«Отчет по ПЗ 1.1. Динамические массивы (одномерные)» Фаустова Анастасия \*\*\*. Вариант № 20.

1. Задание: Найти отдельные суммы *S*1, *S*2 и количества *n*1, *n*2 элементов *Xk*, значения которых соответственно больше *A* и меньше *–A*.

1. Постановка задачи.

Входные данные + ограничения:

А) Переменная N. Тип int. Размер массива (кол-во чисел).

Исключения: буква, буква с цифрой, строка, вещественное, меньше или равно 0, большие числа (>100).

Б) Переменные matrix. Тип \*float[]. Динамический массив чисел.

Исключения: буква, буква с цифрой, строка.

В) Переменные A. Тип float. Сравниваемое число.

Исключения: буква, буква с цифрой, строка.

Выходные данные + ограничения:

А) Переменная sum1. Тип float. Сумма чисел больше A.

Исключения: ограничения значения заданными пределами операционной системы устройства для типа float.

Б) Переменная sum2. Тип float. Сумма чисел меньше -A.

Исключения: ограничения значения заданными пределами операционной системы устройства для типа float.

В) Переменная n1. Тип int. Количество чисел меньше -A.

Исключения: ограничения значения заданными пределами операционной системы устройства для типа int.

Г) Переменная n2. Тип int. Количество чисел меньше -A.

Исключения: ограничения значения заданными пределами операционной системы устройства для типа int.

Д) Переменная matrix[i]. Тип float. Элементы массива.

Исключения: буква, буква с цифрой, строка.

Промежуточные данные + ограничения:

А) Переменная key. Тип int. Флаг переменная для сигнализирования наличия ошибок.

Исключения: буква, буква с цифрой, строка, вещественное число, числа не равные 0 или 1.

Б) Переменная N. Тип float. Промежуточная переменная для проверки на целостность размеров массивов.

Исключения: буква, буква с цифрой, строка.

Б) Переменная a. Тип char. Промежуточная переменная для проверки на отсутствие буквенных значений в размерах массивов

Исключения: нет.

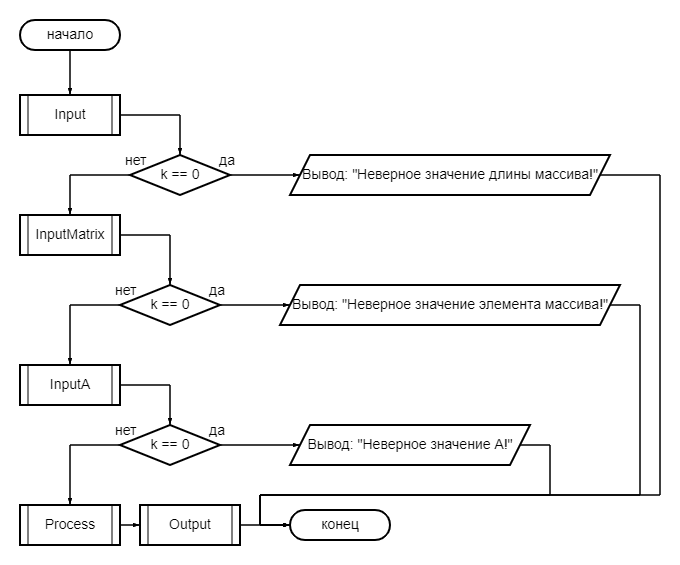
В) Переменная Z. Тип char. Промежуточная переменная для проверки на отсутствие буквенных значений в размерах массивов

Исключения: нет. .

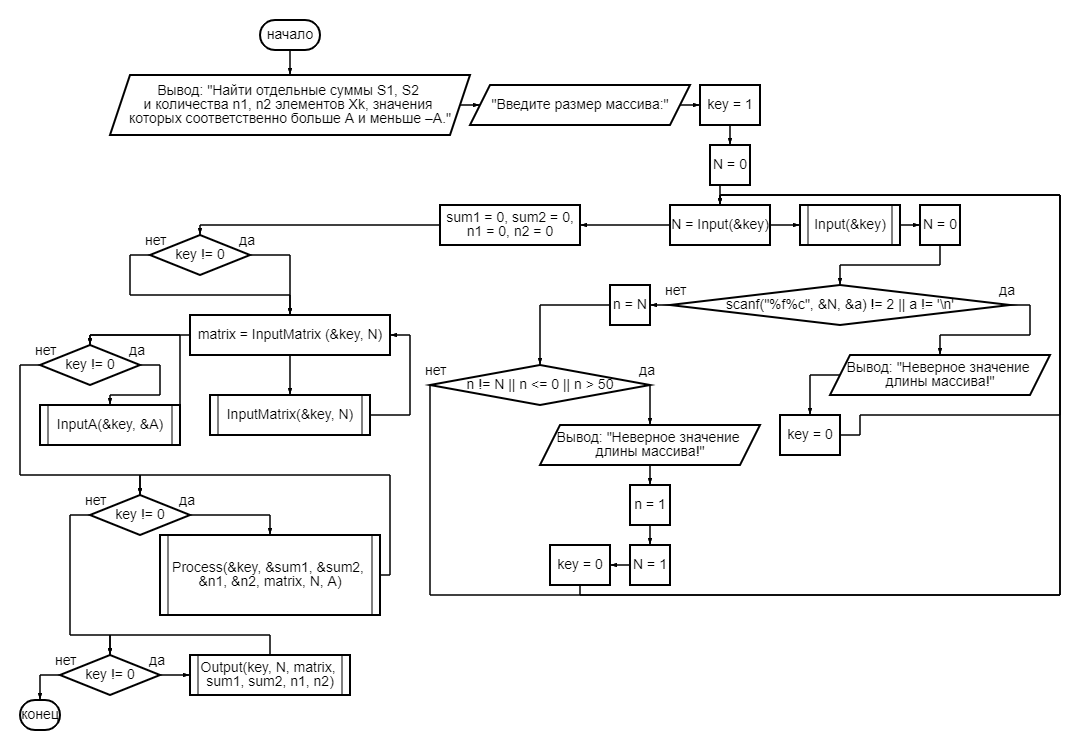
Среда разработки: Visual Studio 2022.

