**ПРАВИТЕЛЬСТВО РОССИЙСКОЙ ФЕДЕРАЦИИ**

**НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ**

**«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»**

Факультет компьютерных наук

Образовательная программа бакалавриата «Программная инженерия»

|  |  |  |
| --- | --- | --- |
| СОГЛАСОВАНО  Научный руководитель,  доцент департамента программной инженерии факультета компьютерных наук, кандидат технических наук    \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Р.А.Нестеров  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_11\_мая\_ 2023 г. |  | УТВЕРЖДАЮ  Академический руководитель образовательной программы «Программная инженерия»  профессор департамента программной инженерии, кандидат технических наук    \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ В.В. Шилов  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_11\_мая\_\_2023 г. |

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | ***Подп. и дата*** |  | | ***Инв. № дубл.*** |  | | ***Взам. Инв. №*** |  | | ***Подп. и дата*** |  | | ***Инв. № подл.*** |  | | **ВСТРАИВАЕМЫЙ ПРОФИЛИРОВЩИК**  **ПРОГРАММНОГО КОДА НА ЯЗЫКЕ С++**  **Программа и методика испытаний**  **ЛИСТ УТВЕРЖДЕНИЯ**  **RU.17701729.04.04-01 01-1-ЛУ**  Исполнитель  студент группы БПИ214  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/ Е.К.Фортов/  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_11\_мая\_2023 г. | |
|  |  |

УТВЕРЖДЕН

RU.17701729.04.04-01 01-1-ЛУ

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

**ВСТРАИВАЕМЫЙ ПРОФИЛИРОВЩИК ПРОГРАММНОГО КОДА НА ЯЗЫКЕ С++**

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

**Листов 18**

Оглавление

[1. ОБЪЕКТ ИСПЫТАНИЙ](#_Toc72270487) 3

[1.1. Наименование программы](#_Toc72270488) 3

[1.2. Краткая характеристика области применения](#_Toc72270489) 3

[2. ЦЕЛЬ ИСПЫТАНИЙ](#_Toc72270490) 4

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

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

[3.1.1. Требования к составу выполняемых функций 6](#_Toc72270493)

[3.1.2. Требования к организации входных данных](#_Toc72270494) 6

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

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

[3.3. Требования к надежности](#_Toc72270497) 7

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

[5. 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. МЕТОДЫ ИСПЫТАНИЙ 9](#_Toc72270504)

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

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

[6.2.1. Начало/конец замера времени, вывод результатов в консоль, в файлы логов вместе с базовой аналитикой полученных результатов](#_Toc72270507) 9

[6.2.2. Вывод справки по командам](#_Toc72270508) 11

[6.2.3. Удаление логов (как локально, так и глобально)](#_Toc72270509) 12

[6.3. Испытание выполнения требований к интерфейсу](#_Toc72270527) 16

[6.4. Испытание выполнения требований к надёжности](#_Toc72270528) 16

[СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ 1](#_Toc72270529)7

# 1. ОБЪЕКТ ИСПЫТАНИЙ

## 1.1. Наименование программы

Наименование программы – «Встраиваемый Профилировщик Программного Кода на Языке С++«FAST\_PROFILE»» («Embedded Profiler of C++ Program Code «FAST\_PROFILE»»).

## 1.2. Краткая характеристика области применения

«Встраиваемый Профилировщик Программного Кода на Языке С++«FAST\_PROFILE»» - программа для профилирования любых программ на С++. Высоконагруженные серверы, графика в играх, ML-задачи и другие профильные и непрофильные программы — сферы применения данного ПО.

# 2. ЦЕЛЬ ИСПЫТАНИЙ

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

# 3. ТРЕБОВАНИЯ К ПРОГРАММЕ

## 3.1. Требования к функциональным характеристикам

### 3.1.1. Требования к составу выполняемых функций

Программа должна давать пользователю возможность выполнять следующие функции:

* начинать замер времени;
* заканчивать замер времени;
* выводить результаты замеров в консоли;
* выводить результаты замеров в отдельный файл (логировать в .txt, .json, .csv);
* создавать файлы с результатами проведенных измерений (на некоторых различных входных данных, код программы остается таким же) для дальнейшего построения графиков на основе этих данных в форматах .csv, .json;
* считать и выводить базовую аналитику результатов (выделение наибольшего и наименьшего времени исполнения того или иного участка кода);
* подсвеченный синтаксис результатов, чтобы отличить вывод программы от вывода логов профайлера
* указывать точность замеров времени (вывод в секундах, миллисекундах, микросекундах, наносекундах);
* добавлять комментарии к i-ому замеру времени при вызове той или иной функции профилировщика;
* выводить справку по командам;
* дополнять функциональность путем наследования главного класса профайлера;
* подключать весь профилировщик путем добавления одного файла с помощью директивы include;
* удалять логи (как локально, так и глобально)

### 3.1.2. Требования к организации входных данных

Входные данные — это исходный код, в который встраиваются нужные команды профилироващика (исходный код может быть представлен в файлах с расширениями .cpp & .h). Исходных код должен отрабатывать корректно, без предупреждений компилятора и ошибок.

В свою очередь, команды профилировщика, внедряемые в код тестируемой программы, — это макросы или отдельные функции, которые могут принимать дополнительную информацию (например, комментарии к тому или иному замеру времени) в виде строк и целочисленных значений.

Касательно требований к входным данным, пользователю необходимо ознакомиться с внутренней справкой профайлера, которая описывает, какие входные данные принимает та или иная функция.

### 3.1.3. Требования к организации выходных данных

Результат работы профайлера на тестируемом коде должен содержать следующее (цвет вывода — по умолчанию стандартный, если не был указан другой в основном коде) :

1. вывод результатов проведенных замеров (вывод в консоль + .csv & .json файлы для дальнейшей постройки графиков в сторонних сервисах)
2. вывод базовой аналитики (максимальное/минимальное время тестов)
3. вывод во внешний .txt файл отчета о проведенном тестировании (комментарии)
4. вывод справки по командам (при вызове соответствующей команды профилировщика)

Если программа, на которой проводились замеры времени, отработала с некорректным завершением (код возврата не 0), профайлер может ничего не вывести и не сохранить проведенные исследования во внешний файл.

## 3.2. Требования к интерфейсу

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

Справка по командам профайлера содержится в прилагаемом .txt файле, также возможен вызов справки из исходного кода в консоль с помощью соответствующей команды в исходном файле.

**3.3. Требования к надежности**

Программа не должна аварийно завершаться при любом наборе входных данных (если не подразумевается отладка приложения).

# 4. ТРЕБОВАНИЯ К ПРОГРАММНОЙ ДОКУМЕНТАЦИИ

Состав программной документации:

1. «Встраиваемый Профилировщик Программного Кода на Языке С++». Техническое задание (ГОСТ 19.201-78 [2])
2. «Встраиваемый Профилировщик Программного Кода на Языке С++». Программа и методика испытаний (ГОСТ 19.301-79 [3])
3. «Встраиваемый Профилировщик Программного Кода на Языке С++». Текст программы (ГОСТ 19.401-78 [4])
4. «Встраиваемый Профилировщик Программного Кода на Языке С++». Пояснительная записка (ГОСТ 19.404-79 [5])
5. «Встраиваемый Профилировщик Программного Кода на Языке С++». 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 — test.cpp. В одну директорию с ней положены заголовочный файл profiler.h и директория include со всеми зависимостями.

# 6. МЕТОДЫ ИСПЫТАНИЙ

## 6.1. Испытание выполнения требований к программной документации

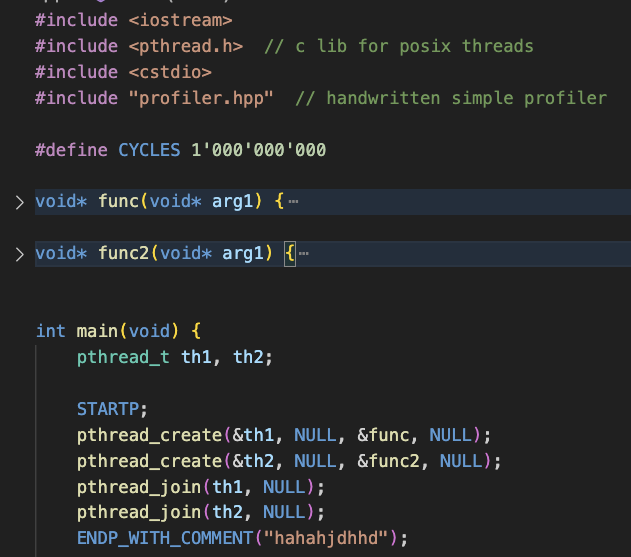
Состав программной документации проверяется визуально, проверяется наличие программной документации в системе LMS. Также визуально проверяется соответствие документации требованиям ГОСТ.

