МИНОБРНАУКИ РФ

ФГБОУ ВО «ИжГТУ имени М.Т. Калашникова»

Кафедра «Программное обеспечение»

Отчет

по лабораторной работе №3-4 по ОС

Выполнил студент группы Б20-191-2: Широбоков А.А.

Принял: Тарасов В.Г.

Ижевск 2022

1. Настройка проекта

Пройдя дальнейшие шаги настройки проекта получим CMake проект, использующий Vcpkg для управления зависимостями.

Перевод [руководства](https://github.com/HexDecimal/libtcod-vcpkg-template)(READ.md) для развертки CMake шаблона:

Вам нужен CMake и компилятор, который CMake может распознать. Vcpkg настраивается как подмодуль, и его не нужно устанавливать отдельно.

Компиляция этой программы из IDE, такой как VSCode, вызовет CMake, который, в свою очередь, вызовет Vcpkg. Vcpkg извлекает и создает все зависимости, необходимые для проекта, а CMake скомпилирует и свяжет их. Это означает, что вам не нужно вручную обрабатывать зависимости, поскольку все они обрабатываются цепочкой инструментов.

Каталог src/ содержит программу hello world. CMake настроен на компиляцию всех файлов \*.cpp в этом каталоге для создания новой программы. Вы должны отредактировать или заменить исходники в этом каталоге, чтобы создать свою программу. Если вы не хотите использовать этот каталог, вы можете изменить, какие файлы собираются в CMakeLists.txt.

По адресу .github/workflows/cmake.yml существует рабочий процесс, который автоматически компилирует программу для Windows, macOS и Ubuntu Linux. Рабочий процесс просто вызывает CMake напрямую, поскольку эта настройка не зависит от конкретной IDE для работы. Действия, которые проходят в обязательном порядке, предоставят архивированные исполняемые файлы для тестирования, они являются временными и загружаются из действия прохождения в автоматизированных сборках. Чтобы опубликовать эти сборки на постоянной основе, вы можете добавить аннотированный тег с именем версии сборки, например 1.0.0 или 2000.12.30.

* Для начала нужно установить необходимые инструменты: Visual Studio, Visual Code, последнюю версию [CMake](https://cmake.org/download/), [Git](https://git-scm.com/downloads).
* Создаем аккаунт Github если его ещё нету, [создаем новый репозиторий](https://docs.github.com/en/repositories/creating-and-managing-repositories/creating-a-repository-from-a-template#creating-a-repository-from-a-template) по этому [шаблону](https://github.com/HexDecimal/libtcod-vcpkg-template).
* [Клонируем](https://docs.github.com/en/github/creating-cloning-and-archiving-repositories/cloning-a-repository-from-github/cloning-a-repository) созданный репозиторий в любую папку: ПКМ - git clone %ссылка на ваш репозиторий%.
* [Инициализируем](https://git-scm.com/book/en/v2/Git-Tools-Submodules) vcpkg, для этого нужно зайти в пустую папку vcpkg, нажать ПКМ – Git Bash Here, затем прописать последовательно:

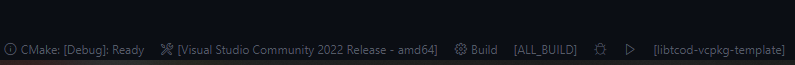
git submodule init

git submodule update

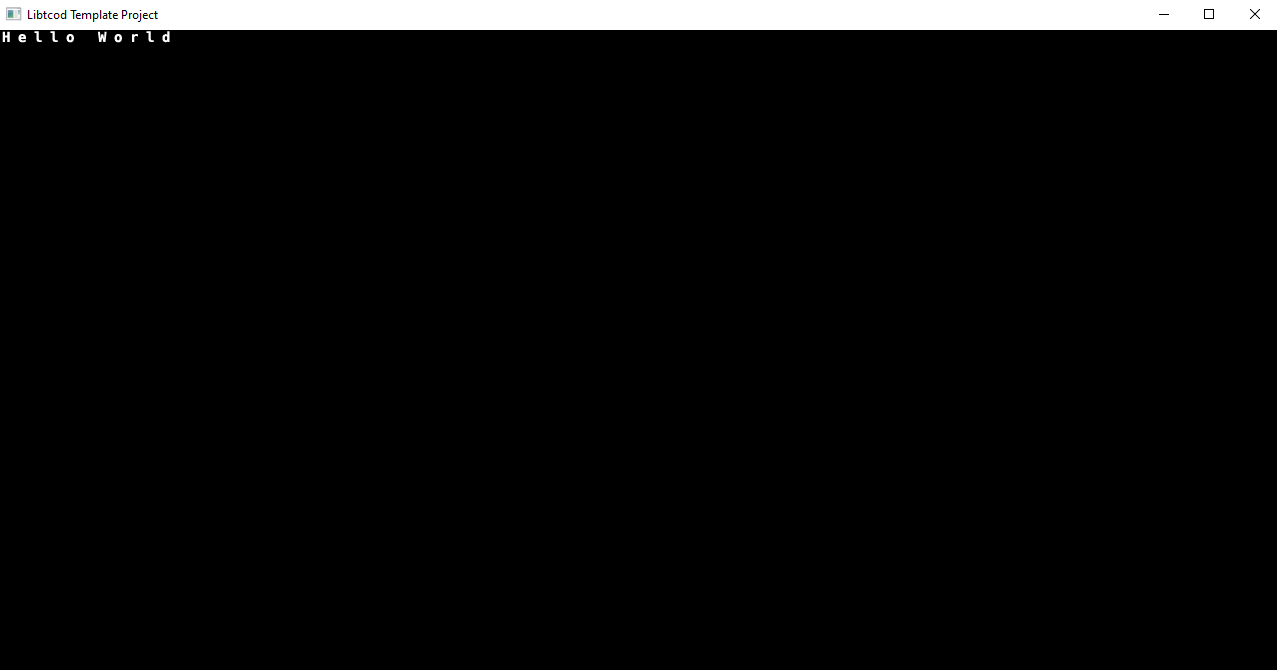
Подождать окончания клонирования vcpkg репозитория.

* Заходим в VScode, открываем папку с нашим проектом, устанавливаем плагины: *C/C++*, *C++ Intellisense*, and *CMake Tools.*
* Когда CMake предлагает выбрать инструмент выбираем (Visual Studio Community ... - amd64). Если уведомление не появилось, в нижней части экрана находим нажимаем на CMake.
* Когда CMake спрашивает про проект выбираем (libtcod-vcpkg-template). Найти данную опцию можно также в нижней части экрана.

В итоге должно выглядеть так:

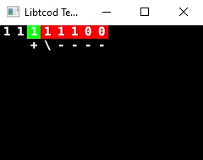


* Ждем пока скачаются внешние библиотеки и нажимаем F5, должны увидеть консоль и в ней Hello World.



* Если после нажатия F5 консоль не появляется, а после остановки программы вылазит ошибка – перезаходим в VScode.
* Для работы с библиотеками SDL и Libtcod – читаем соответствующие документации. Документация: для [SDL](https://wiki.libsdl.org/SDL2/APIByCategory), для [libtcod](https://libtcod.readthedocs.io/en/latest/index.html).
* Можно изменить libtcod-vcpkg-template в CMakeLists.txt и .github/workflows/cmake.yml на название вашего проекта. Возможно, вы захотите отредактировать и другие части скрипта CMake, например, включенные исходные файлы. Настройте стиль редактора в .clang-format и .editorconfig.

22. Результаты работы

33. Код программы(c комментариями)

#include <SDL.h>

#include <cstdlib>

#include <filesystem>

#include <iostream>

#include <libtcod.hpp>

#include <sstream>

#include <string>

#if defined(\_MSC\_VER)

#pragma warning(disable : 4297) *// Allow "throw" in main().  Letting the compiler handle termination.*

#endif

*/// Return the data directory.*

auto get\_data\_dir() -> std::filesystem::path {

    static auto root\_directory = std::filesystem::path{"."}; *// Begin at the working directory.*

    while (!std::filesystem::exists(root\_directory / "data")) {

*// If the current working directory is missing the data dir then it will assume it exists in any parent*

*// directory.*

        root\_directory /= "..";

        if (!std::filesystem::exists(root\_directory)) {

            throw std::runtime\_error("Could not find the data directory.");

        }

    }

    return root\_directory / "data";

};

static tcod::Console g\_console; *// The global console object.*

static tcod::Context g\_context; *// The global libtcod context.*

int i, n, t = 0, k = 0, minind = 0;

std::string s;

bool input\_flag = false;

bool do\_logic = false;

const Uint8\* keys = SDL\_GetKeyboardState(NULL);

void nload(int j) { //функция для анимации загрузки

    for (int h = 0; h < 2; ++h) {

        tcod::print(g\_console, {j, 1}, "-", TCOD\_white, std::nullopt);

        g\_context.present(g\_console);

*// Sleep(50);*

        tcod::print(g\_console, {j, 1}, "\\", TCOD\_white, std::nullopt);

        g\_context.present(g\_console);

*//  Sleep(50);*

        tcod::print(g\_console, {j, 1}, "|", TCOD\_white, std::nullopt);

        g\_context.present(g\_console);

*//  Sleep(50);*

        tcod::print(g\_console, {j, 1}, "/", TCOD\_white, std::nullopt);

        g\_context.present(g\_console);

*//   Sleep(50);*

        tcod::print(g\_console, {j, 1}, "-", TCOD\_white, std::nullopt);

        g\_context.present(g\_console);

