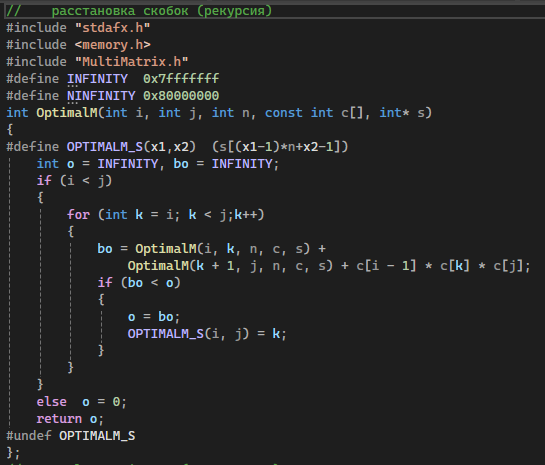
28. L(«Со», «Домик») = min (6, 3, 5) = 3

29. L(«Сом», «Домик») = min (4, 4, 3) = 3

**5 Выполнить сравнительный анализ времени затраченного на решение задачи об оптимальной расстановке скобок при умножении нескольких матриц для двух методов решения (рекурсивное решение, динамическое программирование).**

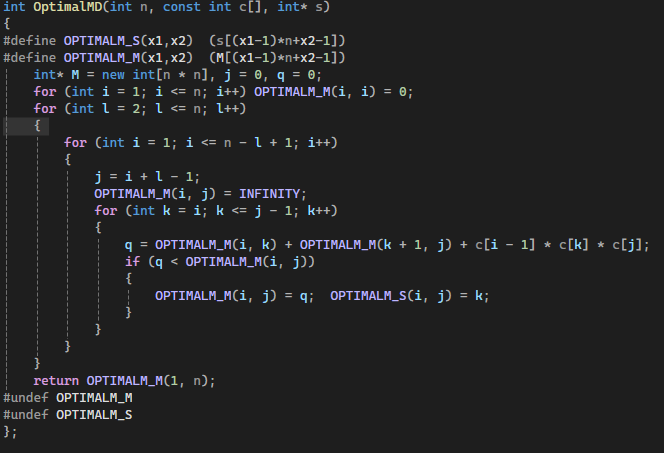


Реализация через рекурсию:



Листинг 5.1. Реализация через рекурсию

Реализация через динамическое программирование:



Листинг 5.2. Реализация через динамическое программирование

Результат:

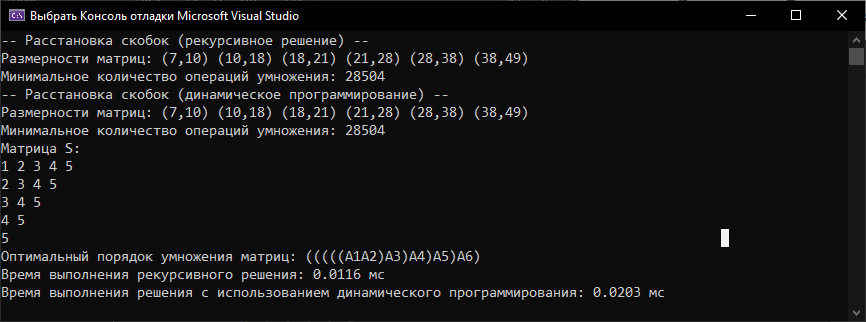


Рисунок 4. Результаты динамического программирования и рекурсии

Принцип расстановки скобок по итоговой матрице:

Скобки расставляются по принципу «сначала внешние – затем внутренние». Имеется 6 матриц. Их размерность:

А1= 60\*15,

А2= 15\*50,

А3= 50\*53,

А4 =53\*10,

А5 =10\*30,

А6 =30\*11.

**Вывод:** освоил общие принципы решения задач методом динамического программирования, сравнил полученные решения задач с рекурсивным методом.