



Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования
«Московский государственный технический университет
имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ ИНФОРМАТИКА И СИСТЕМЫ УПРАВЛЕНИЯ
КАФЕДРА СИСТЕМЫ ОБРАБОТКИ ИНФОРМАЦИИ И УПРАВЛЕНИЯ

**ОТЧЕТ
ПО ЛАБОРАТОРНОЙ РАБОТЕ
*НА ТЕМУ:***

Жизненный цикл ПО полного цикла на примере RPN-калькулятора на C++

Студент

ИУ5-14Б

(группа)

Г.И. Корнеев

(И.О. Фамилия)

Руководитель курсовой
работы

(подпись, дата)

М.И. Колосов

(подпись, дата)

(И.О. Фамилия)

2025 г.

Цель: освоить на практике полный жизненный цикл разработки современного C++ приложения, от настройки окружения и написания кода до организации CI/CD, контейнеризации и документирования.

Задачи:

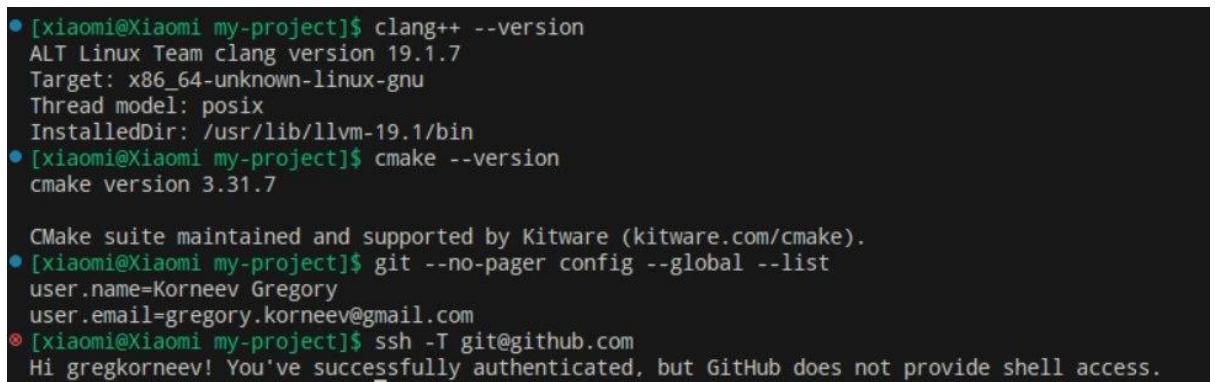
- Выполнить реализацию калькулятора;
- Выполнить реализацию одного из 30 заданий на выбор (pow – возвведение в степень).

Выполнение работы

Проверим корректность настроек инструментальной цепи разработки. Для этого выполним в терминале WSL следующие команды:

- clang++ --version — проверка компилятора
- cmake --version — проверка системы сборки
- git config --global --list — проверка настроек Git
- ssh -T git@github.com — проверка подключения к GitHub

В результате выполнения команд мы должны увидеть установленные версии компилятора и CMake, данные пользователя Git и сообщение об успешной аутентификации на GitHub. Скриншот представлен на рисунке 1.



```
● [xiaomi@Xiaomi my-project]$ clang++ --version
ALT Linux Team clang version 19.1.7
Target: x86_64-unknown-linux-gnu
Thread model: posix
InstalledDir: /usr/lib/llvm-19.1/bin
● [xiaomi@Xiaomi my-project]$ cmake --version
cmake version 3.31.7

CMake suite maintained and supported by Kitware (kitware.com/cmake).
● [xiaomi@Xiaomi my-project]$ git --no-pager config --global --list
user.name=Korneev Gregory
user.email=gregory.korneev@gmail.com
● [xiaomi@Xiaomi my-project]$ ssh -T git@github.com
Hi gregkorneev! You've successfully authenticated, but GitHub does not provide shell access.
```

Рисунок 1 — Скриншот терминала с результатами проверки окружения

Настроим систему сборки CMake. Для этого выполним команду конфигурации проекта:

cmake -B build

Данная команда создаст папку build и сгенерирует в ней все необходимые файлы для сборки проекта. В процессе выполняется поиск компилятора, определяются его возможности и производится загрузка и настройка библиотеки для модульного тестирования Google Test.

В результате мы должны получить сообщения Configuring done и Generating done, что свидетельствует об успешной настройке системы сборки. Скриншот представлен на рисунке 2.

```
[xiaomi@Xiaomi rpn_calculator]$ cmake -B build
CMake Deprecation Warning at build/_deps/googletest-src/CMakeLists.txt:4 (cmake_minimum_required):
  Compatibility with CMake < 3.10 will be removed from a future version of
  CMake.

  Update the VERSION argument <min> value. Or, use the <min>...<max> syntax
  to tell CMake that the project requires at least <min> but has been updated
  to work with policies introduced by <max> or earlier.

CMake Deprecation Warning at build/_deps/googletest-src/googlemock/CMakeLists.txt:39 (cmake_minimum_required):
  Compatibility with CMake < 3.10 will be removed from a future version of
  CMake.

  Update the VERSION argument <min> value. Or, use the <min>...<max> syntax
  to tell CMake that the project requires at least <min> but has been updated
  to work with policies introduced by <max> or earlier.

CMake Deprecation Warning at build/_deps/googletest-src/googletest/CMakeLists.txt:49 (cmake_minimum_required):
  Compatibility with CMake < 3.10 will be removed from a future version of
  CMake.

  Update the VERSION argument <min> value. Or, use the <min>...<max> syntax
  to tell CMake that the project requires at least <min> but has been updated
  to work with policies introduced by <max> or earlier.

-- Configuring done (0.4s)
-- Generating done (0.0s)
-- Build files have been written to: /home/xiaomi/my-project/rpn_calculator/build
```

Рисунок 2 — Результат успешной конфигурации системы сборки проекта с автоматическим скачиванием и настройкой Google Test framework

После реализации базовой логики калькулятора (операции +, -, *, /) убедимся в его работоспособности. Соберём проект командой cmake --build build и запустим полученный исполняемый файл:

```
./build/rpn_calculator
```

Введём простое выражение в обратной польской нотации для проверки сложения:
5 3 +

Программа должна корректно обработать ввод и вывести результат вычисления. Скриншот представлен на рисунке 3.

```
[xiaomi@Xiaomi rpn_calculator]$ ./build/rpn_calculator
Enter RPN expression: 5 3 +
Result: 8
[xiaomi@Xiaomi rpn_calculator]$ ]
```

Рисунок 3 — Результат успешного вычисления выражения в обратной польской нотации «5 3 +»

Реализуем модульные тесты с использованием Google Test Framework для проверки корректности работы всех операций и обработки ошибок. После добавления тестов запустим их с помощью утилиты CTest:

```
cd build && ctest --verbose
```

В результате мы должны увидеть подробный вывод о прохождении всех тестовых случаев. Итоговое сообщение 100% tests passed подтверждает, что все функции калькулятора, включая новую операцию pow, работают корректно, а ошибки обрабатываются должным образом. Скриншот представлен на рисунке 4.

```
[xiaomi@Xiaomi rpn_calculator]$ cd build && ctest --verbose
UpdateCTestConfiguration from :/home/xiaomi/my-project/rpn_calculator/build/DartConfiguration.tcl
UpdateCTestConfiguration from :/home/xiaomi/my-project/rpn_calculator/build/DartConfiguration.tcl
Test project /home/xiaomi/my-project/rpn_calculator/build
Constructing a list of tests
Done constructing a list of tests
Updating test list for fixtures
Added 0 tests to meet fixture requirements
Checking test dependency graph...
Checking test dependency graph end
test 1
    Start 1: rpn_tests

1: Test command: /home/xiaomi/my-project/rpn_calculator/build/tests/rpn_tests
1: Working Directory: /home/xiaomi/my-project/rpn_calculator/build/tests
1: Test timeout computed to be: 10000000
1: Running main() from /home/xiaomi/my-project/rpn_calculator/build/_deps/googletest-src/googletest/src/gtest_main.cc
1: [=====] Running 4 tests from 1 test suite.
1: [-----] Global test environment set-up.
1: [-----] 4 tests from RPNTTest
1: [ RUN   ] RPNTTest.BasicAddition
1: [ OK    ] RPNTTest.BasicAddition (0 ms)
1: [ RUN   ] RPNTTest.ComplexExpression
1: [ OK    ] RPNTTest.ComplexExpression (0 ms)
1: [ RUN   ] RPNTTest.DivisionByZero
1: [ OK    ] RPNTTest.DivisionByZero (0 ms)
1: [ RUN   ] RPNTTest.InvalidExpression
1: [ OK    ] RPNTTest.InvalidExpression (0 ms)
1: [-----] 4 tests from RPNTTest (0 ms total)
1:
1: [-----] Global test environment tear-down
1: [=====] 4 tests from 1 test suite ran. (0 ms total)
1: [ PASSED ] 4 tests.

1/1 Test #1: rpn_tests ..... Passed 0.01 sec
```

Рисунок 4 — Финальный результат выполнения полного набора модульных тестов, подтверждающий корректность работы базовых операций и новой функции извлечения квадратного корня

Настроим автоматическую сборку и тестирование проекта на платформе GitHub Actions. Для этого был создан файл workflow (.github/workflows/ci.yml).

После его добавления и отправки кода в удалённый репозиторий процесс запустился автоматически.

В результате мы должны убедиться, что пайплайн завершился успешно. Для этого на странице репозитория на GitHub во вкладке Actions должно отображаться успешное выполнение задачи (зелёная галочка), что означает: проект успешно собирается и все тесты проходят на удалённом сервере. Скриншот представлен на рисунке 5.



