МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Воронежский государственный технический университет»

**Факультет** экономики, менеджмента и информационных технологий

**Кафедра** систем управления и информационных технологий в строительстве

**Отчет по лабораторной работе**

Тема: **«Передача массива в функцию»**

По дисциплине: Основы программирования и алгоритмизации

Выполнил студент: Гладнева Евгения Геннадьевна

Группа: бИСТ-222

Руководитель: доцент, к.т.н. Курипта О. В.

Работа защищена « » \_\_\_\_\_\_\_\_\_ 2022г.

С оценкой \_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись)

Воронеж 2022

**Постановка задачи**

**Условие задачи:** написать программу с двумя функциями, возвращающими индекс и значение последнего положительного элемента.

**Исходные данные:**

XMIN -1000, XMAX 1000

**Алгоритм решения:**

1. Спрашиваем у пользователя количество элементов в массиве и выделяем соответствующее количество памяти.
2. Заполняем массив рандомными значениями.
3. В обоих функциях в обратном цикле ищем первое положительное значение и возвращаем индекс или значение.
4. Выводим элемент.

**Словесный алгоритм**

**Алгоритм представлен пошаговой детализацией:**

**Шаг 1:** установка директивы препроцессора для функции scanf, подключение необходимых заголовочных файлов, определение констант и прототипов функций

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <stdio.h>

#include <stdlib.h>

#include <locale.h>

#include <time.h>

#define XMIN -1000

#define XMAX 1000

void full\_elements(double\* ptr\_array, int n);

void put\_elements(double\* ptr\_array, int n);

int i\_last\_positive(double\* ptr\_array, int n);

double last\_positive(double\* ptr\_array, int n);

**Шаг 2:** объявление функции main

void main(void) {};

**Шаг 3**: функция заполнения массива

void full\_elements(double\* ptr\_array, int n) {

int stime = (unsigned)time(NULL) / 2;

srand(stime);

for (int i = 0; i < n; i++)

ptr\_array[i] = XMIN + 1.f \* (XMAX - XMIN) \* rand() / RAND\_MAX;

}

**Шаг 4**: функция печать массива

void put\_elements(double\* ptr\_array, int n){

for (int i = 0; i < n; i++)

printf("A[%d]: %lf\n", i, ptr\_array[i]);

}

**Шаг 5:** функция поиска индекса последнего положительного элемента

int i\_last\_positive(double\* ptr\_array, int n) {

for (int i = n - 1; i >= 0; i--)

{

if (ptr\_array[i] > 0)

return i;

}

return -1;

}

**Шаг 6:** функция поиска значения последнего положительного элемента

double last\_positive(double\* ptr\_array, int n) {

for (int i = n - 1; i >= 0; i--)

{

if (ptr\_array[i] > 0)

return ptr\_array[i];

}

return -1;

}

**Шаг 7:** установка локализации в функции main

setlocale(LC\_ALL, "RUS");

**Шаг 8:** объявление переменных

int size, index;

double elem;

**Шаг 9:** ввод кол-ва элементов массива и выделение памяти

double\* array = malloc(size \* sizeof(double));

**Шаг 10**: заполнение массива значениями и печать

full\_elements(array, size);

put\_elements(array, size);

**Шаг 11**: взаимодействие с функциями

index = i\_last\_positive(array, size);

if (index != -1)

{

elem = last\_positive(array, size);

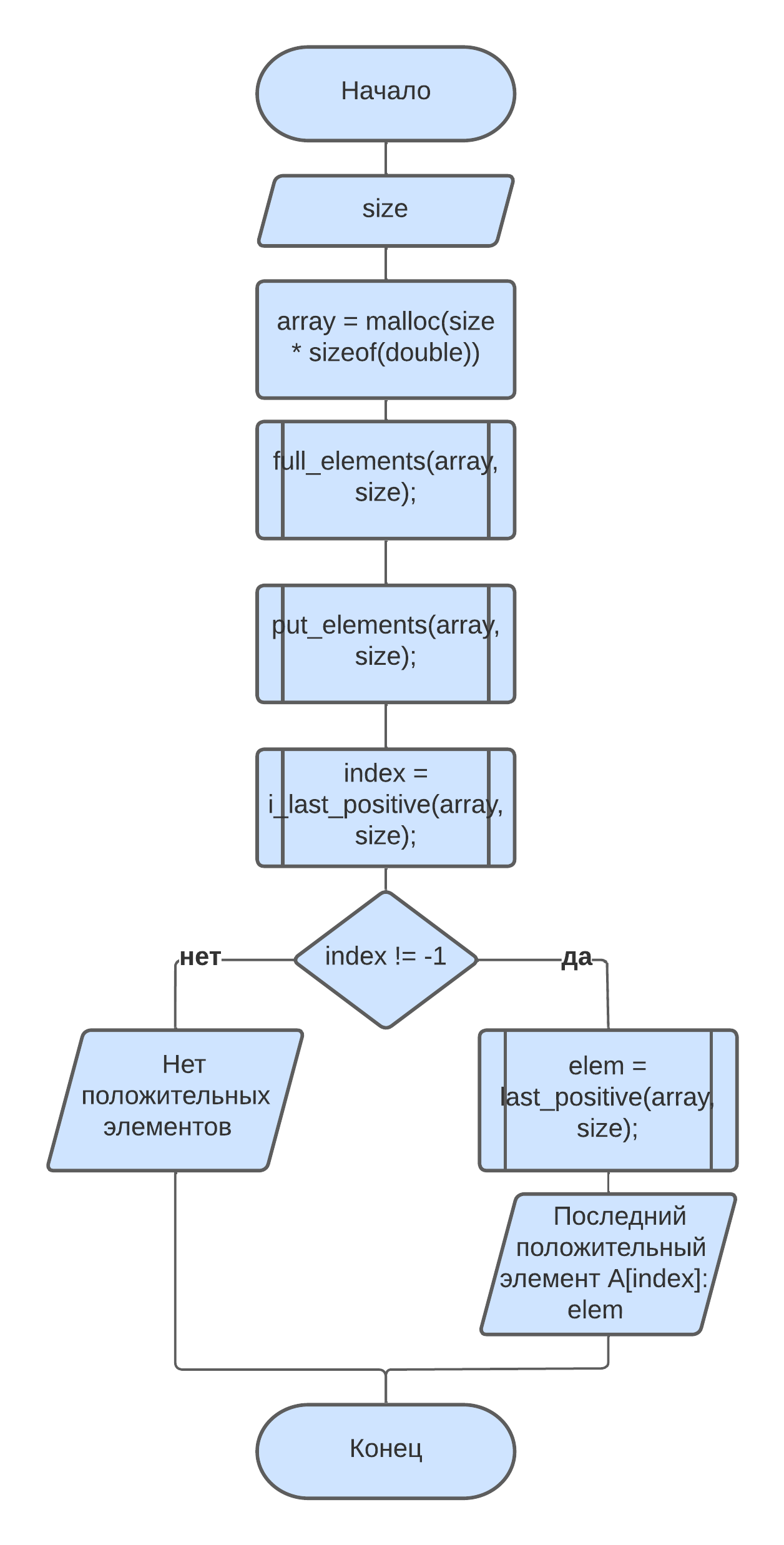
printf("Последний положительный элемент А[%d]: %lf\n", index, elem);