*//  Sleep(50);*

    }

}

*/// Game loop.*

void main\_loop() {

*// Rendering.*

*//     std::cout << s[i];*

*// }*

*// Handle input.*

    SDL\_Event event;

    while (SDL\_PollEvent(&event)) {

        switch (event.type) {

            case SDL\_QUIT:

                std::exit(EXIT\_SUCCESS);

                break;

            case SDL\_KEYDOWN: // если нажата 1 или 0, прибавляем к строке, если enter

// запрещаем ввод и запускаем алгоритм в do\_logic

                if (event.key.keysym.sym == SDLK\_1 && !input\_flag)

                    s += "1";

                else if (event.key.keysym.sym == SDLK\_0 && !input\_flag)

                    s += "0";

                else if (event.key.keysym.sym == SDLK\_RETURN && !input\_flag) {

                    input\_flag = true;

                    do\_logic = true;

                }

                break;

            default:

                break;

        }

        if (do\_logic) { // после нажатия enter перейдем сюда, здесь алгоритм следующего

// сочетания

            n = s.length();

            TCOD\_console\_set\_char\_background(g\_console.get(), n - 1, 0, TCOD\_red, TCOD\_BKGND\_OVERLAY);

            for (i = n - 1; i > 1; --i) {

                if (s[i - 1] < s[i]) {

                    tcod::print(g\_console, {i, 1}, "<", TCOD\_white, std::nullopt);

                    g\_context.present(g\_console);

                    nload(i - 1);

                    tcod::print(g\_console, {i - 1, 1}, ">", TCOD\_white, std::nullopt);

                    TCOD\_console\_set\_char\_background(g\_console.get(), i - 1, 0, TCOD\_green, TCOD\_BKGND\_OVERLAY);

                    g\_context.present(g\_console);

                    t = i - 1;

                    break;

                } else {

                    tcod::print(g\_console, {i, 1}, "<", TCOD\_white, std::nullopt);

                    g\_context.present(g\_console);

                    nload(i - 1);

                    tcod::print(g\_console, {i, 1}, "-", TCOD\_white, std::nullopt);

                    TCOD\_console\_set\_char\_background(g\_console.get(), i - 1, 0, TCOD\_red, TCOD\_BKGND\_OVERLAY);

                    g\_context.present(g\_console);

                    t = 0;

                }

            }

            if (t == 0) {

            } else {

                k = s[t + 1];

                s[t + 1] = s[t];

                s[t] = k;

                nload(t);

                tcod::print(g\_console, {t, 1}, "+", TCOD\_white, std::nullopt);

                g\_context.present(g\_console);

                tcod::print(g\_console, {t, 0}, std::string{s[t]}, TCOD\_white, std::nullopt);

                g\_context.present(g\_console);

                nload(t + 1);

                tcod::print(g\_console, {t + 1, 0}, std::string{s[t + 1]}, TCOD\_white, std::nullopt);

                TCOD\_console\_set\_char\_background(g\_console.get(), t + 1, 0, TCOD\_green, TCOD\_BKGND\_OVERLAY);

                g\_context.present(g\_console);

                std::sort(s.begin() + t + 1, s.end());

            }

            for (i = t + 1; i < n; ++i) {

                nload(i);

                tcod::print(g\_console, {i, 1}, "+", TCOD\_white, std::nullopt);

                g\_context.present(g\_console);

                tcod::print(g\_console, {i, 0}, std::string{s[i]}, TCOD\_white, std::nullopt);

                TCOD\_console\_set\_char\_background(g\_console.get(), i, 0, TCOD\_green, TCOD\_BKGND\_OVERLAY);

                g\_context.present(g\_console);

            }

            do\_logic = false;

        }

*// g\_console.clear();*

        if (!input\_flag) {

            tcod::print(g\_console, {0, 0}, s, TCOD\_white, std::nullopt);

            g\_context.present(g\_console);

        }

    }

}

*/// Main program entry point.*

int main(int argc, char\*\* argv) {

    try {

        auto params = TCOD\_ContextParams{};

        params.tcod\_version = TCOD\_COMPILEDVERSION;

        params.argc = argc;

        params.argv = argv;

        params.renderer\_type = TCOD\_RENDERER\_SDL2;

        params.vsync = 1;

        params.sdl\_window\_flags = SDL\_WINDOW\_RESIZABLE;

        params.window\_title = "Libtcod Template Project";

        auto tileset = tcod::load\_tilesheet(get\_data\_dir() / "dejavu16x16\_gs\_tc.png", {32, 8}, tcod::CHARMAP\_TCOD);

        params.tileset = tileset.get();

        g\_console = tcod::Console{15, 10};

        params.console = g\_console.get();

        g\_context = tcod::Context(params);

        while (true) main\_loop();

    } catch (const std::exception& exc) {

        std::cerr << exc.what() << "\n";

        throw;

    }

}