МИНИСТЕРСТВО ЦИФРОВОГО РАЗВИТИЯ, СВЯЗИ И МАССОВЫХ КОММУНИКАЦИЙ РОССИЙСКОЙ ФЕДЕРАЦИИ

Ордена Трудового Красного Знамени федеральное государственное бюджетное образовательное учреждение высшего образования

Московский технический университет связи и информатики

Кафедра «Информатика»

Лабораторный практикум по дисциплине

Введение в информационные технологии

Лабораторная работа № 1

«Инструментальные компоненты экосистемы C++: конфигурирование среды разработки и сборки кроссплатформенных приложений»

Выполнил:

студент группы БПС2601

Миронов М.А.

Проверил:

доц. Рюмин Ю.А.

Москва 2026

СОДЕРЖАНИЕ

1 Цель работы 3

2 Общее задание 4

3 Индивидуальное задание 5

4 Схема алгоритма решения задачи 6

5 Код программного проекта и порядок сборки 7

5.1 Среда разработки и структура проектов 7

5.2 Консольное приложение HelloCMake 7

5.3 GUI-приложение HelloQtCreator 10

6 Результаты выполнения программы 13

7 Вывод 14

1 Цель работы

Ознакомиться с инструментальными компонентами экосистемы C++, изучить технологию разработки консольных приложений и приложений с графическим пользовательским интерфейсом (GUI), а также реализовать и собрать проекты в операционной системе Linux с использованием Clang, CMake, Qt и Qt Creator.

2 Общее задание

1. Изучить основные средства и технологию разработки консольных и GUI-программных проектов в экосистеме C++ (Clang, GCC, CMake, Qt, Qt Creator, GitVerse и др.).
2. Выполнить все шаги по созданию, сборке и запуску проектов.

3 Индивидуальное задание

Номер варианта: 16.

В операционной системе Linux необходимо создать кроссплатформенное консольное приложение HelloCMake и GUI-приложение HelloQtCreator. Первый проект должен собираться с помощью CMake и выводить приветствие, сведения о компиляторе, стандарте C++, ФИО студента, группу и номер варианта. Второй проект должен быть создан в Qt Creator по шаблону «Qt Widgets Application» и отображать индивидуальные данные студента в графическом окне.

4 Схема алгоритма решения задачи

Консольное приложение имеет линейный алгоритм. После запуска устанавливается локаль, затем последовательно выводятся приветствие, сведения о компиляторе и стандарте языка, а также индивидуальные данные студента. После этого программа завершает работу с нулевым кодом возврата. Блок-схема алгоритма представлена на рисунке 1.

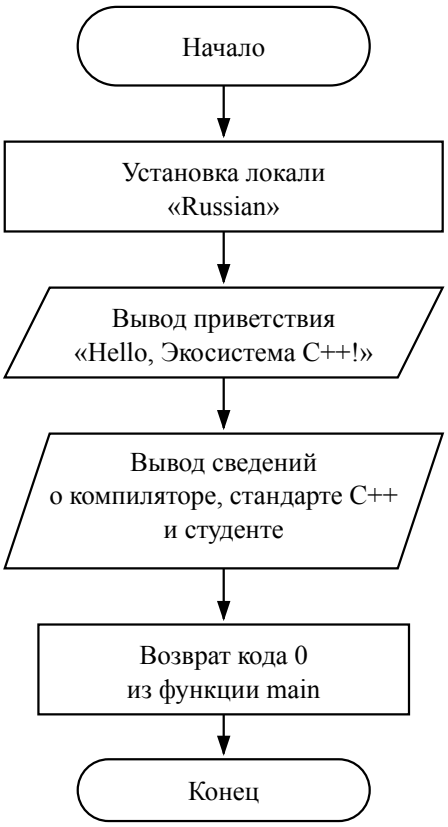


Рисунок 1 — Блок-схема алгоритма консольного приложения

GUI-приложение использует аналогичные индивидуальные данные, но вместо вывода в консоль создаёт объект QApplication, настраивает текстовую метку QLabel, отображает её и запускает цикл обработки событий Qt. Завершение цикла происходит после закрытия окна пользователем.

5 Код программного проекта и порядок сборки

5.1 Среда разработки и структура проектов

Работа выполнялась в Linux. Для консольного приложения использовались компилятор Clang 21.1.8, CMake и стандарт C++20. Графическое приложение создавалось в Qt Creator с Qt 6.10.2, qmake и стандартом C++17.

Файлы работы разделены на два независимых каталога. В part1 размещены HelloCMake.cpp, HelloCMake.h и CMakeLists.txt, а в part2 — файлы проекта Qt Creator: main.cpp, HelloQtCreator.pro, mainwindow.cpp, mainwindow.h и mainwindow.ui. Такое разделение позволяет использовать для проектов разные системы сборки.

5.2 Консольное приложение HelloCMake

Исходный код консольного приложения приведён в листинге 1. Функция setlocale устанавливает локаль для работы с текстом. Макрос __VERSION__ содержит версию компилятора Clang, а макрос __cplusplus — числовое обозначение используемого стандарта C++.

```cpp
// ===
// Лабораторная работа №1
// Консольное приложение "Hello, Экосистема C++"
// ===

#include <clocale>
#include <iostream>

using namespace std;

int main() {
    setlocale(LC_ALL, "Russian");
    cout << "Hello, Экосистема C++" << endl;
    cout << "Компилятор: Clang " << __VERSION__ << endl;
    cout << "Стандарт C++: " << __cplusplus << endl;
    cout << "Студент: Миронов М.А., группа БПС2601, вариант 16" <<
endl;

    return 0;
}
```

Листинг 1 — Исходный код файла HelloCMake.cpp

Конфигурация сборки приведена в листинге 2. Файл CMakeLists.txt объявляет проект и стандарт C++20, создаёт исполняемую цель HelloCMake и добавляет параметры предупреждений для компиляторов GCC и Clang. Команды message выводят сведения о конфигурации проекта.

```
cmake_minimum_required(VERSION 3.8)
project(HelloCMake
    VERSION 1.0
    DESCRIPTION "Демонстрационное приложение для ЛР №1"
    LANGUAGES CXX)

set(CMAKE_CXX_STANDARD 20)
set(CMAKE_CXX_STANDARD_REQUIRED ON)
set(CMAKE_CXX_EXTENSIONS OFF)

add_executable(HelloCMake
    "HelloCMake.cpp"
    "HelloCMake.h")

target_include_directories(HelloCMake PRIVATE
    ${CMAKE_CURRENT_SOURCE_DIR})

if(CMAKE_CXX_COMPILER_ID MATCHES "GNU|Clang")
    target_compile_options(HelloCMake PRIVATE
        -Wall -Wextra -Wpedantic)
endif()

set(CMAKE_RUNTIME_OUTPUT_DIRECTORY ${CMAKE_BINARY_DIR}/bin)

message(STATUS "========================================")
message(STATUS "Проект: ${PROJECT_NAME}")
message(STATUS "Версия: ${PROJECT_VERSION}")
message(STATUS "Стандарт C++: ${CMAKE_CXX_STANDARD}")
message(STATUS "Компилятор C++: ${CMAKE_CXX_COMPILER_ID}")
message(STATUS "Сборочный каталог: ${CMAKE_BINARY_DIR}")
message(STATUS "Исходный каталог: ${CMAKE_CURRENT_SOURCE_DIR}")
message(STATUS "Выходной каталог: ${CMAKE_RUNTIME_OUTPUT_DIRECTORY}")
message(STATUS "Платформа: ${CMAKE_SYSTEM_NAME}")
message(STATUS "========================================")
```

Листинг 2 — Основные параметры файла CMakeLists.txt

Для конфигурации и сборки в терминале Linux выполняются команды из листинга 3. Переменные окружения CC и CXX задают компилятор Clang. Команда cmake -S . -B build создаёт каталог сборки, а cmake --build build выполняет компиляцию и компоновку проекта.

```
cd part1
CC=clang CXX=clang++ cmake -S . -B build
cmake --build build
./build/HelloCMake
```

Листинг 3 — Команды сборки и запуска HelloCMake в Linux

Процесс конфигурации, сборки и запуска проекта показан на рисунке 2.

```
[zethange@tecno:~/Repositories/mtuci/introduction-to-it/solution/lab1]$ cd part1

[zethange@tecno:~/Repositories/mtuci/introduction-to-it/solution/lab1/part1]$ CC=clang CXX=clang++ cmake -S . -B build
CMake Warning (deprecated) at CMakeLists.txt:1 (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.
This warning is for project developers.  Use -Wno-author or -Wno-deprecated
to suppress it.

-- The CXX compiler identification is Clang 21.1.8
-- Detecting CXX compiler ABI info
-- Detecting CXX compiler ABI info - done
-- Check for working CXX compiler: /run/current-system/sw/bin/clang++ - skipped
-- Detecting CXX compile features
-- Detecting CXX compile features - done
-- =====================================
-- Проект: HelloCMake
-- Версия: 1.0
-- Стандарт C++: 20
-- Компилятор C++: Clang
-- Сборочный каталог: /home/zethange/Repositories/mtuci/introduction-to-it/solution/lab1/part1/build
-- Исходный каталог: /home/zethange/Repositories/mtuci/introduction-to-it/solution/lab1/part1
-- Выходной каталог: /home/zethange/Repositories/mtuci/introduction-to-it/solution/lab1/part1/build/bin
-- Платформа: Linux
-- =====================================
-- Configuring done (1.5s)
-- Generating done (0.0s)
-- Build files have been written to: /home/zethange/Repositories/mtuci/introduction-to-it/solution/lab1/part1/build

[zethange@tecno:~/Repositories/mtuci/introduction-to-it/solution/lab1/part1]$ cmake --build build
[ 50%] Building CXX object CMakeFiles/HelloCMake.dir/HelloCMake.cpp.o
[100%] Linking CXX executable HelloCMake
[100%] Built target HelloCMake

[zethange@tecno:~/Repositories/mtuci/introduction-to-it/solution/lab1/part1]$ ./build/HelloCMake
Hello, Экосистема C++
Компилятор: Clang Clang 21.1.8
Стандарт C++: 202002
Студент: Миронов М.А., группа БПС2601, вариант 16

[zethange@tecno:~/Repositories/mtuci/introduction-to-it/solution/lab1/part1]$
```

