


**ПРАВИТЕЛЬСТВО РОССИЙСКОЙ ФЕДЕРАЦИИ
НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ
«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»**

Факультет компьютерных наук
Образовательная программа бакалавриата «Программная инженерия»


СОГЛАСОВАНО

Научный руководитель,
преподаватель департамента
программной инженерии ФКН,
кандидат компьютерных 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.


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

Руководство оператора

ЛИСТ УТВЕРЖДЕНИЯ

RU.17701729.04.04-01 34 01-1-ЛУ

Исполнитель
студент группы БПИ214

—  — / Е.К.Фортов/
— 11 мая 2023 г.

Подп. и дата	
Инв. № дубл.	
Взам. Инв. №	
Подп. и дата	
Инв. № подл.	

УТВЕРЖДЕН

Москва 2023

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

Руководство оператора

RU.17701729.04.04-01 34 01-1-ЛУ

Листов 17

<i>Инв. № подл</i>	
<i>Подп. и дата</i>	
<i>Взам. инв. №</i>	
<i>Инв. № дубл.</i>	
<i>Подп. и дата</i>	

Оглавление

1. НАЗНАЧЕНИЕ ПРОГРАММЫ.....	3
1.1. Функциональное назначение программы.....	3
1.2. Эксплуатационное назначение.....	3
1.3. Состав функций.....	4
2. УСЛОВИЯ ВЫПОЛНЕНИЯ ПРОГРАММЫ.....	5
2.1. Минимальный состав аппаратных средств.....	6
2.2. Минимальный состав программных средств.....	6
2.3. Требования к персоналу (пользователю).....	6
3. ВЫПОЛНЕНИЕ ПРОГРАММЫ.....	7
3.1. Загрузка программы.....	7
3.2. Запуск программы.....	7
3.3. Выполнение программы.....	7
3.3.1. Получение справки по командам до начала пользования.....	7
3.3.2. Начало замера времени.....	7
3.3.3. Конец замера времени.....	8
3.3.4. Вывод результата последнего замера времени (вручную).....	9
3.3.5. Получение информации о профайлере.....	10
3.3.6. Вывод локальных логов в консоль.....	11
3.3.7. Очистка логов.....	12
3.3.8. Изменение размерности логов.....	13
3.4. Завершение программы.....	14
4. СООБЩЕНИЯ ОПЕРАТОРУ.....	15
СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ.....	16

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.04-01 34				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

1. НАЗНАЧЕНИЕ ПРОГРАММЫ

1.1. Функциональное назначение программы

Функциональным назначением программы является предоставление программисту возможности быстро и удобно проводить временные замеры частей своей программы, написанной на C++, во время выполнения программы. Кроме того, разрабатываемая программа создает файлы с результатами проведенных измерений для дальнейшего построения графиков в сторонней программе. Эти данные можно использовать для построения статистики работы исследуемой программы.

1.2. Эксплуатационное назначение

При написании программы по технической документации разработчик должен оценивать время исполнения программного кода на разных входных данных. Для этого программисту иногда удобно использовать готовый профилировщик кода - программу, которая упрощает проведение замеров времени исполнения отдельных частей кода. К сожалению, использование существующих профилировщиков может приводить к затруднениям по причине довольно длительной настройки, а также внедрения большого числа дополнительных инструкций и команд.

Более того, большинство из них предлагают только графический интерфейс, что делает невозможным тестирование программного кода в консоли, например, если программа тестируется на удаленном сервере с доступом только по протоколу ssh. «FAST_PROFILE» решает эти проблемы следующим образом:

- 1) Подключается одной командой через заголовочный файл
- 2) Для начала/окончания замеров времени, а также для других функций, таких как построение графиков проведенных замеров и вывод в консоль/файлы логов, используются макросы.

Встраиваемый профилировщик кода упрощает проведение временных тестов программного кода. В результате экономится время как разработчика, так и тестировщика.

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.06.02-01 34				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

1.3. Состав функций

Программа даёт 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 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;

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.04-01 34				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

2. УСЛОВИЯ ВЫПОЛНЕНИЯ ПРОГРАММЫ

2.1. Минимальный состав аппаратурных средств

Для бесперебойной работы программного продукта требуется компьютер с:

1. Объемом свободной встроенной памяти не меньше 55 МБ,
2. Объёмом оперативной памяти не меньше 1 ГБ.

2.2. Минимальный состав программных средств

Для бесперебойной работы программного продукта требуется компьютер с:

1. Установленной версией компилятора gcc - 14, clang — 3.4
2. Операционной системой со стабильной сборкой, выпущенной не позднее 2015 года

2.3. Требования к персоналу (пользователю)

Необходимое количество персонала – 1 человек.

Необходимая квалификация персонала – пользователь.

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.04-01 34				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

3. ВЫПОЛНЕНИЕ ПРОГРАММЫ

В данном разделе описан пример работы с программой.

3.1. Загрузка программы

Для загрузки данного ПО надо скачать файл profiler.hpp и папку include и положить эти файлы в ту же директорию, где лежит код, который будет профилирован.

3.2. Запуск программы

Для подключения профайлера к основной программе необходимо прописать #include «profiler.hpp» (см. рис. 1):

```
#include "profiler.hpp" // handwritten simple profiler
```

3.3. Выполнение программы

3.3.1. Получение справки по командам до начала пользования

Настоятельно рекомендую ознакомиться с основными командами профайлера, чтобы избежать в дальнейшем ошибок в замерах времени. Для получения справки по командам надо написать в коде (как можно раньше) HELP:

```
int main(void) {  
    HELP;  
    pthread_t th1, th2;
```

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.04-01 34				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

```

////////////////// PROFILER HELP //////////////////
Key commands:
1. START - triggers profiler to start count time
2. STARTP - START + printing line that profiler started
3. START_PRINT_LINE,
   START_PRINT = STARTP
4. END - triggers profiler to finish timer that measured current piece of code
5. ENDP - the same as previous + print time into console
6. END_PRINT = ENDP
7. END_WITH_COMMENT(str) - END + string comment
8. END_COMMENT(str) = END_WITH_COMMENT(str)
9. ENDP_WITH_COMMENT(str) - ENDP + string comment
10. END_PRINT_WITH_COMMENT(str),
    ENDP_COMMENT(str),
    END_PRINT_COMMENT(str) = ENDP_WITH_COMMENT(str)
11. PRINT_TIME - shows you the result of your measurement in milisecs
12. PRINT_TIME_IN_MILISECS = PRINT_TIME
13. PRINT_TIME_IN_NANOSECS,
    PRINT_TIME_IN_MICROSECS,
    PRINT_TIME_IN_SECS - similarly, but with another dimension
14. PRINT_TIME_WITH_COMMENT(str) - PRINT_TIME + comment in brackets (temporary comment,
    NOT be shown in log files)
15. COMMENT(str) = PRINT_TIME_WITH_COMMENT(str)
16. PRINT_TIME_IN_NANOSECS_WITH_COMMENT(str),
    PRINT_TIME_IN_MICROSECS_WITH_COMMENT(str),
    PRINT_TIME_IN_SECS_WITH_COMMENT(str) - similarly, but with another dimension
17. INFO - shows basic info about this profiler
18. HELP - shows current file
19. SHOW_ALL_LOGS - shows all previous logs
20. CLEAR_LOGS - clears all logs in current program
21. CLEAR_ALL_FILE_LOGS - clears all log files
22. CLEAR_ALL_LOGS - clear all log files & log in the current program
23. ANALYZE - prepares files for analyzing (plug, needed for further updated versions of
    profiler)
24. CHANGE_DIMENSION(str) - changes the dimension to:
    NANOSECS (default) - when str = "NANOSECS"
    MICROSECS - when str = "MICROSECS"
    MILISECS - when str = "MILISECS"
    SECS - when str = "SECS"

```

3.3.2. Начало замера времени

Теперь необходимо определить часть кода, которая будет тестироваться на предмет времени исполнения. Перед этим куском кода надо поставить один из макросов:

- START — начало замера времени
- STARTP — начало замера времени + вывод на экран комментария об этом
- START_PRINT_LINE = STARTP
- START_PRINT = STARTP

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.04-01 34				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата


```
int main(void) {
    pthread_t th1, th2;

    STARTP;
    pthread_create(&th1, NULL, &func, NULL);
    pthread_create(&th2, NULL, &func2, NULL);
    pthread_join(th1, NULL);
    pthread_join(th2, NULL);
}
```

Started...

3.3.3. Конец замера времени

После исследуемого участка кода необходимо поставить макрос END (конец замера времени + подсчет результата + логирование в локальные поля). При использовании команд ENDP / END_PRINT поведение профайлера будет аналогичным команде END, только дополнительно текущий результат будет выведен в консоль (также желтым цветом). Также можно остановить замер времени с комментарием — он будет отображен в логах. Это делается с помощью команд END_WITH_COMMENT(str) / END_COMMENT(str), где str — комментарий. Если требуется и оставить комментарий к замеру времени, и вывести результат на экран, следует воспользоваться одной из команд:

- ENDP_WITH_COMMENT(str)
- END_PRINT_WITH_COMMENT(str)
- ENDP_COMMENT(str)
- END_PRINT_COMMENT(str).

Алгоритм работы данных команд аналогичен командам выше.

```
int main(void) {
    HELP;
    pthread_t th1, th2;

    STARTP;
    pthread_create(&th1, NULL, &func, NULL);
    pthread_create(&th2, NULL, &func2, NULL);
    pthread_join(th1, NULL);
    pthread_join(th2, NULL);
    ENDP_WITH_COMMENT("hahahjdhd");
}
```

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.04-01 34				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

```

----- LOGS -----
DIMENSION: nanosecs
986 hahahjdhhhd

```

3.3.4. Вывод результата последнего замера времени (вручную)

Для (дополнительной) распечатки замеренного результата можно воспользоваться любой из команд:

- PRINT_TIME
- PRINT_TIME_IN_NANOSECS
- PRINT_TIME_IN_MICROSECS
- PRINT_TIME_IN_SECS
- PRINT_TIME_IN_MILISECS

Эти команды возьмут результат последнего замера времени и выведут результат в нужной размерности на экран желтым цветом (вместе с комментарием). По умолчанию используются миллисекунды (команда PRINT_TIME). Для вывода результата с комментарием используются функции:

- PRINT_TIME_IN_MILISECS
- PRINT_TIME_WITH_COMMENT(str)
- PRINT_TIME_IN_NANOSECS_WITH_COMMENT(str)
- PRINT_TIME_IN_MICROSECS_WITH_COMMENT(str)
- PRINT_TIME_IN_SECS_WITH_COMMENT(str)
- COMMENT(str), где str — комментарием (локальный, он будет отображен только в консоли и не будет сохраняться в логах. Для сохранения комментария в логи должна быть использована команда END_WITH_COMMENT(str), описанная выше). Алгоритм работы этих команд аналогичен командам, описанным выше.

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.04-01 34				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

```
int main(void) {  
    HELP;  
    pthread_t th1, th2;  
  
    STARTP;  
    pthread_create(&th1, NULL, &func, NULL);  
    pthread_create(&th2, NULL, &func2, NULL);  
    pthread_join(th1, NULL);  
    pthread_join(th2, NULL);  
    ENDP_WITH_COMMENT("hahahjdhhhd");  
  
    std::cout << "HEHEHE\n";  
    PRINT_TIME;  
    PRINT_TIME_IN_NANOSECS_WITH_COMMENT("done");  
    std::cout << "HEHEHE\n";  
}
```

```
-----  
HEHEHE  
986.376 milisecs | hahahjdhhhd  
Comment: done:  
986376250.000 nanosecs  
HEHEHE
```

3.3.5. Получение информации о профайлере

Для получения информации о данном ПО пользователь может воспользоваться макросом INFO. Этот макрос выведет литералы, описывающие что есть данный продукт, зачем он нужен, базовые команды, а также контакты его разработчика.

```
int main(void) {  
    HELP;  
    pthread_t th1, th2;  
    INFO;  
}
```

```
----- Simple fortov profiler -----  
Simple profiler for measuring execution time in .cpp programmes;  
dev's email: ekfortov@edu.hse.ruStarted...
```

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.04-01 34				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

3.3.6. Вывод локальных логов в консоль

Для вывода всех имеющихся в данной сессии логов можно воспользоваться макросом `SHOW_ALL_LOGS`, который выведет все логи на экран в том порядке, в котором пользователь совершал замеры времени.

```
int main(void) {
    pthread_t th1, th2;

    STARTP;
    pthread_create(&th1, NULL, &func, NULL);
    pthread_create(&th2, NULL, &func2, NULL);
    pthread_join(th1, NULL);
    pthread_join(th2, NULL);
    ENDP_WITH_COMMENT("hahahjd hhd");

    std::cout << "HEHEHE\n";
    PRINT_TIME;
    PRINT_TIME_IN_NANOSECS_WITH_COMMENT("done");
    std::cout << "HEHEHE\n";

    // CLEAR_ALL_LOGS;
    // CLEAR_ALL_FILE_LOGS;
    // CHANGE_DIMENSION("MICROSECS");    CHANGE_DIMENSION("MILISECS")
    START
    func(nullptr);
    func2(nullptr);
    ENDP

    SHOW_ALL_LOGS
    return 0;
}
```

```
ALL LOGS:
982892125
1875881209
Those are all logs!
```

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.04-01 34				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

3.3.7. Очистка логов

Для очистки логов можно воспользоваться макросом `CLEAR_LOGS`, который удалит все логи в текущей сессии (логи предыдущих сессий останутся в файлах). Для удаления всех файлов логов есть макрос `CLEAR_ALL_FILE_LOGS`, который удаляет файлы `log.txt`, `log.json`, `log.csv`. Для удаления как текущих логов (логов текущей сессии работы программы), так и файлов логов, используется макрос `CLEAR_ALL_LOGS`, который делает то же самое, что и макросы `CLEAR_LOGS` и `CLEAR_ALL_FILE_LOGS` вместе.

```
int main(void) {
    pthread_t th1, th2;

    STARTP;
    pthread_create(&th1, NULL, &func, NULL);
    pthread_create(&th2, NULL, &func2, NULL);
    pthread_join(th1, NULL);
    pthread_join(th2, NULL);
    ENDP_WITH_COMMENT("hahahjdhd");

    std::cout << "HEHEHE\n";
    PRINT_TIME;
    PRINT_TIME_IN_NANOSECS_WITH_COMMENT("done");
    std::cout << "HEHEHE\n";

    CLEAR_ALL_LOGS;
    // CLEAR_ALL_FILE_LOGS;
    // CHANGE_DIMENSION("MICROSECS");      CHANGE_DIMENSION("MILISECS")
    START
    func(nullptr);
    func2(nullptr);
    ENDP

    SHOW_ALL_LOGS
    return 0;
}
```

```
ALL LOGS:
1894647209
Those are all logs!
```

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.04-01 34				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

3.3.8. Изменение размерности логов

Для изменения размерности результатов замеров используется макрос `CHANGE_DIMENSION(str)`, где `str` — нужная размерность логов: "NANOSECS" (значение по умолчанию), "MICROSECS", "MILISECS", "SECS". Замечание: данная команда работает только для внешних логов — в файлах. Причем размерность замеров времени в файлах будет соответствовать аргументу последнего вызова данной команды. В локальных логах (вывод в консоль) для изменения стандартной размерности (миллисекунды) следует пользоваться командами: по умолчанию используются миллисекунды (команда `PRINT_TIME`). Для вывода результата с комментарием используются функции:

- `PRINT_TIME_IN_NANOSECS`
- `PRINT_TIME_IN_MICROSECS`
- `PRINT_TIME_IN_SECS`
- `PRINT_TIME_IN_MILISECS` (описаны выше).

```
int main(void) {
    pthread_t th1, th2;

    STARTP;
    pthread_create(&th1, NULL, &func, NULL);
    pthread_create(&th2, NULL, &func2, NULL);
    pthread_join(th1, NULL);
    pthread_join(th2, NULL);
    ENDP_WITH_COMMENT("hahahjd hhd");

    std::cout << "HEHEHE\n";
    PRINT_TIME;
    PRINT_TIME_IN_NANOSECS_WITH_COMMENT("done");
    std::cout << "HEHEHE\n";

    CLEAR_ALL_LOGS;
    // CLEAR_ALL_FILE_LOGS;
    CHANGE_DIMENSION("MICROSECS");
    CHANGE_DIMENSION("MILISECS")

    START
    func(nullptr);
    func2(nullptr);
    ENDP

    SHOW_ALL_LOGS
    return 0;
}
```

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.04-01 34				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

```
1892.691 milisecs
ALL LOGS:
1892691917
Those are all logs!
```

3.4. Завершение программы

Профилировщик сам завершит свою работу вместе с завершением пользовательской программы.

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.04-01 34				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

4. СООБЩЕНИЯ ОПЕРАТОРУ

В программе допускается Undefined Behaviour, которое может возникать по причине некорректного пользования профайлером.

Например, UB может возникнуть при вводе некорректного комментария в случае смены размерности результатов.

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.04-01 34				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ

1) CSV VS JSON [Электронный ресурс]. Режим доступа:
<https://coresignal.com/blog/json-vs-csv/>, свободный. (дата обращения: 11.05.2023)

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.04-01 34				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

ЛИСТ РЕГИСТРАЦИИ ИЗМЕНЕНИЙ

[illegible]