ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«РОССИЙСКИЙ УНИВЕРСИТЕТ ТРАНСПОРТА»
(РУТ (МИИТ))

Институт транспортной техники и систем управления

Кафедра «Управление и защита информации»

ОТЧЁТ
О ПРАКТИЧЕСКОЙ РАБОТЕ №4-1
По дисциплине «Процедурное программирование»

Выполнил: ст. гр. ТКИ – 112
Потапов А.К.
Проверил: к.т.н., доц.
Васильева М.А.

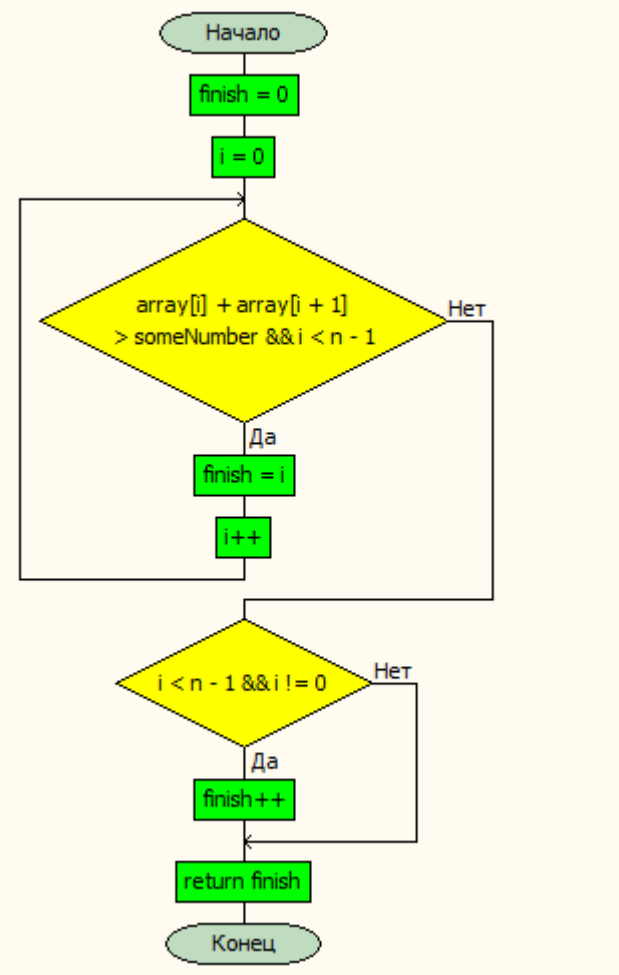
Москва 2021

Задание 4-1

1. Найти сумму элементов, значения которых состоят из одной цифры.
2. Заменить элементы массива между минимальным и максимальным на те же элементы в обратном порядке
3. Найти номер последней пары соседних элементов с одинаковыми знаками, произведение которых меньше заданного числа.

13	1. Заменить второй элемент массива на максимальный среди отрицательных. 2. Найти количество тех элементов, значения которых положительны и по модулю не превосходят заданное число A. 3. Найти номер первой пары соседних элементов, сумма которых меньше заданного числа.	[-10;10]

