Министерство цифрового развития, связи и массовых коммуникаций

Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования «Сибирский государственный университет

телекоммуникаций и информатики» (СибГУТИ)

Кафедра прикладной математики и кибернетики

Лабораторная работа

«Вычисление метрических характеристик реализаций алгоритмов»

Выполнил:

Студент группы ИП-017

Костин А.В.

Работу проверил:

ассистент кафедры ПМиК

Агалаков А.А.

Новосибирск 2023 г.

**Содержание**

Оглавление

[1. Задание 3](#_Toc153310346)

[2. Результаты выполнения 4](#_Toc153310347)

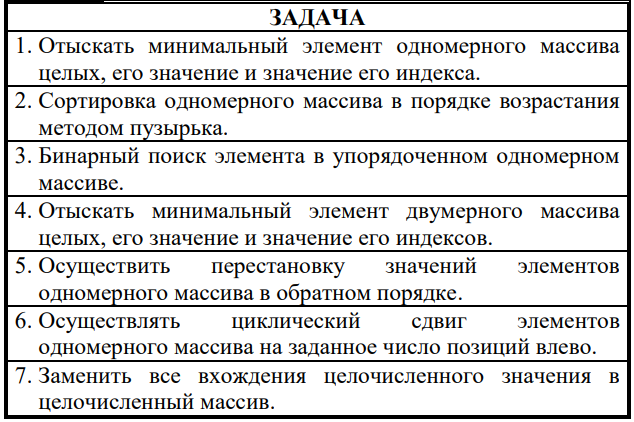
[3. Исходные тексты программ на языке Python 15](#_Toc153310348)

[4. Исходные тексты программ на языке C# 18](#_Toc153310349)

1. **Задание**

1. Написать подпрограммы на двух языках программирования для решения

следующих задач:



1. Для каждой подпрограммы вычислить следующие метрические характеристики:

♦ η\*2 – число единых по смыслу входных и выходных параметров, представленных в сжатой без избыточной формы;

♦ η1 - число отдельных операторов;

♦ η2 - число отдельных операндов;

♦ η - длина словаря реализации;

♦ Ν1 - общее число вхождений всех операторов в реализацию;

♦ Ν2 - общее число вхождений всех операндов в реализацию;

♦ Ν - длина реализации;

♦ Ν^ - предсказанная длина реализации по соотношению Холстеда;

♦ V\* - потенциальный объем реализации:

V\* =(2+ η\*2)\*log2(2 + η\*2)

♦ V - объем реализации: V\* = N\*log2η

♦ L - уровень программы через потенциальный объем: L =V\*/V.

♦ L ^ - уровень программы по реализации: L^= (2/η1)\*(η2/N2).

♦ I - интеллектуальное содержание программы:

I=(2/η1)\*(η2/N2)\*(N1 + N2)\*log2(η2/N2).

1. **Результаты выполнения**

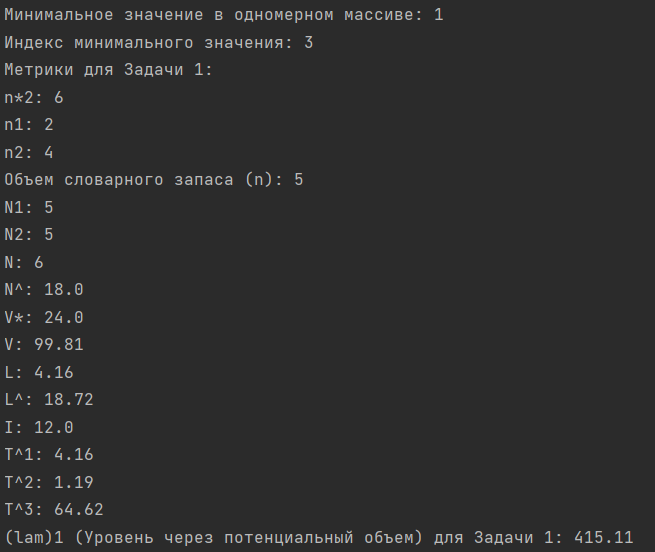
****

Рисунок 1. Метрика для программы Python

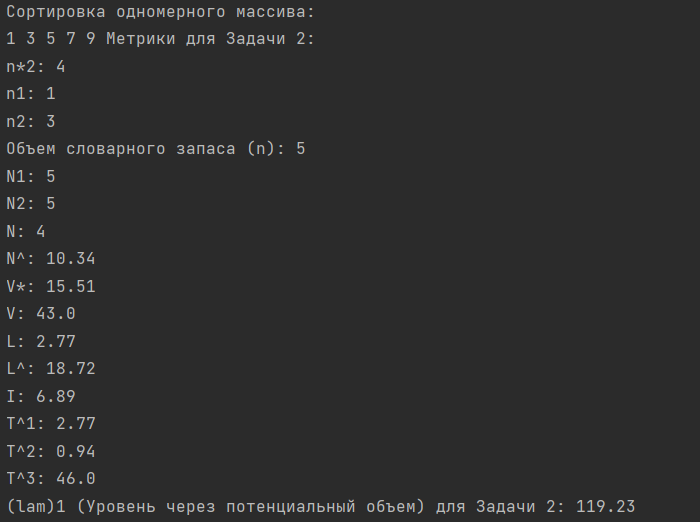
****

Рисунок 2. Метрика для программы Python

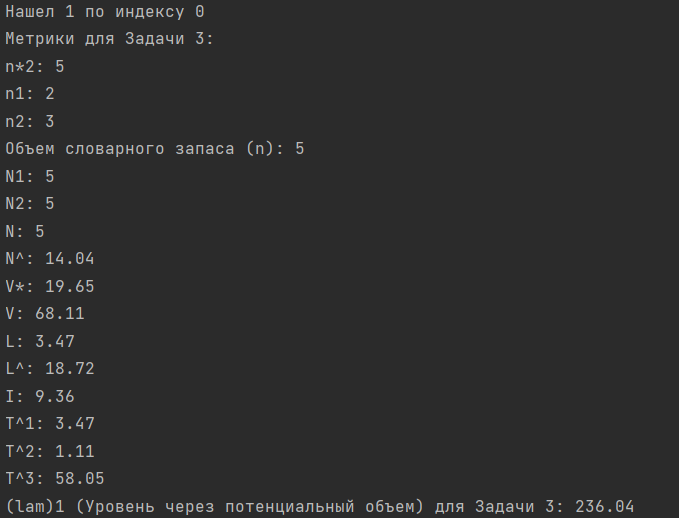


Рисунок 3. Метрика для программы Python

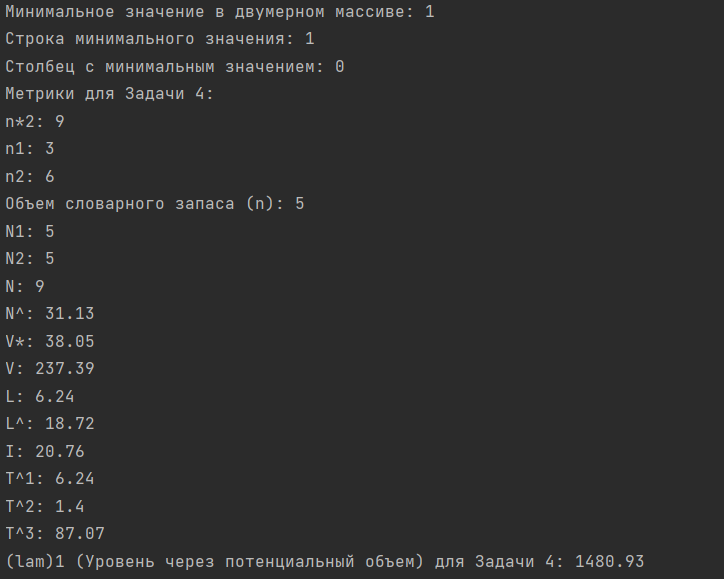


Рисунок 4. Метрика для программы Python

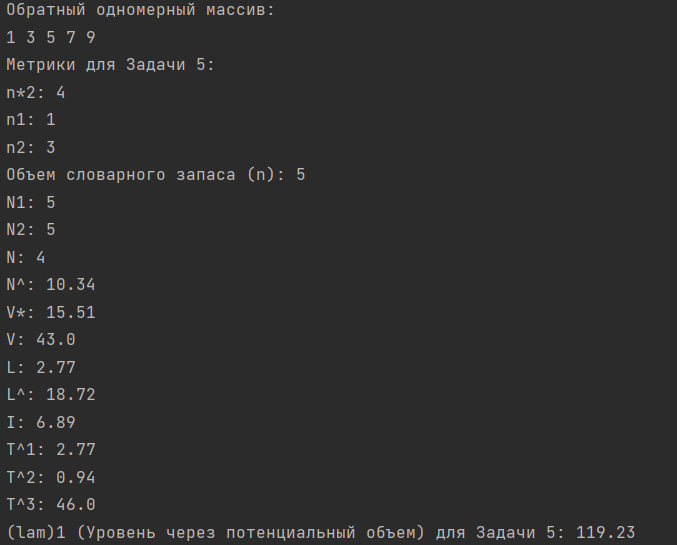


Рисунок 5. Метрика для программы Python

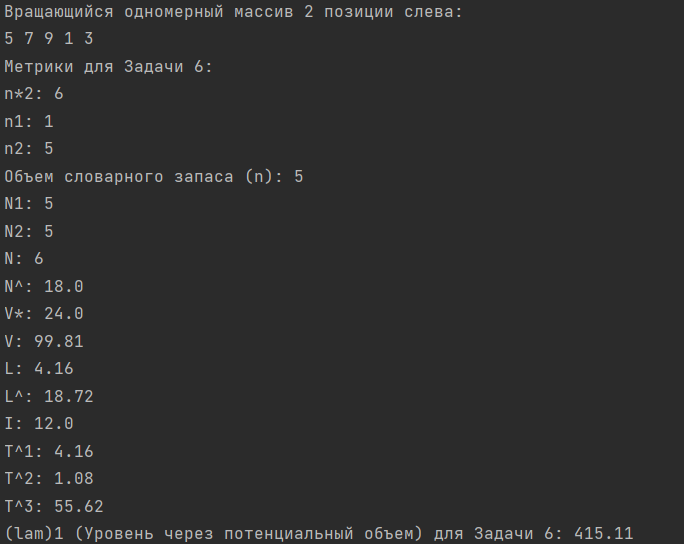


Рисунок 6. Метрика для программы Python

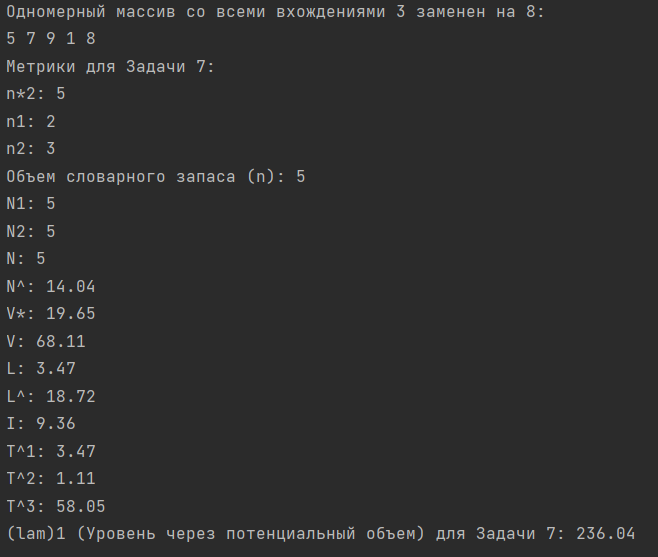


Рисунок 7. Метрика для программы Python

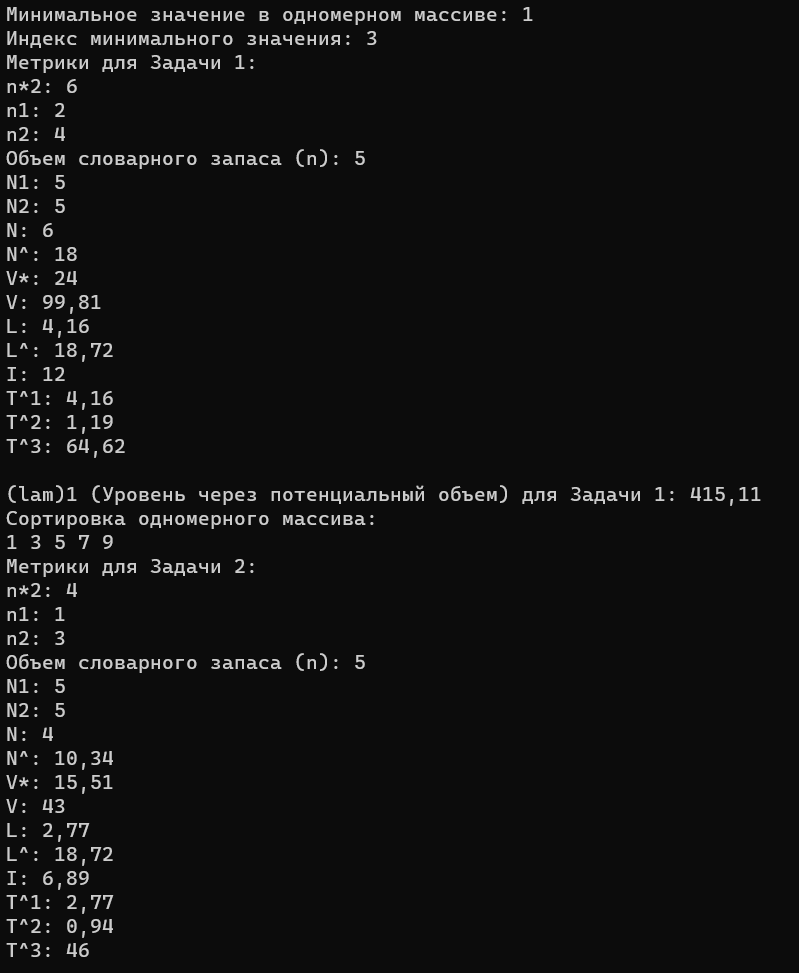
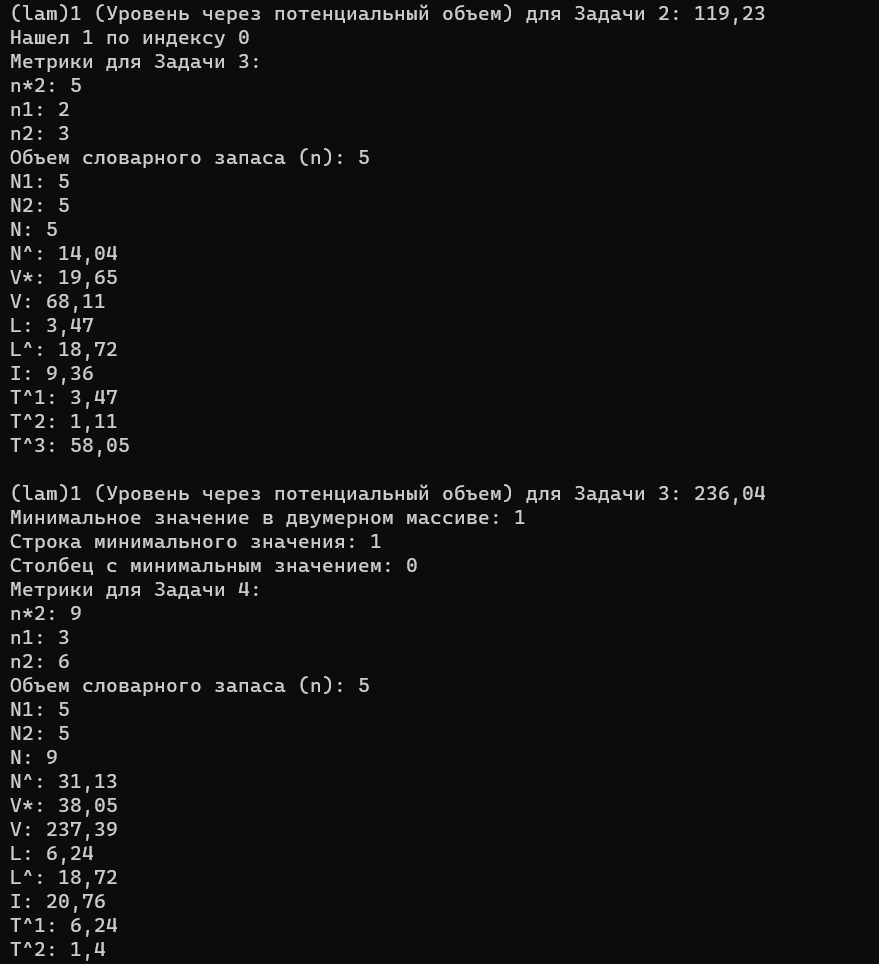
****

Рисунок 8. Метрика для программы С#

 Рисунок 9. Метрика для программы С#

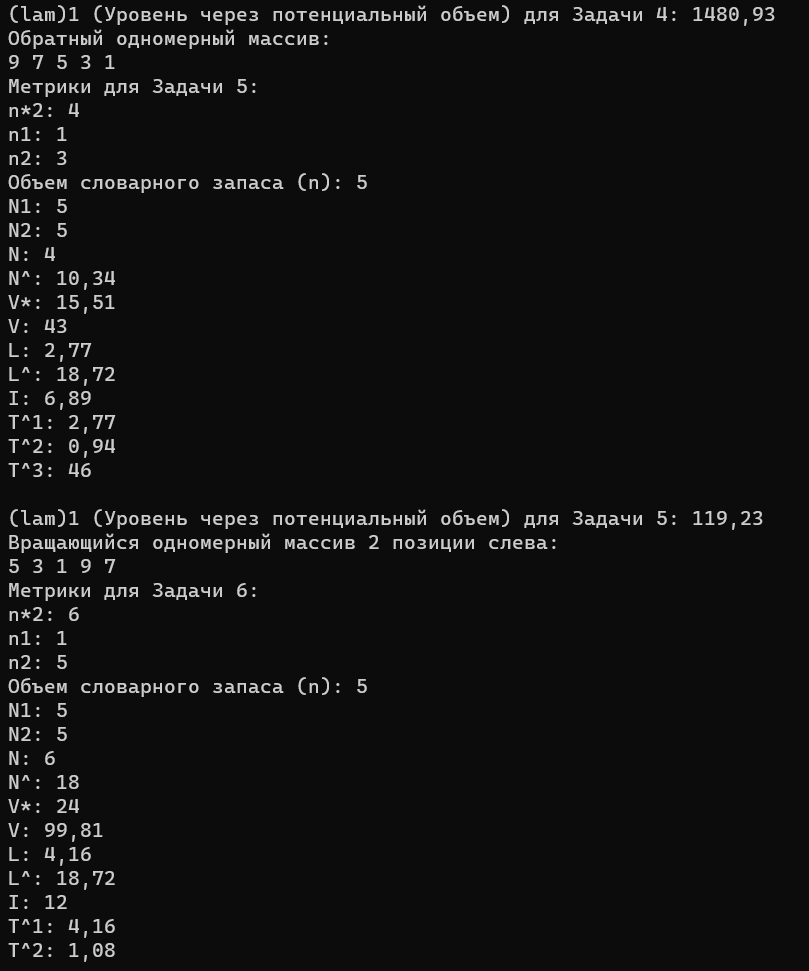
****

Рисунок 10. Метрика для программы C#

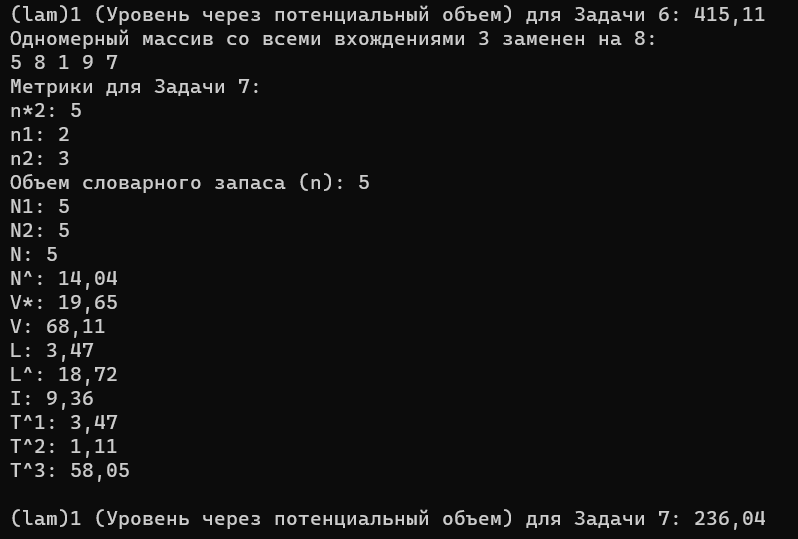
****

Рисунок 11. Метрика для программы C#

**3. Исходные тексты программ на языке Python**

main.py

import math  
  
  
def find\_minimum(array):  
 min\_value = 999999999  
 min\_index = -1  
  
 for i in range(len(array)):  
 if array[i] < min\_value:  
 min\_value = array[i]  
 min\_index = i  
  
 return min\_value, min\_index  
  
  
def buble\_sort(array):  
 for i in range(len(array) - 1):  
 for j in range(len(array) - 1 - i):  
 if array[j] > array[j + 1]:  
 temp = array[j]  
 array[j] = array[j + 1]  
 array[j + 1] = temp  
  
  
def binary\_search(sorted\_array, target):  
 left = 0  
 right = len(sorted\_array) - 1  
 while left <= right:  
  
 mid = int(left + (right - left) / 2)  
  
 if sorted\_array[mid] == target:  
 return mid  
  
 if sorted\_array[mid] < target:  
 left = mid + 1  
 else:  
 right = mid - 1  
  
 return -1  
  
  
def find\_minimum2d(array):  
 min\_value = 999999999  
 min\_row = -1  
 min\_col = -1  
 for row in range(len(array)):  
 for col in range(len(array[row])):  
 if array[row][col] < min\_value:  
 min\_value = array[row][col]  
 min\_row = row  
 min\_col = col  
 return min\_value, min\_row, min\_col  
  
  
def reverse\_array(array):  
 return array[-1]  
  
  
def rotate\_array\_left(array, positions):  
 n = len(array)  
 positions %= n  
  
 temp = [0] \* n  
  
 for i in range(n):  
 new\_position = (n - positions + i) % n  
 temp[new\_position] = array[i]  
  
 for i in range(n):  
 array[i] = temp[i]  
  
  
def replace\_all(array: [], old\_value, new\_value):  
 index = array.index(old\_value)  
 array[index] = new\_value  
  
  
def calculate\_metrics(task\_name, array, operands, operators):  
 eta\_star2 = len(operands) + len(operators)  
 eta1 = len(operators)  
 eta2 = len(operands)  
  
 vocabulary = dict()  
  
 for item in array:  
  
 if item not in vocabulary:  
 vocabulary[item] = 1  
 else:  
 vocabulary[item] += 1  
  
 N1 = 0  
 N2 = len(array)  
 for operand in vocabulary.keys():  
 N1 += vocabulary[operand]  
  
 N = eta1 + eta2  
 NHat = eta\_star2 \* math.log(2 + eta\_star2, 2)  
  
 VStar = (2 + eta\_star2) \* math.log(2 + eta\_star2, 2)  
 V = VStar \* N \* math.log(2, math.e)  
 L = V / VStar  
 LHat = (2 / 3.0) \* (N1 + N2) \* math.log(2 + (N1 + N2) / 2, 2)  
 I = (2 / 3.0) \* (N1 / N2) \* (eta1 + eta2) \* math.log(2 + (N1 / N2) \* (eta1 + eta2), 2)  
  
 T1Hat = V / VStar  
 T2Hat = (math.log(math.log(2 + eta1, 2) \* math.log(2 + eta2, 2), 2) / 2)  
 T3Hat = math.log(2 + eta1, 2) \* math.log(2 + eta2, 2) \* N1 \* N2 / 2  
  
 print("Метрики для " + task\_name + ":")  
 print(f"n\*2: {eta\_star2}")  
 print(f"n1: {eta1}")  
 print(f"n2: {eta2}")  
 print(f"Объем словарного запаса (n): {len(vocabulary)}")  
 print(f"N1: {N1}")  
 print(f"N2: {N2}")  
 print(f"N: {N}")  
 print(f"N^: {round(NHat, 2)}")  
 print(f"V\*: {round(VStar, 2)}")  
 print(f"V: {round(V, 2)}")  
 print(f"L: {round(L, 2)}")  
 print(f"L^: {round(LHat, 2)}")  
 print(f"I: {round(I, 2)}")  
 print(f"T^1: {round(T1Hat, 2)}")  
 print(f"T^2: {round(T2Hat, 2)}")  
 print(f"T^3: {round(T3Hat, 2)}")  
  
 lambda1 = L \* V  
  
 print(f"(lam)1 (Уровень через потенциальный объем) для {task\_name}: {round(lambda1, 2)}")  
  
  
def run():  
 oneDimensionalArray = [5, 3, 9, 1, 7]  
 twoDimensionalArray = [[5, 3, 9], [1, 7, 2]]  
  
 targetValue = 1  
  
 minResult1, minResult2 = find\_minimum(oneDimensionalArray)  
 print(f"Минимальное значение в одномерном массиве: {minResult1}")  
 print(f"Индекс минимального значения: {minResult2}")  
  
 task1Operands = ["minValue", "minIndex", "array", "i"]  
  
 task1Operators = ["for", "if"]  
  
 calculate\_metrics("Задачи 1", oneDimensionalArray, task1Operands, task1Operators);  
  
 buble\_sort(oneDimensionalArray);  
 print("Сортировка одномерного массива:")  
 for num in oneDimensionalArray:  
 print(num , end=" ")  
  
 task2Operands = ["temp", "i", "j"]  
  
 task2Operators = ["for"]  
  
 calculate\_metrics("Задачи 2", oneDimensionalArray, task2Operands, task2Operators)  
  
 binarySearchResult = binary\_search(oneDimensionalArray, targetValue)  
 if binarySearchResult != -1:  
 print(f"Нашел {targetValue} по индексу {binarySearchResult}")  
 else:  
 print(f"{targetValue} не найден в одномерном массиве")  
  
 task3Operands = ["left", "right", "mid"]  
 task3Operators = ["if", "while"]  
  
 calculate\_metrics("Задачи 3", oneDimensionalArray, task3Operands, task3Operators)  
  
 minResult2D1, minResult2D2, minResult2D3 = find\_minimum2d(twoDimensionalArray)  
 print(f"Минимальное значение в двумерном массиве: {minResult2D1}")  
 print(f"Строка минимального значения: {minResult2D2}")  
 print(f"Столбец с минимальным значением: {minResult2D3}")  
  
 task4Operands = ["minValue", "minRow", "minCol", "array", "row", "col"]  
  
 task4Operators = ["if", "for", "nested for"]  
  
 calculate\_metrics("Задачи 4", oneDimensionalArray, task4Operands, task4Operators)  
  
 reverse\_array(oneDimensionalArray)  
 print("Обратный одномерный массив:")  
 for num in oneDimensionalArray:  
 print(num, end=" ")  
 print("")  
  
 task5Operands = ["temp", "i", "j"]  
  
 task5Operators = ["for"]  
  
 calculate\_metrics("Задачи 5", oneDimensionalArray, task5Operands, task5Operators)  
  
 positions = 2  
 rotate\_array\_left(oneDimensionalArray, positions);  
 print(f"Вращающийся одномерный массив {positions} позиции слева:")  
 for num in oneDimensionalArray:  
 print(num, end=" ")  
 print("")  
  
 task6Operands = ["positions", "n", "temp", "i", "newPosition"]  
 task6Operators = ["for"]  
  
 calculate\_metrics("Задачи 6", oneDimensionalArray, task6Operands, task6Operators)  
  
 oldValue = 3  
 newValue = 8  
 replace\_all(oneDimensionalArray, oldValue, newValue)  
 print(f"Одномерный массив со всеми вхождениями {oldValue} заменен на {newValue}:")  
 for num in oneDimensionalArray:  
 print(num, end=" ")  
 print("")  
  
 task7Operands = ["oldValue", "newValue", "i"]  
 task7Operators = ["for", "if"]  
  
 calculate\_metrics("Задачи 7", oneDimensionalArray, task7Operands, task7Operators)

**4. Исходные тексты программ на языке C#**

Program.cs

using System;

using System.Linq;

using System.Collections.Generic;

namespace lab16

{

class Program

{

public static (int, int) FindMinimum(int[] array)

{

int minValue = int.MaxValue;

int minIndex = -1;

for (int i = 0; i < array.Length; i++)

{

if (array[i] < minValue)

{

minValue = array[i];

minIndex = i;

}

}

return (minValue, minIndex);

}

public static void BubbleSort(int[] array)

{

for (int i = 0; i < array.Length - 1; i++)

{

for (int j = 0; j < array.Length - 1 - i; j++)

{

if (array[j] > array[j + 1])

{

int temp = array[j];

array[j] = array[j + 1];

array[j + 1] = temp;

}

}

}

}

public static int BinarySearch(int[] sortedArray, int target)

{

int left = 0;

int right = sortedArray.Length - 1;

while (left <= right)

{

int mid = left + (right - left) / 2;

if (sortedArray[mid] == target)

return mid;

if (sortedArray[mid] < target)

left = mid + 1;

else

right = mid - 1;

}

return -1;

}

public static (int, int, int) FindMinimum2D(int[,] array)

{

int minValue = int.MaxValue;

int minRow = -1;

int minCol = -1;

for (int row = 0; row < array.GetLength(0); row++)

{

for (int col = 0; col < array.GetLength(1); col++)

{

if (array[row, col] < minValue)

{

minValue = array[row, col];

minRow = row;

minCol = col;

}

}

}

return (minValue, minRow, minCol);

}

public static void ReverseArray(int[] array)

{

Array.Reverse(array);

}

public static void RotateArrayLeft(int[] array, int positions)

{

int n = array.Length;

positions %= n;

int[] temp = new int[n];

for (int i = 0; i < n; i++)

{

int newPosition = (i - positions + n) % n;

temp[newPosition] = array[i];

}

for (int i = 0; i < n; i++)

{

array[i] = temp[i];

}

}

public static void ReplaceAll(int[] array, int oldValue, int newValue)

{

for (int i = 0; i < array.Length; i++)

{

if (array[i] == oldValue)

{

array[i] = newValue;

}

}

}

public static void CalculateMetrics(string taskName, int[] array, string[] operands, string[] operators)

{

int etaStar2 = operands.Length + operators.Length;

int eta1 = operators.Length;

int eta2 = operands.Length;

var vocabulary = new Dictionary<int, int>();

foreach (int item in array)

{

if (!vocabulary.ContainsKey(item))

{

vocabulary[item] = 1;

}

else

{

vocabulary[item] += 1;

}

}

int N1 = 0;

int N2 = array.Length;

foreach (int operand in vocabulary.Keys)

{

N1 += vocabulary[operand];

}

int N = eta1 + eta2;

double NHat = etaStar2 \* Math.Log(2 + etaStar2, 2);

double VStar = (2 + etaStar2) \* Math.Log(2 + etaStar2, 2);

double V = VStar \* N \* Math.Log(2, Math.E);

double L = V / VStar;

double LHat = (2 / 3.0) \* (N1 + N2) \* Math.Log(2 + (N1 + N2) / 2, 2);

double I = (2 / 3.0) \* (N1 / N2) \* (eta1 + eta2) \* Math.Log(2 + (N1 / N2) \* (eta1 + eta2), 2);

double T1Hat = V / VStar;

double T2Hat = (Math.Log(Math.Log(2 + eta1, 2) \* Math.Log(2 + eta2, 2), 2) / 2);

double T3Hat = Math.Log(2 + eta1, 2) \* Math.Log(2 + eta2, 2) \* N1 \* N2 / 2;

Console.WriteLine("Метрики для " + taskName + ":");

Console.WriteLine($"n\*2: {etaStar2}");

Console.WriteLine($"n1: {eta1}");

Console.WriteLine($"n2: {eta2}");

Console.WriteLine($"Объем словарного запаса (n): {vocabulary.Count}");

Console.WriteLine($"N1: {N1}");

Console.WriteLine($"N2: {N2}");

Console.WriteLine($"N: {N}");

Console.WriteLine($"N^: {Math.Round(NHat, 2)}");

Console.WriteLine($"V\*: {Math.Round(VStar, 2)}");

Console.WriteLine($"V: {Math.Round(V, 2)}");

Console.WriteLine($"L: {Math.Round(L, 2)}");

Console.WriteLine($"L^: {Math.Round(LHat, 2)}");

Console.WriteLine($"I: {Math.Round(I, 2)}");

Console.WriteLine($"T^1: {Math.Round(T1Hat, 2)}");

Console.WriteLine($"T^2: {Math.Round(T2Hat, 2)}");

Console.WriteLine($"T^3: {Math.Round(T3Hat, 2)}");

Console.WriteLine();

// Вычисление уровня программы через потенциальный объем (λ1)

double lambda1 = L \* V;

Console.WriteLine($"(lam)1 (Уровень через потенциальный объем) для {taskName}: {Math.Round(lambda1, 2)}");

}

public static void Main()

{

int[] oneDimensionalArray = { 5, 3, 9, 1, 7 };

int[,] twoDimensionalArray = { { 5, 3, 9 }, { 1, 7, 2 } };

int targetValue = 1;

// Выполнение задачи 1

var minResult = FindMinimum(oneDimensionalArray);

Console.WriteLine($"Минимальное значение в одномерном массиве: {minResult.Item1}");

Console.WriteLine($"Индекс минимального значения: {minResult.Item2}");

string[] task1Operands = { "minValue", "minIndex", "array", "i" };

string[] task1Operators = { "for", "if" };

CalculateMetrics("Задачи 1", oneDimensionalArray, task1Operands, task1Operators);

// Выполнение задачи 2

BubbleSort(oneDimensionalArray);

Console.WriteLine("Сортировка одномерного массива:");

foreach (var num in oneDimensionalArray)

{

Console.Write(num + " ");

}

Console.WriteLine();

string[] task2Operands = { "temp", "i", "j" };

string[] task2Operators = { "for" };

CalculateMetrics("Задачи 2", oneDimensionalArray, task2Operands, task2Operators);

// Выполнение задачи 3

int binarySearchResult = BinarySearch(oneDimensionalArray, targetValue);

if (binarySearchResult != -1)

{

Console.WriteLine($"Нашел {targetValue} по индексу {binarySearchResult}");

}

else

{

Console.WriteLine($"{targetValue} не найден в одномерном массиве");

}

string[] task3Operands = { "left", "right", "mid" };

string[] task3Operators = { "if", "while" };

CalculateMetrics("Задачи 3", oneDimensionalArray, task3Operands, task3Operators);

// Выполнение задачи 4

var minResult2D = FindMinimum2D(twoDimensionalArray);

Console.WriteLine($"Минимальное значение в двумерном массиве: {minResult2D.Item1}");

Console.WriteLine($"Строка минимального значения: {minResult2D.Item2}");

Console.WriteLine($"Столбец с минимальным значением: {minResult2D.Item3}");

string[] task4Operands = { "minValue", "minRow", "minCol", "array", "row", "col" };

string[] task4Operators = { "if", "for", "nested for" };

CalculateMetrics("Задачи 4", oneDimensionalArray, task4Operands, task4Operators);

// Выполнение задачи 5

ReverseArray(oneDimensionalArray);

Console.WriteLine("Обратный одномерный массив:");

foreach (var num in oneDimensionalArray)

{

Console.Write(num + " ");

}

Console.WriteLine();

string[] task5Operands = { "temp", "i", "j" };

string[] task5Operators = { "for" };

CalculateMetrics("Задачи 5", oneDimensionalArray, task5Operands, task5Operators);

// Выполнение задачи 6

int positions = 2;

RotateArrayLeft(oneDimensionalArray, positions);

Console.WriteLine($"Вращающийся одномерный массив {positions} позиции слева:");

foreach (var num in oneDimensionalArray)

{

Console.Write(num + " ");

}

Console.WriteLine();

string[] task6Operands = { "positions", "n", "temp", "i", "newPosition" };

string[] task6Operators = { "for" };

CalculateMetrics("Задачи 6", oneDimensionalArray, task6Operands, task6Operators);

// Выполнение задачи 7

int oldValue = 3;

int newValue = 8;

ReplaceAll(oneDimensionalArray, oldValue, newValue);

Console.WriteLine($"Одномерный массив со всеми вхождениями {oldValue} заменен на {newValue}:");

foreach (var num in oneDimensionalArray)

{

Console.Write(num + " ");

}

Console.WriteLine();

string[] task7Operands = { "oldValue", "newValue", "i" };

string[] task7Operators = { "for", "if" };

CalculateMetrics("Задачи 7", oneDimensionalArray, task7Operands, task7Operators);

}

}

}