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

По дисциплине «Языки программирования»

Выполнил: ст. гр. ТКИ – 111

Владимиров Макар Алексеевич

Проверил: к.т.н., доц.

Васильева М. А.

Москва 2022

Содержание

1. Формулировка задачи

2. Блок-схема алгоритма

3. Решение задачи на языке программирования C++

4. Решение тестовых примеров

5. Зачет задания в GitHub

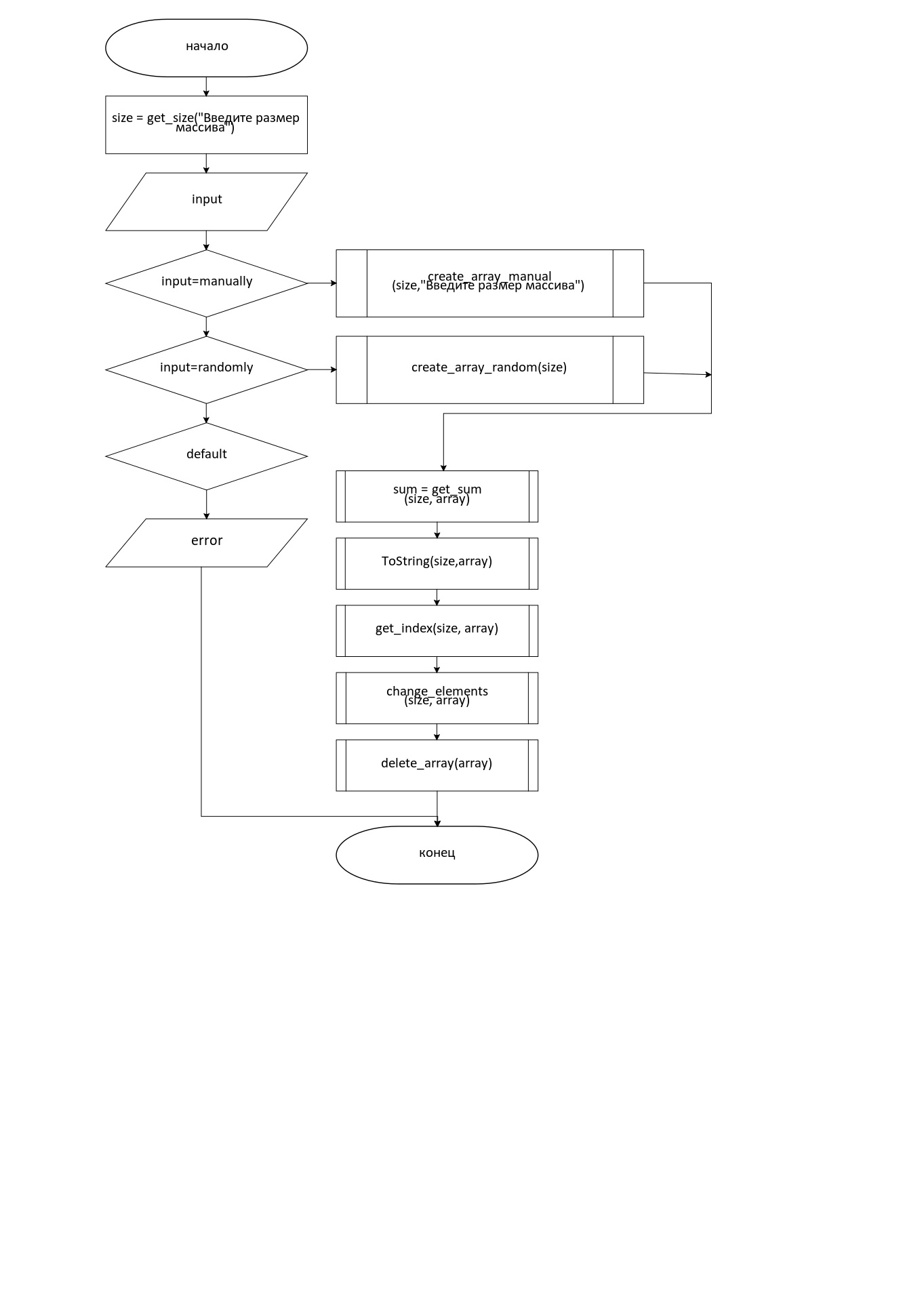
**Задание 4–1**

1. **Формулировка задачи**

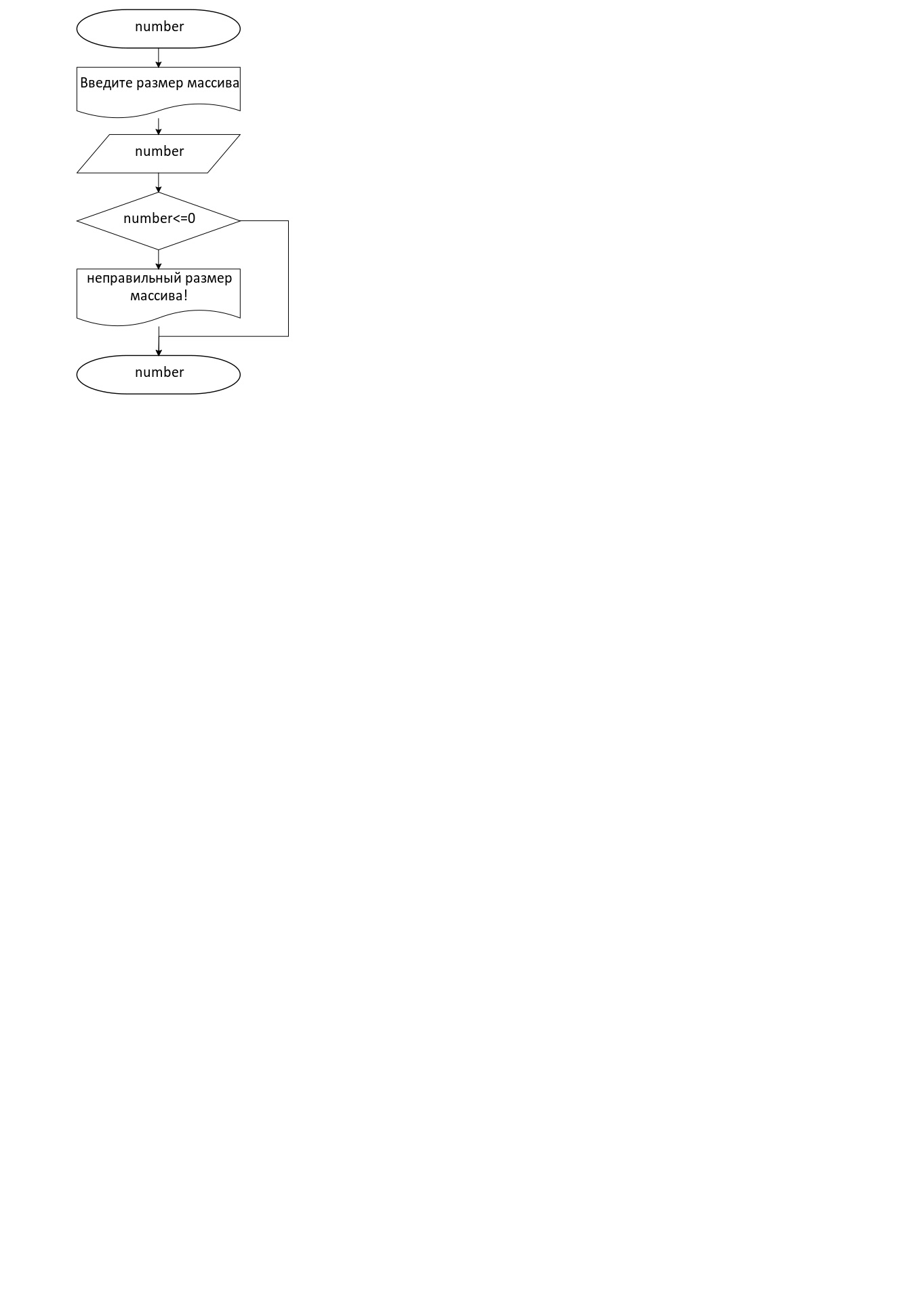
|  |  |  |
| --- | --- | --- |
| **№ варианта** | **Задание** | **Интервал** |
| 7 | 1. Найти сумму чётных отрицательных элементов 2. Вывести индексы тех элементов, значения которых кратны 6 3. Заменить нулями элементы массива между минимальным и максимальным кроме них самих | [-1000;1000] |

1. **Блок-схема алгоритма**

Блок-схемы алгоритмов функций представлены на рисунках: (**Рисунок 1, Рисунок 2, Рисунок 3, Рисунок 4, Рисунок 5, Рисунок 6, Рисунок 7, Рисунок 8, Рисунок 9, Рисунок 10, Рисунок 11**).



**Рисунок 1 – Блок-схема функции main()**

****

**Рисунок 2 – Блок-схема функции get\_size(message)**

****

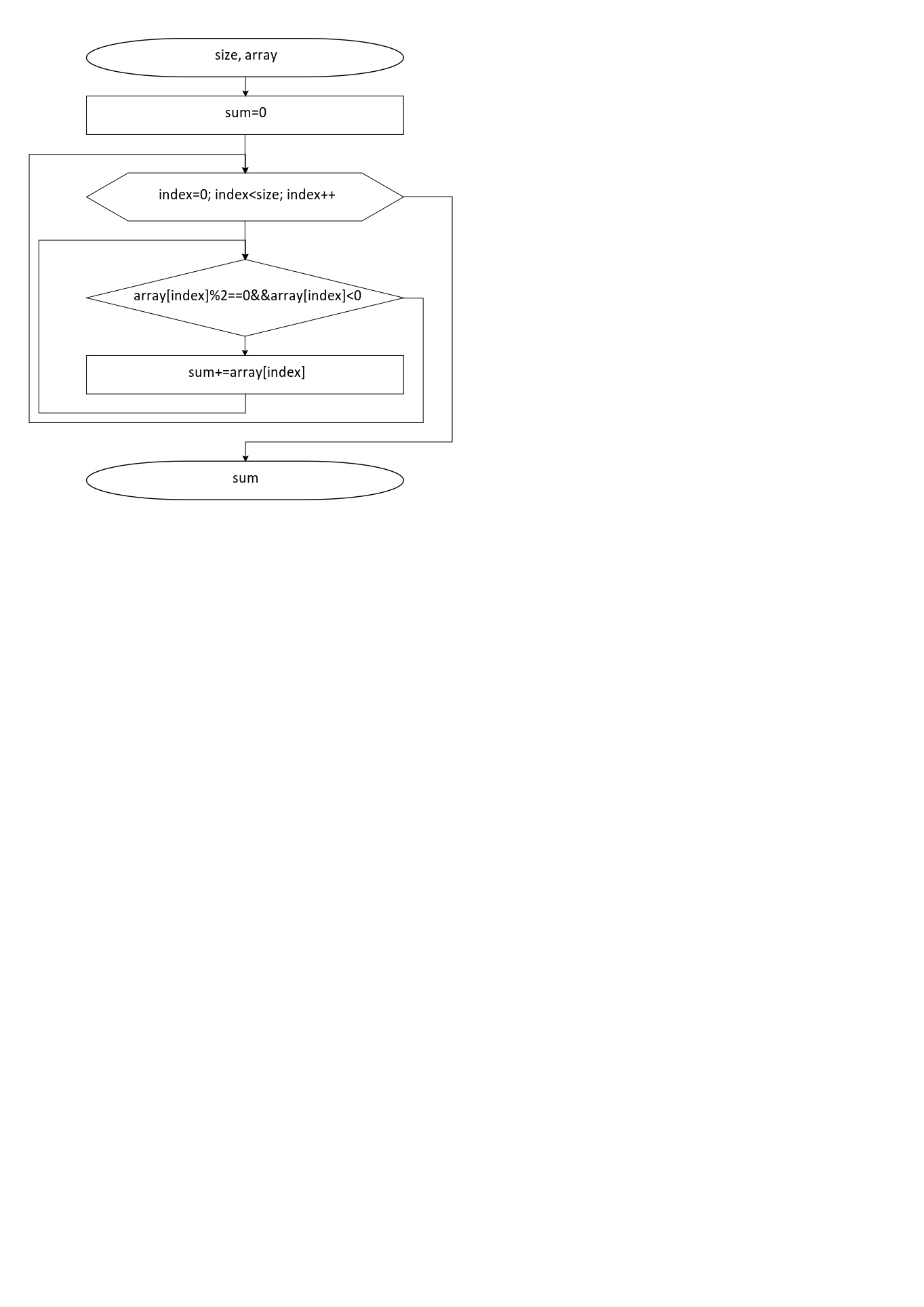
**Рисунок 3 – Блок-схема функции create\_array\_manual(size, message)**

****

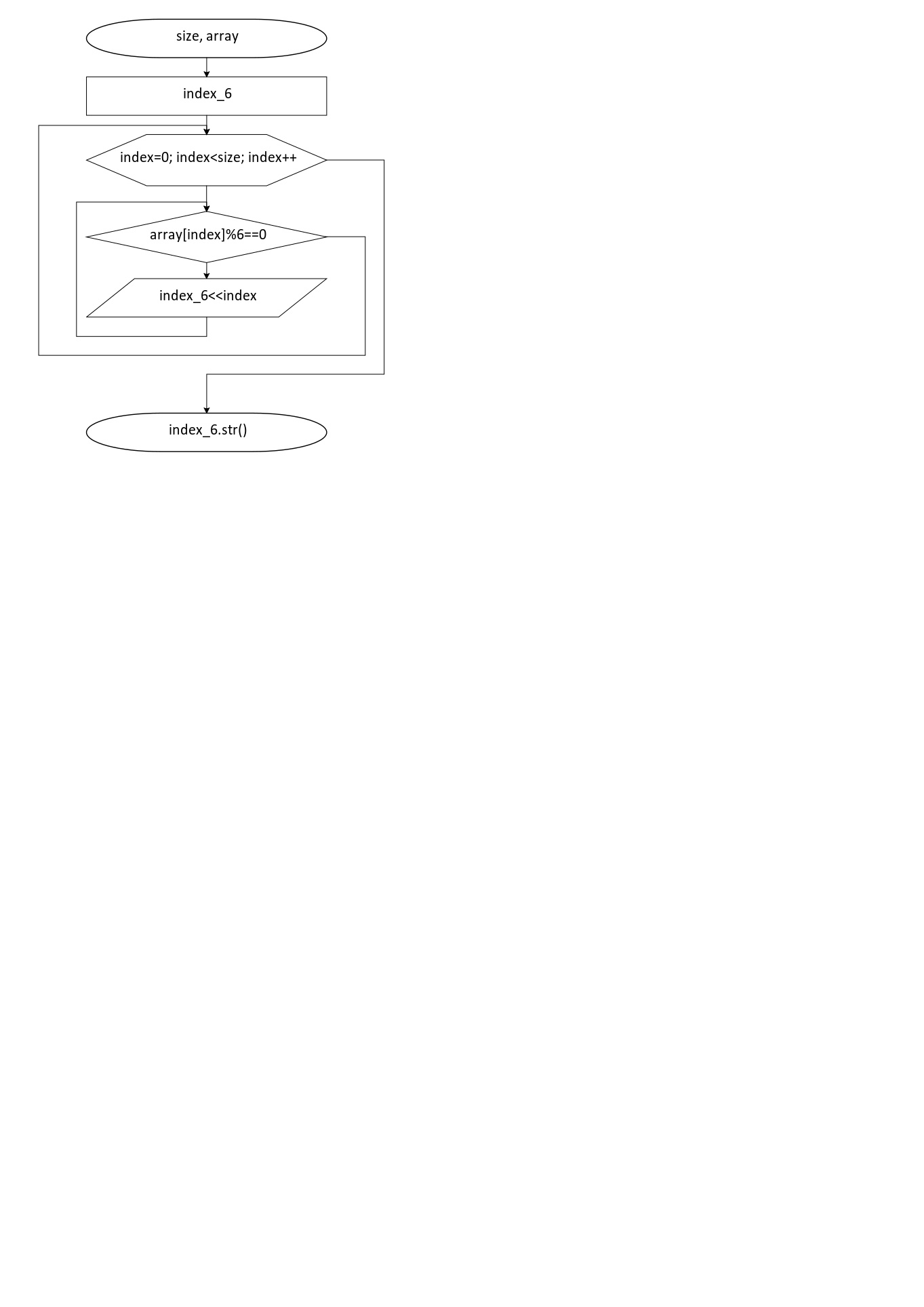
**Рисунок 4 – Блок-схема функции create\_array\_random(size)**

