

МИНОБРНАУКИ РОССИИ

федеральное государственное бюджетное образовательное учреждение высшего образования

«Санкт-Петербургский государственный технологический институт (технический университет)»

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| УГСН | | 09.00.00 | Информатика и вычислительная техника | | |
| Уровень образования | |  | Высшее образование – бакалавриат | | |
| Форма обучения | |  | Очная | | |
| Факультет | |  | Информационных технологий  и управления | | |
| Кафедра | |  | Систем автоматизированного проектирования и управления | | |
| Учебная дисциплина | |  | Информационные технологии  и программирование | | |
| Курс | I | | | Группа | 4304 |

Отчёт по контрольной работе № 3

Вариант № 26

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Исполнитель: |  |  |  |  |
| обучающийся группы 4304 |  |  |  | Рыбник Всеволод Сергеевич |
|  |  | (дата, подпись) |  |  |
|  |  |  |  |  |
| Проверил: |  |  |  | Корниенко Иван Григорьевич |
|  |  | (дата, подпись) |  | Макарук Роман Валерьевич |
|  |  |  |  | Федин Алексей Константинович |

# СОДЕРЖАНИЕ

[1 Задание №1 3](#_Toc51078351)

[1.1 Цель работы 3](#_Toc51078352)

[1.2 Постановка задачи 3](#_Toc51078353)

[1.3 Описание хода выполнения 3](#_Toc51078354)

[1.4 Блок-схема алгоритма решения задачи](#_Toc51078355) 3

[1.5 Исходный код полученного программного решения 5](#_Toc51078356)

[1.6 Тестирование](#_Toc51078357) 9

[1.7 Выводы по заданию №1](#_Toc51078358) 9

[2 Задание №2](#_Toc51078359) 10

[2.1 Цель работы](#_Toc51078360) 10

[2.2 Постановка задачи](#_Toc51078361) 10

[2.3 Описание хода выполнения](#_Toc51078362) 10

[2.4 Блок-схема алгоритма решения задачи](#_Toc51078363) 10

[2.5 Исходный код полученного программного решения](#_Toc51078364) 12

[2.6 Тестирование](#_Toc51078365) 15

[2.7 Выводы по заданию №2](#_Toc51078366) 15

[3 Задание №3](#_Toc51078367) 16

[3.1 Цель работы](#_Toc51078368) 16

[3.2 Постановка задачи](#_Toc51078369) 16

[3.3 Описание хода выполнения](#_Toc51078370) 16

[3.4 Блок-схема алгоритма решения задачи](#_Toc51078371) 16

[3.5 Исходный код полученного программного решения 1](#_Toc51078372)8

[3.6 Тестирование](#_Toc51078373) 20

[3.7 Выводы по заданию №3](#_Toc51078374) 20

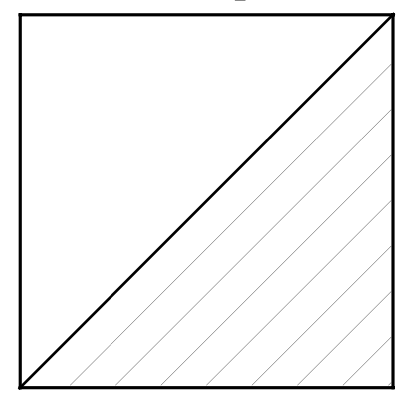
# 1 Задание №1

## 1.1 Цель работы

Динамическое распределение памяти.

## 1.2 Постановка задачи

Даны числа n и m, действительная матрица An×m. Найти и вывести наименьшее Emn и наибольшее Emx из значений элементов, расположенных в заштрихованной части матрицы:



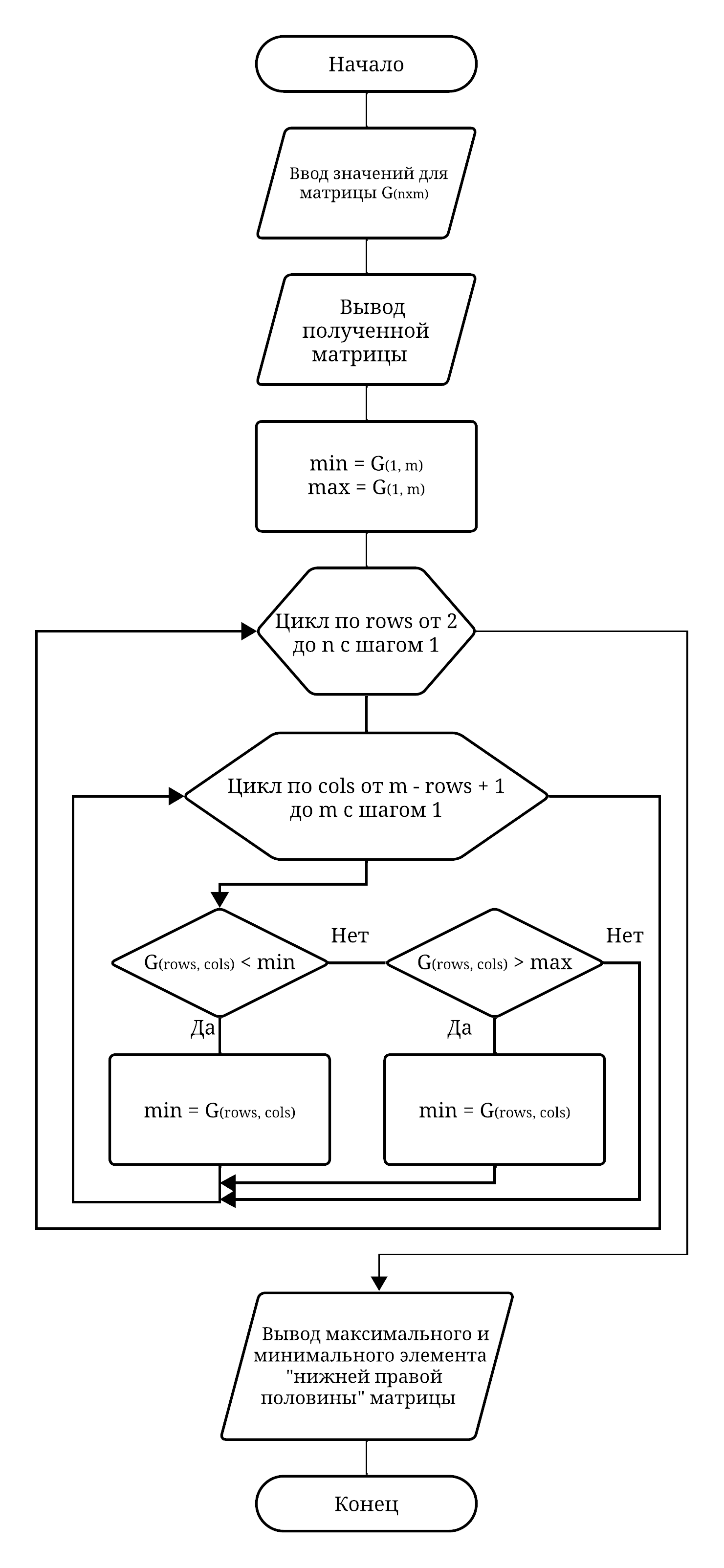
Вывести полученную матрицу A.

## 1.3 Описание хода выполнения

## Для выполнения поставленной задачи была использована структура, выполняющая функции двумерного массива — матрицы.

## 1.4 Блок-схема алгоритма решения задачи

Блок-схема алгоритма №3.1:

Рисунок 1 – Блок-схема алгоритма решения задачи №1

## 1.5 Исходный код полученного программного решения

**Код файла first\_task.c:**

#include <stdlib.h>

#include <time.h>

#include "first\_algorithm.h"

enum MENU {ManualInput=1, RandomInput=2, Quit=3};

int main(void)

{

srand(time(NULL));

int options = 1, loop\_indicator = 1;

int rows = 0, cols = 0;

Array array;

printf(" Vsevolod Rybnik test 3 task 1 var 26\n");

while (loop\_indicator)

{

int\* result;

puts("Specify matrix size: \n Rows: ");

rows = get\_int();

puts(" Cols:");

cols = get\_int();

puts(" 1 - Manual input\n 2 - Random input\n 3 - Quit");

options = get\_int();

switch (options)