Рисунок 2 — Процесс конфигурации, сборки и запуска консольного приложения HelloCMake

5.3 GUI-приложение HelloQtCreator

В Qt Creator выбраны команды «Создать проект» → «Application (Qt)» → «Qt Widgets Application». Проекту присвоено имя HelloQtCreator, в качестве системы сборки выбран qmake. После выбора доступного комплекта Qt Creator создал файлы главного окна и файл конфигурации проекта. Создание проекта показано на рисунке 3.

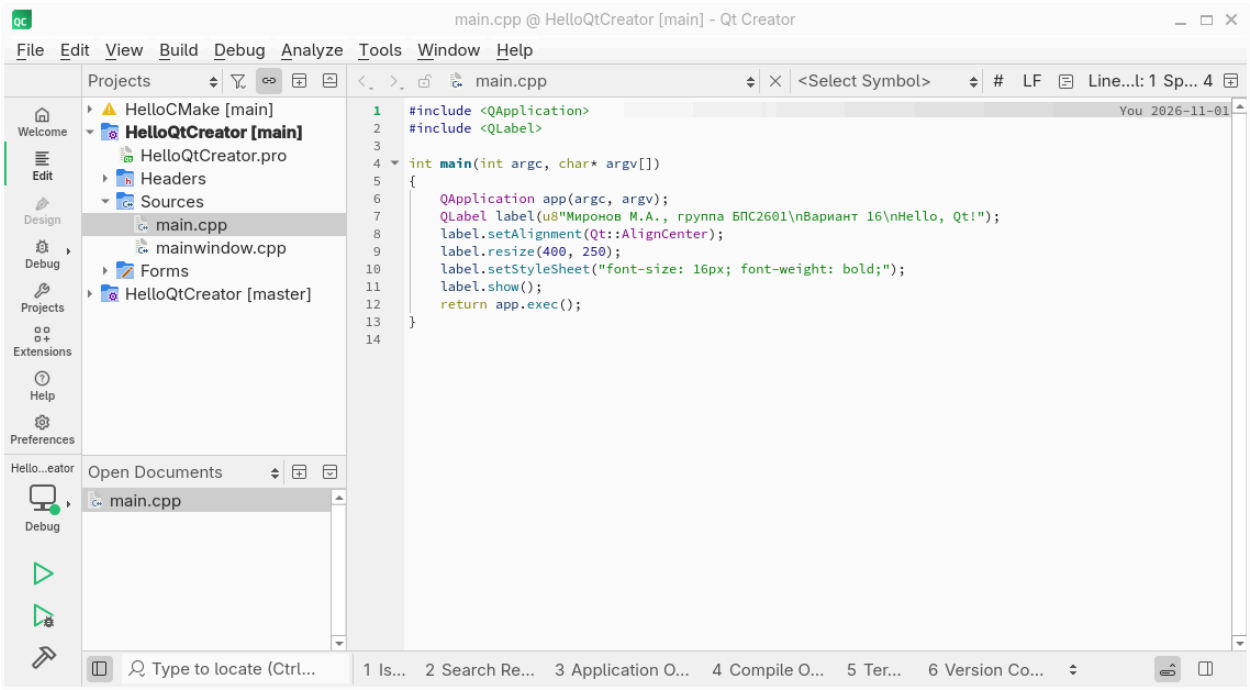


Рисунок 3 — Проект HelloQtCreator в Qt Creator

В файл main.cpp добавлен код из листинга 4. Строка с префиксом u8 содержит индивидуальные данные в UTF-8. Методы setAlignment, resize и setStyleSheet задают выравнивание текста, размеры окна и его оформление. Метод exec запускает цикл обработки событий приложения.

```
#include <QApplication>
#include <QLabel>

int main(int argc, char* argv[])
{
    QApplication app(argc, argv);
    QLabel label(u8"Миронов М.А., группа БПС2601\nВариант 16\nHello,
Qt!");
    label.setAlignment(Qt::AlignCenter);
    label.resize(400, 250);
    label.setStyleSheet("font-size: 16px; font-weight: bold;");
    label.show();
    return app.exec();
}
```

