**ПРАВИТЕЛЬСТВО РОССИЙСКОЙ ФЕДЕРАЦИИ**

**НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ**

**«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»**

Факультет компьютерных наук

Образовательная программа бакалавриата «Программная инженерия»

|  |  |  |
| --- | --- | --- |
| СОГЛАСОВАНО  Преподаватель департамента программной инженерии ФКН, кандидат компьютерных наук  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_С.А.Виденин  \_\_\_\_\_\_\_\_\_\_\_10\_марта 2024 г. |  | УТВЕРЖДАЮ  Академический руководитель образовательной программы «Программная инженерия»  профессор департамента программной инженерии, кандидат технических наук  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Н.А. Павлочев  \_\_\_\_\_\_\_\_\_\_\_\_10\_марта\_2024 г. |

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | ***Подп. и дата*** |  | | ***Инв. № дубл.*** |  | | ***Взам. Инв. №*** |  | | ***Подп. и дата*** |  | | ***Инв. № подл.*** |  | | **ВЫСОКОПРОИЗВОДИТЕЛЬНЫЙ НАСТРАИВАЕМЫЙ HTTP СЕРВЕР**  **Программа и методика испытаний**  **ЛИСТ УТВЕРЖДЕНИЯ**  **RU.17701729.04.04-01 ТЗ 01-1-ЛУ**  Исполнитель  студент группы БПИ214  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/ Е.К.Фортов/  \_\_\_\_\_\_\_\_\_\_\_\_\_10\_марта\_2024 г. | |
|  |  |

УТВЕРЖДЕН

RU.17701729.04.04-01 ТЗ 01-1-ЛУ

|  |  |
| --- | --- |
| ***Подп. и дата*** |  |
| ***Инв. № дубл.*** |  |
| ***Взам. инв. №*** |  |
| ***Подп. и дата*** |  |
| ***Инв. № подл*** |  |

**ВЫСОКОПРОИЗВОДИТЕЛЬНЫЙ НАСТРАИВАЕМЫЙ HTTP СЕРВЕР**

**Программа и методика испытаний  
  
RU.17701729.04.04-01 ТЗ 01-1-ЛУ**

**Листов 15**

Оглавление

[1.](#_Toc72106850) [ОБЪЕКТ ИСПЫТАНИЙ](#_Toc72270487) [3](#_Toc72106850)

[1.1. Наименование программы 3](#_Toc72106851)

[1.2. Краткая характеристика области применения 3](#_Toc72106852)

[2.](#_Toc72106853) [ЦЕЛЬ ИСПЫТАНИЙ](#_Toc72270490) [4](#_Toc72106853)

[3.](#_Toc72106856) [ТРЕБОВАНИЯ К ПРОГРАММЕ](#_Toc72270491) [5](#_Toc72106856)

[3.1.](#_Toc72106857) [Требования к функциональным характеристикам](#_Toc72270492) [5](#_Toc72106857)

[3.1.1. Требования к составу выполняемых функций](#_Toc72270493) 5

[3.1.2. Требования к организации входных данных](#_Toc72270494) 5

[3.1.3. Требования к организации выходных данных](#_Toc72270495) 6

[3.2.](#_Toc72106858) [Требования к интерфейсу](#_Toc72270496)6

[3.3. Требования к надежности](#_Toc72270497) 6

[4.](#_Toc72106859) [ТРЕБОВАНИЯ К ПРОГРАММНОЙ ДОКУМЕНТАЦИИ](#_Toc72270498)7

[5.](#_Toc72106879) [СРЕДСТВА И ПОРЯДОК ИСПЫТАНИЙ](#_Toc72270499)8

[5.1. Технические средства, используемые во время испытаний](#_Toc72270500) 8

[5.2. Программные средства, используемые во время испытаний](#_Toc72270501) 8

[5.3. Порядок проведения испытаний](#_Toc72270502) 8

[5.4. Загрузка программы для испытания](#_Toc72270503) 8

[6.](#_Toc72106882) [МЕТОДЫ ИСПЫТАНИЙ](#_Toc72270504)9

[6.1. Испытание выполнения требований к программной документации](#_Toc72270505) 9