}

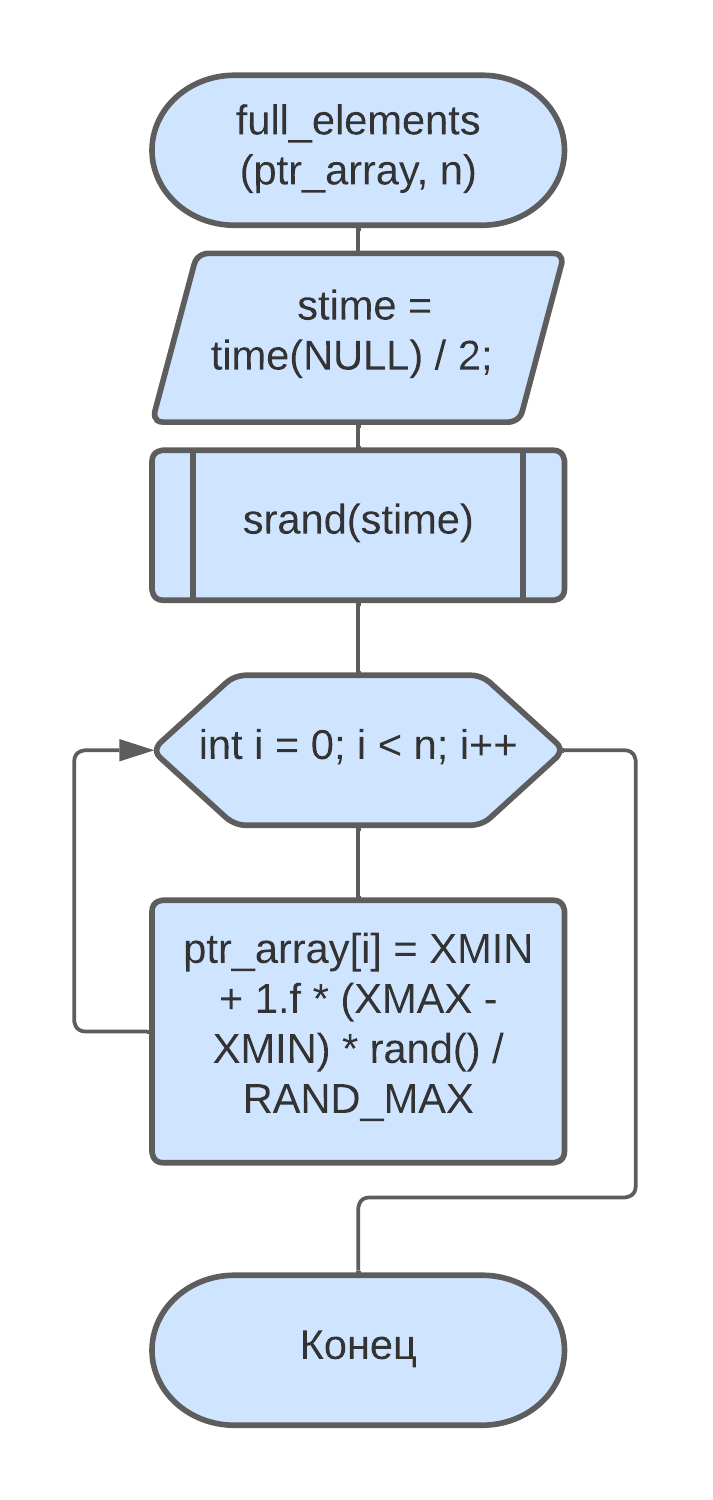
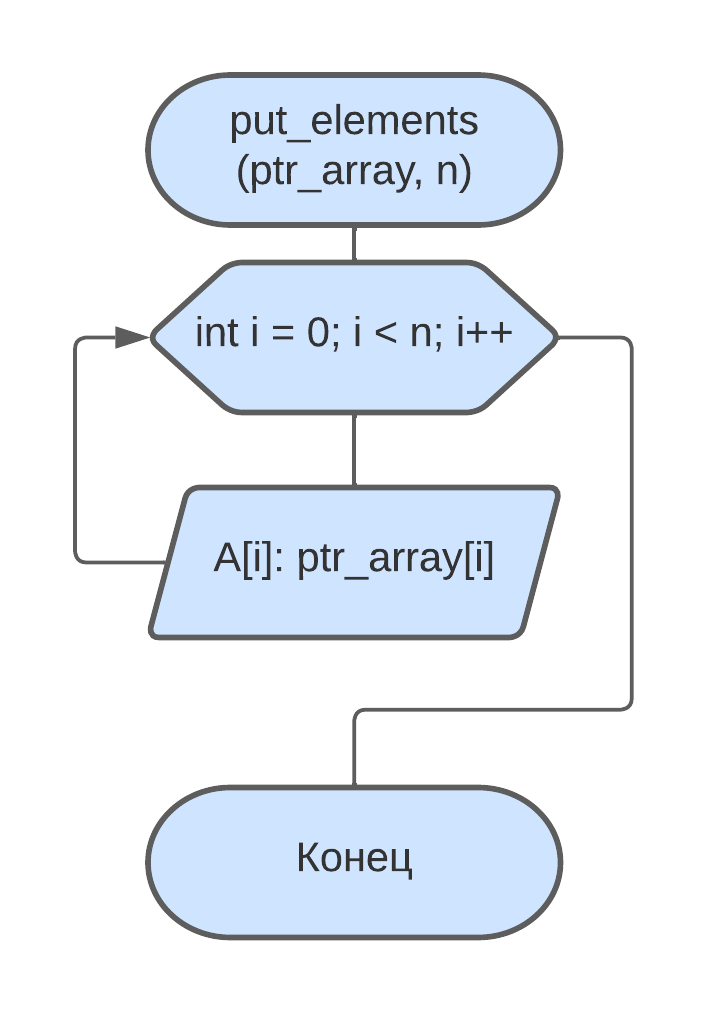
else

printf("Нет положительных элементов\n");

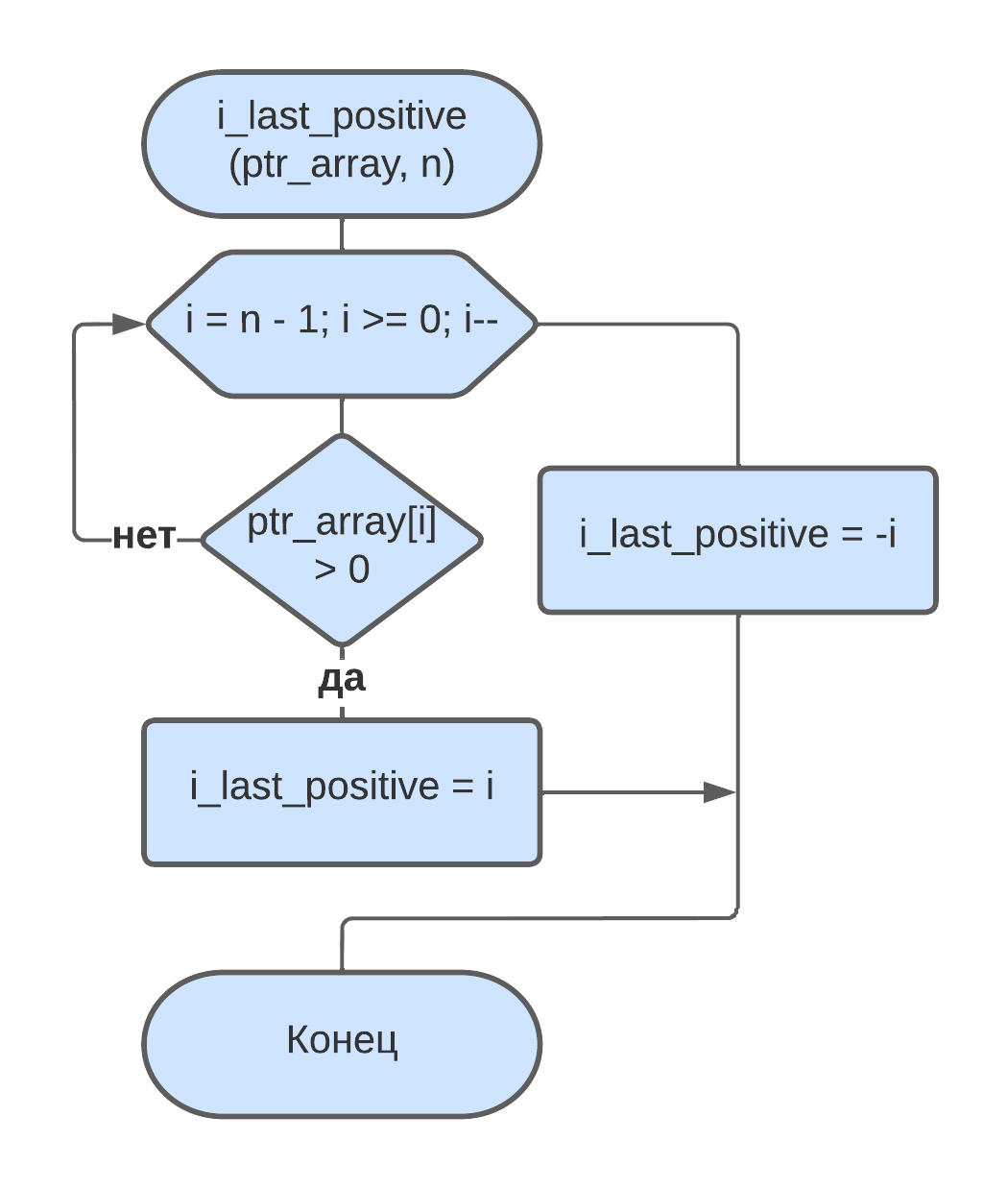