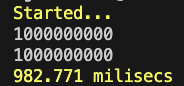
Все документы удовлетворяют представленным требованиям.

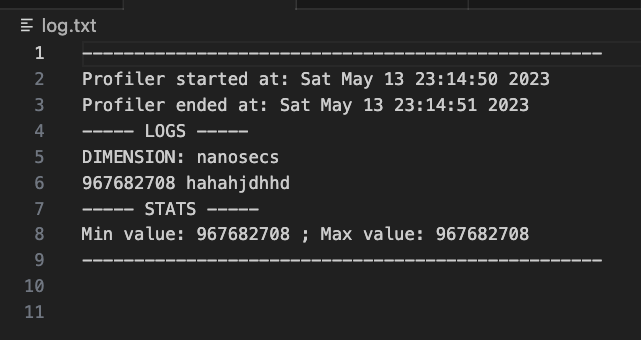
## 6.2. Испытание выполнения требований к функциональным характеристикам

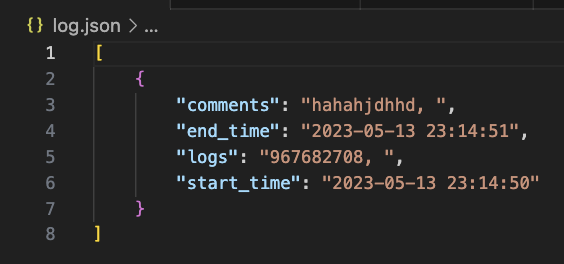
### 6.2.1. Начало/конец замера времени, вывод результатов в консоль, в файлы логов вместе с базовой аналитикой полученных результатов

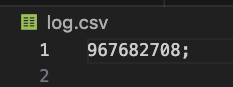
В данном случае для начала замера времени используется макрос STARTP, для окончания — макрос ENDP\_WITH\_COMMENT(str), где str — комментарий. Файлы логов создаются и заполняются автормаически.





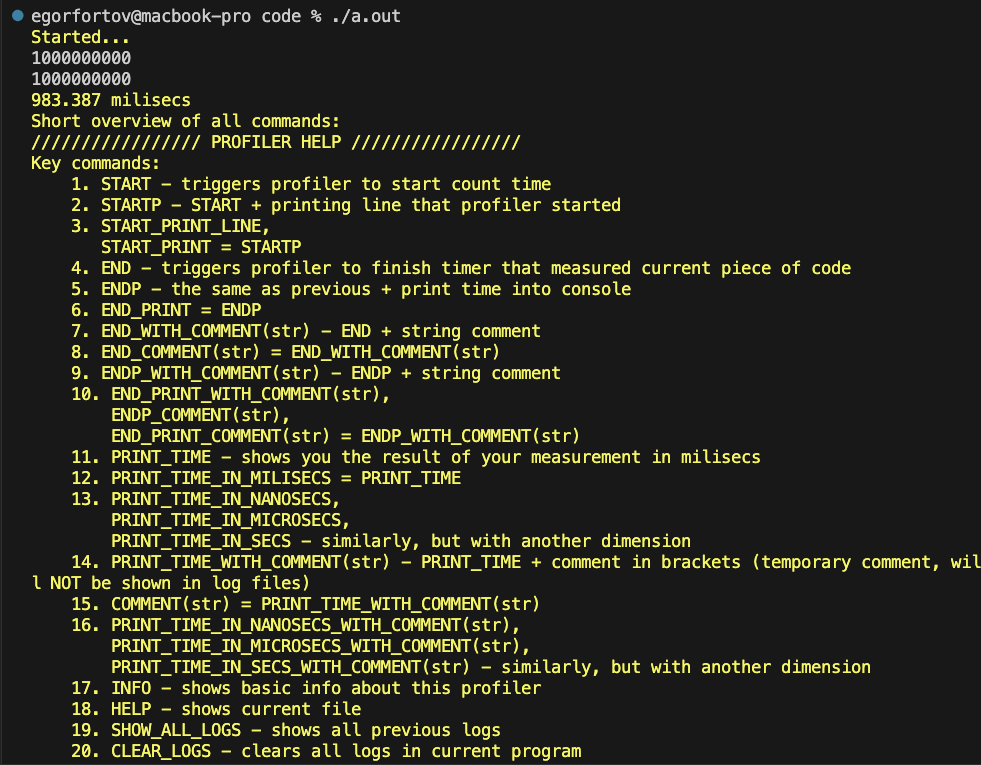
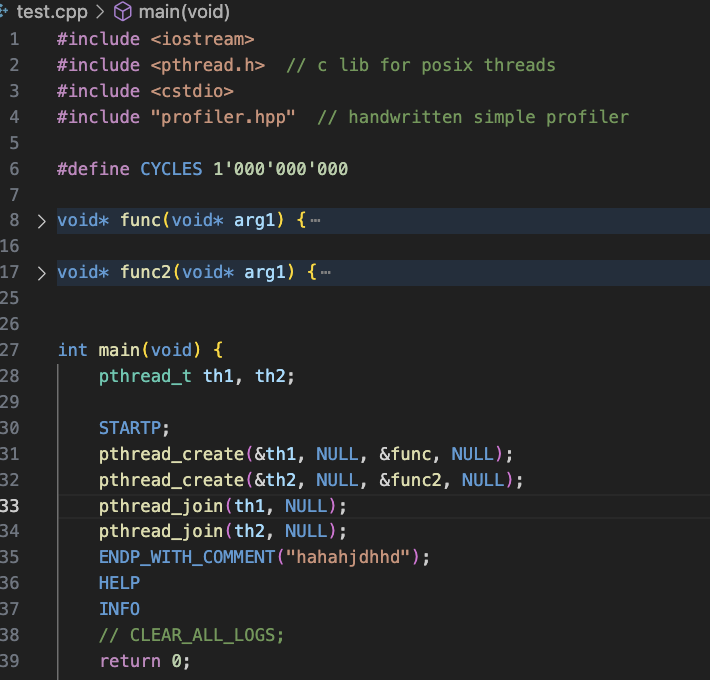