Листинг 4 — Исходный код файла main.cpp приложения HelloQtCreator

Сборка выполняется командой «Сборка» → «Собрать проект» или сочетанием Ctrl+B, а запуск — командой «Сборка» → «Запустить» или

сочетанием Ctrl+R. В результате отображается окно с ФИО студента, группой, номером варианта и приветствием.

6 Результаты выполнения программы

После сборки HelloCMake программа была запущена в терминале Linux. Фактический вывод приведён в листинге 5. Число 202002 соответствует стандарту C++20. Строка версии __VERSION__ в Clang уже содержит название компилятора, поэтому в выводе слово «Clang» повторяется. Программа завершилась с кодом возврата 0.

```
Hello, Экосистема C++
Компилятор: Clang Clang 21.1.8
Стандарт C++: 202002
Студент: Миронов М.А., группа БПС2601, вариант 16
```

Листинг 5 — Результат работы консольного приложения HelloCMake

При запуске HelloQtCreator отображается окно размером 400 × 250 пикселей. Текст располагается по центру и содержит три строки: индивидуальные данные студента, номер варианта и сообщение «Hello, Qt!». Результат работы приложения представлен на рисунке 4.

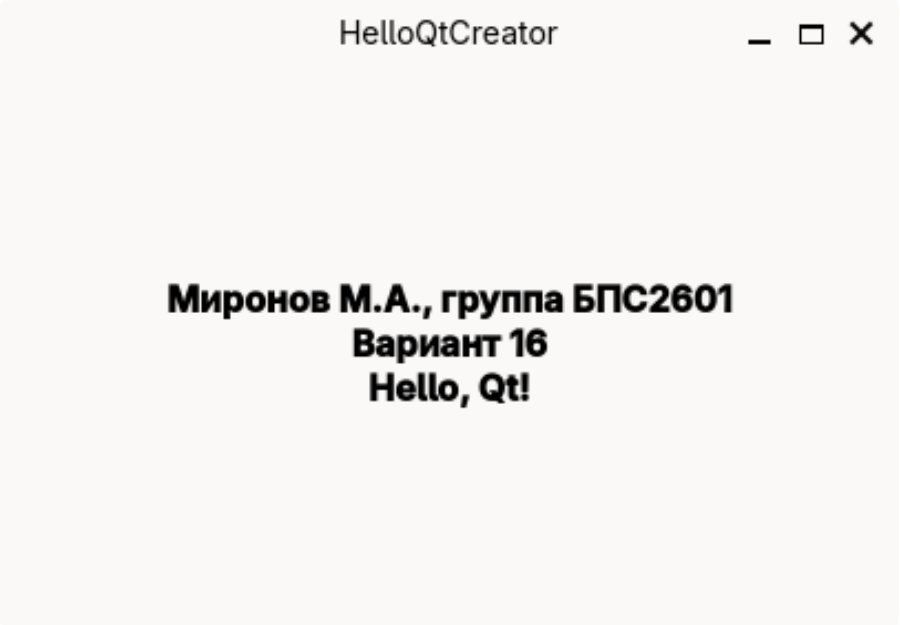


Рисунок 4 — Результат работы GUI-приложения HelloQtCreator

7 Вывод

В ходе выполнения лабораторной работы были изучены инструментальные компоненты экосистемы C++ в Linux и реализованы консольное и GUI-приложения. Для консольного проекта HelloCMake написан код программы, построена блок-схема алгоритма и подготовлена конфигурация CMake со стандартом C++20. Графический проект HelloQtCreator создан в Qt Creator с Qt 6 и системой сборки qmake.

Консольное приложение было собрано компилятором Clang и запущено в терминале. Проверка показала, что вывод содержит требуемые сведения, используется стандарт C++20 и программа завершается с нулевым кодом возврата. В графическом приложении реализовано отображение индивидуальных данных с помощью QLabel.

Исходные файлы были опубликованы в личном форке на GitVerse с использованием системы контроля версий Git. Таким образом, были освоены этапы создания, сборки, запуска, проверки и публикации кроссплатформенных программных проектов, которые составляют основу дальнейшей разработки приложений на C++.