Учреждение образования «Белорусский государственный технологический университет»

Факультет информационных технологий

Кафедра информационных систем и технологий

Лабораторная работа №4

Динамическое программирование

Выполнил

студент 2 курса 8 группы

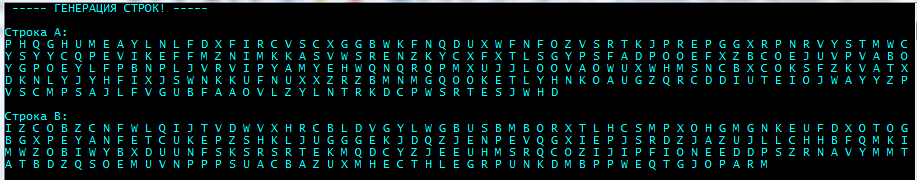
Бондарчик Е.Н

Проверил Бракович А.И

Минск 2017

**ЗАДАНИЕ 1**

Сгенерировать две строки размерами 300 и 250 символов:



**ЗАДАНИЕ 2**

double kmas[] = { 0.04, 0.05, 0.625, 0.1, 0.2, 0.5, 1 };

cout << " ----- ДИСТАНЦИЯ ЛЕВЕНШТЕЙНА! ----- " << endl << endl;

for (int j = 0; j < 3; j++)

{

int res = 0; clock\_t t1 = 0, t2 = 0;

for (int i = 0; i < 300 \* kmas[j]; i++)

cout << A[i] << " "; cout << endl;

for (int i = 0; i < 250 \* kmas[j]; i++)

cout << B[i] << " "; cout << endl;

t1 = clock();

res = levenshtein(300 \* kmas[j], A, 250 \* kmas[j], B); t2 = clock();

cout << "Динамическое программирование" << endl << "Дистанция Левенштейна: " << res << endl;

cout << "Время: " << t2 - t1 << endl;

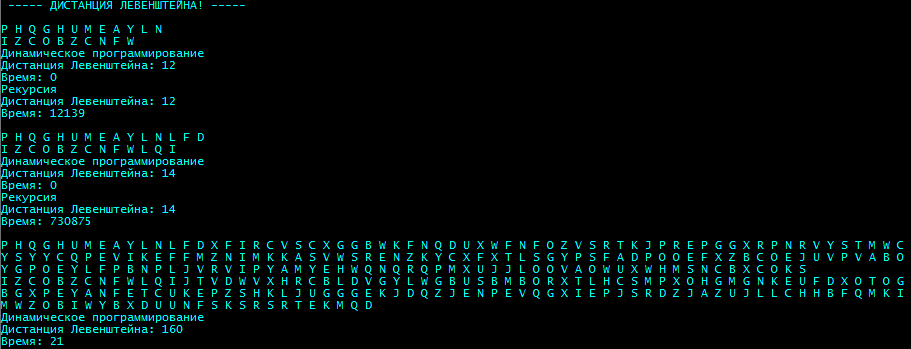
t1 = clock(); res = levenshtein\_r(300 \* kmas[j], A, 250 \* kmas[j], B); t2 = clock();

cout << "Рекурсия" << endl << "Дистанция Левенштейна: " << res << endl;

cout << "Время: " << t2 - t1 << endl << endl;

}

Пример выполнения:



**ЗАДАНИЕ 3**

Как видно из скриншота задания номер 2 и графика, представленного ниже, при самых небольших значениях k, а соответственно, при небольшой длине строк, метод динамического программирования является выигрышным вариантом по сравнению с методом рекурсии, который проигрывает. Это происходит по той причине, что в методе динамического программирования мы должны рассмотреть полиноминальное количество вариантов, пока не найдем решение, а в методе рекурсии перебор является экспоненциальным. При кол-ве букв 160 и 120 рекурсивный метод проработал всю ночь, но так и не выдал решения.

**ЗАДАНИЕ 4**

Найти расстояние Левенштейна между словами «КОЛ» и «СТОЛБ».

1. L(«КОЛ», «СТОЛБ») = min

2. L(«КО», «СТОЛБ») = min

3. L(«КОЛ», «СТОЛ») = min

4. L(«КО», «СТОЛ») = min

5. L(«К», «СТОЛБ») = min

L(«», «СТОЛБ») = 5,

L(«», «СТОЛ») = 4

6. L(«К», «СТОЛ») = min

L(«», «СТОЛ») = 4,

L(«», «СТО») = 3

7. L(«КОЛ», «СТО») = min

8. L(«КО», «СТО») = min

9. L(«КОЛ», «СТ») = min

10. L(«КОЛ», «С») = min

L(«КОЛ», «») = 3,

L(«КО», «») = 2

11. L(«К», «СТО») = min

L(«», «СТО») = 3,

L(«», «СТ») = 2

12. L(«КО», «СТ») = min

L(«К», «С») = 1,

13. L(«К», «СТ») = min

L(«», «СТ») = 2,

L(«К», «С») = 1,

L(«», «С») = 1,

14. L(«КО», «С») = min

L(«К», «С») = 1,

L(«КО», «С») = 2,

L(«К», «») = 1

15. L(«КО», «С») = min (2, 3, 2) = 2

16. L(«К», «СТ») = min (3, 2, 2) = 2

17. L(«КО», «СТ») = min (3, 3, 2) = 2

18. L(«К», «СТО») = min (4, 3, 3) = 3

19. L(«КОЛ», «С») = min (4, 3, 3) = 3

20. L(«КОЛ», «СТ») = min (3, 4, 3) = 3

21. L(«КО», «СТО») = min (4, 3, 2) = 2

22. L(«КОЛ», «СТО») = min (3, 4, 3) = 3

23. L(«К», «СТОЛ») = min (5, 4, 4) = 4

24. L(«К», «СТОЛБ») = min (6, 5, 5) = 5

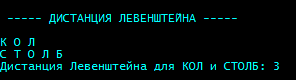
25. L(«КО», «СТОЛ») = min (5, 3, 4) = 3

26. L(«КОЛ», «СТОЛ») = min (4, 3, 3) = 3

27. L(«КО», «СТОЛБ») = min (6, 4, 5) = 4

28. L(«КОЛ», «СТОЛБ») = min (5, 3, 4) = 3

Дистанция Левенштейна для слов «КОЛ» и «СТОЛБ»: 3.

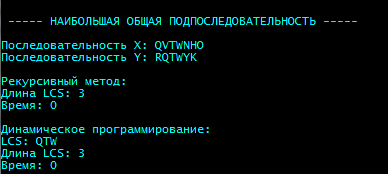


**ЗАДАНИЕ 5**

Даны две последовательности: QVTWNHO и RQTWYK

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | | R | | Q | | T | | W | | Y | | K | |
|  | 0 | | 0 | | 0 | | 0 | | 0 | | 0 | | 0 | |
| Q | 0 | | 0 | | 1 | | 1 | | 1 | | 1 | | 1 | |
| V | 0 | | 0 | | 1 | | 1 | | 1 | | 1 | | 1 | |
| T | 0 | | 0 | | 1 | | 2 | | 2 | | 2 | | 2 | |
| W | 0 | | 0 | | 1 | | 2 | | 3 | | 3 | | 3 | |
| N | 0 | | 0 | | 1 | | 2 | | 3 | | 3 | | 3 | |
| H | 0 | | 0 | | 1 | | 2 | | 3 | | 3 | | 3 | |
| O | 0 | | 0 | | 1 | | 2 | | 3 | | 3 | | 3 | |
|  | |  | | R | | Q | | T | | W | | Y | | K | |
|  | | 0 | | 0 | | 0 | | 0 | | 0 | | 0 | | 0 | |
| Q | | 0 | |  | |  | |  | |  | |  | |  | |
| V | | 0 | |  | |  | |  | |  | |  | |  | |
| T | | 0 | |  | |  | |  | |  | |  | |  | |
| W | | 0 | |  | |  | |  | |  | |  | |  | |
| N | | 0 | |  | |  | |  | |  | |  | |  | |
| H | | 0 | |  | |  | |  | |  | |  | |  | |
| O | | 0 | |  | |  | |  | |  | |  | |  | |

Проверка рекурсивным и динамичным алгоритмами:



Исследуем зависимость времени работы алгоритма от количества символов в строках:



Метод динамического программирования справляется со строками длиной 300 и 250 символов за 3 единицы тактового времени, в то время как рекурсивный метод уже на 14 символах считает почти 7 секунд (7357 единиц тактового времени). Таким образом, можно сделать вывод, что для решения задачи о нахождении наибольшей обшей подподследовательности динамическое программирование будет более эффективным методом, чем рекурсивный метод.

