**УО «Белорусский государственный технологический университет»**

Математическое программирование

Тема: Динамическое программирование

Выполнила студентка ФИТ

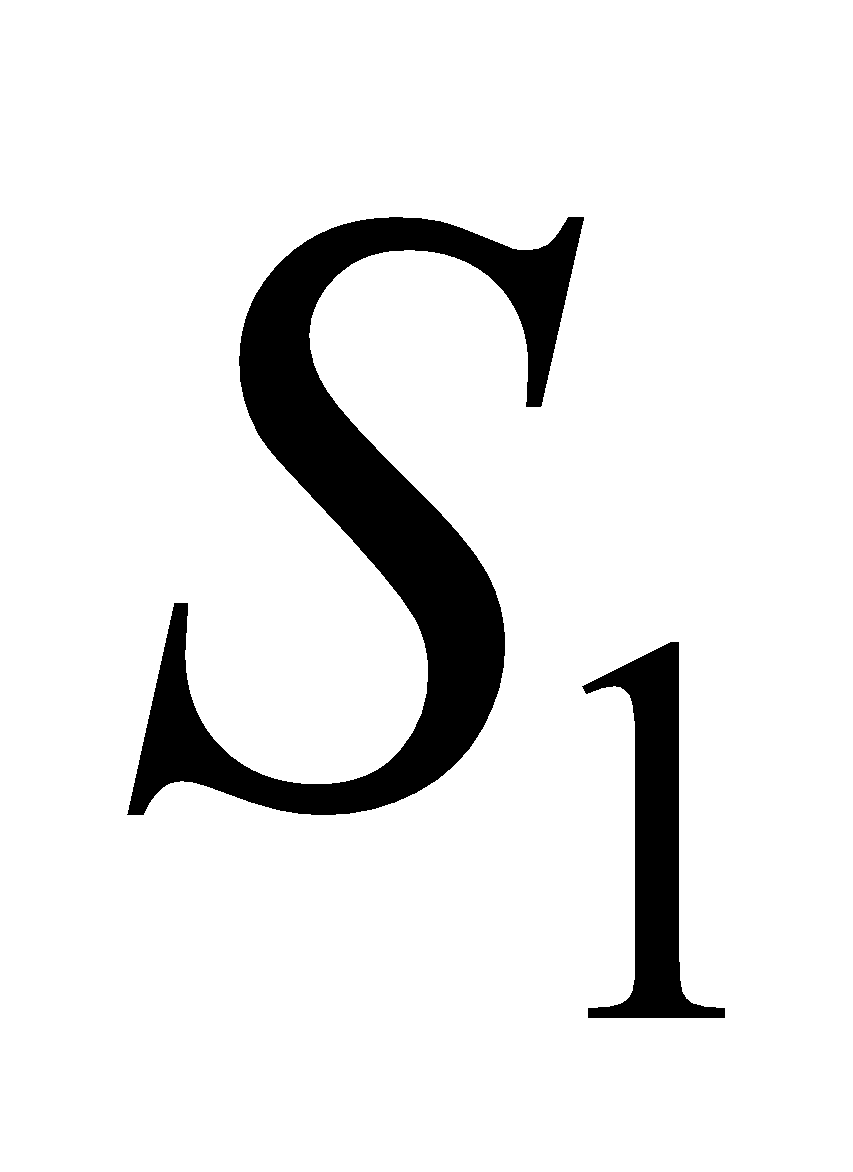
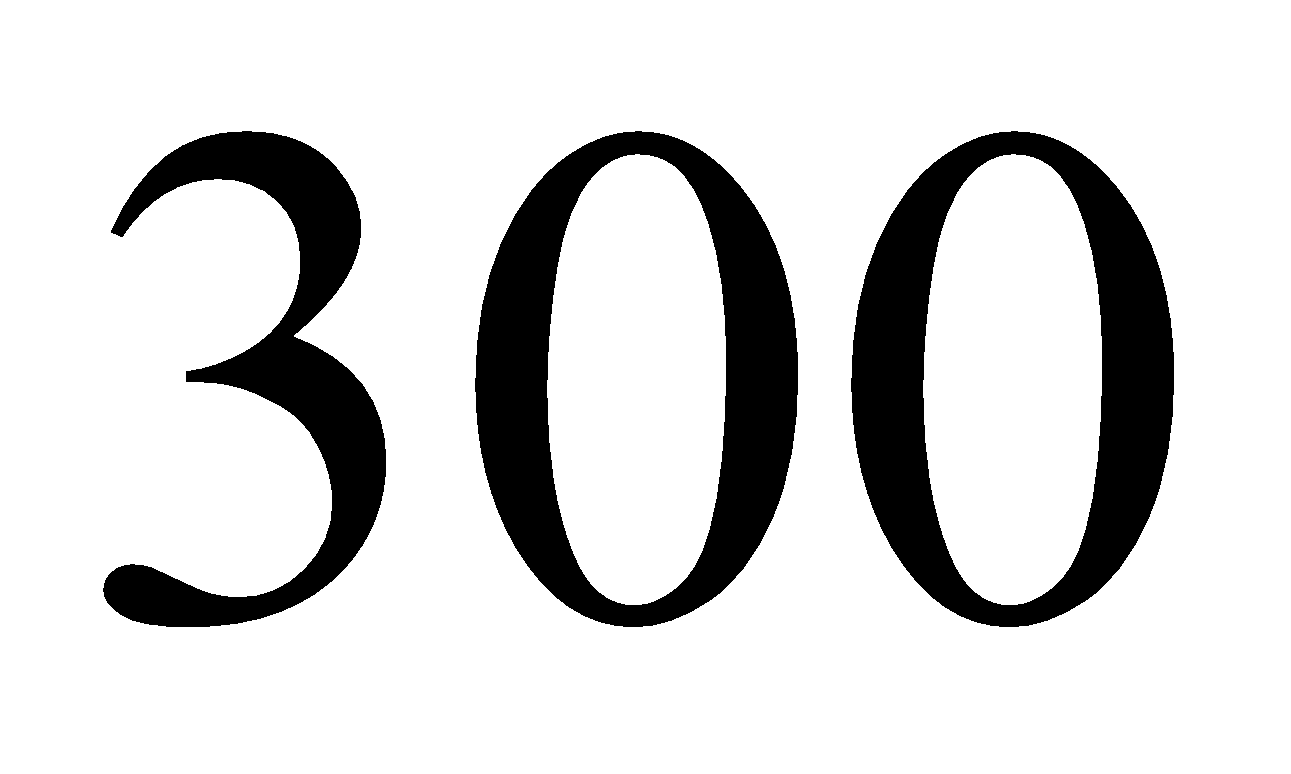
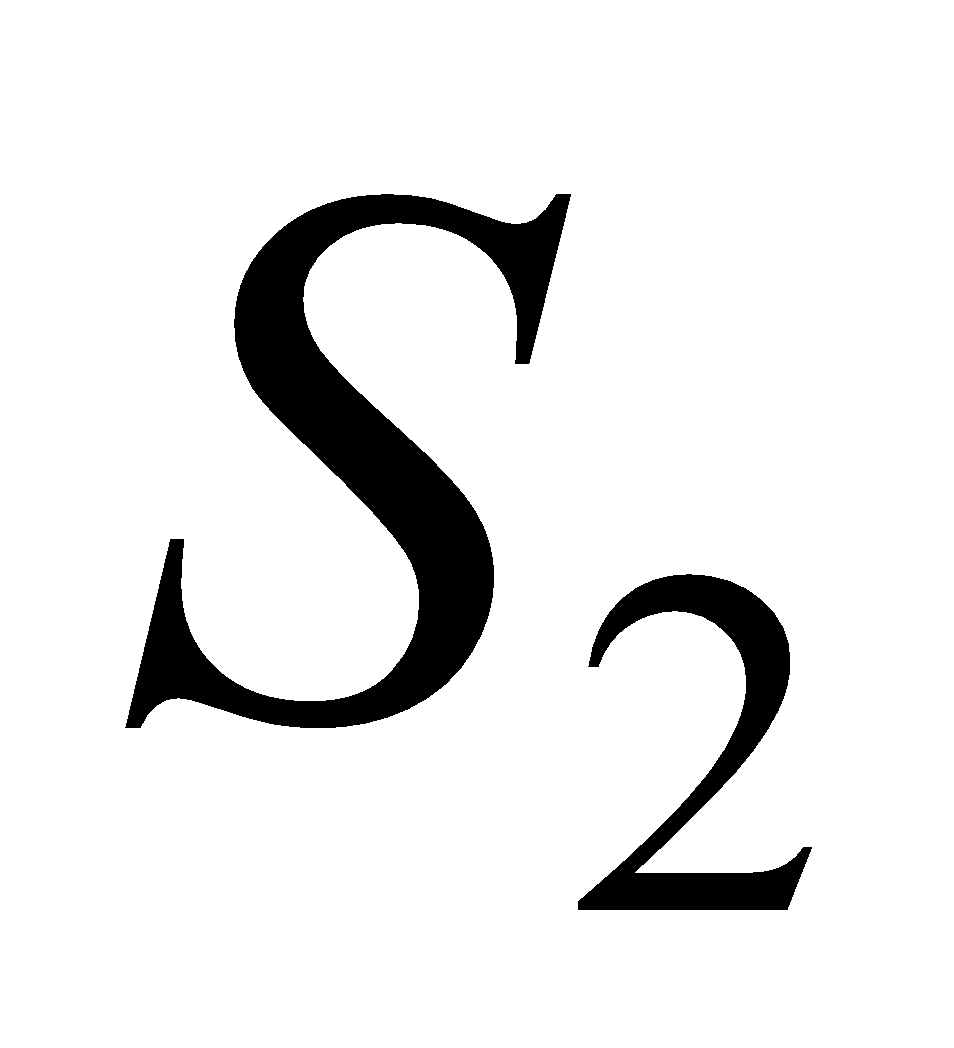
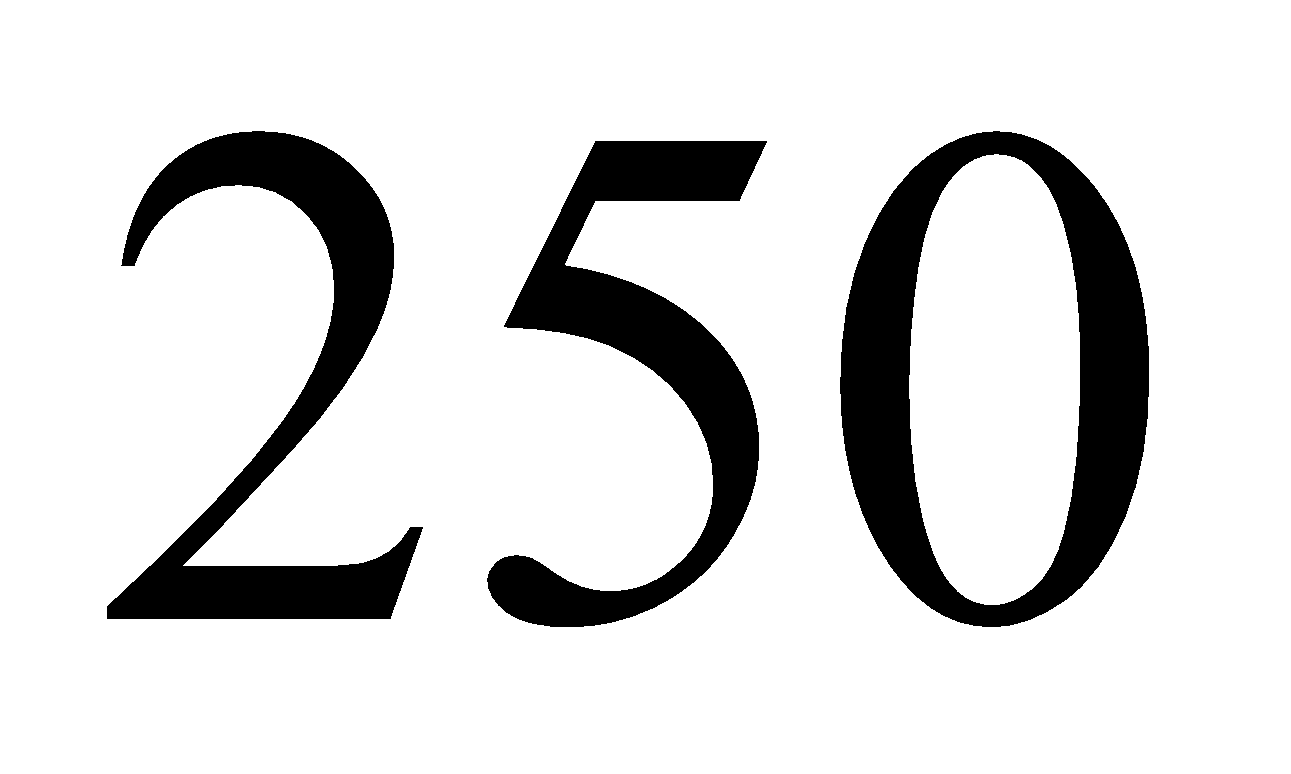
2 курса 1 группы

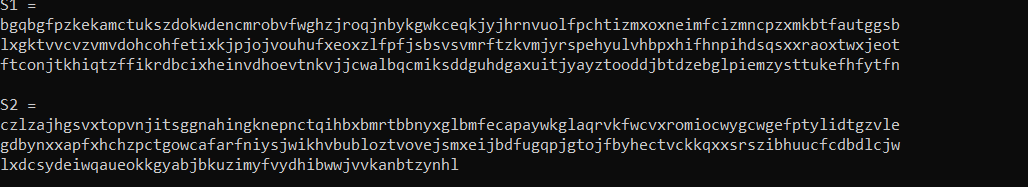
Пригодич Вера

Минск 2022

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

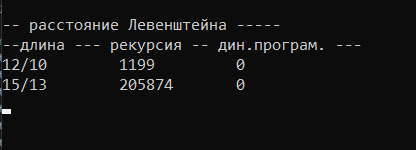
**Задание 1.**

На языке С++ сгенерировать случайным образом строку букв латинского алфавита  длиной  символов и длиной .



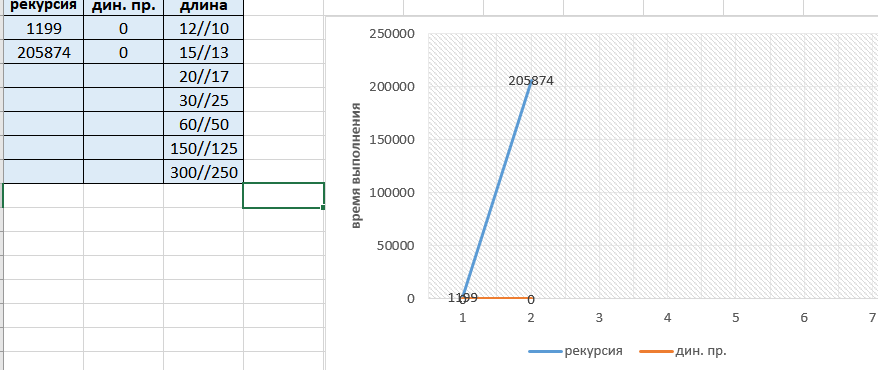
**Задание 2.**

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).

****

**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).



|  |  |  |  |
| --- | --- | --- | --- |
| Вариант | Задание 4 | | Задание 5 |
| 10 | Ель | Дрель | 100\*15, 15\*20, 20\*43, 43\*70, 70\*40, 40\*71 |

**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).

1. **Вычисление дистанции Левенштейна**

По варианту необходимо вычислить , т.е. количество символьных преобразований, которые необходимо совершить, чтобы превратить слово Ель в слово Дрель.

1. L(«ель», «дрель») =
2. L(«ел», «дрель») =
3. L(«ель», «дрел») =
4. L(«ел», «дрел») =
5. L(«е», «дрель») =

L(«е», «дрель») = 5, L(«», «дрел») = 4.

1. L(«е», «дрел») =

L(«», «дрел») = 4, L(«», «дре») = 3.

1. L(«ель», «дре») =
2. L(«ел», «дре») =
3. L(«е», «дре») =

L(«», «дре») = 3, L(«», «др») = 2.

1. L(«ель», «др») =
2. L(«ел», «др») =
3. L(«е», «др») =

L(«», «др») = 2, L(«», «д») = 1.

1. L(«ель», «д») =

L(«ель», «») = 3, L(«ел», «») = 2.

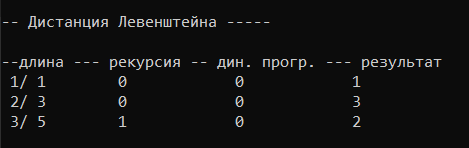
1. L(«ел», «д») =

L(«ел», «») = 2, L(«е», «») = 1.

1. L(«е», «д») =

L(«», «д») = 1, L(«е», «») = 1, L(«», «») = 0.

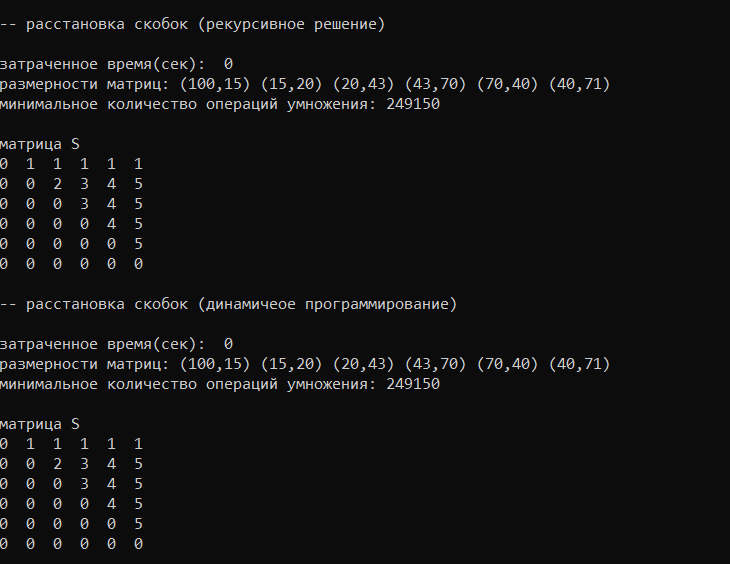
1. L(«е», «д») = min(2,2,1) = 1.
2. L(«ел», «д») = min(3,3,2) = 2.
3. L(«ель», «д») = min(4,4,3) = 3.
4. L(«е», «др») = min(3,3,2) = 2.
5. L(«ел», «др») = min(4,4,3) = 3.
6. L(«ель», «др») = min(5,5,4) = 4.
7. L(«е», «дре») = min(4,4,3) = 3.
8. L(«ел», «дре») = min(3,5,4) = 3.
9. L(«ель», «дре») = min(4,6,5) = 4.
10. L(«е», «дрел») = min(5,3,4) = 3.
11. L(«е», «дрель») = min(6,4,5) = 4.
12. L(«ел», «дрел») = min(4,4,2) = 2.
13. L(«ель», «дрел») = min(3,5,4) = 3.
14. L(«ел», «дрель») = min(5,3,4) = 3.
15. L(«ель», «дрель») = min(4,4,2) = 2.

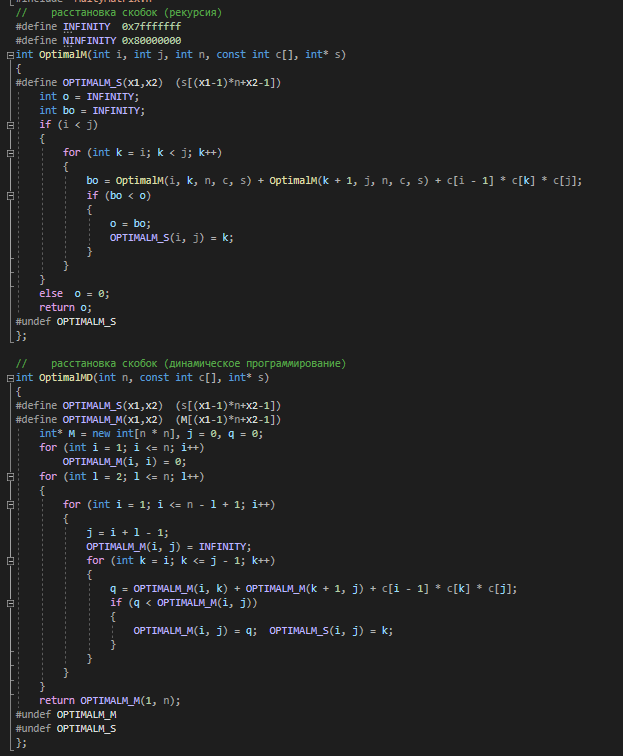


**Задание 5.**

Выполнить сравнительный анализ времени затраченного на решение задачи об оптимальной расстановке скобок при умножении нескольких матриц для двух методов решения (рекурсивное решение, динамическое программирование). Размерность матриц взять в соответствии с вариантом.

Объяснить в отчете принцип расставления скобок по итоговой матрице + код + копии экрана.





Принцип расстановки скобок по итоговой матрице:

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

А1=100\*15,

А2=15\*20,

А3=20\*43,

А4 =43\*70,

А5 =70\*40,

А6 =40\*71.

Матрица S:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **1** | **2** | **3** | **4** | **5** | **6** |
| **1** | 0 | 1 | 1 | 1 | 1 | 1 |
| **2** | 0 | 0 | 2 | 3 | 4 | 5 |
| **3** | 0 | 0 | 0 | 3 | 4 | 5 |
| **4** | 0 | 0 | 0 | 0 | 4 | 5 |
| **5** | 0 | 0 | 0 | 0 | 0 | 5 |
| **6** | 0 | 0 | 0 | 0 | 0 | 0 |

Найдем элемент (1,6) в матрице S, он равен 1. Это означает, что точка разрыва между 1-ой и 6-ой матрицей находится после 1-ой матрицы. Что позволяет расставить скобки следующим образом:

A1\*(A2\*A3\*A4\*A5\*A6)

Точку разрыва между второй и шестой матрицей определяет элемент (2,6). Он равен 5. Следовательно разрыв будет после 5-ой матрицы.

A1\*((A2\*A3\*A4\*A5) \*A6)

Далее берем элемент (2,5) и получаем, что он равен 4. Следовательно получаем:

A1\*(((A2\*A3\*A4) \*A5) \*A6)

И на последнем шаге мы возьмем элемент (2,4) и он равен 3:

A1\*((((A2\*A3) \*A4) \*A5) \*A6)

Это выражение и есть конечное.

Полученная расстановка скобок позволяет получить минимальное количество операций умножения, равное 249150.

**Вывод:** освоила общие принципы решения задач методом динамического программирования, сравнила полученные решения задач с рекурсивным методом.