1. Лабораторная работа № 3

«Программирование алгоритмов обработки одномерных массивов»

* 1. Цель работы

Изучение особенностей представления и обработки одномерных массивов в языках С/С++ с учетом связи указателей и массивов. Получение практических навыков реализации алгоритмов обработки одномерных массивов средствами языков С/С++. Исследование особенностей обработки одномерных динамических массивов.

## Постановка задачи

Получить навыки работы с одномерными массивами, с указателями в языках программирования C/C++. Найти взаимосвязь между указателем и массивом.

* 1. Ход выполнения работы

Вариант выполнения задания – 18.

* + 1. Описание алгоритма решения задачи
       1. Определение входных и выходных данных

В главном меню программы подаётся число от 1 до 5. При вводе другого числа или символа программа выдаст ошибку.

При вводе массива от пользователя ожидается ввод натурального числа n – количество элементов в массиве – и n целых чисел.

При сортировке массива программа принимает от пользователя натуральные числа p и q, соответствующие условию: .

При выводе массива программа выводит значения ячеек массива в табличном виде.

* + - 1. Структурная схема алгоритма

Структурная схема программы представлена на Рисунках 3.1 – 3.3 .

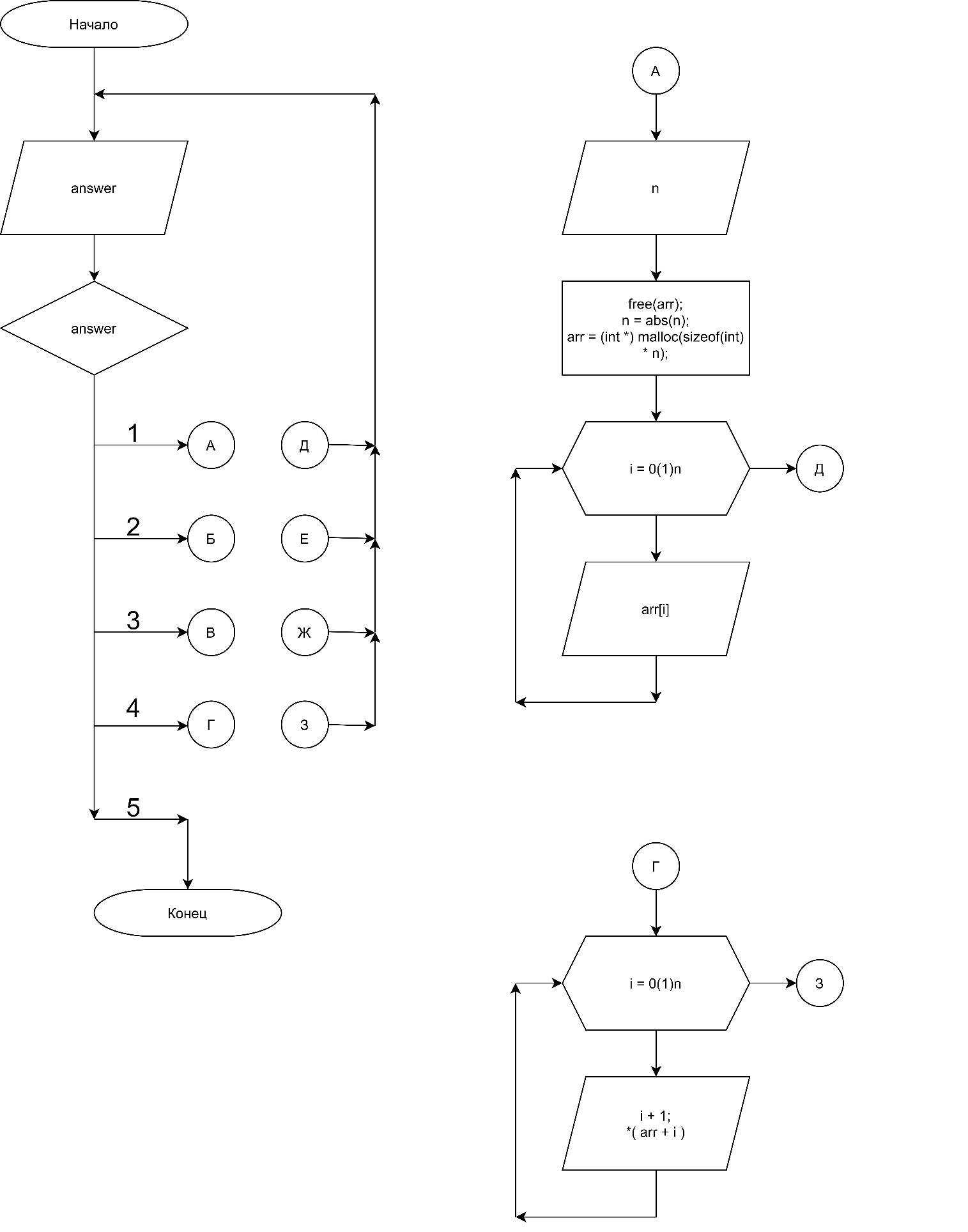


Рисунок 3.1 – Структурная схема программы, часть 1

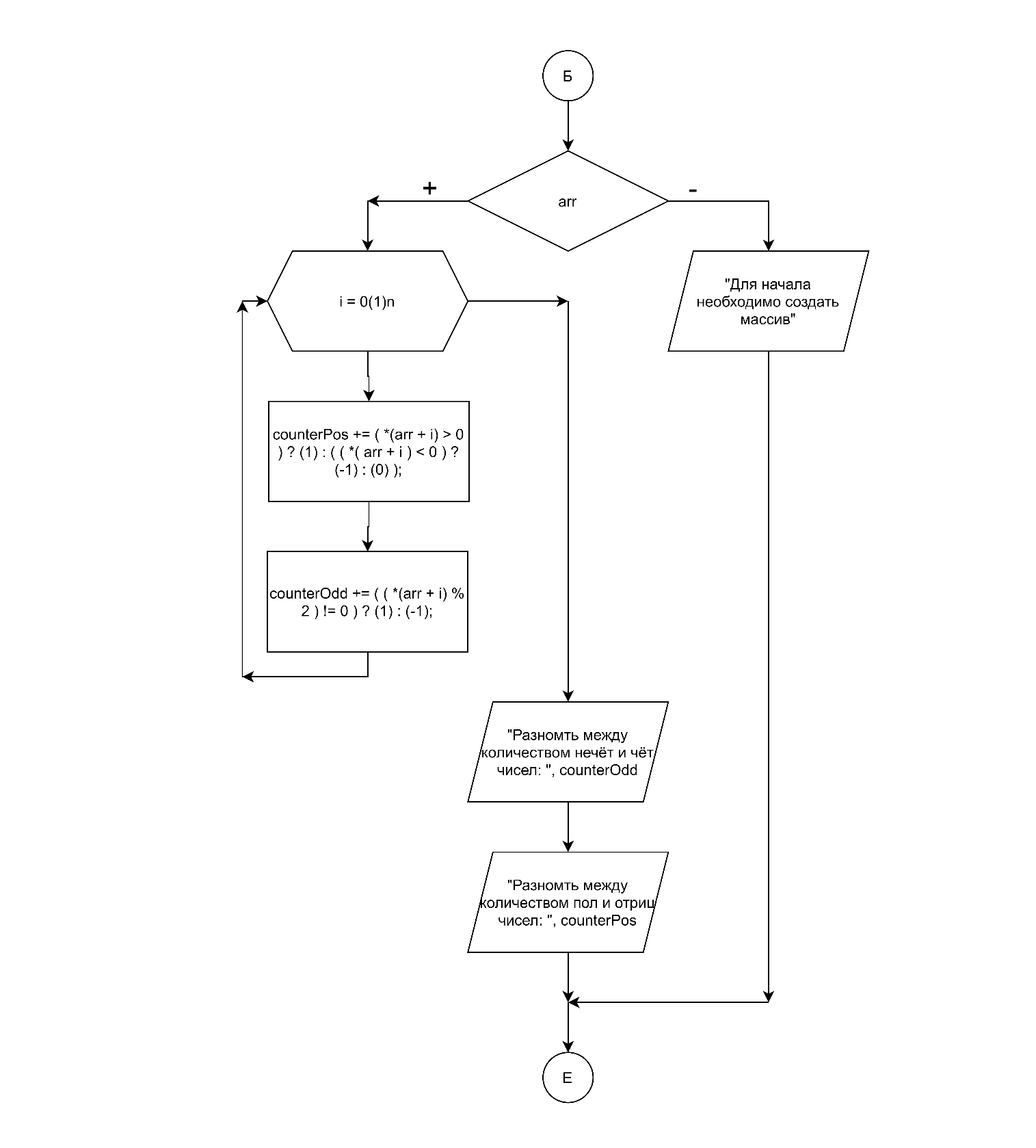


Рисунок 3.2 – Структурная схема программа, часть 2

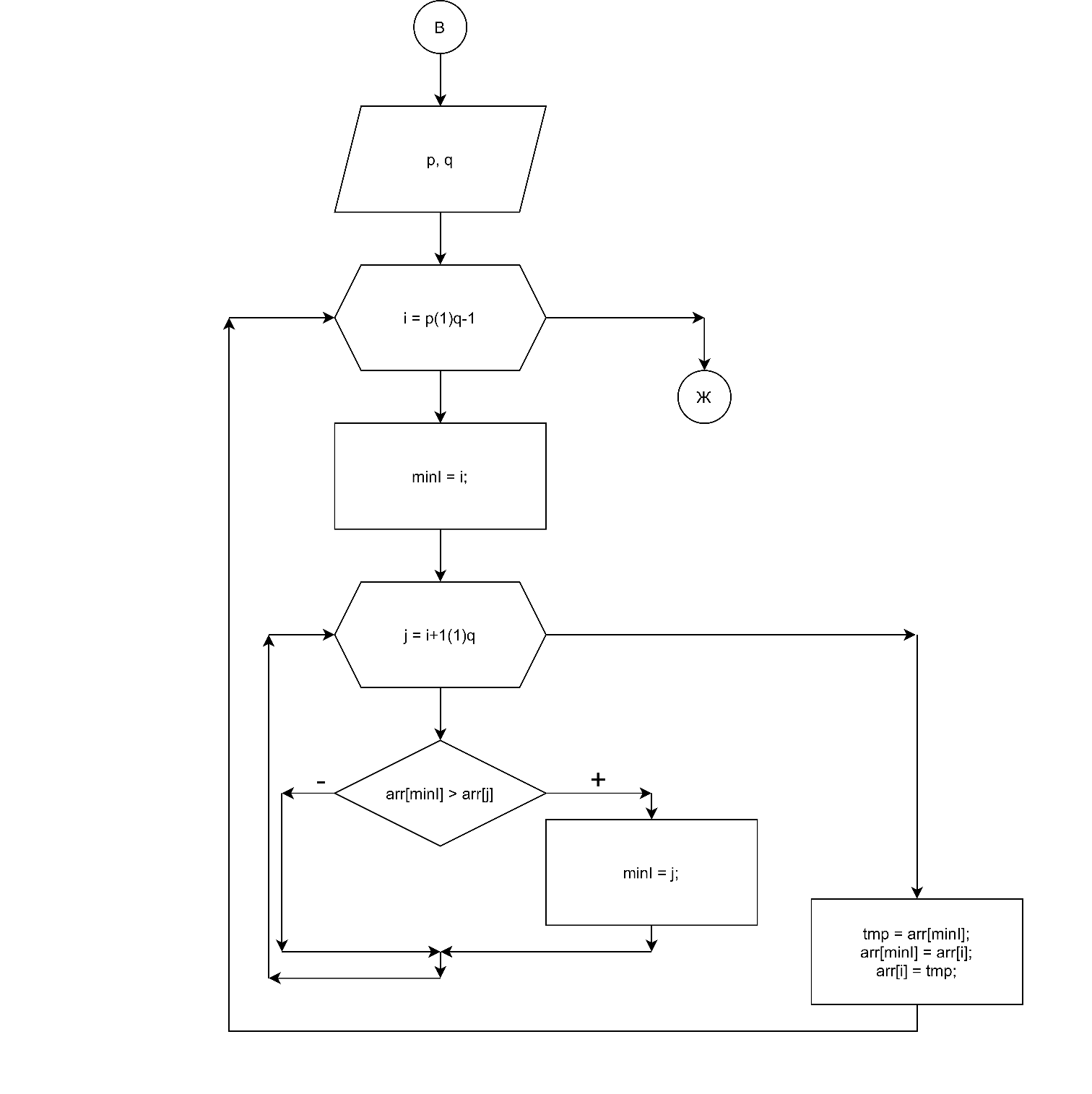


Рисунок 3.3 – Структурная схема программы, часть 3

* + 1. Текст программы (ЯП C, операционная система GNU/Linux Ubuntu)

#include <stdio.h>

#include <stdlib.h>

#define CLRSCR() fflush(NULL); system("clear")

#define pause(a) fflush(NULL); printf("Для продолжения нажмите %c ...\n", a); while (getchar() != a);

#define printTitle(title) printf("Запущена функция: %s ...\n\n", title)

int main ()

{

int n = 0, counterPos = 0, counterOdd = 0;

int p = 0, q = 0;

int \*arr = 0;

char answer[64];