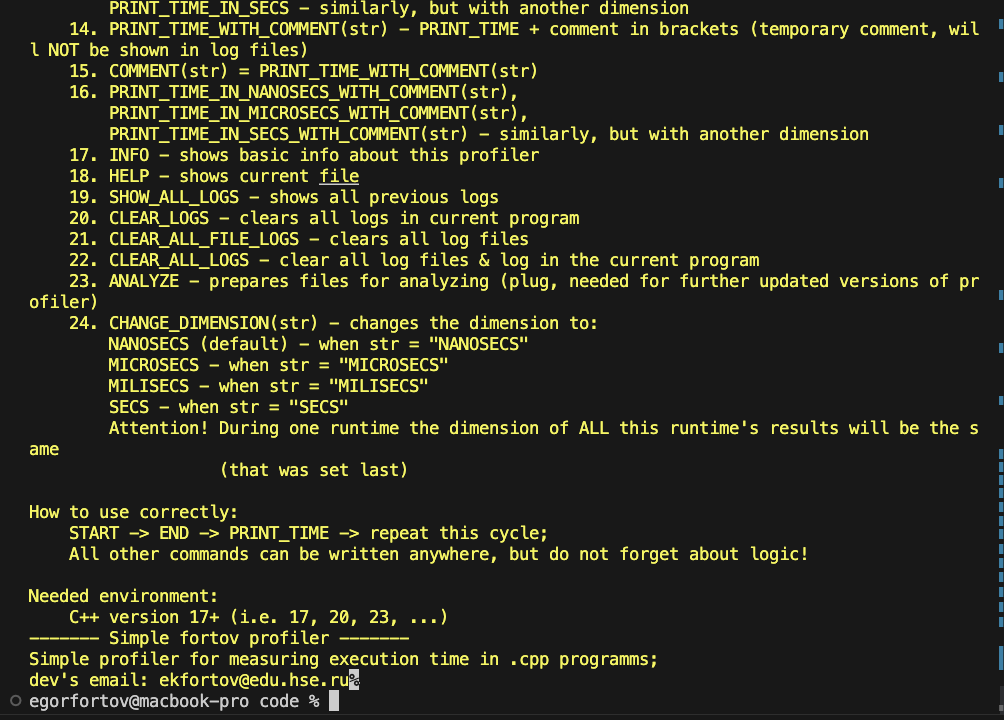




### 6.2.2. Вывод справки по командам

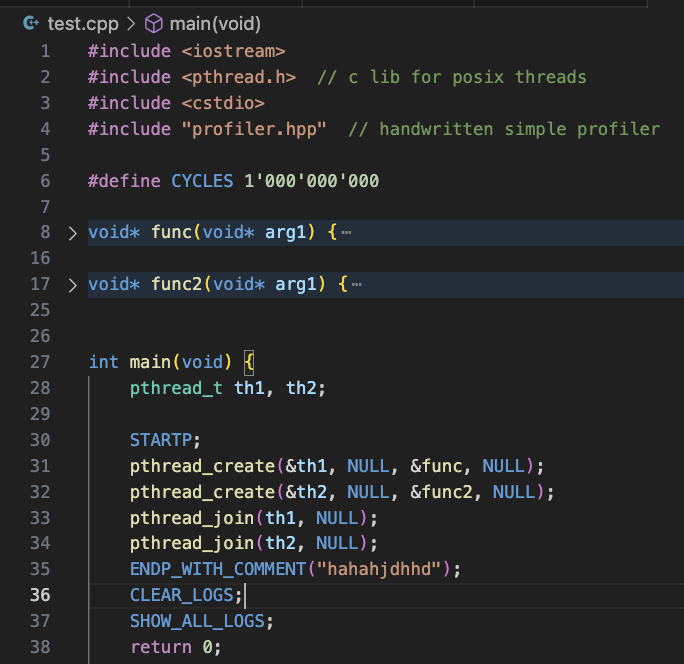
Используя макрос HELP, выведем справку по командам. Используя макрос INFO, посмотрим информацию о профайлере.

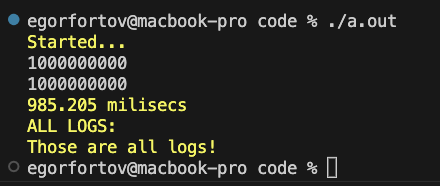


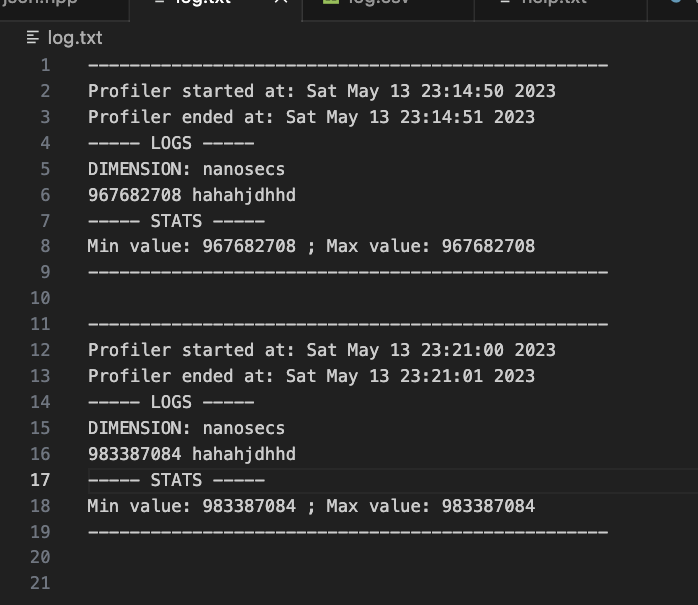


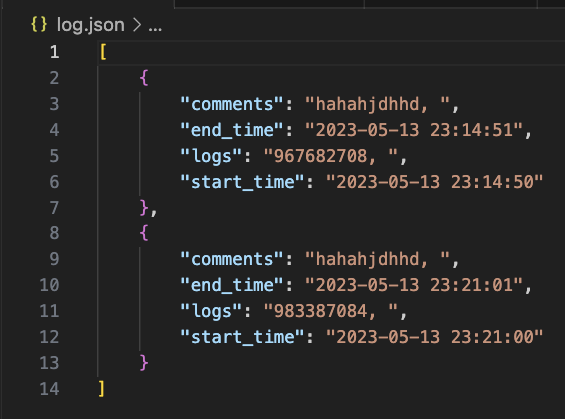
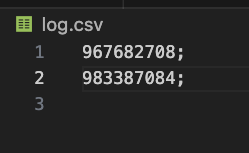
### 6.2.3. Удаление логов (как локально, так и глобально)

С помощью команды CLEAR\_LOGS удалим локальные логи, то есть логи текущей сессии работы программы. При этом ранее полученные логи (в файлах) останутся.

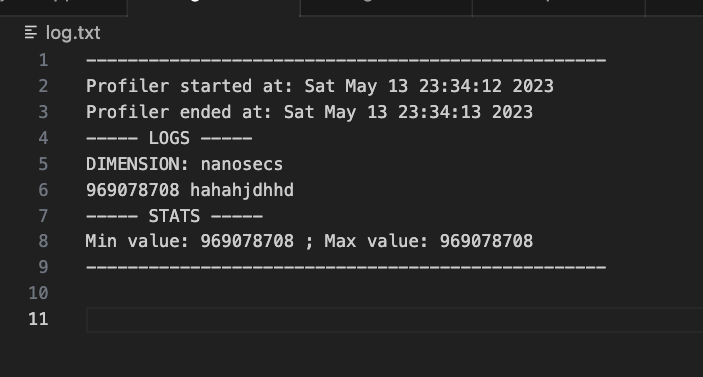
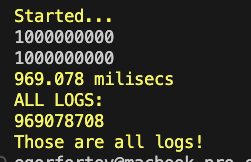
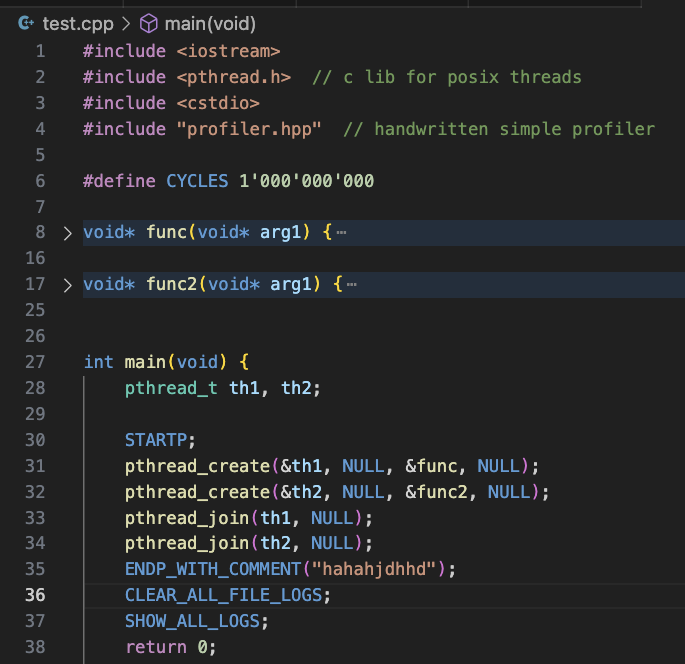








С помощью команды CLEAR\_ALL\_FILE\_LOGS удалим логи в файлах (фактически, просто удалим предыдущие логи).



### 

### 

При выполнении команды CLEAR\_ALL\_LOGS очистятся все логи, то есть фактически вызовутся две описанные выше команды: CLEAR\_ALL\_FILE\_LOGS & CLEAR\_LOGS.

## 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 > 100 раз на разных тестовых кодах. Ошибок не обнаружено.

**СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ**

1. ГОСТ 19.106-78 Требования к программным документам, выполненным печатным способом. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
2. ГОСТ 19.201-78 Техническое задание. Требования к содержанию и оформлению. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
3. ГОСТ 19.301-79 Программа и методика испытаний. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
4. ГОСТ 19.401-78 Текст программы. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
5. ГОСТ 19.404-79 Пояснительная записка. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
6. ГОСТ 19.505-79 Руководство оператора. // 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 |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |