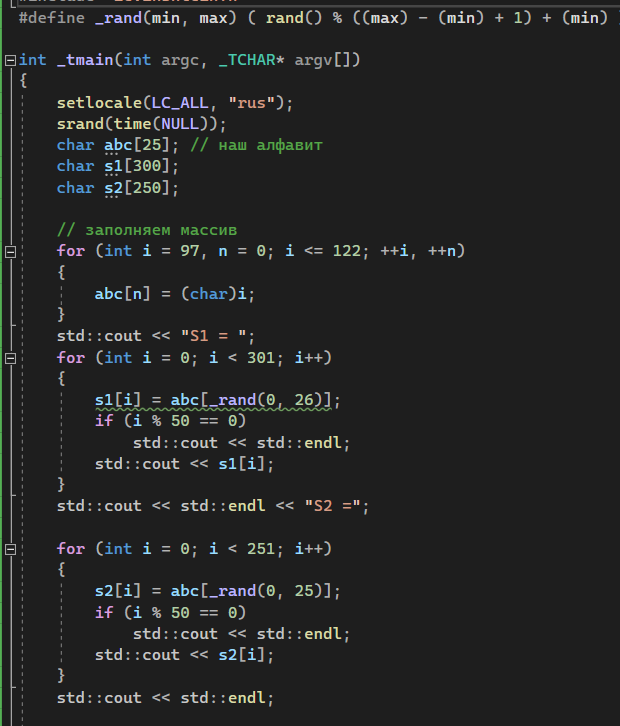
Минск 2022

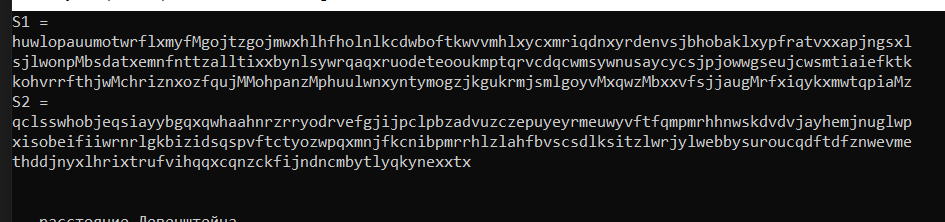
**Цель работы:** освоить общие принципы решения задач методом динамического программирования, сравнить полученные решения задач с рекурсивным методом.

**Практическая часть**

**Задание 1.**

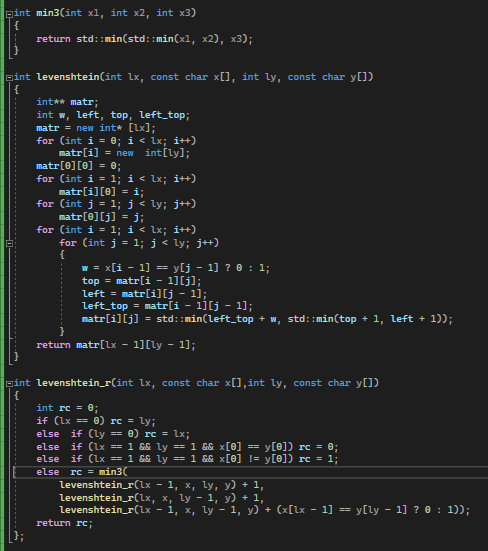
На языке С++ сгенерировать случайным образом строку букв латинского алфавита *S1* длиной 300 символов и *S2*длиной 250.

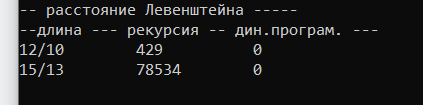




**Задание 2.**

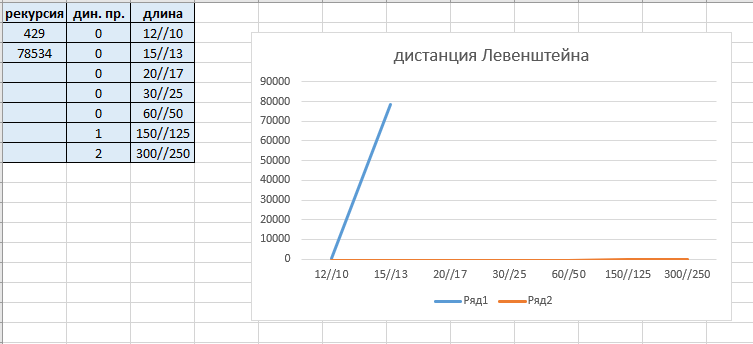
Вычислить двумя способами (рекурсивно и с помощью динамического программирования) – Левенштейна для , где - длина строки , - строка, состоящая из первых символов строки . (копии экрана и код вставить в отчет).

****



**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).



**Задание 4.**

Реализовать вручную пример вычисления дистанции Левенштейна при помощи рекурсивного алгоритма (в соответствии с вариантом).

|  |  |
| --- | --- |
| Задание 4 | |
| Сон | Донор |



= 5.

= 4.



= 4.

= 3.



= 3.

= 2.



= 2.

= 1.



= 3.

= 2.



= 2.

= 1.



= 1.

= 1.

= 0.



**Задание 5.**

Выполнить сравнительный анализ времени затраченного на решение задачи о наибольшей общей подпоследовательности для двух методов решения (рекурсивное решение, динамическое программирование). Две последовательности взять в соответствии с вариантом. Построить графики зависимости времени вычисления от . Отобразить ход решения в отчете(по примеру из лекции) + код и копии экрана.

Сейчас мы для W ищем совпадение:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| l | H | R | W | C | Y | U | J |
| W | 0 | 0 | 1 | 1 | 1 | 1 | 1 |
| L |  |  |  |  |  |  |  |
| P |  |  |  |  |  |  |  |
| C |  |  |  |  |  |  |  |
| A |  |  |  |  |  |  |  |
| U |  |  |  |  |  |  |  |

2.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| l | H | R | W | C | Y | U | J |
| W | 0 | 0 | 1 | 1 | 1 | 1 | 1 |
| L | 0 | 0 | 1 | 1 | 1 | 1 | 1 |
| P |  |  |  |  |  |  |  |
| C |  |  |  |  |  |  |  |
| A |  |  |  |  |  |  |  |
| U |  |  |  |  |  |  |  |

3.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| l | H | R | W | C | Y | U | J |
| W | 0 | 0 | 1 | 1 | 1 | 1 | 1 |
| L | 0 | 0 | 1 | 1 | 1 | 1 | 1 |
| P | 0 | 0 | 1 | 1 | 1 | 1 | 1 |
| C | 0 | 0 | 1 | 2 | 2 | 2 | 2 |
| A | 0 | 0 | 1 | 2 | 2 | 2 | 2 |
| U | 0 | 0 | 1 | 2 | 2 | 3 | 3 |

Крайний элемент в нижнем правом углу показывает нам наибольшую общую последовательность.

График зависимости для сравнение скорости выполнения рекурсивного алгоритма и динамического программирования