// Начало программы

printf("Вас приветствует программа обработки одномерных массивов.\n");

pause('=');

do

{

CLRSCR();

printf("Выберите действие:");

printf("\n%2d) %s.", 1, "Заполнить массив");

printf("\n%2d) %s.", 2, "\*my\* Выполнить обработку");

printf("\n%2d) %s.", 3, "\*my\* Выполнить сортировку");

printf("\n%2d) %s.", 4, "Вывести массив на экран");

printf("\n%2d) %s.", 5, "Удалить массив из памяти и выйти");

printf("\n%s ", "$:");

scanf("%64s", answer);

printf("\n<=============================>\n\n");

switch ( \*answer )

{

case '1':

/\* Заполнение массива \*/

free(arr);

printTitle("Создание одномерного массива");

printf("Для начала введите количество элементов в массиве: "); scanf("%d", &n);

n = abs(n);

arr = (int \*) malloc(sizeof(int) \* n);

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

{

printf("Введите %d элемент массива: ", i + 1);

scanf("%d", arr + i);

}

printf("Отлично! Создан новый массив, имеющий размер: %d (байт).\n", (int) sizeof(int) \* n);

break;

case '2':

/\* Выполнение обработки \*/

printTitle("Обработка массива (по варианту 18)");

if (arr)

{

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

{

counterPos += ( \*(arr + i) > 0 ) ? (1) : ( ( \*( arr + i ) < 0 ) ? (-1) : (0) );

counterOdd += ( ( \*(arr + i) % 2 ) != 0 ) ? (1) : (-1);

}

printf("Была произведена обработка массива. Известно, что:\n");

// Вывод результата с полож/отриц

if (counterPos > 0)

{

printf(" \* положительных чисел больше отрицательных на %d.\n", counterPos);

}

else if (counterPos < 0)

{

printf(" \* положительных чисел меньше отрицательных на %d.\n", abs(counterPos));

}

else

{

printf(" \* количество положительных чисел равно количеству отрицательных.\n");

}

// Вывод результата с чёт/нечёт

if (counterOdd > 0)

{

printf(" \* нечётных чисел больше на %d.\n", counterOdd);

}

else if (counterOdd < 0)

{

printf(" \* нечётных чисел меньше на %d.\n", abs(counterOdd) );

}

else

{

printf(" \* количество нечётных равно количеству чётных.\n");

}

} else {

printf("Входящий массив пуст. Рекомендуется для начала создать массив.\n");

}

break;

case '3':

/\* Выполнение сортировки\*/

printTitle("Сортировка массива методом прямого выбора");

if (n >= 4)

{

printf("Для начала введите число p (1 <= p < %d): ", n);

scanf("%d", &p);

while ( (p < 1) || (n < p) )

{

printf(" \* Повторяю: введите число p в интервале 1 <= p < %d: ", n); scanf("%d", &p);

}

printf("Теперь введите число q (%d < q <= %d): ", p, n);

scanf("%d", &q);

while ( (q < p) || (n <= q) )

{

printf(" \* Повторяю: введите число q в интервале %d < q <= %d: ", p, n); scanf("%d", &q);

}

q -= 1; p -= 1;

int minI, tmp;

for (int i = p; i < q - 1; i++)

{

minI = i;

for (int j = i + 1; j < q; j++)

{

if ( \*( arr + minI ) > \*( arr + j ) )

{

minI = j;

}

}

tmp = \*( arr + minI );

\*( arr + minI ) = \*( arr + i );

\*( arr + i ) = tmp;

}

printf("Массив был успешно отсортирован.\n");

}

else

{

printf("Невозможно обработать массив: массив должен состоять как минимум из четырёх элементов.");

}

break;

case '4':

/\* Вывод массива \*/

printTitle("Печать массива в виде массива");

if (arr)

{

printf("+-----+------------+--------+--------+\n");

printf("|Номер| Знач ячейки|Положит?| Чётное?|\n");

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

{

printf("+-----+------------+--------+--------+\n");

printf("| %3d | %10d | %s | %s |\n",

i + 1,

\*( arr + i ),

( \*(arr + i) > 0 ) ? (" Да ") : ( ( \*( arr + i ) < 0 ) ? (" Нет ") : (" Нуль ") ),

( ( \*(arr + i) % 2 ) == 0 ) ? (" Да ") : (" Нет ")

);

}

printf("+-----+------КОНЕЦ-+--------+--------+\n");

}

else

{

printf("Входящий массив пуст. Заполните его!\n");

}

break;

case '5':

/\* Выход из программы \*/

printf("Выход из программы...\n");

break;

default:

printf("\nВведён неизвестный символ: \"%c\". Повторите ввод.\n", \*answer);

break;

}

pause('=');

} while ( (\*answer) != '5');

CLRSCR();

}

* + 1. Описание тестовых примеров

Тестовые примеры продемонстрированы на Рисунках 3.4 ­– 3.9 .

На Рисунке 3.4 показано главное меню программы, где пользователь выбирает действие над массивом.

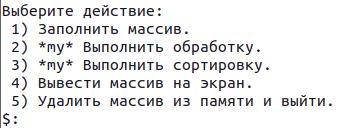


Рисунок 3.4 – Главное меню программы

На Рисунке 3.5 Продемонстрировано заполнение массива.

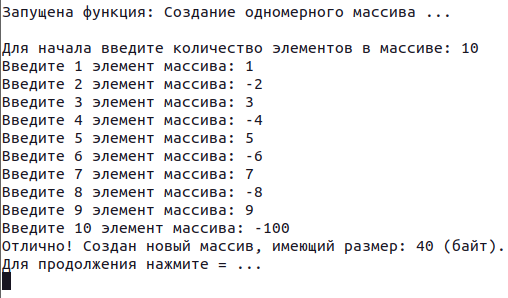


Рисунок 3.5 – Заполнение одномерного массива данными

На Рисунке 3.6 продемонстрирован вывод введённого массива в табличном виде. Таблица выводит значение ячейки массива и характеристику числа, записанного в данную ячейку.

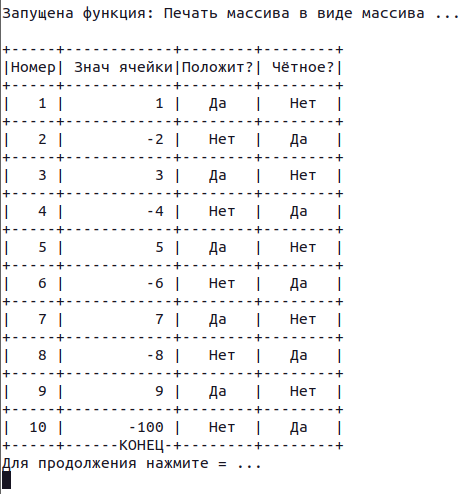


Рисунок 3.6 – Вывод массива на экран

На Рисунке 3.7 Продемонстрирована работоспособность функции обработки массива – сравнение количества чётных и нечётных элементов, количества положительных и отрицательных.

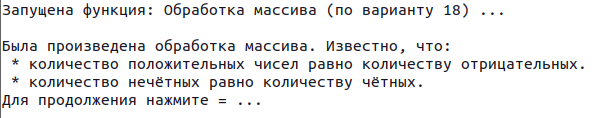


Рисунок 3.7 – Обработка данных в массиве

На Рисунке 3.8 Продемонстрировано окно ввода дополнительных данных для сортировки массива, а на Рисунке 3.9 ­– печать результата сортировки.

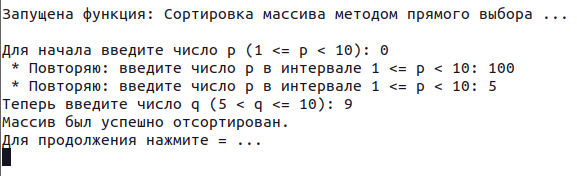


Рисунок 3.8 – Окно взаимодействия с пользователем для сортировки массива

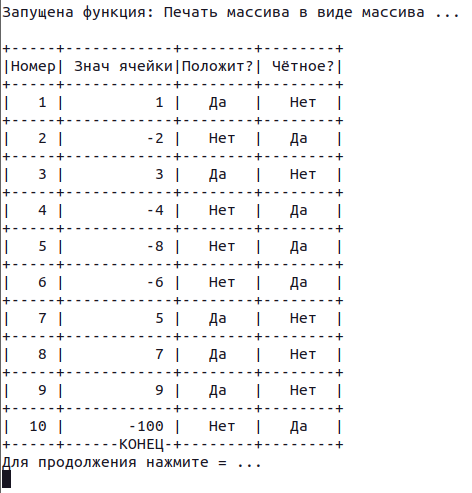


Рисунок 3.9 – Печать массива, у которого отсортированы по возрастанию элементы с 5 по 9

## Вывод

При выполнении данной лабораторной работы были получены навыки работы с одномерными массивами данных и указателями. Практическим способом выявлена взаимосвязь указателей с элементами массива. Были закреплены навыки создания меню программы, сортировки данных, в частности повторно освоена сортировка методом прямого выбора. Полученные во время разработки навыки помогут разрабатывать более эффективные алгоритмы сортировки данных, оптимизировать производительность программы за счёт использования указателей.