[6.2. Испытание выполнения требований к функциональным характеристикам](#_Toc72270506) 9

[6.3. Испытание выполнения требований к интерфейсу](#_Toc72270527) 13

[6.4.](#_Toc72270527) [Испытание выполнения требований к надёжности](#_Toc72270528)13

[СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ](#_Toc72106890) 14

# 1. [Error: Reference source not found](#_Toc72270487)

## 1.1. Наименование программы

Наименование программы – «Высокопроизводительный Настраиваемый HTTP Сервер» («High Perfomance Customizable HTTP Server»).

## 1.2. Краткая характеристика области применения

Данный IT продукт представляет из себя высокоуровневую C++ библиотеку, которая дает возможность быстро проектировать и разворачивать REST API на языке С++, минуя такие низкоуровневые детали, как сокеты, потоки, контексты и т.д.

# 2. [Error: Reference source not found](#_Toc72270490)

Целью испытаний является проверка корректности выполнения программой функций, перечисленных в разделе «Требования к программе».

# 3. [Error: Reference source not found](#_Toc72270491)

## 3.1. [Error: Reference source not found](#_Toc72270492)

## [Error: Reference source not found](#_Toc72270493)

Программа должна давать пользователю возможность выполнять следующие функции:

* инстанцировать объект http сервера
* базово конфигурировать http сервер
* наследовать класс http сервера под свои нужды
* создавать status line http ответа из готовых шаблонов: определять методы GET и POST, код возвращаемого значения
* создавать заголовки http ответа из готовых шаблонов: content-type, content-length и т. д.
* создавать тела http ответа из готовых шаблонов, отдельных html файлов
* настраивать кастомное логирование с разными уровнями в отдельный файл
* настраивать кастомное логирование с разными уровнями в syslog
* кешировать http ответы
* обрабатывать ошибки
* читать комментарии в коде сервера, которых будет достаточно для использования всех возможностей сервера
* подключать сервер через .hpp файл с отдельной папкой (где будут все остальные файлы-зависимости лежать), настройка необходимых зависимостей идет через готовый CmakeLists.txt

[Error: Reference source not found](#_Toc72270494)

[Error: Reference source not found](#_Toc72270494)

Входные данные — это исходный код на С++, в который портируется http сервер (исходный код может быть представлен в файлах с расширениями .cpp, .h, .hpp). В качестве системы автоматизации сборки проекта рекомендуется использовать CMake, так как в таком случае будет намного проще настроить зависимости, необходимые для подключаемого http сервера. Исходный код проекта до подключения данного фреймворка должен компилироваться успешно и проект должен собираться корректно.

В свою очередь, после подключения фреймворка (после успешного подключения всех необходимых для его работы файлов и успешной настройки необходимых зависимостей) в исходном коде создается объект http сервера. При проектировании REST API для создания очередного эндпоинта необходимо воспользоваться соответствующим методом созданного http сервера.

Касательно требований к входным данным, программисту необходимо ознакомиться с внутренней справкой / документацией http сервера, которая исчерпывающе описывает, как с помощью него проектировать REST API.

### 3.1.3. Требования к организации выходных данных

Результат работы сервера должен представлять собой действующий REST API, а также файлы с логированием. Отследить корректность работы можно с помощью логов, настроенных на максимально возможный уровень — DEBUG, а также с помощью непосредственно функционального тестирования написанного REST API. Если в логах была обнаружена хоть одна ошибка, сервер имеет неопределенное поведение.

## 3.2. Требования к интерфейсу

Графический интерфейс у данного сервиса фактически отсутствует, так как все команды прописываются именно в исходном файле.

**3.3. Требования к надежности**

Дополнительные требования к надежности не предъявляются, кроме тех, которые предъявляются к использующимся библиотекам. При корректном использовании фреймворка (то есть при обеспечении необходимых условий для корректной отработки используемых функций библиотек) программа не должна завершаться аварийно.

# 4. ТРЕБОВАНИЯ К ПРОГРАММНОЙ ДОКУМЕНТАЦИИ

Состав программной документации:

* «Высокопроизводительный Настраиваемый HTTP Сервер». Техническое задание (ГОСТ 19.201-78 [2])
* «Высокопроизводительный Настраиваемый HTTP Сервер». Программа и методика испытаний (ГОСТ 19.301-79 [3])
* «Высокопроизводительный Настраиваемый HTTP Сервер». Текст программы (ГОСТ 19.401-78 [4])
* «Высокопроизводительный Настраиваемый HTTP Сервер». Пояснительная записка (ГОСТ 19.404-79 [5])
* «Высокопроизводительный Настраиваемый HTTP Сервер». Руководство оператора (ГОСТ 19.505-79 [6])

# 5. СРЕДСТВА И ПОРЯДОК ИСПЫТАНИЙ

## 5.1. Технические средства, используемые во время испытаний

Ноутбук Macbook Pro 14 дюймов, M1.

## 5.2. Программные средства, используемые во время испытаний

На ноутбуке имеется:

- ОС Mac OS Ventura 13.2.1

- ОЗУ 16 ГБ

- компилятор clang последней стабильной версии

## 5.3. Порядок проведения испытаний

Испытания должны проводиться в следующем порядке:

- проверка требований к программной документации,

- проверка требований к функциональным характеристикам,

- проверка требований к интерфейсу,

- проверка требований к надёжности.

**5.4. Загрузка программы для испытания**

Тестируемая программа — main.cpp. В нее подключен ServeMe.hpp, настроен корректно CmakeLists.txt, проект компилируется и собирается успешно.

# 6. МЕТОДЫ ИСПЫТАНИЙ

## 6.1. Испытание выполнения требований к программной документации

Состав программной документации проверяется визуально, проверяется наличие программной документации в системе LMS. Также визуально проверяется соответствие документации требованиям ГОСТ.

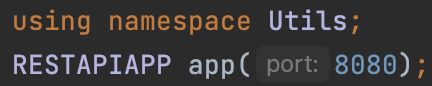
Все документы удовлетворяют представленным требованиям.

## 6.2. Испытание выполнения требований к функциональным характеристикам

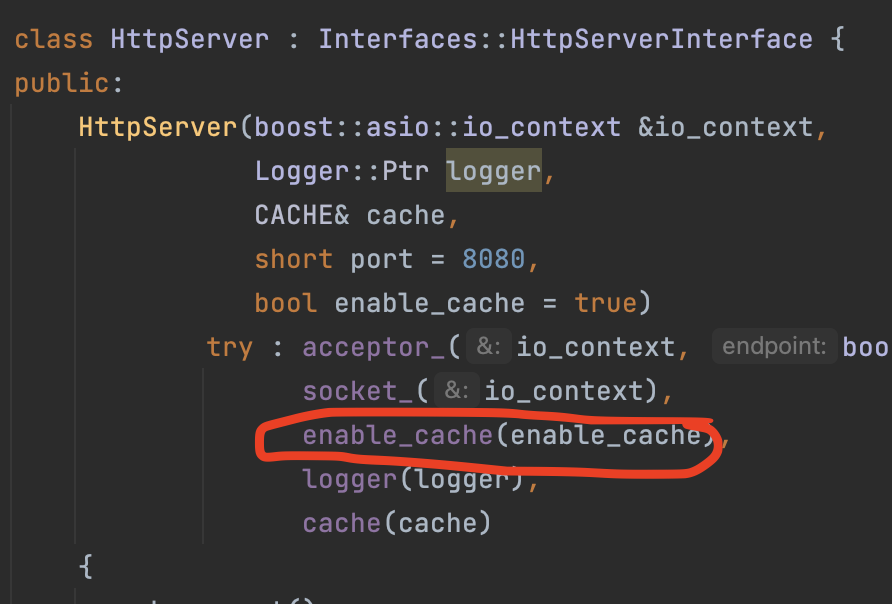
Из технического задания имеем:

Программа должна давать пользователю возможность выполнять следующие функции:

* инстанцировать объект http сервера:



* базово конфигурировать http сервер:

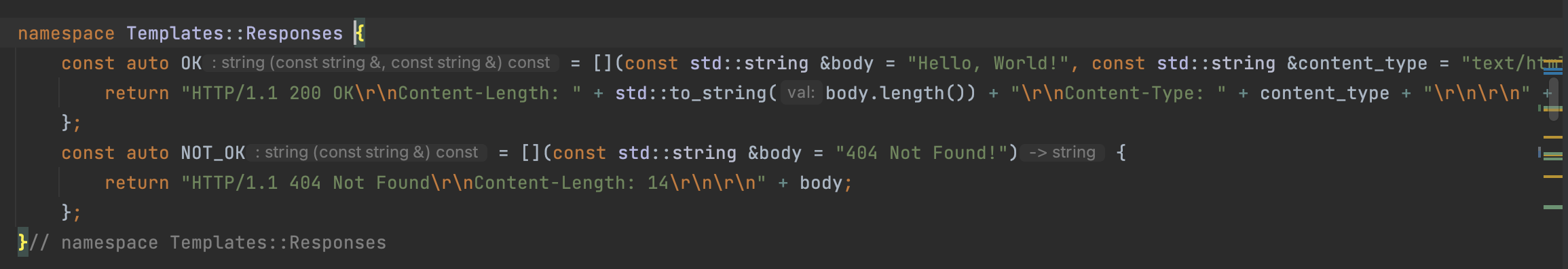


* наследовать класс http сервера под свои нужды:



* создавать status line http ответа из готовых шаблонов: определять методы GET и POST, код возвращаемого значения:

Эта возможность обеспечивается готовыми шаблонами:



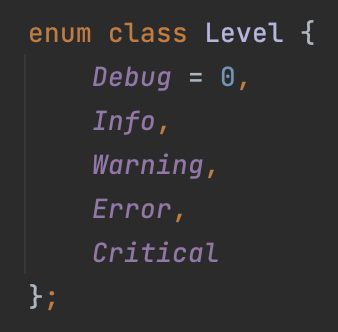
* создавать заголовки http ответа из готовых шаблонов: content-type, content-length: аналогично предыдущему пункту
* создавать тела http ответа из готовых шаблонов, отдельных html файлов:

Есть возможность указать конкретный .html-файл:



* настраивать кастомное логирование с разными уровнями в отдельный файл:

Уровни логирования:



Функция, которая отвечает за логирование в отдельный файл:



* настраивать кастомное логирование с разными уровнями в syslog:

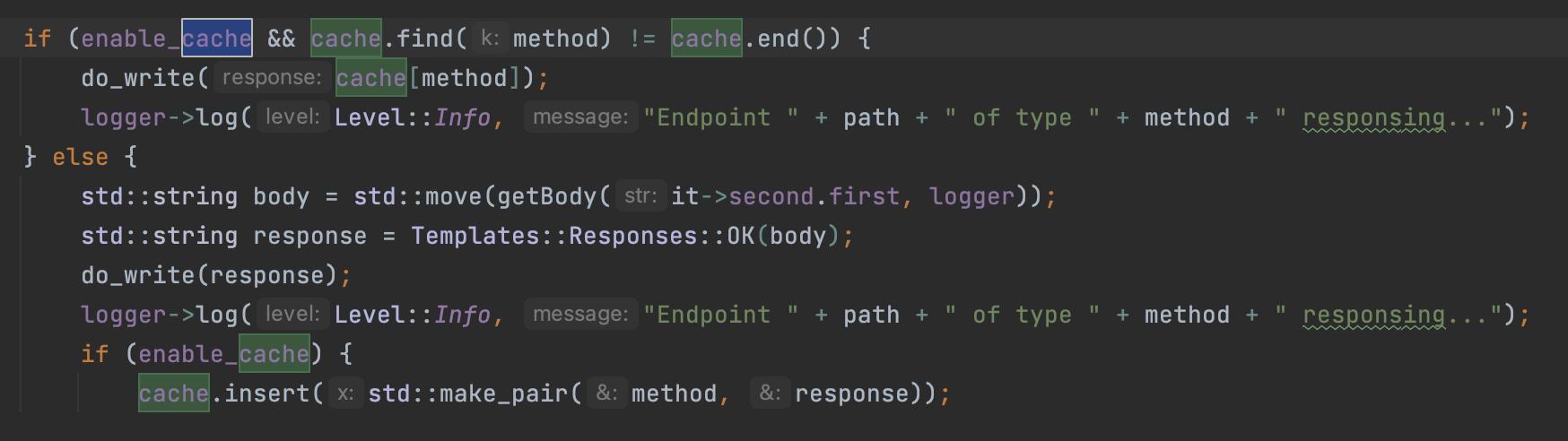
Функция, которая отвечает за логирование в syslog:



* кешировать http ответы:

Для кеширования создан отдельный объект «cache», в который можно сохранять запросы-ответы и читать из него:





* обрабатывать ошибки:

Обработка ошибок (в частности, исключений) встречается очень часто во всех компонентах REST API.

* читать комментарии в коде сервера, которых будет достаточно для использования всех возможностей сервера:

Некоторые строки кода REST API снабжены исчерпывающими комментариями. Более того, до кода в .hpp файле приведена документация, которая раскрывает возможности данного сервера и помогает пользователю быстрее вникнуть в суть проекта и быстро использовать/переиспользовать приведенный код.

