Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное учреждение

высшего образования

«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт цифрового развития

Кафедра инфокоммуникаций

**ОТЧЕТ**

**ПО ЛАБОРАТОРНОЙ РАБОТЕ №12**

**дисциплины «Алгоритмизация»**

**Вариант\_\_\_**

|  |  |  |  |
| --- | --- | --- | --- |
|  | | Выполнил:  Иващенко Олег Андреевич  2 курс, группа ИВТ-б-о-22-1,  09.03.02 «Информационные и вычислительные машины», направленность (профиль) «Программное обеспечение средств вычислительной техники и автоматизированных систем»  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись) | |
|  | | Руководитель практики:  Доцент кафедры инфокоммуникации  Воронкин Роман Александрович  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись) | |
|  | |  | |

Отчет защищен с оценкой \_\_\_\_\_\_\_\_\_\_\_ Дата защиты\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Ставрополь, 2023 г.

**Тема**: «Алгоритм Левенштейна»

Порядок выполнения работы

Задача 1. Реализация функция вычисления числа Фибоначчи с помощью динамического программирования.

Таблица 1 – Код программы

|  |
| --- |
| using System;  class LevenshteinDistance  {  static void Main()  {  Console.WriteLine("Введите первое слово:");  Console.Write(">>> ");  string str1 = Console.ReadLine();  Console.WriteLine("Введите второе слово:");  Console.Write(">>> ");  string str2 = Console.ReadLine();  int distance = CalculateLevenshteinDistance(str1, str2);  Console.WriteLine($"Расстояние Левенштейна между '{str1}' и '{str2}' - {distance}");  Console.WriteLine("\nДля завершения работы программы нажмите любую клавишу...");  Console.ReadKey();  }  static int CalculateLevenshteinDistance(string str1, string str2)  {  int m = str1.Length;  int n = str2.Length;  int[,] dp = new int[m + 1, n + 1];  for (int i = 0; i <= m; i++)  for (int j = 0; j <= n; j++)  {  if (i == 0) dp[i, j] = j;  else if (j == 0) dp[i, j] = i;  else  {  int cost = (str1[i - 1] == str2[j - 1]) ? 0 : 1;  dp[i, j] = Math.Min(Math.Min(dp[i - 1, j] + 1, dp[i, j - 1] + 1), dp[i - 1, j - 1] + cost);  }  }  return dp[m, n];  }  } |

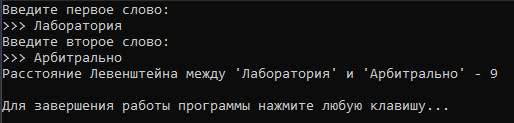


Рисунок 12 – Результат выполнения программы