2. Структура программы.

2.1 Структура программы



2.2 Блок-схема алгоритма



3. Тесты.

|  |  |  |
| --- | --- | --- |
| Входные данные | Результат программы | Смысл |
| n = w | Неверное значение длины массива! | Проверка на букву |
| n = 3asa | Неверное значение длины массива! | Проверка на число c буквой |
| n = 3,8 | Неверное значение длины массива! | Проверка на целое число |
| n = 99999999999 | Неверное значение длины массива! | Выделение памяти массивам |
| n = 0 | Неверное значение длины массива! | Валидность размера массива |
| n = bebra | Неверное значение длины массивов! | Проверка на строку |
| matrix[i] = d | Неверное значение элемента массива! | Проверка на букву |
| matrix[i] = 3a | Неверное значение элемента массива! | Проверка на число c буквой |
| matrix[i] = dsadas | Неверное значение элемента массива! | Проверка на строку |
| A = d | Неверное значение A! | Проверка на букву |
| A = 3d | Неверное значение A! | Проверка на число c буквой |
| A = dsaklsa | Неверное значение A! | Проверка на строку |
| n = 1  matrix[i] = 1  A = 1 | Элементы вашего массива:  Элемента массива [0] = 1  Сумма элементов > A = 0  Сумма элементов <= -A = 0  Количество элементов >= A = 0  Количество элементов <= -A = 0 | Проверка на корректность работы в случае, когда нет подходящих элементов |
| n = 3  matrix[i] = 1  matrix[i+1] = 2  matrix[i+2] = 3  A = 2 | Элементы вашего массива:  Элемента массива [0] = 1  Элемента массива [1] = 2  Элемента массива [2] = 3  Сумма элементов > A = 3  Сумма элементов <= -A = 0  Количество элементов >= A = 1  Количество элементов <= -A = 0 | Проверка корректности работы программы при наличии положителных чисел |
| n = 3  matrix[i] = 1  matrix[i+1] = -2  matrix[i+2] = 3  A = 1 | Элементы вашего массива:  Элемента массива [0] = 1  Элемента массива [1] = -2  Элемента массива [2] = 3  Сумма элементов > A = 3  Сумма элементов <= -A = -2  Количество элементов >= A = 1  Количество элементов <= -A = 1 | Проверка корректности работы программы при наличии отрицательных и положительных чисел |

4. Код:

// ConsoleApplication20.cpp : Этот файл содержит функцию "main". Здесь начинается и заканчивается выполнение программы.

#define \_CRT\_SECURE\_NO\_WARNINGS // Директива препроцесса для устранения предупреждений

#include <iostream>

#include <stdio.h>

using namespace std; // Использование стандартного пространства имён

int Input(int\*); // Прототип функции ввода размера массива

float\* InputMatrix(int\*, int); // Прототип функции ввода элементов массива

void InputA(int\*, float\*); // Прототип функции ввода сравниваемого числа

void Process(int\*, float\*, float\*, int\*, int\*, float\*, int, float); // Прототип функции обработки массива

void Output(int, int, float\*, float, float, int, int); // Прототип функции вывода результатов

int main()

{

setlocale(LC\_ALL, "RUS"); // Установка для консоли русского языка

cout << "\tНайти отдельные суммы S1, S2 и количества n1, n2 элементов Xk,\n";

cout << "\tзначения которых соответственно больше A и меньше –A.\n\n";

cout << "Введите размер массива:\n";

int key = 1; // Временная переменная-ключ

int N = 0; // Переменная - размер массива

N = Input(&key); // Вызов функции ввода размера массива и последующие присваивание результата в переменную отвечающую за размер массива

float sum1 = 0, sum2 = 0; // Переменные: сумма чисел > A, сумма чисел < A

int n1 = 0, n2 = 0; // Количество чисел > A, количество чисел < A

float\* matrix = new float[N]; // Создание одномерного динамического массива

if (key != 0) // Проверка наличия ошибок на предыдущих стадиях

{

matrix = InputMatrix(&key, N); // Вызов функции заполнения матрицы

}

float A; // Сравниваемое число

if (key != 0) // Проверка наличия ошибок на предыдущих стадиях

{

InputA(&key, &A); // Вызов функции ввода сравниваемого числа

}

if (key != 0) // Проверка наличия ошибок на предыдущих стадиях

{

Process(&key, &sum1, &sum2, &n1, &n2, matrix, N, A); // Вызов функции вычислений

}

if (key != 0) // Проверка наличия ошибок на предыдущих стадиях

{

Output(key, N, matrix, sum1, sum2, n1, n2); // Вызов функции вывода результатов программы

}

return 0; // Сообщение процессору об удачной работе программы

}

int Input(int\* key) // Описание функции ввода размера массива

{

int n; // Размер массивов

float N = 0; // Промежуточная переменная для проверки на целостность размеров массивов

char a; // Промежуточная переменная для проверки на отсутствие буквенных значений в раз-мерах массивов

if (scanf("%f%c", &N, &a) != 2 || a != '\n') // Считывание размера массива и одновременная проверка на буквенное значение

{

cout << "Неверное значение длины массива!";

\*key = 0; // Переключаем переменную-флаг для сигнализировании ошибки

}

else

{

n = N;

if (n != N || n <= 0 || n > 50) // Проверка корректности ввода размера массива и наличие вещественного ввода

{

cout << "Неверное значение длины массива!";

n = 1; // Промежуточное действие-костяк

N = 1; // Промежуточное действие-костяк

\*key = 0; // Переключаем переменную-флаг для сигнализировании ошибки

}

}

cout << endl;

return N;

}

float\* InputMatrix(int\* key, int N) // Функция ввода элементов матрицы

{

static float\* matrix; // Создание матрицы внутри функции

if (\*key != 0) // Проверка наличия ошибок на предыдущих стадиях

{

matrix = new float[N]; // Выделение памяти под новую матрицу

char Z; // Временная переменная для проверки на буквенное значение

cout << "Введите значение элементов массива:\n";

for (int i = 0; i < N; i++) // Цикл для прогона по элементам массива

{

if (\*key != 0) // Проверка наличия ошибок на предыдущих стадиях

{

cout << "Элемент" << " [" << i << "]:" << endl; // Вывод i-го элемента массива

}

if (\*key != 0) // Проверка наличия ошибок на предыдущих стадиях

{

if (scanf("%f%c", &matrix[i], &Z) != 2 || Z != '\n') // Считывание элемента матрицы и одновременная проверка на буквенное значение

{

cout << "Неверное значение элемента массива!";

\*key = 0; // Переключаем переменную-флаг для сигнализировании ошибки

break; // Преждевременный выход из цикла

}

}

}

}

cout << endl;

return matrix; // Возврат заполненой матрицы

}

void InputA(int\* key, float\* A) // Функция ввода сравниваемого элемента

{

if (\*key != 0) // Проверка наличия ошибок на предыдущих стадиях

{

char Z; // Временная переменная для проверки на буквенное значение

cout << "Введите значение A:\n";

if (\*key != 0) // Проверка наличия ошибок на предыдущих стадиях

{

if (scanf("%f%c", A, &Z) != 2 || Z != '\n') // Ввода сравниваемого числа и проверка на буквенное значение

{

cout << "Неверное значение A!";

\*key = 0; // Переключаем переменную-флаг для сигнализировании ошибки

}

}

}

}

void Process(int\* key, float\* sum1, float\* sum2, int\* n1, int\* n2, float\* matrix, int N, float A) // Функция обработки данных

{

if (\*key != 0) // Проверка наличия ошибок на предыдущих стадиях

{

for (int i = 0; i < N; i++) // Цикл для прогона по элементам массива

{

if (matrix[i] > A) // Проверка элемента на первое условие

{

++\* n1; // Инкрементируем первый счётчик

\*sum1 += matrix[i]; // Прибавляем к первой сумме

}

if (matrix[i] < -A) // Проверка элемента на первое условие

{

++\* n2; // Инкрементируем второй счётчик

\*sum2 += matrix[i]; // Прибавляем ко второй сумме

}

}

}

}

void Output(int key, int N, float\* matrix, float sum1, float sum2, int n1, int n2) // Описание функции вывода результатов

{

cout << endl;

cout << "Элементы вашего массива:" << endl;

for (int i = 0; i < N; i++) // Цикл для прогона по элементам массива

{

cout << "Элемента массива " << "[" << i << "] = " << matrix[i] << endl; // Вывод элементов массива

}

cout << "Сумма элементов > A = " << sum1 << endl;

cout << "Сумма элементов <= -A = " << sum2 << endl;

cout << "Количество элементов >= A = " << n1 << endl;

cout << "Количество элементов <= -A = " << n2 << endl;

if (key != 0) // Проверка наличия ошибок на предыдущих стадиях

{

free(matrix); // Очистка памяти из под массива

}

cout << endl;

}