Блок схема



Код программы

```
#include <iostream>
#include <random>
/**
*\brief Функция возвращает целочисленное число, введённое пользователем
*\return Возвращает число, введённое пользователем
*/
int InputInt(const std::string& message);
/**
*\brief Функция заполняет массив случайными числами
*\param array массив
*\param n размер Массива
*/
void RandomArray(int* array, const size_t n);
/**
*\brief Функция выводит массив в консоль
*\param array массив
*\param n размер Массива
*/
void PrintArray(int* array, const size_t n);
/**
*\brief Функция позврляет заполнить массив вручную
*\param array массив
*\param n размер Массива
*/
void ManualArray(int* array, const size_t n);
/**
*\brief Функция меняет второй элемент массива на максимальный отрицательный в массиве
*\param array массив
*\param n размер Массива
*/
void MaxNegative(int* array, const size_t n);
/**
```

```
*\brief Функция считает количество элементов в массиве, положительных и по модулю не превосходящих заданное число
*\param array массив
*\param n размер Массива
*\return Возвращает количество элементов
*/
int FindNumberOf(int* array, const size_t n);
/**
*\brief Функция ищет индексы первой пары чисел в массиве, сумма которых меньше заданного числа
*\param array массив
*\param n размер Массива
*\return Возвращает индекс первого элемента пары в случае успеха
*/
int FirstPair(int* array, const size_t n);
/**
*\brief Класс констант метода ввода
*/
enum class Input
{
	random,
	manual
};
/**
*\brief Точка входа в программу
*\return 0 в случае успеха
*/
int main()
{
	size_t n = InputInt("Enter number of elements in array");
	std::cout << static_cast<int>(Input::random) << " — random\n"
		<< static_cast<int>(Input::manual) << " — manual";
	int* array = new int[n];
	size_t choose = InputInt("");
	try {
		auto chose = static_cast<Input>(choose);
		switch (chose)
		{
		case Input::random:
		{
			RandomArray(array, n);
			PrintArray(array, n);
			break;
		}
		case Input::manual:
```

```
                {
                        ManualArray(array, n);
                        PrintArray(array, n);
                        break;
                }
                }
        }
        catch (std::out_of_range&) {
                return 1;
        }

        std::cout << "Number of elements, which positive and abs < some
number\n" << FindNumberOf(array, n) << " — nuber of elements\n";
        std::cout << "Numbers of first pair, which sum < some number\n";
        int temp = FirstPair(array, n);
        std::cout << "Indexes of first pair — " << temp << ", " << temp + 1 <<
std::endl;
        std::cout << "Replace 2nd element in array by maximum negative\n";
        MaxNegative(array, n);
        PrintArray(array, n);
        return 0;
}

void PrintArray(int* array, const size_t n)
{
        std::cout << "Array: ";
        for (size_t i = 0; i < n; i++)
        {
                std::cout << array[i] << " ";
        }
        std::cout << std::endl;
}

void RandomArray(int* array, const size_t n)
{
        std::random_device rnd;
        std::mt19937 gen(rnd());
        std::uniform_int_distribution<> Uniform_int_distribution(-10, 10);
        for (size_t i = 0; i < n; i++)
        {
                array[i] = Uniform_int_distribution(gen);
        }
}

void ManualArray(int* array, const size_t n)
```

```
{
	for (size_t i = 0; i < n; i++)
	{
		std::cin >> array[i];
	}
}

void MaxNegative(int* array, const size_t n) {
	int maximumNegative = 0;
	for (size_t i = 0; i < n; i++) {
		if (array[i] < maximumNegative) {
			maximumNegative = array[i];
		}
	}
	if (maximumNegative != 0) {
		array[1] = maximumNegative;
	}
}

int FindNumberOf(int* array, const size_t n) {
	std::cout << "Enter number A: ";
	int someNumber;
	std::cin >> someNumber;
	size_t count = 0;
	for (size_t i = 0; i < n; i++) {
		if (array[i] >= 0 && abs(array[i]) < someNumber) {
			count++;
		}
	}
	return count;
}

int FirstPair(int* array, const size_t n) {
	std::cout << "Enter number A: ";
	int someNumber;
	std::cin >> someNumber;
	int finish = 0; //индекс первого элемента пары
	size_t i = 0;
	while (array[i] + array[i + 1] > someNumber && i < n - 1) {
		finish = i;
		i++;
	}
	if (i < n - 1 && i != 0) {
		finish++;
	}
```

```
	return finish;
}

int InputInt(const std::string& message)
{
	std::cout << message << std::endl;
	int input = -1;
	std::cin >> input;
	if (input < 0) {
		throw (std::out_of_range("Incorrect size"));
	}
	return input;
}
```