****

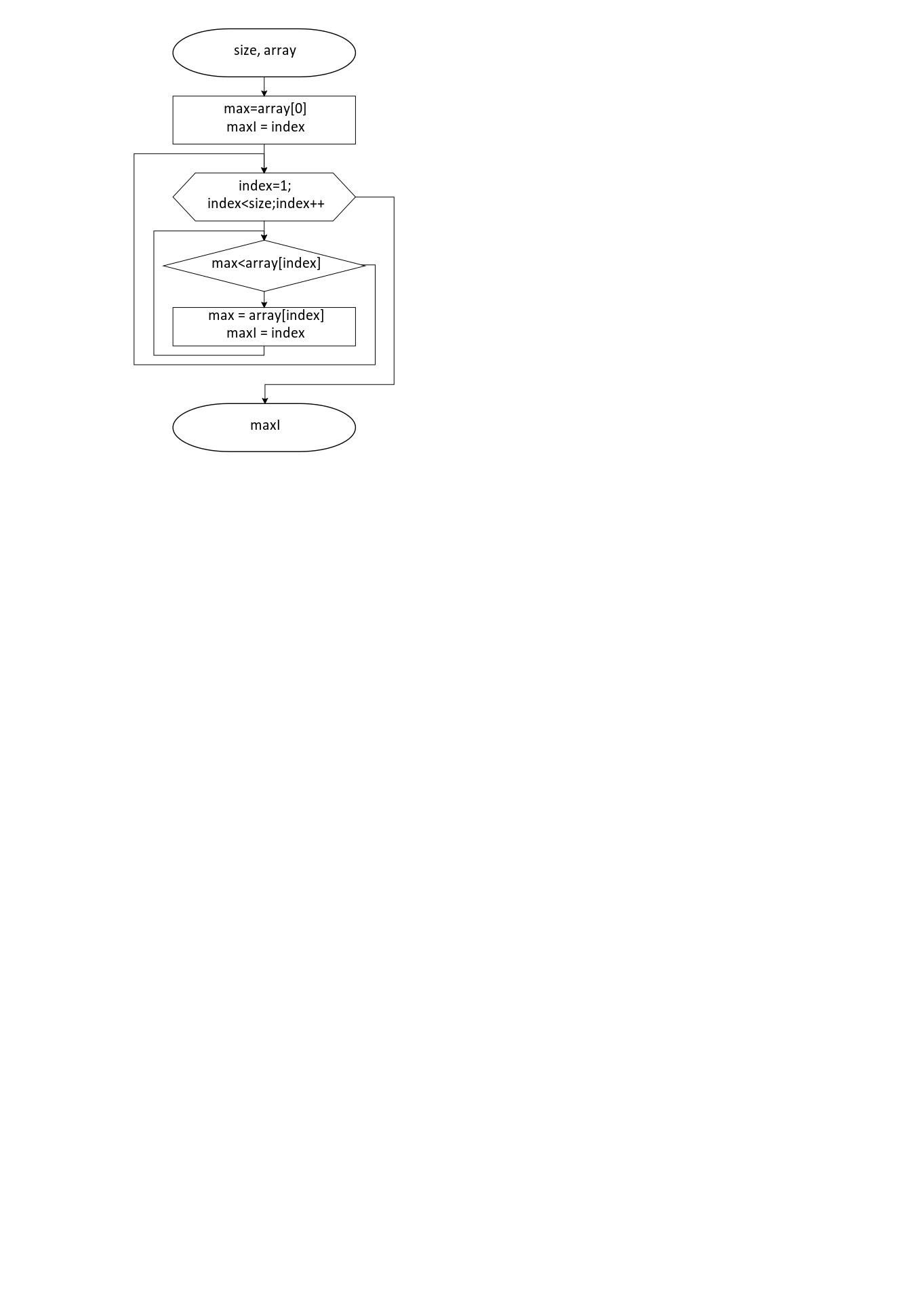
**Рисунок 5 – Блок-схема функции ToString(size,array)**

****

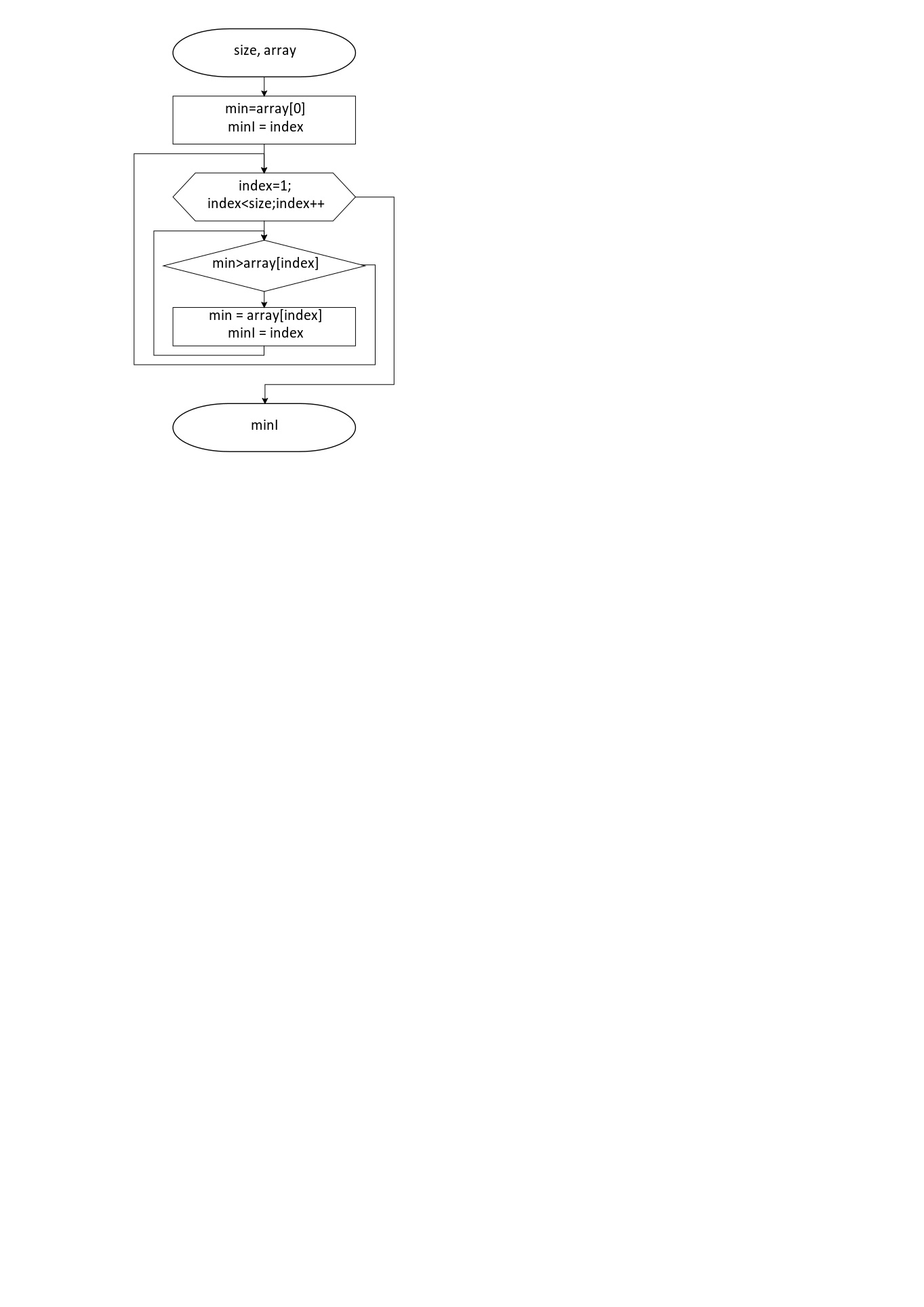
**Рисунок 6 – Блок-схема функции get\_sum(size,array)**

****

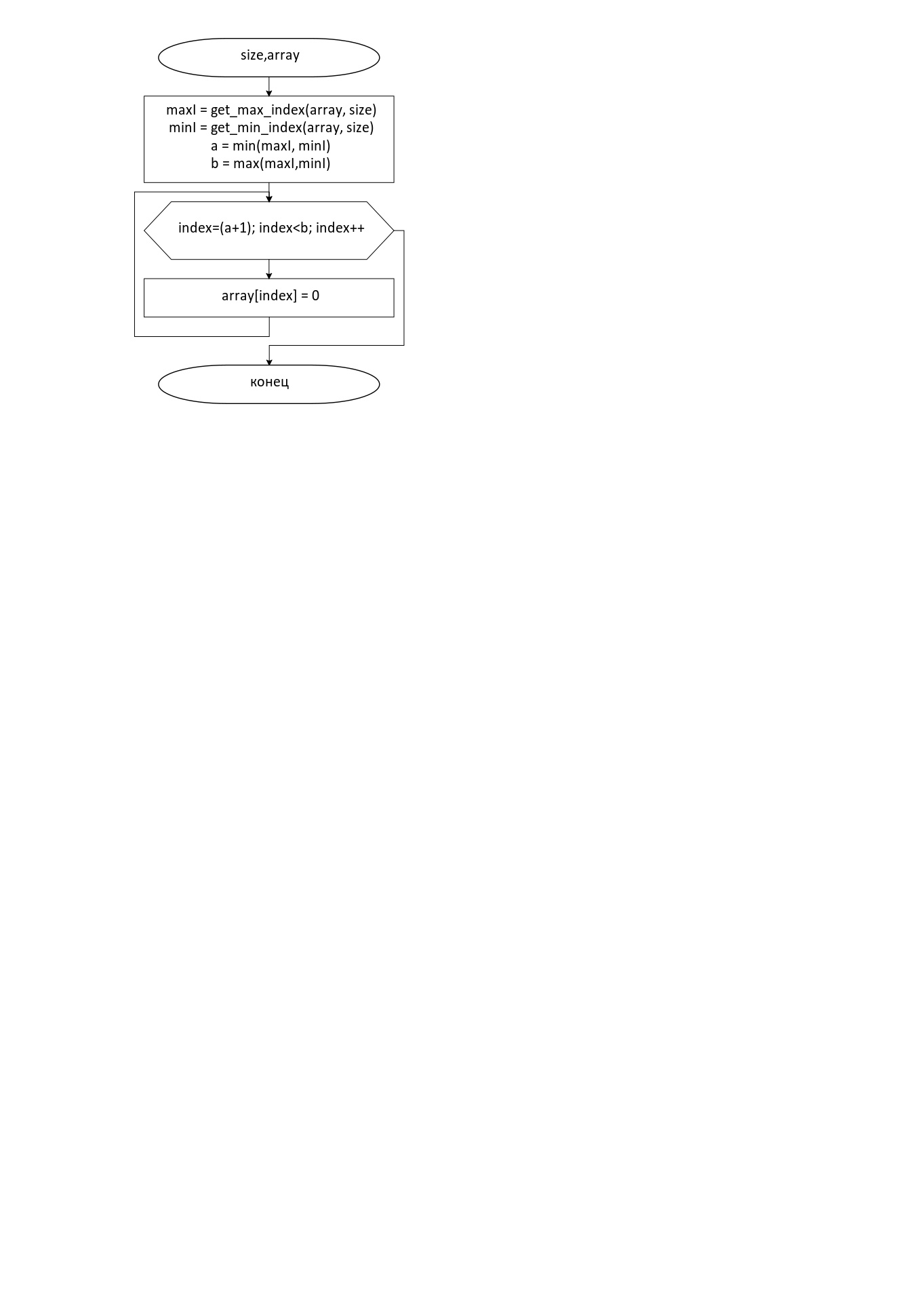
**Рисунок 7 – Блок-схема функции get\_index(size,array)**

****

**Рисунок 8 – Блок-схема функции get\_max\_index(size,array)**

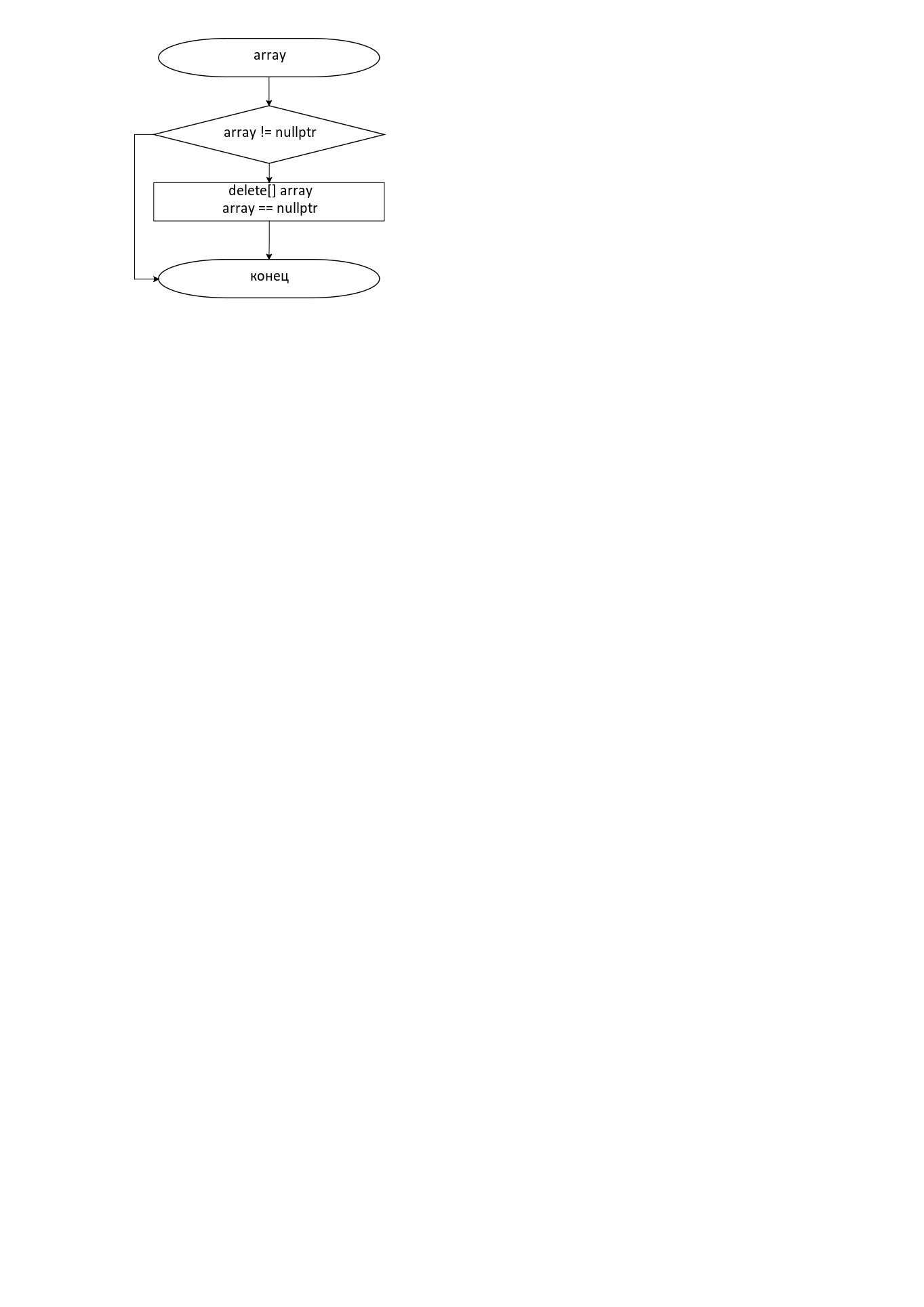
****

**Рисунок 9 – Блок-схема функции get\_min\_index(size,array)**

****

**Рисунок 10 – Блок-схема функции**

**change\_elements(size,array)**

****

**Рисунок 11 – Блок-схема функции delete\_array(array)**

1. **Решение задачи на языке программирования C++**

#include <string>

#include <iostream>

#include <exception>

#include <random>

#include <sstream>

using namespace std;

/\*\*

\* \brief

\* \Функция заменяющая нулём элементы между максимальным и минимальным

\*/

void change\_elements(const size\_t size, int\* array);

/\*\*

\* \brief Ввод размера

\* \param message Сообщение пользователю

\* \exception При вводе 0

\* \return Размер

\*/

size\_t get\_size(const string& message = nullptr );

/\*\*

\* \brief Создание массива и заполнение вручную

\* \param size Размер массива

\* \param message Сообщение пользователю

\* \return Указатель на массив

\*/

int\* create\_array\_manual(const size\_t size, const string& message = nullptr);

/\*\*

\* \brief Создание массива и заполнение случайными числами

\* \param size Размер массива

\* \param message Сообщение пользователю

\* \return Указатель на массив

\*/

int\* create\_array\_random(

const size\_t size,

const int max = 1000,

const int min = -1000);

/\*\*

\* \brief

\* \param size Размер массива

\* \param array Указатель на массив

\* \return Массив в виде строки

\*/

string ToString(const size\_t size, int\* array);

/\*\*

\* \brief

\* \Вариант заполнения массива

\*/

enum class user\_choice

{

manually,

randomly

};

/\*\*

\* \brief

\* \Удаление массива

\* \param array Указатель на массив

\*/

void delete\_array(int\*& array);

/\*\*

\* \brief Функция нахождения суммы чётных отрицательных элементов

\* \param size Размер массива

\* \param array Указатель на массив

\* \return Сумма чётных отрицательных элементов

\*/

int get\_sum(const size\_t size, int\* array);

/\*\*

\* \brief Функция нахождения индексов элементов значения которых кратны 3 и 6

\* \param size Размер массива

\* \param array Указатель на массив элементов значения которых кратны 3 и 6

\* \return Индексы

\*/

string get\_index(const size\_t size, int\* array);

/\*\*

\* \brief

\* \param size Размер массива

\* \param array Указатель на массив

\* \return Индекс максимального элемента

\*/

int get\_max\_index(int\* array, const size\_t size);

/\*\*

\* \brief

\* \param size Размер массива

\* \param array Указатель на массив

\* \return Индекс минимального элемента

\*/

int get\_min\_index(int\* array, const size\_t size);

/\*\*

\* \brief

\* \Точка входа в программу

\* \return 0 в случае успеха

\*/

int main()

{

setlocale(LC\_ALL, "Rus");

try

{

const size\_t size = get\_size("Введите размер массива");

cout<<"Выберите способ заполнения массива\n"<<

static\_cast<int>(user\_choice::manually)<<

" - вручную\n"<<

static\_cast<int>(user\_choice::randomly)<<

" - случайно\n";

auto user\_choice1 = -1;

cin>> user\_choice1;

auto choice = static\_cast<user\_choice>(user\_choice1);

int\* array = nullptr;

switch(choice)

{

case user\_choice::manually:

{

array = create\_array\_manual(size, "Введите массив:\n");

break;

}

case user\_choice::randomly:

{

array = create\_array\_random(size);

break;

}

default:

{

cerr << "неправильный выбор" << endl;

return 1;

}

}

cout << "Наш массив\n" ;

cout << ToString(size, array);

int sum = get\_sum(size,array);

if (sum >= 0)

{

cout << "В массиве нет чётных отрицательных элементов" << endl;

}

else

{

cout << "Сумма отрицательных чётных элементов = " << sum << endl;

}

cout << "Индексы элементов кратных 6: " << get\_index(size, array);

change\_elements(size, array);

cout << "Наш массив\n";

cout << ToString(size, array);

delete\_array(array);

return 0;

}

catch (exception& error)

{

cerr << error.what();

return 1;

}

}

size\_t get\_size(const string& message)

{

int number = -1;

cout << message;

cin >> number;

if (number <= 0)

{

throw out\_of\_range("Неправильный размер!");

}

return static\_cast<size\_t>(number);

}

int\* create\_array\_manual(const size\_t size, const string& message)

{

cout << message;

int\* array = new int[size];

for (size\_t index = 0; index < size; index ++)

{

cin >> array[index];

}

return array;

}

int\* create\_array\_random(const size\_t size, const int max, const int min)

{

//Will be used to obtain a seed for the random number engine

std::random\_device rd;

//Standard mersenne\_twister\_engine seeded with rd()

std::mt19937 gen(rd());

std::uniform\_int\_distribution<> uniformIntDistribution(min, max);

auto\* array = new int[size];

for (size\_t index = 0; index < size; index ++)

{

array[index] = uniformIntDistribution(gen);

}

return array;

}

string ToString(const size\_t size, int\* array)

{

stringstream buffer;

for (size\_t index = 0 ; index < size - 1; index ++)

{

buffer << array[index]<< ", ";

}

buffer << array[size - 1] << "\n";

return buffer.str();

}

void delete\_array(int \*& array)

{

if (array != nullptr)

{

delete[] array;

array == nullptr;

}

}

int get\_sum(const size\_t size, int\* array)

{

int sum = 0;

for (size\_t index = 0 ; index < size; index ++)

{

if(array[index] % 2 == 0 && array[index] < 0)

{

sum += array[index];

}

}

return sum;

}

string get\_index(const size\_t size, int\* array)

{

stringstream index\_6;

for (size\_t index = 0 ; index < size; index ++)

{

if(array[index] % 6 == 0)

{

index\_6 << index << ", ";

}

}

return index\_6.str();

}

int get\_max\_index(int\* array, const size\_t size)

{

int max = array[0];

int maxI = 0;

for (size\_t index = 1 ; index < size; index ++)

{

if(max < array[index])

{

max = array[index];

maxI= index;

}

}

return maxI;

}

int get\_min\_index(int\* array, const size\_t size)

{

int min = array[0];

int minI = 0;

for (size\_t index = 1 ; index < size; index ++)

{

if(min > array[index])

{

min = array[index];

minI= index;

}

}

return minI;

}

void change\_elements(const size\_t size, int\* array)

{

int maxI = get\_max\_index(array, size);

int minI = get\_min\_index(array, size);

int a = std :: min(maxI, minI);

int b = std :: max(maxI, minI);

for (size\_t index = (a + 1) ; index < b; index++)

{

array[index] = 0;

}

}

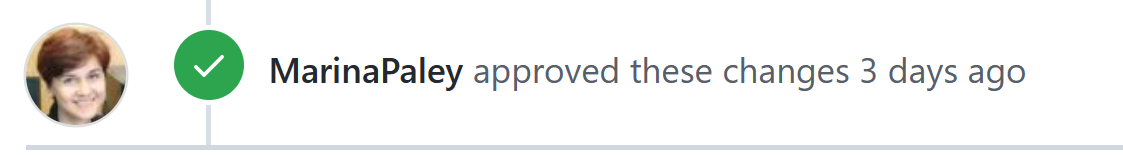
1. **Решение тестовых примеров**

Изображение выглядит как текст

Автоматически созданное описание

**Рисунок 12 – Расчёты**

1. **Зачет задания в GitHub**



**Рисунок 13 – Зачет задания в GitHub**