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

Современные технологии программирования

Практическая работа №12

«Вычисление метрических характеристик реализаций алгоритмов»

Выполнил: студент 4 курса
группы ИП-111
Кузьменок Денис Витальевич

Проверил преподаватель:
Зайцев Михаил Георгиевич

Новосибирск, 2024 г.

Задание

Задание

1. Написать подпрограммы на двух языках программирования для решения следующих задач:

ЗАДАЧА
1. Отыскать минимальный элемент одномерного массива целых, его значение и значение его индекса.
2. Сортировка одномерного массива в порядке возрастания методом пузырька.
3. Бинарный поиск элемента в упорядоченном одномерном массиве.
4. Отыскать минимальный элемент двумерного массива целых, его значение и значение его индексов.
5. Осуществить перестановку значений элементов одномерного массива в обратном порядке.
6. Осуществлять циклический сдвиг элементов одномерного массива на заданное число позиций влево.
7. Заменить все вхождения целочисленного значения в целочисленный массив.

2. Для каждой подпрограммы вычислить следующие метрические характеристики:

- η^*_2 – число единых по смыслу входных и выходных параметров, представленных в сжатой без избыточной форме;
- η_1 - число отдельных операторов;
- η_2 - число отдельных операндов;
- η - длина словаря реализации;
- N_1 - общее число вхождений всех операторов в реализацию;
- N_2 - общее число вхождений всех операндов в реализацию;
- N - длина реализации;
- $\hat{N}$ - предсказанная длина реализации по соотношению Холстеда;
- V^* - потенциальный объем реализации:

$$V^* = (2+\eta_2^*) * \log_2(2+\eta_2^*).$$

- V - объем реализации:

$$V = N * \log_2 \eta.$$

- L - уровень программы через потенциальный объем:

$$L = V^* / V.$$

- $\hat{L}$ - уровень программы по реализации:

$$\hat{L} = (2/\eta_1) * (\eta_2 / N_2).$$

- I - интеллектуальное содержание программы:

$$I = (2/\eta_1) * (\eta_2 / N_2) * (N_1 + N_2) * \log_2(\eta_1 + \eta_2).$$

- $\hat{T}_1$ - прогнозируемое время написания программы, выраженное через потенциальный объем:

$$\hat{T} = \frac{V^2}{S * V^*}.$$

- $\hat{T}_2$ - прогнозируемое время написания программы, выраженное через длину реализации, найденную по Холстеду (т.е. в предположении, что программа совершенна):

$$\hat{T} = \frac{\eta_1 \times \mathrm{N}_2 \times (\eta_1 \log_2 \eta_1 + \eta_2 \log_2 \eta_2) \times \log_2 \eta}{2 \times S \times \eta_2}.$$

- $\hat{T}_3$ - прогнозируемое время написания программы, выраженное через метрические характеристики реализации:

$$\hat{T} = \frac{\eta_1 \times \mathrm{N}_2 \times \mathrm{N} \times \log_2 \eta}{2 \times S \times \eta_2}.$$

3. По всем реализациям алгоритмов определить средние значения уровней языков программирования λ:

$$\lambda_1 = \hat{L}^2 \times V,$$

$$\lambda_2 = \frac{V^{*2}}{V}.$$

Ход работы

Программа на языке C# показала следующие результаты:

η^*_2	η_1	η_2	η	N_1	N_2	N	N^	V*	V	L	I^	I	T^	T^	T^	λ_1	λ_2
2	8	8	16	17	23	40	48	8	160	0.05	0.0869565	13.913	320	220.8	184	1.20983	0.4
1	8	7	15	24	25	49	43.6515	4.75489	191.438	0.0248378	0.07	13.4006	770.752	243.631	273.482	0.938044	0.118101
3	12	9	21	21	25	46	71.5489	11.6096	202.047	0.0574602	0.06	12.1228	351.629	523.776	336.744	0.727368	0.667092
2	10	10	20	29	41	70	66.4386	8	302.535	0.0264432	0.0487805	14.7578	1144.09	588.643	620.197	0.719893	0.211546
1	7	6	13	14	19	33	35.1613	4.75489	122.115	0.0389379	0.0902256	11.0179	313.613	144.208	135.344	0.994092	0.185146
2	11	6	17	14	19	33	53.5635	8	134.886	0.0593092	0.0574163	7.74467	227.429	381.319	234.927	0.44467	0.474474
3	6	4	10	6	6	12	23.5098	11.6096	39.8631	0.291238	0.222222	8.85847	13.6875	35.144	17.9384	1.96855	3.38116

Программа на языке Python показала следующие результаты:

η^*_2	η_1	η_2	η	N_1	N_2	N	N^	V*	V	L	I^	I	T^	T^	T^	λ_1	λ_2
1	6	7	13	9	12	21	35.1613	4.7549	77.7092	0.0612	0.19444	15.1101	127.0004	66.9148	39.9647	2.9381	0.2909
1	4	6	10	7	7	14	23.5098	4.7549	46.5070	0.1022	0.4286	19.9317	45.4879	18.2228	10.8516	8.5421	0.4861
3	8	9	17	16	21	37	52.5293	11.6096	151.2361	0.0767	0.1071	16.2039	197.0118	200.3976	141.1537	1.7361	0.8912
2	7	9	16	13	18	31	48.1808	8.0	124.0	0.0645	0.1429	17.7143	192.2	134.9063	86.8	2.5306	0.5161
1	1	1	2	1	2	3	0.0	4.7549	3.0	1.5850	1.0	3.0	0.1893	0.0	0.3	3.0	7.5363
2	4	3	7	4	7	11	12.7549	8.0	30.8809	0.2590	0.2143	6.6173	11.9203	16.7102	14.4110	1.4180	2.0725
3	4	4	8	5	6	11	16.0	11.6096	33.0	0.3518	0.3333	11.0	9.3801	14.4	9.9	3.6667	4.0844

Вывод результатов программы на C#:

```
=================Проверка кода на C#=================
Запуск программы поиска минимального элемента в массиве и его индекса:

Предсказанная длина реализации по соотношению Холстеда (N^) = 48
Потенциальный объём реализации (V*) = 8
Объём реализации (V) = 160
Уровень программы = 0,05
Уровень программы по реализации (L^) = 0,0869565217391304
Интеллектуальное содержание программы (I) = 13,9130434782609
Прогназируемое время написания программы (T1) = 320
Прогназируемое время написания программы по Холстеду (T2) = 220,8
Прогназируемое время написания программы (T3) = 184
Среднее значение уровня языков программирования lamda1 = 1,20982986767486
Среднее значение уровня языков программирования lamda2 = 0,4

Запуск программы пузырьковой сортировки:

Предсказанная длина реализации по соотношению Холстеда (N^) = 43,6514844544032
Потенциальный объём реализации (V*) = 4,75488750216347
Объём реализации (V) = 191,437639184817
Уровень программы = 0,0248377880254416
Уровень программы по реализации (L^) = 0,07
Интеллектуальное содержание программы (I) = 13,4006347429372
Прогназируемое время написания программы (T1) = 770,751562050235
Прогназируемое время написания программы по Холстеду (T2) = 243,630820141799
Прогназируемое время написания программы (T3) = 273,482341692596
Среднее значение уровня языков программирования lamda1 = 0,938044432005606
Среднее значение уровня языков программирования lamda2 = 0,118100887863558
```

```
Запуск программы бинарного поиска:

Предсказанная длина реализации по соотношению Холстеда (N^) = 71,5488750216347
Потенциальный объём реализации (V*) = 11,6096404744368
Объём реализации (V) = 202,046601447823
Уровень программы = 0,0574602116108096
Уровень программы по реализации (L^) = 0,06
Интеллектуальное содержание программы (I) = 12,1227960868694
Прогназируемое время написания программы (T1) = 351,628711039786
Прогназируемое время написания программы по Холстеду (T2) = 523,775617229577
Прогназируемое время написания программы (T3) = 336,744335746372
Среднее значение уровня языков программирования lamda1 = 0,727367765212163
Среднее значение уровня языков программирования lamda2 = 0,667092398386559

Запуск программы поиска минимального элемента в двумерном массиве и его индекса:

Предсказанная длина реализации по соотношению Холстеда (N^) = 66,4385618977472
Потенциальный объём реализации (V*) = 8
Объём реализации (V) = 302,534966642115
Уровень программы = 0,0264432243611153
Уровень программы по реализации (L^) = 0,0487804878048781
Интеллектуальное содержание программы (I) = 14,7578032508349
Прогназируемое время написания программы (T1) = 1144,09257551432
Прогназируемое время написания программы по Холстеду (T2) = 588,642508862063
Прогназируемое время написания программы (T3) = 620,196681616337
Среднее значение уровня языков программирования lamda1 = 0,719892841504141
Среднее значение уровня языков программирования lamda2 = 0,211545794888923
```

```
Запуск программы переворачивания массива:

Предсказанная длина реализации по соотношению Холстеда (N^) = 35,1612594587302
Потенциальный объём реализации (V*) = 4,75488750216347
Объём реализации (V) = 122,114510698656
Уровень программы = 0,0389379400937632
Уровень программы по реализации (L^) = 0,0902255639097744
Интеллектуальное содержание программы (I) = 11,0178505893524
Прогназируемое время написания программы (T1) = 313,613176261
Прогназируемое время написания программы по Холстеду (T2) = 144,207600820385
Прогназируемое время написания программы (T3) = 135,34358269101
Среднее значение уровня языков программирования lamda1 = 0,994091782497963
Среднее значение уровня языков программирования lamda2 = 0,185145524711824

Запуск программы циклического сдвига:

Предсказанная длина реализации по соотношению Холстеда (N^) = 53,5635228093372
Потенциальный объём реализации (V*) = 8
Объём реализации (V) = 134,886273761261
Уровень программы = 0,0593092223316912
Уровень программы по реализации (L^) = 0,0574162679425837
Интеллектуальное содержание программы (I) = 7,74466643605328
Прогназируемое время написания программы (T1) = 227,428835614974
Прогназируемое время написания программы по Холстеду (T2) = 381,31860006744
Прогназируемое время написания программы (T3) = 234,926926800863
Среднее значение уровня языков программирования lamda1 = 0,44466984321837
Среднее значение уровня языков программирования lamda2 = 0,474473778653529
```

```
Запуск программы замены элемента в массиве на новое:

Предсказанная длина реализации по соотношению Холстеда (N^) = 23,5097750043269
Потенциальный объём реализации (V*) = 11,6096404744368
Объём реализации (V) = 39,8631371386484
Уровень программы = 0,291237501806674
Уровень программы по реализации (L^) = 0,222222222222222
Интеллектуальное содержание программы (I) = 8,85847491969963
Прогназируемое время написания программы (T1) = 13,687501400527
Прогназируемое время написания программы по Холстеду (T2) = 35,1440019411094
Прогназируемое время написания программы (T3) = 17,9384117123918
Среднее значение уровня языков программирования lamda1 = 1,96854998215547
Среднее значение уровня языков программирования lamda2 = 3,38116268864863
```

Вывод программы на Python:

```
=================Проверка кода на Python=================
Запуск программы поиска минимального элемента в массиве и его индекса:

Предсказанная длина реализации по соотношению Холстеда (N^) = 35.161259458730164
Потенциальный объём реализации (V*) = 4.754887502163468
Объём реализации (V) = 77.70923408096293
Уровень программы = 0.06118819157591353
Уровень программы по реализации (L^) = 0.19444444444444445
Интеллектуальное содержание программы (I) = 15.110128849076125
Прогназируемое время написания программы (T1) = 127.00037716354545
Прогназируемое время написания программы по Холстеду (T2) = 66.9148051067739
Прогназируемое время написания программы (T3) = 39.964748955923795
Среднее значение уровня языков программирования lamda1 = 2.9380806095425798
Среднее значение уровня языков программирования lamda2 = 0.2909429674042952
Запуск программы пузырьковой сортировки:
```

```
Предсказанная длина реализации по соотношению Холстеда (N^) = 23.509775004326936
Потенциальный объём реализации (V*) = 4.754887502163468
Объём реализации (V) = 46.50699332842307
Уровень программы = 0.10224026886849909
Уровень программы по реализации (L^) = 0.42857142857142855
Интеллектуальное содержание программы (I) = 19.931568569324174
Прогназируемое время написания программы (T1) = 45.48794114405168
Прогназируемое время написания программы по Холстеду (T2) = 18.222815821316008
Прогназируемое время написания программы (T3) = 10.85163177663205
Среднее значение уровня языков программирования lamda1 = 8.542100815424645
Среднее значение уровня языков программирования lamda2 = 0.48614097666065903
Запуск программы бинарного поиска:

Предсказанная длина реализации по соотношению Холстеда (N^) = 52.529325012980806
Потенциальный объём реализации (V*) = 11.60964047443681
Объём реализации (V) = 151.23612512626255
Уровень программы = 0.07676499556401796
Уровень программы по реализации (L^) = 0.10714285714285714
Интеллектуальное содержание программы (I) = 16.203870549242414
Прогназируемое время написания программы (T1) = 197.01183334289337
Прогназируемое время написания программы по Холстеду (T2) = 200.39755312875297
Прогназируемое время написания программы (T3) = 141.15371678451172
Среднее значение уровня языков программирования lamda1 = 1.7361289874188301
Среднее значение уровня языков программирования lamda2 = 0.8912139995199851
Запуск программы поиска минимального элемента в двумерном массиве и его индекса:
```

```
Предсказанная длина реализации по соотношению Холстеда (N^) = 48.18080946738404
Потенциальный объём реализации (V*) = 8.0
Объём реализации (V) = 124.0
Уровень программы = 0.06451612903225806
Уровень программы по реализации (L^) = 0.14285714285714285
Интеллектуальное содержание программы (I) = 17.71428571428571
Прогназируемое время написания программы (T1) = 192.2
Прогназируемое время написания программы по Холстеду (T2) = 134.90626650867532
Прогназируемое время написания программы (T3) = 86.8
Среднее значение уровня языков программирования lamda1 = 2.530612244897959
Среднее значение уровня языков программирования lamda2 = 0.5161290322580645
Запуск программы переворачивания массива:

Предсказанная длина реализации по соотношению Холстеда (N^) = 0.0
Потенциальный объём реализации (V*) = 4.754887502163468
Объём реализации (V) = 3.0
Уровень программы = 1.584962500721156
Уровень программы по реализации (L^) = 1.0
Интеллектуальное содержание программы (I) = 3.0
Прогназируемое время написания программы (T1) = 0.18927892607143723
Прогназируемое время написания программы по Холстеду (T2) = 0.0
Прогназируемое время написания программы (T3) = 0.3
Среднее значение уровня языков программирования lamda1 = 3.0
Среднее значение уровня языков программирования lamda2 = 7.536318386076782
Запуск программы циклического сдвига:
```

```
Предсказанная длина реализации по соотношению Холстеда (N^) = 12.754887502163468
Потенциальный объём реализации (V*) = 8.0
Объём реализации (V) = 30.880904142633646
Уровень программы = 0.2590597724421979
Уровень программы по реализации (L^) = 0.21428571428571427
Интеллектуальное содержание программы (I) = 6.617336601992925
Прогназируемое время написания программы (T1) = 11.9203780083316
Прогназируемое время написания программы по Холстеду (T2) = 16.71016489776183
Прогназируемое время написания программы (T3) = 14.4110885998957
Среднее значение уровня языков программирования lamda1 = 1.418000700427055
Среднее значение уровня языков программирования lamda2 = 2.0724781795375833
Запуск программы замены элемента в массиве на новое:

Предсказанная длина реализации по соотношению Холстеда (N^) = 16.0
Потенциальный объём реализации (V*) = 11.60964047443681
Объём реализации (V) = 33.0
Уровень программы = 0.35180728710414577
Уровень программы по реализации (L^) = 0.3333333333333333
Интеллектуальное содержание программы (I) = 11.0
Прогназируемое время написания программы (T1) = 9.380135434838502
Прогназируемое время написания программы по Холстеду (T2) = 14.4
Прогназируемое время написания программы (T3) = 9.9
Среднее значение уровня языков программирования lamda1 = 3.6666666666666665
Среднее значение уровня языков программирования lamda2 = 4.084356119566102
```

Вывод:

Лабораторная работа позволила глубже понять процесс разработки программных модулей на разных языках программирования, а также оценить их сложность и прогнозируемое время разработки с помощью метрик Холстеда. Методы оценки сложности программного кода являются полезными инструментами для оценки эффективности программного обеспечения, а также для планирования времени разработки и оптимизации кода.

Листинг программ

Program.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Runtime.Remoting.Messaging;
using System.Text;
using System.Threading.Tasks;

namespace lab12
{
    class Program
    {
        private static void Main(string[] args)
        {
            Console.WriteLine("=================Проверка кода на
C#=================");
            Console.WriteLine("Запуск программы поиска минимального элемента в
массиве и его индекса:\n");
            Calculate(2, 8, 8, 17, 23);
            Console.WriteLine("Запуск программы пузырьковой сортировки:\n");
            Calculate(1, 8, 7, 24, 25);
            Console.WriteLine("Запуск программы бинарного поиска:\n");
            Calculate(3, 12, 9, 21, 25);
            Console.WriteLine("Запуск программы поиска минимального элемента в
двумерном массиве и его индекса:\n");
            Calculate(2, 10, 10, 29, 41);
            Console.WriteLine("Запуск программы переворачивания массива:\n");
            Calculate(1, 7, 6, 14, 19);
            Console.WriteLine("Запуск программы циклического сдвига:\n");
            Calculate(2, 11, 6, 14, 19);
            Console.WriteLine("Запуск программы замены элемента в массиве на
новое:\n");
            Calculate(3, 6, 4, 6, 6);
        }

        private static void Calculate(double udot, double u1, double u2, double
N1, double N2)
        {
            int S = 10;
            double V = (N1 + N2) * Math.Log((u1 + u2), 2);
            double Vdot = (2 + udot) * Math.Log((2 + udot), 2);
            double Lgalka = (2 / u1) * (u2 / N2);

            Console.WriteLine($"Предсказанная длина реализации по соотношению
Холстеда (N^) = {u1 * Math.Log(u1, 2) + u2 * Math.Log(u2, 2)}");
            Console.WriteLine($"Потенциальный объём реализации (V*) = {Vdot}");
            Console.WriteLine($"Объём реализации (V) = {V}");
            Console.WriteLine($"Уровень программы = {Vdot / V}");
            Console.WriteLine($"Уровень программы по реализации (L^) =
{Lgalka}");
            Console.WriteLine($"Интеллектуальное содержание программы (I) = {(2
/ u1) * (u2 / N2) * (N1 + N2) * Math.Log((u1 + u2), 2)}");
            Console.WriteLine($"Прогназируемое время написания программы (T1) =
{(V * V / (S * Vdot))}");
```

```
            Console.WriteLine($"Прогназируемое время написания программы по
Холстеду (T2) = {(u1 * N2 * (u1 * Math.Log(u1, 2) + u2 * Math.Log(u2, 2)) *
Math.Log((u1 + u2), 2)) / (2 * S * u2)}");
            Console.WriteLine($"Прогназируемое время написания программы (T3) =
{(u1 * N2 * (N1 + N2) * Math.Log((u1 + u2), 2)) / (2 * S * u2)}");
            Console.WriteLine($"Среднее значение уровня языков программирования
lamda1 = {Lgalka * Lgalka * V}");
            Console.WriteLine($"Среднее значение уровня языков программирования
lamda2 = {Vdot * Vdot / V}\n");
        }

        private static (int , int) FindMin(List<int> a)
        {
            if (a.Count == 0) return (int.MaxValue, -1);

            int minVal = a[0];
            int minIndex = 0;

            for(int i = 1; i < a.Count; i++)
            {
                if (a[i] < minVal)
                {
                    minVal = a[i];
                    minIndex = i;
                }
            }

            return (minVal, minIndex);
        }

        private static void BubbleSort(List<int> a)
        {
            for(int i = 0; i <  a.Count - 1; i++)
            {
                for(int j = a.Count - 1; j > i; j--)
                {
                    if (a[j] < a[j - 1])
                    {
                        int temp = a[j];
                        a[j] = a[j - 1];
                        a[j - 1] = temp;
                    }
                }
            }
        }

        private static int BinarySearch(List<int> a, int target)
        {
            int left = 0, right = a.Count - 1;

            while(left <= right)
            {
                int mid = (left + right) / 2;
                if (a[mid] == target) return mid;
                if (a[mid] < target) left = mid + 1;
                else right = mid - 1;
            }

            return -1;
```

```
        }

        private static (int, int, int) FindMinMatrix(List<List<int>> a)
        {
            if (a.Count == 0 || a[0].Count == 0) return (int.MaxValue, -1, 1);

            int minVal = a[0][0], minRow = 0, minCol = 0;

            for(int i = 0; i < a.Count; i++)
            {
                for(int j = 0; j < a[i].Count; j++)
                {
                    if (a[i][j] < minVal)
                    {
                        minVal = a[i][j];
                        minRow = i;
                        minCol = j;
                    }
                }
            }

            return (minVal, minRow, minCol);
        }

        private static void Reverse(List<int> a)
        {
            int left = 0, right = a.Count - 1;

            while(left < right)
            {
                int temp = a[left];
                a[left] = a[right];
                a[right] = temp;
                left++;
                right--;
            }
        }

        private static void CycleShift(List<int> a, int count)
        {
            count %= a.Count;
            if(count == 0) return;

            List<int> temp = new List<int>(a);
            for(int i = 0; i < a.Count; i++)
            {
                a[i] = temp[(i + count) % a.Count];
            }
        }

        private static void ReplaceValue(List<int> a, int oldValue, int
newValue)
        {
            for(int i = 0; i < a.Count; i++)
            {
                if (a[i] == oldValue)
                {
                    a[i] = newValue;
                }
```

```
                    }
                }
            }
        }
```

Main.py

```
import math

def calculate(udot, u1, u2, N1, N2):
    S = 10
    V = (N1 + N2) * math.log2(u1 + u2)
    Vdot = (2 + udot) * math.log2(2 + udot)
    Lgalka = (2 / u1) * (u2 / N2)

    print(f"Предсказанная длина реализации по соотношению Холстеда (N^) = {u1 * math.log2(u1) + u2 * math.log2(u2)}")
    print(f"Потенциальный объём реализации (V*) = {Vdot}")
    print(f"Объём реализации (V) = {V}")
    print(f"Уровень программы = {Vdot / V}")
    print(f"Уровень программы по реализации (L^) = {Lgalka}")
    print(f"Интеллектуальное содержание программы (I) = {(2 / u1) * (u2 / N2) * (N1 + N2) * math.log2(u1 + u2)}")
    print(f"Прогназируемое время написания программы (T1) = {V * V / (S * Vdot)}")
    print(f"Прогназируемое время написания программы по Холстеду (T2) = {(u1 * N2 * (u1 * math.log2(u1) + u2 * math.log2(u2)) * math.log2(u1 + u2)) / (2 * S * u2)}")
    print(f"Прогназируемое время написания программы (T3) = {(u1 * N2 * (N1 + N2) * math.log2(u1 + u2)) / (2 * S * u2)}")
    print(f"Среднее значение уровня языков программирования lamda1 = {Lgalka * Lgalka * V}")
    print(f"Среднее значение уровня языков программирования lamda2 = {Vdot * Vdot / V}")


def find_min(a):
    if not a:
        return (float('inf'), -1)

    min_val = a[0]
    min_index = 0
    for i in range(1, len(a)):
        if a[i] < min_val:
            min_val = a[i]
            min_index = i
    return (min_val, min_index)


def bubble_sort(a):
    n = len(a)
    for i in range(n-1):
        for j in range(n-i-1):
            if a[j] > a[j+1]:
                a[j], a[j+1] = a[j+1], a[j]


def binary_search(a, target):
    left, right = 0, len(a) - 1
    while left <= right:
        mid = (left + right) // 2
        if a[mid] == target:
            return mid
```

```
        elif a[mid] < target:
            left = mid + 1
        else:
            right = mid - 1
    return -1


def find_min_matrix(a):
    if not a or not a[0]:
        return (float('inf'), -1, -1)

    min_val = a[0][0]
    min_row, min_col = 0, 0
    for i in range(len(a)):
        for j in range(len(a[i])):
            if a[i][j] < min_val:
                min_val = a[i][j]
                min_row, min_col = i, j
    return (min_val, min_row, min_col)

def reverse(a):
    a.reverse()


def cycle_shift(a, positions):
    positions %= len(a)
    if positions == 0:
        return
    a[:] = a[-positions:] + a[:-positions]

def replace_value(a, old_value, new_value):
    for i in range(len(a)):
        if a[i] == old_value:
            a[i] = new_value

if __name__ == "__main__":
    print("=================Проверка кода на Python================")
    print("Запуск программы поиска минимального элемента в массиве и его
индекса:\n")
    calculate(1, 6, 7, 9, 12)

    print("Запуск программы пузырьковой сортировки:\n")
    calculate(1, 4, 6, 7, 7)

    print("Запуск программы бинарного поиска:\n")
    calculate(3, 8, 9, 16, 21)

    print("Запуск программы поиска минимального элемента в двумерном массиве и
его индекса:\n")
    calculate(2, 7, 9, 13, 18)

    print("Запуск программы переворачивания массива:\n")
    calculate(1, 1, 1, 1, 2)

    print("Запуск программы циклического сдвига:\n")
    calculate(2, 4, 3, 4, 7)

    print("Запуск программы замены элемента в массиве на новое:\n")
    calculate(3, 4, 4, 5, 6)
```