МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное автономное образовательное учреждение высшего образования
"Омский государственный технический университет"

Кафедра «Автоматизированные системы обработки информации и управления»

ПРАКТИЧЕСКАЯ РАБОТА №2

по дисциплине

«Измерительные средства аналитики программных систем и технологий»

по теме

«Метрики Мак-Кейба.»

Выполнил: Шмидт А.В.
гр. ИВТ - 244
Проверил: Зубарев А.А.
к.т.н., доцент

Омск 2025

ЗАДАНИЕ

1. Выбрать программу, в которой ОБЯЗАТЕЛЬНО должны быть ВЕТВЛЕНИЯ, или ЦИКЛЫ (или то и другое). Представить графически схему алгоритма программы.
2. Представить поток управления в виде графа, сделав оценку свойств передачи управления в программе («дерево обязательного предшествования»).
3. Определить цикломатическую сложность графа потока управления.
4. Сделать выводы по работе.

ПРИМЕР ОПРЕДЕЛЕНИЯ МЕТРИК ПО МАК-КЕЙБУ

Программа на C++ считывает количество элементов массива, запрашивает у пользователя значения и сохраняет их во внутренний динамический массив. Для каждого числа программа определяет знак, чётность и вычисляет сумму его цифр с помощью отдельной подпрограммы. В процессе обработки формируются статистические показатели: количество положительных, отрицательных, чётных и нечётных элементов. После завершения цикла программа выводит сводную статистику и сравнивает, каких чисел больше — положительных или отрицательных. В финале консоль ожидает нажатия клавиши, что предотвращает её мгновенное закрытие. Ниже представлен листинг программы.

```
#include <iostream>
#include <vector>
using namespace std;

int sumOfDigits(int x) {
    x = abs(x);
    int s = 0;
    while (x > 0) {
        s += x % 10;
        x /= 10;
    }
    return s;
}

int main() {
    int n;
    cout << "Enter amount: ";
    cin >> n;

    vector<int> arr(n);

    for (int i = 0; i < n; i++) {
        cout << "Enter element " << i + 1 << ": ";
        cin >> arr[i];
    }

    int positiveCount = 0;
    int negativeCount = 0;
    int evenCount = 0;
    int oddCount  = 0;

    for (int i = 0; i < n; i++) {
        int val = arr[i];

        if (val > 0) {
            positiveCount++;
        } else if (val < 0) {
            negativeCount++;
```

```
        }

        if (val % 2 == 0) {
            evenCount++;
        } else {
            oddCount++;
        }

        int s = sumOfDigits(val);
        cout << "The sum of digits of " << val << " = " << s << "\n";
    }

    cout << "Positive numbers: " << positiveCount << "\n";
    cout << "Negative numbers: " << negativeCount << "\n";
    cout << "Even numbers: " << evenCount << "\n";
    cout << "Odd numbers: " << oddCount << "\n";


    cout << "\nPress Enter to exit...";
    cin.ignore();
    cin.get();
    return 0;
}
```

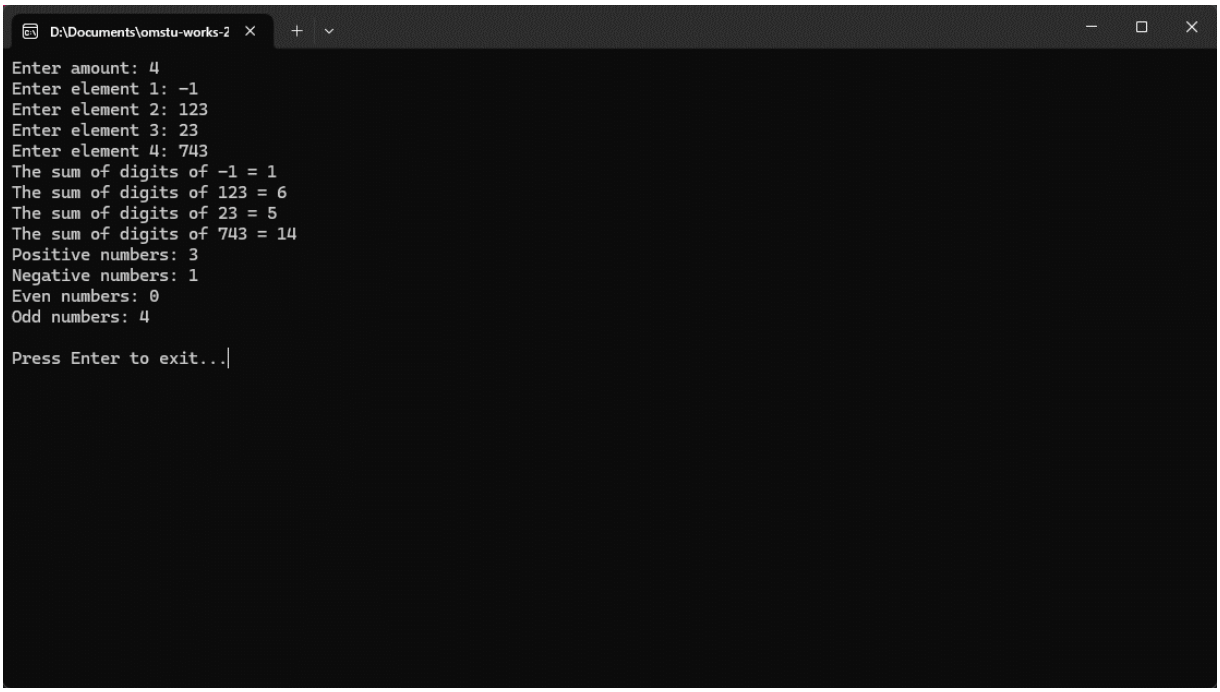

Рисунок 1 – Результат работы программы.

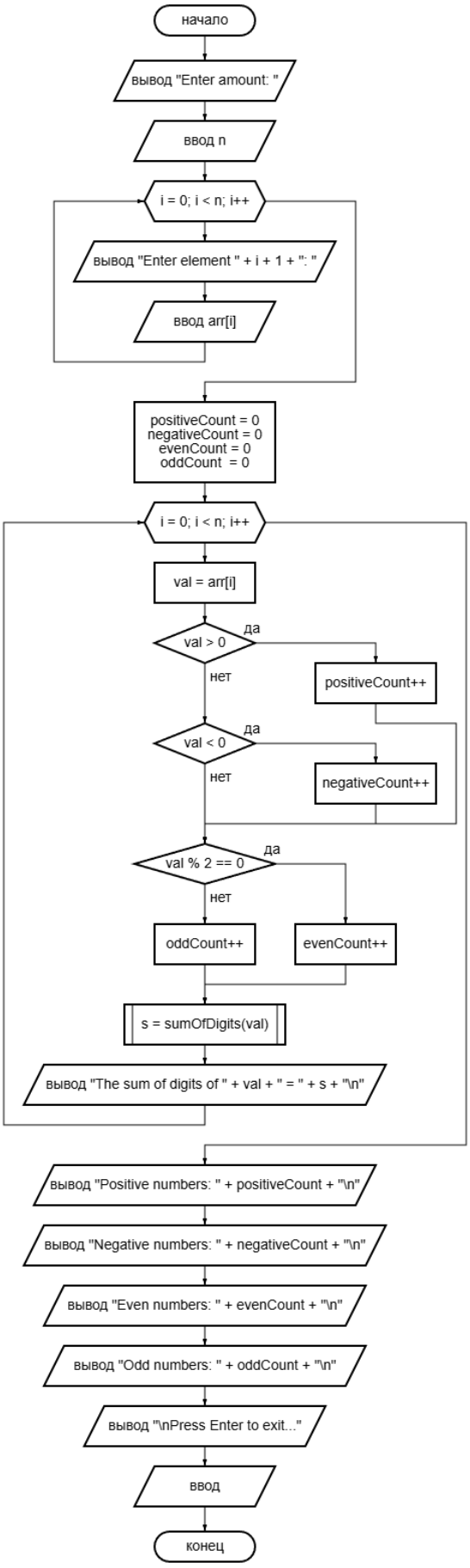

Рисунок 2 – Схема алгоритма главной функции.

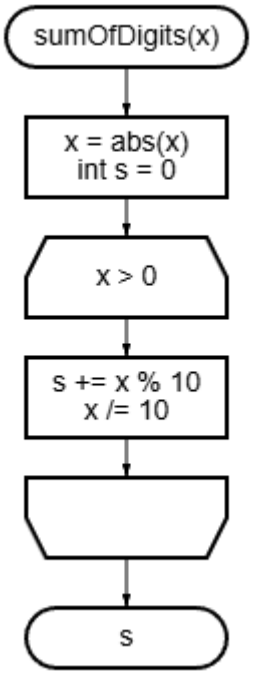

Рисунок 3 – Схема алгоритма подпрограммы для подсчёта суммы цифр.

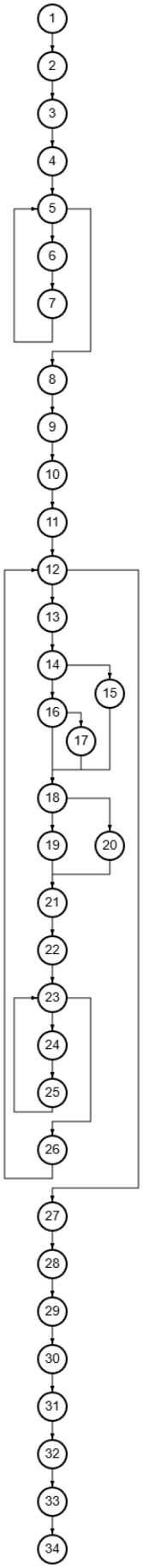

Рисунок 4 – Граф потока управления.

Таблица 1 – Пояснение к шагам графа.

Номер шага графа	Действие шага
1	int n;
2	cout << "Enter amount: ";
3	cin >> n;
4	vector<int> arr(n);
5	for (int i = 0; i < n; i++)
6	cout << "Enter element " << i + 1 << ": ";
7	cin >> arr[i];
8	int positiveCount = 0;
9	int negativeCount = 0;
10	int evenCount = 0;
11	int oddCount = 0;
12	for (int i = 0; i < n; i++)
13	int val = arr[i];
14	if (val > 0)
15	positiveCount++;
16	else if (val < 0)
17	negativeCount++;
18	if (val % 2 == 0)
19	evenCount++;
20	oddCount++;
21	x = abs(x);
22	int s = 0;
23	while (x > 0)
24	s += x % 10;
25	x /= 10;
26	cout << "The sum of digits of " << val << " = " << s << "\n";
27	cout << "Positive numbers: " << positiveCount << "\n";
28	cout << "Negative numbers: " << negativeCount << "\n";
29	cout << "Even numbers: " << evenCount << "\n";
30	cout << "Odd numbers: " << oddCount << "\n";
31	cout << "\nPress Enter to exit...";
32	cin.ignore();
33	cin.get();

Продолжение таблицы 2 – Пояснение к шагам графа.

34	return 0;

Формула для вычисления цикломатического числа Мак-Кейба (цикломатической сложности) Z(G):

Z(G) = e - v + 2p, где:

e - количество рёбер в графе потока управления;

v – количество узлов (вершин);

p – количество связных компонент (здесь p = 1 — одна программа).

В нашем случае имеем:

Z(G) = 38 - 34 + 2*1 = 6

ВЫВОД

В ходе работы проанализирована программа на C++, которая принимает набор чисел, определяет для каждого знак, чётность и сумму цифр, подсчитывает статистику по введённым значениям и выводит итоговое сравнение. Построен граф потока управления. По формуле Мак-Кейба $Z(G) = e - v + 2p$ цикломатическая сложность программы составила 6 ($p = 1$).

Значение не превышает порог 10, что указывает на нормальную сложность и программа не требует рефакторинга.

Метрика Мак-Кейба подтвердила свою актуальность: она наглядно показала минимальное количество тестовых прогонов (6), необходимых для полного покрытия всех ветвей и контуров программы, и позволила объективно оценить структурную сложность кода.