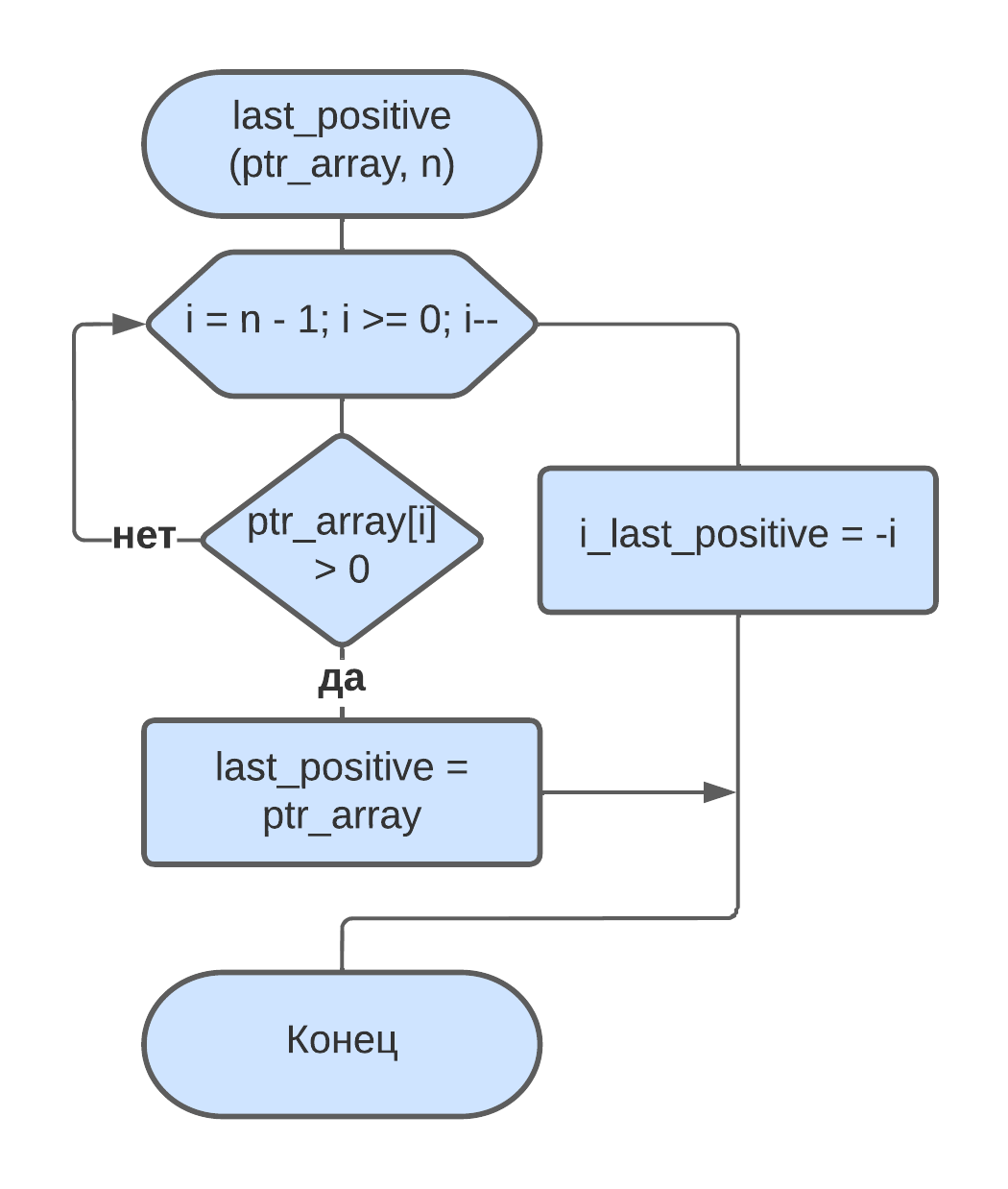
**Блок – схема программы**

Блок – схема программы представлена на рисунке 1.

а)



в)

 б)

г) д)

а) — блок-схема головного алгоритма;

б) — блок-схема функции full\_elements;

в) — блок-схема функции put\_elements;

г) — блок-схема функции i\_last\_positive;

д) — блок-схема функции last\_positive;

Рисунок 1 — Блок-схема программы

**Результат работы программы**

На рисунке 2 представлено окно ввода данных

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 2 — Окно ввода данных

На рисунке 3 представлена работа программы.

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 3 — Окно с выполненной программой

**Код программы**

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <stdio.h>

#include <stdlib.h>

#include <locale.h>

void full\_elements(double\* ptr\_array, int n); // ф-ция берётся из файла заданий

void put\_elements(double\* ptr\_array, int n); // ф-ция берётся из файла заданий

int i\_last\_positive(double\* ptr\_array, int n);

double last\_positive(double\* ptr\_array, int n);

void main()

{

setlocale(LC\_ALL, "RUS");

int size, index;

double elem;

printf("Введите размер массива\n>");

scanf("%d", &size);

double\* array = malloc(size \* sizeof(double));

full\_elements(array, size);

put\_elements(array, size);

index = i\_last\_positive(array, size);

if (index != -1)

{

elem = last\_positive(array, size);

printf("Последний положительный элемент А[%d]: %lf\n", index, elem);

}

else

printf("Нет положительных элементов\n");

free(array);

}

// поиск индекса последнего положительного элемента

int i\_last\_positive(double\* ptr\_array, int n)

{

for (int i = n - 1; i >= 0; i--)

{

if (ptr\_array[i] > 0)

return i;

}

return -1;

}

// поиск значения последнего положительного элемента

double last\_positive(double\* ptr\_array, int n)

{

for (int i = n - 1; i >= 0; i--)

{

if (ptr\_array[i] > 0)

return ptr\_array[i];

}

return -1;

}

// заполнение массива случайными значениями

void full\_elements(double\* ptr\_array, int n)

{

int stime = (unsigned)time(NULL) / 2;

srand(stime);

for (int i = 0; i < n; i++)

ptr\_array[i] = XMIN + 1.f \* (XMAX - XMIN) \* rand() / RAND\_MAX;

}

// печать элементов массива

void put\_elements(double\* ptr\_array, int n)

{

for (int i = 0; i < n; i++)

printf("A[%d]: %lf\n", i, ptr\_array[i]);

}