содержание

ВВЕДЕНИЕ. АНАЛИЗ ЗАДАЧ ПРОЕКТИРОВАНИЯ 4

1 РАЗРАБОТКА И ОПИСАНИЕ ПОСЛЕДОВАТЕЛЬНОГО АЛГОРИТМА ПРОГРАМ-МНОЙ СИСТЕМЫ 5

1.1 Общий алгоритм программы 5

1.2 Алгоритм выделения границ черно-белого изображения 5

2 РАЗРАБОТКА ПРОГРАММНОЙ СРЕДЫ, РЕАЛИЗУЮЩЕЙ ПОСЛЕДОВАТЕЛЬ-НЫЙ АЛГОРИТМ ОБРАБОТКИ 7

3 РАЗРАБОТКА И ОБОСНОВАНИЕ ВАРИАНТА СХЕМЫ ПАРАЛЛЕЛИЗАЦИИ АЛ-ГОРИТМА 9

4 РАЗРАБОТКА ПРОГРАММНЫХ МОДУЛЕЦ СИСТЕМЫ ПАРАЛЛЕЛЬНОЙ ОБРА-БОТКИ 10

5 ТЕСТИРОВАНИЕ И СРАВНИТЕЛЬНЫЙ АНАЛИЗ ПРОИЗВОДИТЕЛЬНОСТИ ПРОГРАММЫ С ПОСЛЕДОВАТЕЛЬНОЙ И ПАРАЛЛЕЛЬНОЙ ОБРАБОТКОЙ 12

ЗАКЛЮЧЕНИЕ 15

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 16

ПРИЛОЖЕНИЕ А ТЕКСТ ПРОГРАММЫ

ПРИЛОЖЕНИЕ Б ГРАФИЧЕСКИЙ МАТЕРИАЛ. СХЕМА АЛГОРИТМА ПРОГРАМ-МЫ ПОСЛЕДОВАТЕЛЬНОЙ ОБРАБОТКИ

ПРИЛОЖЕНИЕ В ГРАФИЧЕСКИЙ МАТЕРИАЛ. СХЕМА АЛГОРИТМА ПРОГРАМ-МЫ ПАРАЛЛЕЛЬНОЙ ОБРАБОТКИ

ПРИЛОЖЕНИЕ Г ГРАФИЧЕСКИЙ МАТЕРИАЛ. СХЕМА РЕСУРСОВ СИСТЕМЫ ПА-РАЛЛЕЛЬНОЙ ОБРАБОТКИ

# ВВЕДЕНИЕ. анализ задач проектирования

Ввиду постоянного совершенствования многопроцессорных систем и увеличения числа ядер в современных процессорах возрастает роль параллельных алгоритмов.

Параллельный алгоритм — алгоритм, который может быть реализован по частям на множестве различных вычислительных устройств с последующим объединением полученных результатов и получением корректного результата.

Целью данного курсового проекта было сравнение производительности компьютерных систем последовательной и параллельной обработки данных. Оценка производилась на основе затраченного времени на обработку изображения и качества обработки изображения.

В качестве входных данных было выбрано монохромное (черно-белое) изображение размером 4000 на 4000 пикселей в формате bmp. К изображению примелись следующие фильтры: выделение границ черно-белого изображения на основе дифференциального оператора Робертса (дифференциальный фильтр I). Для параллельной системы использовалось 6 процессов.

Ожидается прирост производительности без ухудшения качества выходного изображения при использовании параллельной системы обработки изображения по сравнению с использованием последовательной системы обработки изображения.

# 1 Разработка и описание последовательного алго-ритма программной системы.

* 1. **Общий алгоритм программы.**

Исходные данные: изображение для обработки.

Алгоритм:

1. Открытие исходного изображения.
2. Начало отсчета времени выполнения алгоритмов.
3. Создание матрицы пикселей изображения.
4. Выделение границ элементов черно-белого графического изображения на основе дифференциального оператора I.
5. Преобразование полученной матрицы изображения в выходное изображение.
6. Конец отсчета времени выполнения алгоритма.
7. Сохранение обработанного изображения.
8. Закрытие изображения.

Выходные данные: обработанное изображение.

**1.2. Алгоритм выделения границ элементов черно-белого графического изображения на основе дифференциального оператора (дифференциальный фильтр I).**

Исходные данные: матрица изображения для обработки.

Алгоритм:

1. Цикл по всем пикселям матрицы изображения:
   1. Пиксель = |A - D| + |C - B | (рис. 1.1).
2. Конец алгоритма.

Выходные данные: матрица обработанного изображения по алгоритму выделения границ черно-белого белого графического изображения на основе дифференциального оператора.

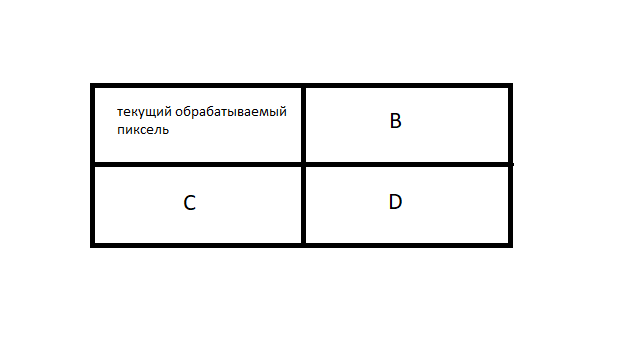


Рисунок 1.1 – Расположение пикселей B, C, D относительно текущего обрабатываемого пикселя

# 2 Разработка программной системы, реализующей последовательный алгоритм обработки.

Среда разработки: Microsoft Visual Studio 2022 Community.

Задание выполнялось на языке программирования С++.

Операционная система: Windows 10

Для выполнения задачи потребовалось подключение стандартных библиотек iostream, time, memory, string, stdio и сторонней библиотеки opencv.hpp:

iostream – стандартная библиотека ввода-вывода;

time – стандартная библиотека, содержащая функции для работы со временем и датой;

string – класс с методами и переменными для организации работы со строками в языке программирования C++, включенный в стандартную библиотеку;

opencv.hpp – сторонняя библиотека OpenCV для работы с изображениями;

**Реализованные пользовательские функции:**

Mat img – матрица входного изображения.

Mat outputImage – матрица выходного изображения.

Mat ConvolutionOperationRobert(Mat& src);

Входные параметры: имя исходного изображения.

Назначение: выделение границ элементов черно-белого графического изображения на основе дифференциального оператора (дифференциальный фильтр I).

Возвращаемые данные: нет.

# 3 Разработка и обоснование варианта схемы парал-лелизации алгоритма

Исходные данные: изображение для обработки.

Алгоритм:

1. Открытие исходного изображения.
2. Начало отсчета времени выполнения алгоритмов.
3. Создание матрицы пикселей изображения.
4. Создание 6 процессов.
5. Разделение изображения на 6 равных частей.
6. Для каждой из 6 частей изображения в отдельном процессе (рис. 3.1):
   1. Выделение границ элементов черно-белого графического изображения на основе дифференциального оператора.
7. Преобразование полученной матрицы изображения в выходное изображение.
8. Закрытие потоков.
9. Конец отсчета времени выполнения алгоритма.
10. Сохранение обработанного изображения.
11. Закрытие изображения.

Выходные данные: обработанное изображение.



Рисунок 3.1 – Схема параллелизации алгоритма

# 4 Разработка программных модулей системы пара-ллельной обработки данных.

Среда разработки: Microsoft Visual Studio 2022 Community.

Задание выполнялось на языке программирования С++.

Операционная система: Windows 10

Для выполнения задачи потребовалось подключение стандартных библиотек iostream, time, string и сторонней библиотеки opencv.hpp, mpi.h:

iostream – стандартная библиотека ввода-вывода;

time – стандартная библиотека, содержащая функции для работы со временем и датой;

string – класс с методами и переменными для организации работы со строками в языке программирования C++, включенный в стандартную библиотеку;

opencv.hpp – сторонняя библиотека OpenCV для работы с изображениями;

mpi.h – библиотека, добавляющая поддержку механизма передачи сообщений в стандартные языки программирования,

**Реализованные пользовательские данные и функции:**

Mat image – матрица входного изображения.

Mat outImage – матрица выходного изображения.

Size\_t imageTotalSize – общий размер матрицы входного изображения.

Size\_t imagePartialSize – размер матрицы подаваемый каждому процессу.

void ConvolutionOperationRobert(uchar\*& partialBuffer, size\_t imagePartialSize, int channels);

Входные параметры: матрица, передаваемая в каждый процесс, размер матрицы, количество каналов.

Назначение: выделение границ элементов черно-белого графического изображения на основе дифференциального оператора (дифференциальный фильтр I).

Возвращаемые данные: нет.

MPI\_Bcast(&imagePartialSize, 1, MPI\_UNSIGNED\_LONG\_LONG, 0, MPI\_COMM\_WORLD) – отправляет размер “imagePartialSize” каждому процессу.

MPI\_Barrier(MPI\_COMM\_WORLD) – синхронизирует процессы.

MPI\_Scatter(image.data, imagePartialSize, MPI\_UNSIGNED\_CHAR, partialBuffer, imagePartialSize, MPI\_UNSIGNED\_CHAR, 0, MPI\_COMM\_WORLD) – разбрасывает изображение между процессами.

MPI\_Gather(partialBuffer, imagePartialSize, MPI\_UNSIGNED\_CHAR, outImage.data, imagePartialSize, MPI\_UNSIGNED\_CHAR, 0, MPI\_COMM\_WORLD) – собирает полное изображение.

# 5 Тестирование и сравнительный анализ произво-дительности программы систем с последовательной и параллельной обработкой.

Среда тестирование – ноутбук, процессор Intel Core i7-8750H с частотой 2.2 ГГц, ОЗУ 16 ГБ, тип системы: 64-разрядная OC Windows10.

**Тест 1:** «Выделение границ черно-белого изображения (последовательная обработка)»

Ожидаемый результат: изображение, обработанное по алгоритму выделения границ черно-белого изображения за промежуток времени больший, чем при параллельной обработке

Описание: тестирование правильности обработки исходного изображения (рис 4.1, рис. 4.2) и времени выполнения обработки изображения при последовательной обработке (рис 4.3).

Полученный результат:



Рисунок 4.1 – Исходное изображение



Рисунок 4.2 – Выходное изображение

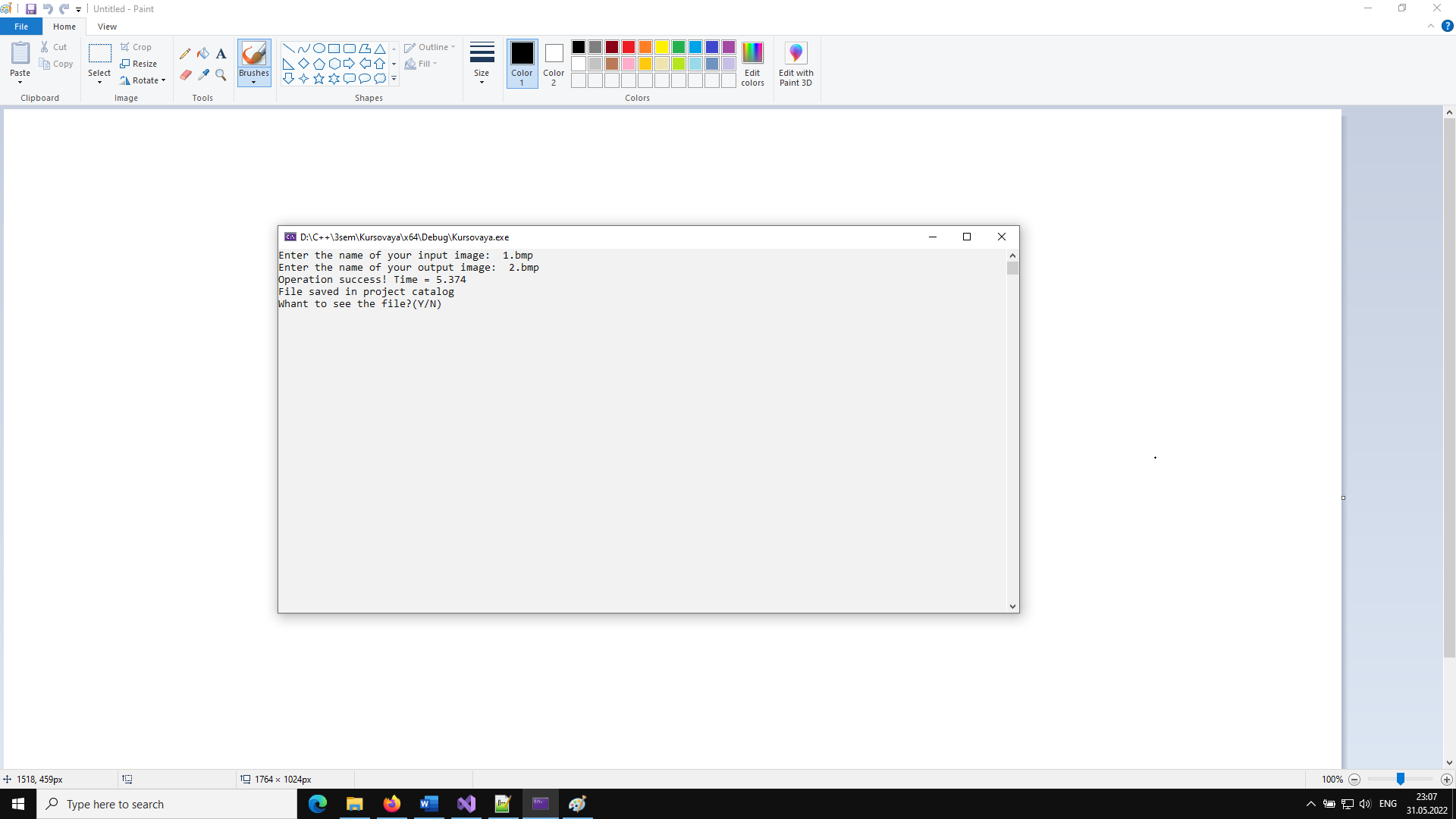


Рисунок 4.3 – Результаты тестирования времени выполнения

Вывод: результат соответствует ожиданиям.

**Тест 2:** «Выделение границ черно-белого изображения (параллельная обработка)»

Ожидаемый результат: изображение, обработанное по алгоритму выделения границ черно-белого изображения за промежуток времени меньший, чем при последовательной обработке

Описание: тестирование правильности обработки исходного изображения (рис 4.1, рис. 4.4) и времени выполнения обработки изображения при параллельной обработке (рис 4.5).

Полученный результат:



Рисунок 4.4 – Выходное изображение

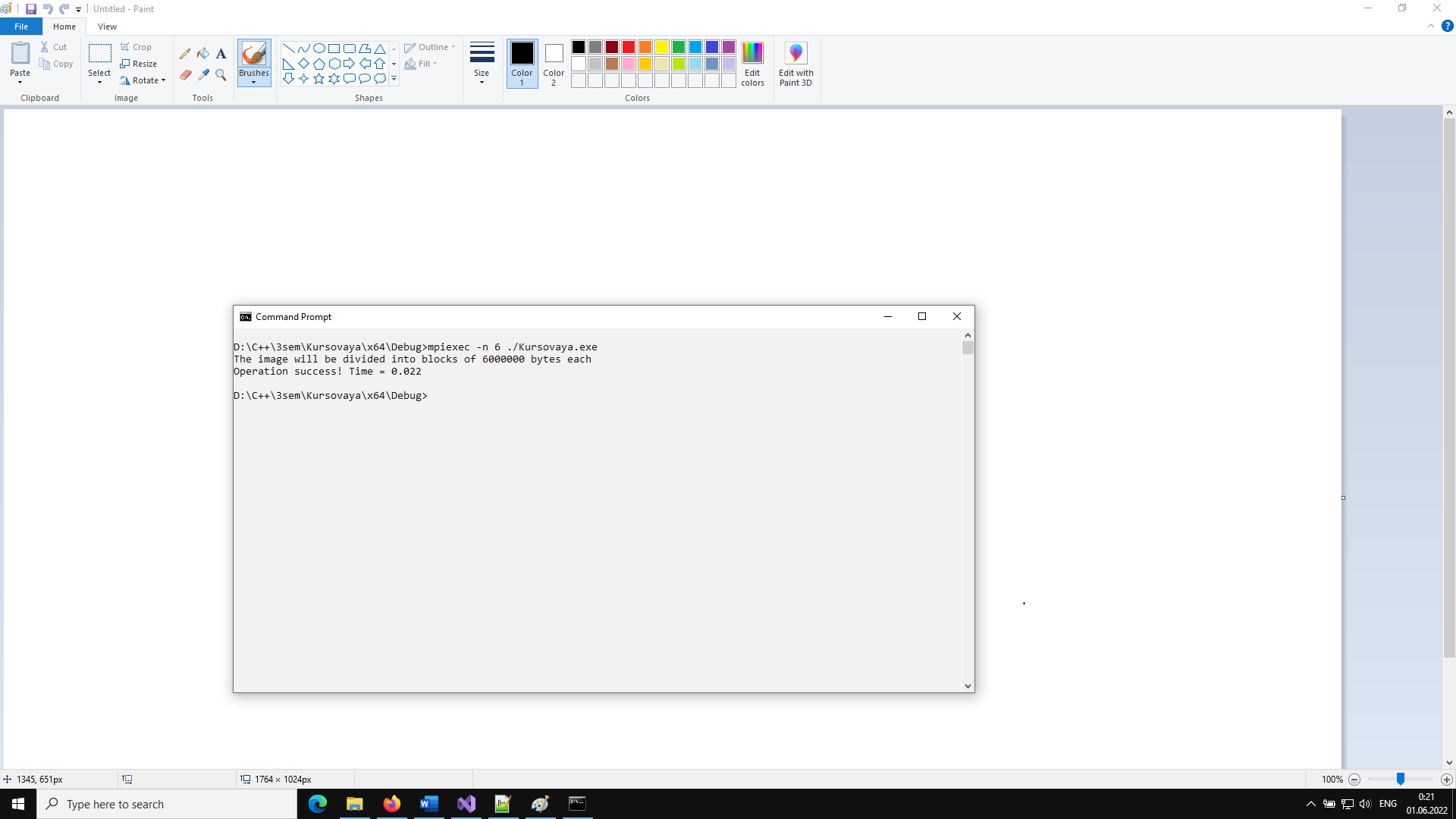


Рисунок 4.5 – Результаты тестирования времени выполнения

Вывод: результат соответствует ожиданиям. Параллельная обработка изображения отнимает меньше времени на выполнение алгоритмов, при этом качество обработки выходного изображения остается, не падает.

# Заключение

В результате выполнения курсового проекта было проведено сравнение производительности компьютерных систем последовательной и параллельной обработки черно-белого изображения.

Выходные изображения, полученные в результате последовательной и параллельной обработки идентичны. Время, затраченное на обработку изображения при параллельной обработке, как ожидалось, ниже, чем при последовательной. Но при последовательной обработке тратится время на открытие, закрытие потоков, переключение между ними, формирование выходного изображения на основании обработанных сегментов (их соединение).

Следовательно, использование компьютерной системы параллельной обработки данных является более эффективным, чем использование системы последовательной обработки, особенно при решении трудоемких задач.

# Список используемых источников

1. ГОСТ 2.105-95. Единая система конструкторской документации (ЕСКД). Общие требования к текстовым документам.
2. ГОСТ 19.504-79. Единая система программной документации ЕСПД. Руководство программиста. Требования к содержанию и оформлению.
3. ГОСТ 19.701-90. ЕСПД. Схемы алгоритмов, программ, данных и систем. Обозначения условные и правила выполнения.
4. ГОСТ 19.005-85. ЕСПД.

Р-схемы алгоритмов и программ. Обозначения условные графические и правила выполнения.

1. ГОСТ 19.101-77. ЕСПД. Виды программ и программных документов.
2. ГОСТ 19.102-77. ЕСПД. Стадии разработки.
3. ГОСТ 19.103-77. ЕСПД. Обозначения программ и программных документов.
4. ГОСТ 19.401-78. ЕСПД. Текст программы. Требования к содержанию и оформлению.
5. ГОСТ 19.402-78. ЕСПД. Описание программы.
6. ГОСТ 7.1-2003. Система стандартов по информации, библиотечному и издательскому делу. Библиографическая запись. Библиографическое описание. Общие требования и правила составления.