{

default:

puts("Dis value is not akceptabele\n");

case Quit:

puts("Bye, see you later");

return EXIT\_SUCCESS;

case ManualInput:

array = make\_array(rows, cols, false);

output(array);

result = max\_min\_finder(array);

printf("Min Element: %d\nMax Element: %d\n", result[0], result[1]);

clean\_space(&array);

continue;

case RandomInput:

array = make\_array(rows, cols, true);

output(array);

result = max\_min\_finder(array);

printf("Min Element: %d\nMax Element: %d\n", result[0], result[1]);

clean\_space(&array);

continue;

}

}

}

**Код файла first\_struct.c:**

#include <math.h>

#include <locale.h>

#include <stdlib.h>

#include "first\_interface.h"

#define upper\_left\_border 218

#define upper\_right\_border 191

#define lower\_left\_border 192

#define lower\_right\_border 217

#define underline 196

#define aside\_border 179

struct TwoDimensionalArray {

int rows;

int cols;

int \*\*data;

};

typedef struct TwoDimensionalArray Array;

Array make\_array(int rows, int cols, bool rand\_man\_indicator)

{

Array array;

array.rows = rows;

array.cols = cols;

array.data = (int \*\*) calloc(array.rows, sizeof(int \*)); // https://ufchgu.ru/blog/realloc-malloc-calloc-chem-zapolnjaet#:~:text=Функция%20malloc%20принимает%20один%20аргумент,количество%20элементов%20и%20их%20размер.

for (int row = 0; row < array.rows; row++) {

array.data[row] = (int \*) calloc(array.cols, sizeof(int));

}

if (rand\_man\_indicator){

for (int row = 0; row < array.rows; row++) {

for (int col = 0; col < array.cols; col++) {

array.data[row][col] = -99 + rand()%(100 + 98);

}

}

}

else

{

for (int row = 0; row < array.rows; row++) {

for (int col = 0; col < array.cols; col++) {

printf(" - Specify %d %d element of Matrix: ", row+1, col+1);

array.data[row][col] = get\_double();

}

}

}

return array;

}

void clean\_space(Array \*array)

{

for (int row\_index = 0; row\_index < array->rows; row\_index++) {

free(array->data[row\_index]);

array->data[row\_index] = NULL;

}

free(array->data);

array->data = NULL;

}

void output(Array array)

{

printf(" %c", upper\_left\_border);

int underline\_amount = 6 \* array.cols + 2;

for (int i = 1; i < underline\_amount; i++)

printf("%c", underline);

printf("%c\n", upper\_right\_border);

for (int row = 0; row < array.rows; row++) {

for (int col = 0; col < array.cols; col++) {

if (col == 0)

printf(" %c", aside\_border);

if (array.data[row][col] >= 0)

printf(" ");

double digits = floor(log10(abs(array.data[row][col]))) + 1;

printf("%\*s", digits == 2? 2 : digits == 3? 1 : 3, " ");

printf("%d ", array.data[row][col]);

}

printf(" %c\n", aside\_border);

}

printf(" %c", lower\_left\_border);

for (int i = 1; i < underline\_amount; i++)

printf("%c", underline);

printf("%c\n", lower\_right\_border);

}

**Код файла first\_interface.c:**

#include <stdio.h>

#include <stdbool.h>

double get\_double(void)

{

char temprem, tempclear; // временный остаток

double input = 0;

while(true)

{

temprem = 0;

tempclear = 0;

if((!scanf("%lf%c",&input ,&temprem))|| temprem != '\n')

{

printf(" - Error: Invalid value for double variables.\n - One more time: ");

while(tempclear != '\n')

scanf("%c",&tempclear);

}

else

return input;

}

}

int get\_int(void)

{

char temprem, tempclear;

int input = 0;

while(true)

{

temprem=0;

tempclear=0;

if((!scanf("%d%c",&input ,&temprem))|| temprem != '\n')

{

printf(" - Error: Invalid value for int variables.\nOne more time: ");

while(tempclear != '\n')

scanf("%c",&tempclear);

}

else

return input;

}

}

**Код файла first\_algorithm.c:**

#include "first\_struct.h"

int\* max\_min\_finder(Array array)

{

int\* result = (int\*)calloc(2, sizeof(int)); // 0 элемент - min, 1 - max

result[0] = array.data[array.rows-1][array.cols-1];

result[1] = array.data[array.rows-1][array.cols-1];

for (int rows = 1; rows < array.rows; rows++)

{

for (int cols = array.cols - (rows + 1); cols < array.cols; cols++)

{

result[0] = (array.data[rows][cols] < result[0])? array.data[rows][cols] : result[0];

result[1] = (array.data[rows][cols] > result[1])? array.data[rows][cols] : result[1];

}

}

return result;

}

**Код файла first\_algorithm.h:**

#include "first\_struct.h"

int\* max\_min\_finder(Array array);

**Код файла first\_interface.h:**

#include <stdio.h>

#include <stdbool.h>

double get\_double(void);

int get\_int(void);

**Код файла first\_struct.h:**

#include <stdlib.h>

#include "first\_interface.h"

typedef struct TwoDimensionalArray {

int rows;

int cols;

int \*\*data;

} Array;

Array random\_values\_for\_array(Array array);

Array make\_array(int rows, int cols, bool rand\_man\_indicator);

void clean\_space(Array \*array);

void output(Array array);

## 1.6 Тестирование

Результат тестирования приведён на рисунке 2.

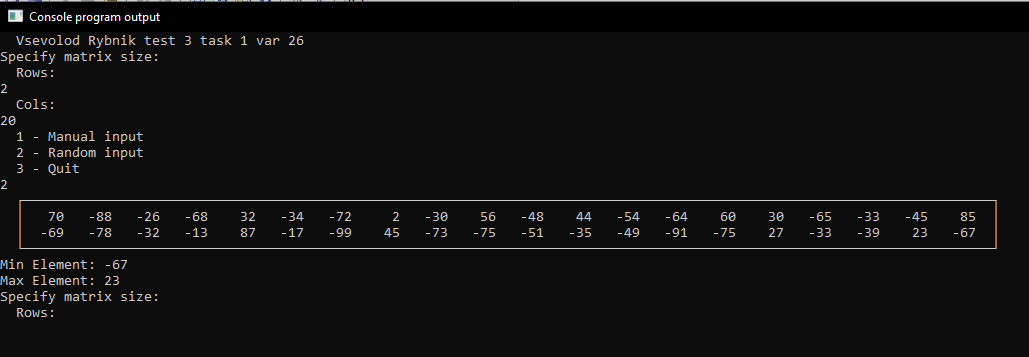


Рисунок 2 – Экранная копия результата работы разработанной программы задания №1

## 1.7 Выводы по заданию №1

В ходе выполнения поставленной задачи была улучшена функция вывода значений матрицы из предыдущей контрольной так, чтобы она поддерживала матрицы любого формата, также был освоен метод возвращения нескольких значений из метода без использования хэш-таблицы.

# 2 Задание №2

## 2.1 Цель работы

Работа с битами.

## 2.2 Постановка задачи

Дана последовательность из 8 символов. Сравнить их младший и старший биты. Если они равны, то заменить старший нулём, младший – единицей, иначе заменить старший бит единицей, младший – нулём.

## 2.3 Описание хода выполнения

Для выполнения данного задания была изучена новая тема «Кодировки Символов», освоены методы работы с битами в C.

## 2.4 Блок-схема алгоритма решения задачи

На рисунке 3 представлена блок-схема алгоритма решения задачи №2.

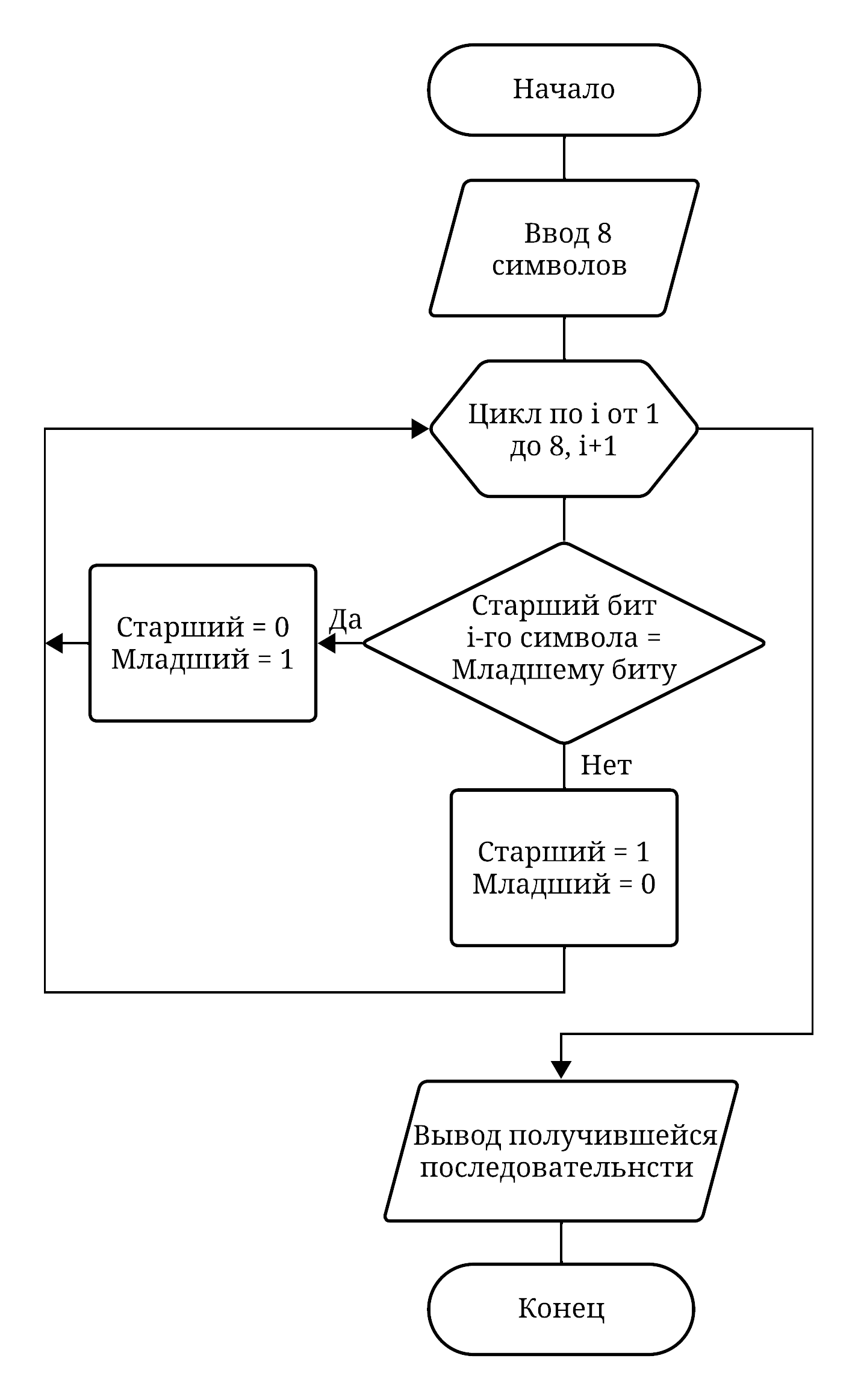


Рисунок 3 – Блок-схема алгоритма решения задачи №2

## 2.5 Исходный код полученного программного решения

**Код основного файла — second\_task.c:**

#include <locale.h>

#include <time.h>

#include "second\_interface.h"

#include "second\_algorithm.h"

#define EL\_AMOUNT 8

enum MENU {Man=1, Rand=2, Quit=3};

int main(void)

{

setlocale (LC\_CTYPE, "RU");

srand(time(NULL));

int loop\_indicator = 1, options = 1;

printf(" Vsevolod Rybnik test 3 task 2 var 26\n");

while (loop\_indicator)

{

char sequence[EL\_AMOUNT];

char\* bits;

puts(" 1 - Manual input\n 2 - Random input\n 3 - Quit");

options = get\_int();

switch (options)

{

case Man:

puts("Specify your symbols sequence: ");

get\_sequence(sequence, EL\_AMOUNT);

bits = algorithm(sequence, EL\_AMOUNT);

printf("...And result is -> %s\n", bits);

printf("In ASCII - %s\n", back\_to\_char(bits, EL\_AMOUNT));

free(bits);

continue;

case Rand:

printf(

"Your random symbol sequence is %s\n",

get\_random\_sequence(sequence, EL\_AMOUNT)

);

bits = algorithm(sequence, EL\_AMOUNT);

printf("...And result is -> %s\n", bits);

printf("In ASCII - %s\n", back\_to\_char(bits, EL\_AMOUNT));

free(bits);

continue;

case Quit:

puts(" Bye, see you later!");

return EXIT\_SUCCESS;

default:

puts(" Dis value is not akceptabele\n");

}

}

}

**Код файла second\_algorithm.c:**

#include <stdio.h>

#include <string.h>

#include <stdlib.h>

char\* algorithm(char\* sequence, int seq\_len) {

char\* bits = (char\*)calloc(64, sizeof(int)); // массив для хранения битов

for (int i = 0; i < seq\_len; i++) {

char c = sequence[i];

sprintf(bits + i \* 8, "%08d", c);

}

for (int i = 0; i < seq\_len; i++) {

int firstBit = bits[i \* 8] - '0';

int lastBit = bits[i \* 8 + 7] - '0';

if (firstBit == lastBit) {

bits[i \* 8] = '0';

bits[i \* 8 + 7] = '1';

} else {

bits[i \* 8] = '1';

bits[i \* 8 + 7] = '0';

}

}

return bits;

}

**Код файла second\_interface.c:**

#include <stdio.h>

#include <stdlib.h>

#include <stdbool.h>

#include <string.h>

int get\_int(void)

{

char temprem, tempclear;

int input = 0;

while(true)

{

temprem=0;

tempclear=0;

if((!scanf("%d%c",&input ,&temprem))|| temprem != '\n')

{

printf(" - Error: Invalid value for int variables.\nOne more time: ");

while(tempclear != '\n')

scanf("%c",&tempclear);

}

else

return input;

}

}

char\* get\_sequence(char\* sequence, int length){

char temprem, tempclear;

while(true)

{

temprem=0;

tempclear=0;

sequence[0]='\0';

if((!scanf("%s",sequence))|| (int)strlen(sequence) != length)

{

printf(" - Error: Invalid value for 8 length string sequence.\nOne more time: ");

while(tempclear != '\n')

scanf("%c",&tempclear);

}

else

return sequence;

}

}

char\* get\_random\_sequence(char\* sequence, int length)

{

for (int i = 0; i < length; i++) {

sequence[i] = rand() % 26 + 'A';

}

sequence[8] = '\0';

return sequence;

}

char\* back\_to\_char(char\* bits, int seq\_len)

{

char\* result = (char\*) malloc(seq\_len + 1);

for (int i = 0; i < seq\_len; i++) {

char c = 0;

for (int j = 0; j < 8; j++) {

c = (c << 1) | (bits[i \* 8 + j] - '0');

}

result[i] = c;

}

result[seq\_len] = '\0';

return result;

}

**Код файла second\_interface.h:**

#include <stdio.h>

#include <stdlib.h>

#include <stdbool.h>

int get\_int(void);

char\* get\_random\_sequence(char\* sequence, int length);

char\* back\_to\_char(char\* bits, int seq\_len);

char\* get\_sequence(char\* sequence, int length);

**Код файла second\_algorithm.h:**

#include <stdio.h>

#include <string.h>

char\* algorithm(char\* sequence, int seq\_len);

## 2.6 Тестирование

Тестирование было проведено на рандомных значениях:

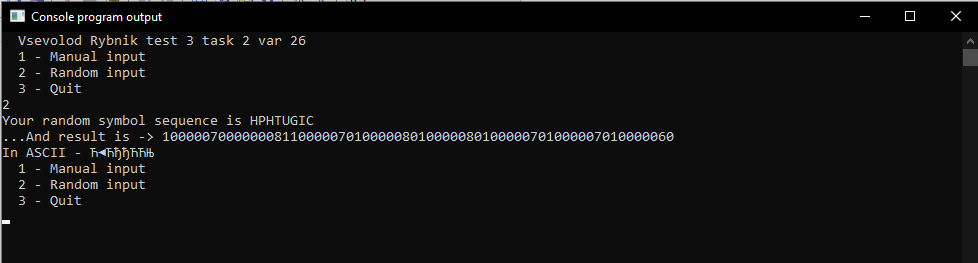


Рисунок 4 – Экранная копия результата работы разработанной программы задания 2

## 2.7 Выводы по заданию №2

В ходе выполнения задания были изучены новые темы и понятия в программировании, способы работы с битами символов в C.

# 3 Задание №3

## 3.1 Цель работы

Разобраться в функционале указателей на функции.

## 3.2 Постановка задачи

Функция fold суммирует все числа в массиве целочисленных чисел с использованием функции sum, получает в качестве аргументов указатель на исходный массив, размер массива, указатель на функцию sum от двух аргументов и возвращает сумму всех элементов массива. Функция sum производит сложение двух аргументов и возвращает их сумму.

## 3.3 Описание хода выполнения

Для реализации поставленной задачи были изучены указатели на функции в C.

## 3.4 Блок-схема алгоритма решения задачи

Блок схема к задаче №3:

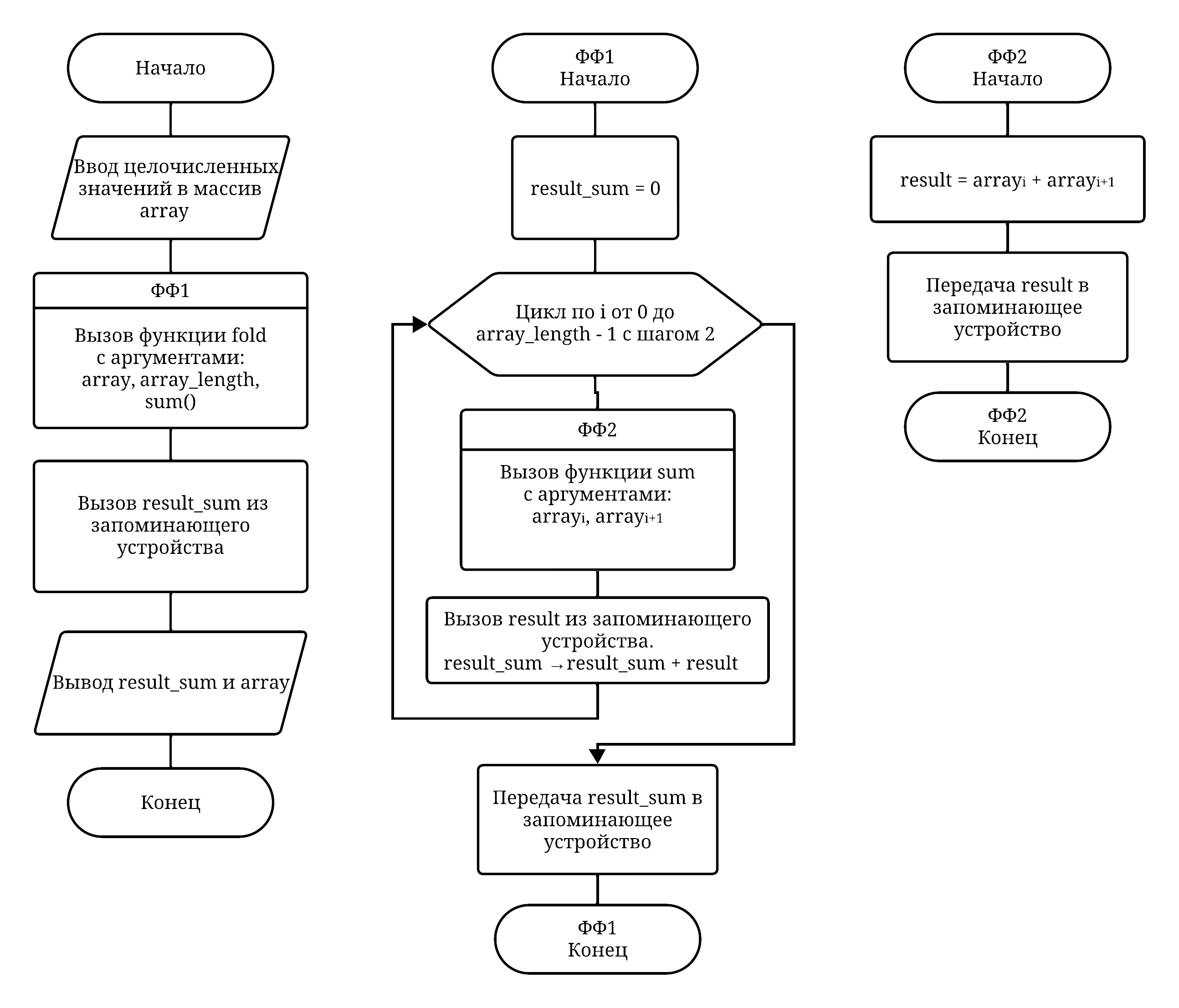


Рисунок 5 – Блок-схема алгоритма решения задачи №3

## 3.5 Исходный код полученного программного решения

**Код основного файла third\_task.c:**

#include <time.h>

#include "third\_interface.h"

#include "third\_algorithm.h"

enum MENU {Man=1, Rand=2, Quit=3};

int main(void)

{

srand(time(NULL));

int ARRAY\_LEN = 0;

int loop\_indicator = 1, options = 1;

int\* array = (int\*)calloc(ARRAY\_LEN, sizeof(int));

printf(" Vsevolod Rybnik test 3 task 3 var 26\n");

while (loop\_indicator)

{

puts("Specfiy array length: ");

ARRAY\_LEN = get\_int();

puts(" 1 - Manual input\n 2 - Random input\n 3 - Quit");

options = get\_int();

switch (options)

{

case Man:

bind\_values(array, false, ARRAY\_LEN);

printf("Result: %d \n", fold(array, ARRAY\_LEN, algorithm));

output(array, ARRAY\_LEN);

continue;

case Rand:

bind\_values(array, true, ARRAY\_LEN);

output(array, ARRAY\_LEN);

printf("Result: %d \n", fold(array, ARRAY\_LEN, algorithm));

continue;

case Quit:

puts(" Bye, see you later!");

return EXIT\_SUCCESS;

default:

puts(" Dis value is not akceptabele\n");

}

}

}

**Код файла third\_interface.c:**

#include <stdio.h>

#include <stdlib.h>

#include <stdbool.h>

int get\_int(void)

{

char temprem, tempclear;

int input = 0;

while(true)

{

temprem=0;

tempclear=0;

if((!scanf("%d%c",&input ,&temprem))|| temprem != '\n')

{

printf(" - Error: Invalid value for int variables.\nOne more time: ");

while(tempclear != '\n')

scanf("%c",&tempclear);

}

else

return input;

}

}

void output(int array[], int ARRAY\_LEN)

{

printf("Array: [ %d", array[0]);

for (int i = 1; i < ARRAY\_LEN; i++)

{

printf(", ");

printf("%d", array[i]);

}

printf("]\n");

}

int\* bind\_values(int array[], bool rand\_indicator, int ARRAY\_LEN)

{

for (int i = 0; i < ARRAY\_LEN; i++){

if (rand\_indicator){

array[i] = -99 + rand()%(100+98);

} else {

printf(" - Specify %d element of Array: ", i+1);

array[i] = get\_int();

}

}

return array;

}

**Код файла third\_algorithm.c:**

int algorithm(int first\_value, int second\_value)

{

return first\_value + second\_value;

}

int fold(int array[], int array\_len, int (\*sum)(int, int))

{

int result\_sum = 0;

for (int i = 0; i < array\_len - 1; i+=2)

{

result\_sum += sum(array[i], array[i+1]);

}

return result\_sum;

}

**Код файла third\_algorithm.h:**

int fold(int array[], int array\_len, int (\*sum)(int, int));

int algorithm(int first\_value, int second\_value);

**Код файла third\_interface.h:**

#include <stdbool.h>

#include <stdlib.h>

#include <stdio.h>

int get\_int(void);

void output(int array[], int ARRAY\_LEN);

int\* bind\_values(int array[], bool rand\_indicator, int ARRAY\_LEN);

## 3.6 Тестирование

Тестирование было проведено на рандомных значениях:

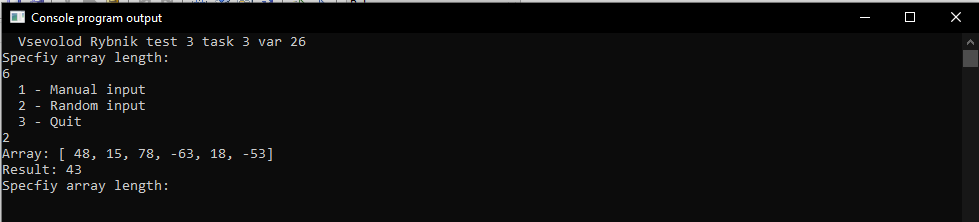


Рисунок 6 – Экранная копия результата работы разработанной программы задания №3

## 3.7 Выводы по заданию №3

В ходе выполнения задания были изучены способы передачи функции в качестве аргумента в другом методе.