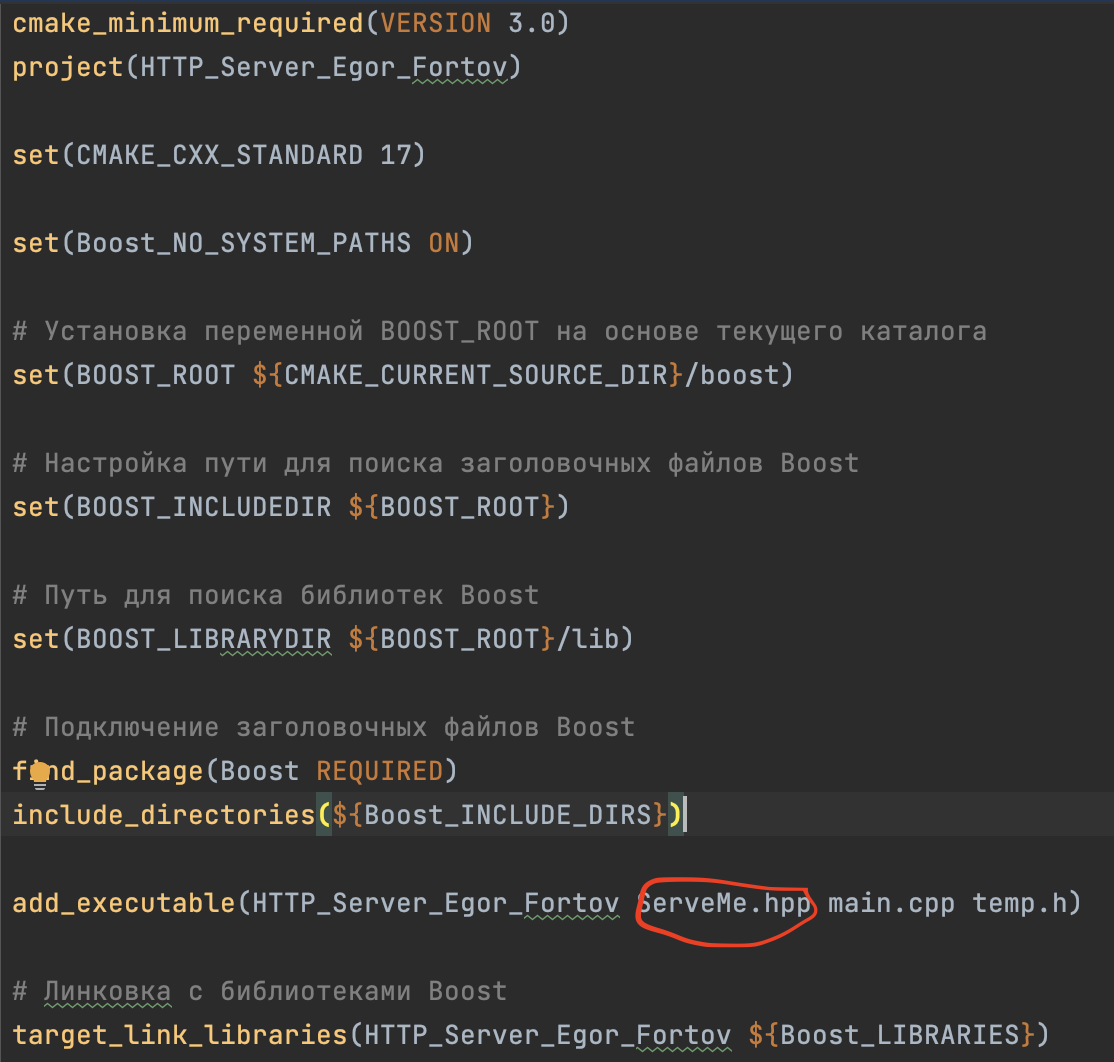
* подключать сервер через .hpp файл с отдельной папкой (где будут все остальные файлы-зависимости лежать), настройка необходимых зависимостей идет через готовый CmakeLists.txt:

Было принято решение поместить весь код программы в один .hpp-файл с целью простоты подключения и использования.

Подключение файла:



В CmakeLists.txt можно также настроить подключение этого файла к проекту:



**6.3. Испытание выполнения требований к интерфейсу**

Требования к интерфейсу проверяются визуально. Представленный интерфейс приложения в предыдущем разделе удовлетворяет всем требованиям.

## 6.4. Испытание выполнения требований к надёжности

Программа была протестирована > 20 раз на разных тестовых кодах. Ошибок не обнаружено.

**СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ**

1. ГОСТ 15150-69 Машины, приборы и другие технические изделия. Исполнения для различных климатических районов. Категории, условия эксплуатации, хранения и транспортирования в части воздействия климатических факторов внешней среды. – М.: Изд-во стандартов, 1997.
2. ГОСТ 19.101-77 Виды программ и программных документов. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
3. ГОСТ 19.102-77 Стадии разработки. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
4. ГОСТ 19.106-78 Требования к программным документам, выполненным печатным способом. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
5. ГОСТ 19.201-78 Техническое задание. Требования к содержанию и оформлению. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
6. ГОСТ 19.301-79 Программа и методика испытаний. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
7. ГОСТ 19.401-78 Текст программы. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
8. ГОСТ 19.404-79 Пояснительная записка. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
9. ГОСТ 19.505-79 Руководство оператора. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
10. ГОСТ 19.602-78 Правила дублирования, учета и хранения программных документов, выполненных печатным способом. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.

# ЛИСТ РЕГИСТРАЦИИ ИЗМЕНЕНИЙ

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Изм. | Номера листов (страниц) | | | | Всего листов (страниц) в документе | № документа | Входящий № сопроводительного документа и дата | Подпись | Дата |
| измененных | замененных | новых | аннулированных |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |