|  |
| --- |
|  |
| МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ |
| Федеральное государственное бюджетное образовательное учреждение высшего образования  "**МИРЭА - Российский технологический университет"**  **РТУ МИРЭА** |

Институт искусственного интеллекта

Кафедра общей информатики

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ОТЧЕТ**  **ПО ПРАКТИЧЕСКОЙ РАБОТЕ № 12:**  **элементы алгоритмизации и процедурного программирования** | | | | |
| **по дисциплине** | | | |  |
| **«**ИНФОРМАТИКА**»** | | | |  |
| Выполнил студент группы *ИКБО-08-21* | | | *Пономарев М.Д.* | |
| Принял  *Старший преподаватель* | | | *Смирнов С.С.* | |
| Практическая | « » 2021 г. | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | | |
| работа выполнена |  | (подпись студента) | | |
| «Зачтено» | « » 2021 г. | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | | |
|  |  | (подпись руководителя) | | |

Москва 2021

СОДЕРЖАНИЕ

[**1** **ПОСТАНОВКА ЗАДАЧИ** 3](#_Toc89809617)

[**2** **ПРОЕКТИРОВАНИЕ И РЕАЛИЗАЦИЯ** 4](#_Toc89809618)

[**2.1 Блок-схемы алгоритмов программы (по ГОСТу)** 4](#_Toc89809619)

[**2.2 Структурированный код программы с комментариями** 7](#_Toc89809620)

[**2.3 Примеры тестирования, доказывающие работоспособность программы** 10](#_Toc89809621)

[**3** **ВЫВОДЫ** 12](#_Toc89809622)

[**4** **ИНФОРМАЦИОННЫЕ ИСТОЧНИКИ** 13](#_Toc89809623)

1. **ПОСТАНОВКА ЗАДАЧИ**

Требуется разработать блок-схему алгоритма и написать программу обработки данных в соответствии с вариантом №24. При этом требуется контролировать типы и диапазоны вводимых данных, а также предусмотреть обработку других исключительных ситуаций (если они есть). Блок-схема должна быть полной, т. е. должна описывать и процесс диалога с пользователем, и контроль вводимых данных, и подпрограммы вычислений с обработкой возможных исключительных операций. Блок-схема должна изображаться по ГОСТу. При обнаружении ошибки ввода или ошибки вычислений программа должна информативно уведомлять пользователя о причине ошибки. Если ошибка произошла на этапе ввода данных, то программа должна просить пользователя повторить ввод.

Условие задачи:

Создать квадратную матрицу размера 8 на 8. Матрица заполняется случайными целыми числами в диапазоне от 1 до 100. На матрицу накладывается разметка, соответствующая шахматной доске. Пользователь выбирает цвет клеток, с которыми будет происходить дальнейшая работа. Требуется из элементов матрицы, стоящих на клетках заданного цвета, сформировать одномерный массив и упорядочить его методом быстрой сортировки по возрастанию. Результаты преобразований вывести на экран.

1. **ПРОЕКТИРОВАНИЕ И РЕАЛИЗАЦИЯ**

**2.1 Блок-схемы алгоритмов программы (по ГОСТу)**

Составим блок-схемы алгоритмов заданной программы в соответствии со стандартом ГОСТ. На схемах присутствуют сопроводительные комментарии. Изображения блок-схем приведены на рис. 1–3.

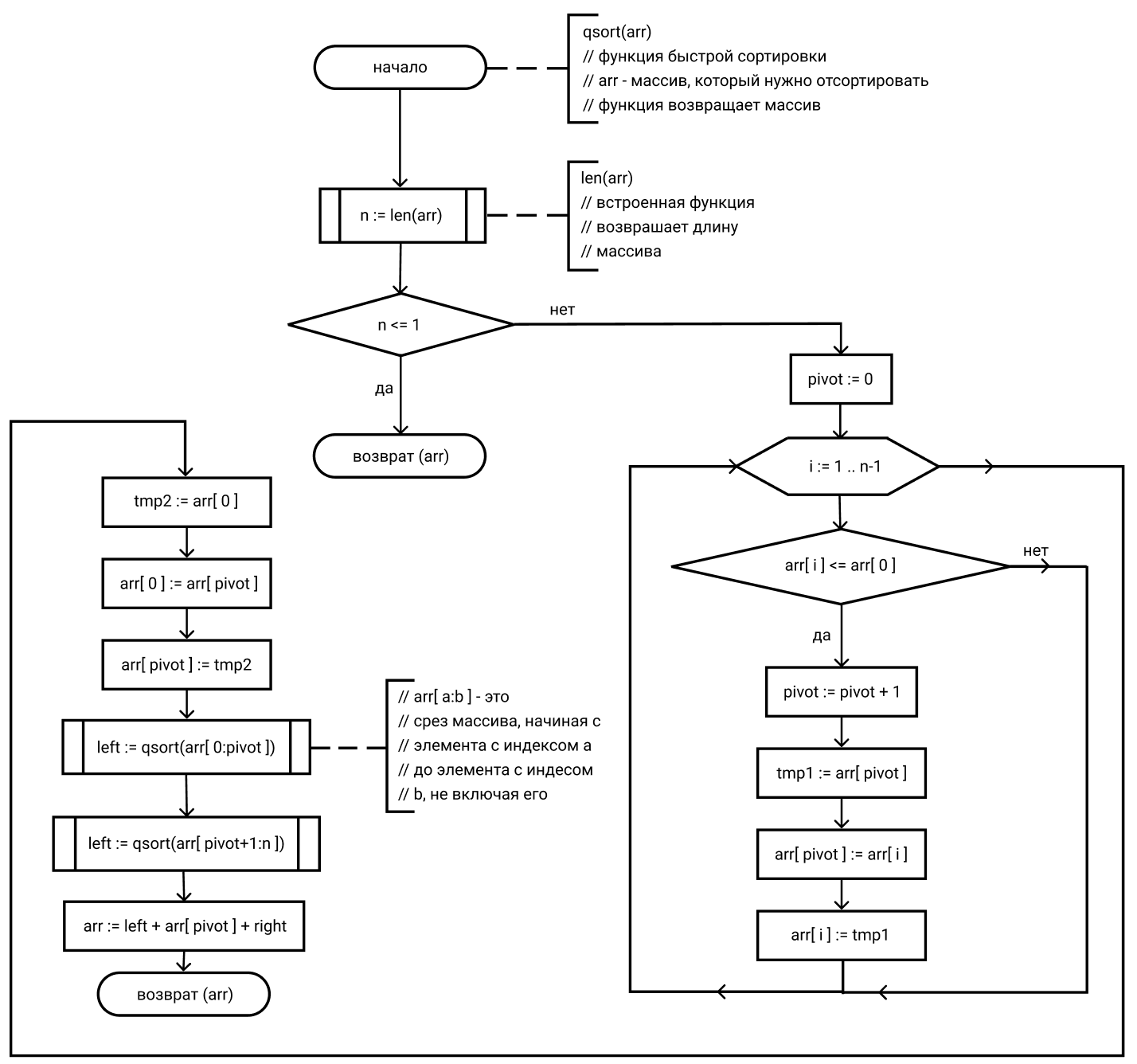
******

Рисунок 1 – Блок-схемы алгоритмов программы (часть 1)

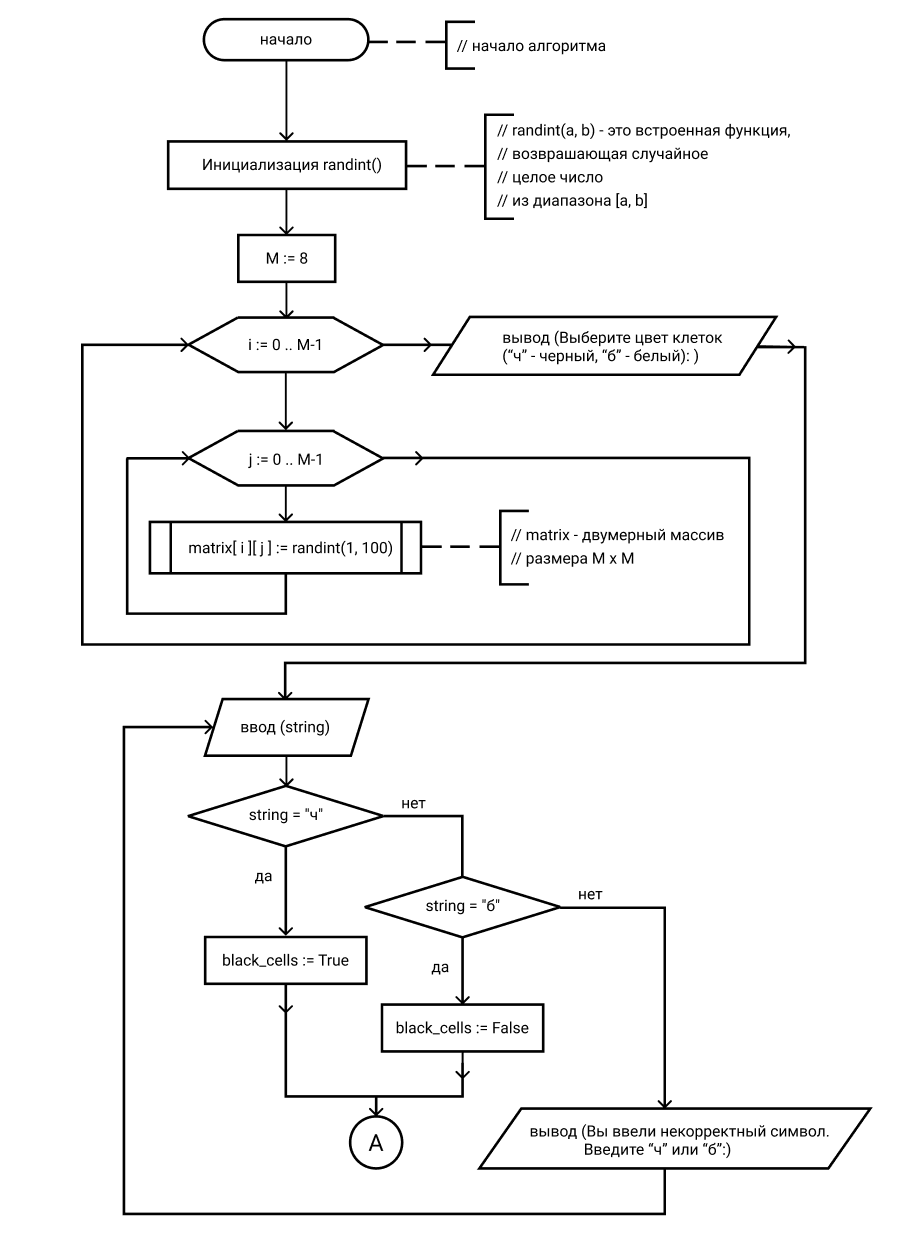


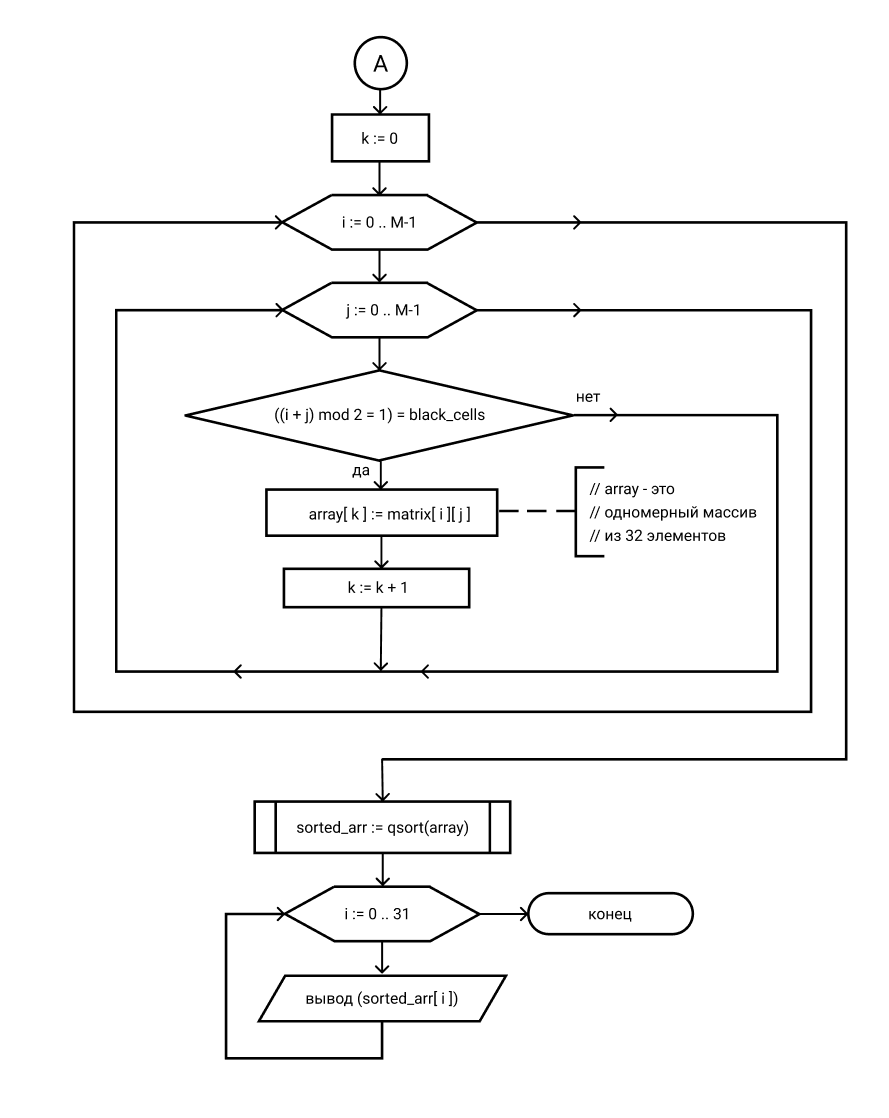
Рисунок 2 – Блок-схемы алгоритмов программы (часть 2)******

Рисунок 3 – Блок-схемы алгоритмов программы (часть 3)

**2.2 Структурированный код программы с комментариями**

Далее продемонстрируем структурированный код данной программы, сопровождающийся пояснительными комментариями. Код реализован на языке программирования Python, поддерживающем структурно-модульный подход.

Описание программы.

Вначале подключим функцию «randint» из библиотеки «random». Затем создадим подпрограмму-функцию «qsort», выполняющую сортировку произвольного одномерного массива методом быстрой сортировки.

Функция «qsort» реализована следующим образом: сначала проверяется длина массива, который функция получает на вход, и если она меньше 2, то функция возвращает массив без изменений. Иначе создается переменная «pivot», равная 0. Будем называть элемент с индексом «pivot» опорным. Затем с помощью цикла мы пробегаем по всем элементам массива, и если встречаем элемент меньше первого элемента массива, то увеличиваем «pivot» на 1 и меняем местами вышеуказанный элемент с опорным. После выхода из цикла меняем местами опорный элемент и первый элемент массива. Затем рекурсивно вызывается функция «qsort» сначала для среза исходного массива с первого элемента по опорный, не включая его (назовем массив, который она вернет левым), а затем для среза, начиная с элемента, идущего после опорного, и до конца массива (его назовем правым массивом). После возвращения массивов из рекурсивных вызовов функция «qsort» возвращает массив, «склеенный» из левого массива, опорного элемента и правого массива.

Часть алгоритма, описанного выше, представлена на рис. 4.

***Изображение выглядит как текст

Автоматически созданное описание***

Рисунок 4 – Код программы (часть 1)

Далее создается двумерный массив (квадратная матрица) размером 8 на 8. С помощью функции «randint» матрица заполняется случайными целыми числами в диапазоне от 1 до 100.

После этого пользователя просят ввести цвет клеток (символ «ч» для черного цвета, «б» для белого), на которых стоят числа из матрицы, если наложить на нее разметку, соответствующую шахматной доске, которые впоследствии будут сформированы в отдельный массив. Если пользователь вводит символ «ч», то создается булева переменная «black\_cells», равная «True», означающая, что пользователь ввел черный цвет. Иначе, если пользователь вводит символ «б», то переменная «black\_cells» создается, равная «False». При ошибке ввода программа уведомляет об этом пользователя, выводит причину ошибки и запрашивает повторный ввод пока не будет введено корректный символ (см. рис. 5).

***Изображение выглядит как текст

Автоматически созданное описание***

Рисунок 5 – Код программы (часть 2)

Далее формируется одномерный массив из элементов, стоящих на клетках, цвет которых совпадает с цветом, введенным пользователем. Для этого программа пробегает по заполненной случайными числами матрице и выбирает те, что стоят на клетке, сумма индексов которой четна (если выбраны белые клетки) или нечетна (если выбраны черные клетки). Выбранные элементы помещаются в одномерный массив «array».

Затем вызывается функция «qsort», на вход которой подается массив «array». После этого результат работы «qsort» (отсортированный массив) выводится на экран (см. рис. 6).

***Изображение выглядит как текст

Автоматически созданное описание***

Рисунок 6 – Код программы (часть 3)

**2.3 Примеры тестирования, доказывающие работоспособность программы**

Далее представим иллюстрации тестирования программы.

В первом и во втором случае (рис. 7 и 8 соответственно) пользователь выбирает цвет, и при попытках некорректного ввода, программа уведомляет его об ошибке и ее причине.

В первом случае показано, что при вводе символа «ч» программа вывела на экран массив, состоящий из элементов, стоящих на клетках черного цвета.

Во втором случае при вводе символа «б» программа вывела на экран массив, состоящий из элементов, стоящих на клетках белого цвета.

Результаты работы программы на обоих тестированиях выведены на экран, программа работает корректно.

***Изображение выглядит как текст

Автоматически созданное описание***

Рисунок 7 – Тестирование программы (часть 1)

***Изображение выглядит как текст

Автоматически созданное описание***

Рисунок 8 – Тестирование программы (часть 2)

1. **ВЫВОДЫ**

В ходе данной практической работы был разработан алгоритм выполнения заданной программы. Сначала были построены блок-схемы в соответствии с ГОСТом, описывающие суть алгоритма, а затем был написан код программы на языке программирования Python, поддерживающим процедурно-модульный подход.

В конце работы были проведены тестирования, показавшие корректную работу программы в зависимости от различных вводимых значений.

1. **ИНФОРМАЦИОННЫЕ ИСТОЧНИКИ**
2. Информатика: Методические указания по выполнению практических работ / С.С. Смирнов, Д.А. Карпов — М., МИРЭА — Российский технологический университет, 2020. – 102 с