Рисунок 5 — Успешное выполнение CI/CD пайплайна в GitHub Actions с зелёной галочкой статуса

Вывод: в результате выполнения работы был освоен на практике полный жизненный цикл разработки современного C++ приложения, от настройки окружения и написания кода до организации CI/CD, контейнеризации и документирования.

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. Быков, А. Ю. Решение задач на языках программирования Си и Си++ : методические указания / А. Ю. Быков. — Москва : МГТУ им. Н.Э. Баумана, 2017. — 248 с. — ISBN 978-5-7038-4577-6. — Текст : электронный // Лань : электронно-библиотечная система. — URL: <https://e.lanbook.com/book/103505>
2. Каширин, И. Ю. От Си к Си++ : учебное пособие / И. Ю. Каширин, В. С. Новичков. — 2-е изд., стер. — Москва : Горячая линия-Телеком, 2012. — 334 с. — ISBN 978-5-9912-0259-6. — Текст : электронный // Лань : электронно-библиотечная система. — URL: <https://e.lanbook.com/book/5161>
3. Быков А. Ю. Решение задач на языках программирования Си и Си++ : метод. указания к выполнению лаб. работ / Быков А. Ю. ; МГТУ им. Н. Э. Баумана. - М. : Изд-во МГТУ им. Н. Э. Баумана, 2017. - 244 с. : ил. - ISBN 978-5-7038-4577-6.
4. Иванова Г. С., Ничушкина Т. Н. Объектно-ориентированное программирование : учебник для вузов / Иванова Г. С., Ничушкина Т. Н. ; общ. ред. Иванова Г. С. - М. : Изд-во МГТУ Н.Э.Баумана, 2002. <http://progbook.ru/technologiya-programmirovaniya/582-ivanova-tehnologiya-programmirovaniya.html>
5. Иванова Г. С., Ничушкина Т. Н. Объектно-ориентированное программирование : учебник для вузов / Иванова Г. С., Ничушкина Т. Н. ; общ. ред. Иванова Г. С. - М. : Изд-во МГТУ им. Н. Э. Баумана, 2014. - 455 с. : ил. - Библиогр.: с. 450. - ISBN 978-5-7038-3921-8.
6. Подбельский В.В. Язык Си++: Учебное пособие. – М.: Финансы и статистика, 2003.
<http://progbook.ru/c/737-podbelskii-programmiovanie-na-yazyke-si.html>.
7. Акулов О.А., Медведев Н.В. Информатика. Базовый курс. – М.: Омега-Л, 2006. <http://razym.ru/naukaobraz/obrazov/151874-akulov-oa-medvedev-nv-informatika-bazovyy-kurs.html>
8. Вычислительные методы и программирование. МГУ им. М.В. Ломоносова. ISSN 1726-3522. Журнал входит в 1-й уровень Белого списка научных журналов Минобрнауки России. <https://num-meth.ru/index.php/journal/index>

Дополнительные материалы

1. Иванова Г.С. Технология программирования: Учебник для вузов. – М.: Изд. МГТУ им. Ахо А.В., Хопкрофт Д.Э., Ульман Д.Д. Структуры данных и алгоритмы. – М., Вильямс, 2003. <http://razym.ru/naukaobraz/obrazov/181547-aho-a-ulman-d-hopcroft-d-struktury-dannyh-i-algoritmy.html>
2. Дейтел Х.М., Дейтел П.Дж. Как программировать на C++. – М.: Бином, 2001.
3. Т. Кормен Т., Лейзерсон Ч., Ривест Р. Алгоритмы: построение и анализ. – М. МЦНМО, 2005. <http://rutracker.org/forum/viewtopic.php?t=533181>
4. Джосьютис Н. С++ Стандартная библиотека для профессионалов. – СПб.: Питер, 2004. http://progbook.ru/c/178-dzhosyutis_c_standartnaya_biblioteka.html
5. Подбельский В.В. Стандартный Си++: Учебное пособие. – М.: Финансы и статистика, 2008.
6. Объектно-ориентированное программирование в C++: пер. с англ. / Лафоре Р. - 4-е изд. - СПб.: Питер, 2004. - 923 с. - (Классика computer science). - ISBN 5-94723-302-9.
7. Т. Кормен Т., Лейзерсон Ч., Ривест Р. Алгоритмы: построение и анализ. – М. МЦНМО, 2005.
8. Г. Шилдт. С++. Базовый курс, 3-е издание: Пер с англ. – М.: Издательский дом «Вильямс», 2011. – 624 с.
9. Павловская Т. А. С/C++. Программирование на языке высокого уровня: учебник для вузов / Павловская Т. А. - СПб.: Питер, 2003. - 460 с. - (Учебник для вузов). - ISBN 5-94723-568-4.
10. Бесплатные образовательные программы партнера (VK):
<https://education.vk.com/students>