

### МИНОБРНАУКИ РОССИИ

### федеральное государственное бюджетное образовательное учреждение высшего образования

### «Московский государственный технологический университет «СТАНКИН» (ФГБОУ ВО «МГТУ «СТАНКИН»)

### Институт информационных систем и технологий

### Кафедра инженерной графики

**Отчет по выполнению лабораторной работы №1**

## по дисциплине «Геометрическое моделирование и компьютерная графика»

## Направление подготовки: 09.03.03 «Прикладная информатика»

## Профиль: «Управление данными»

## Руководитель: Бейреш А. М.

Оценка Подпись Дата

## Студент:

## Хусниярова А. Р.

## Группа ИДБ-20-11

## 

Подпись Дата

## Москва, 2023

**СОДЕРЖАНИЕ**

[Задание 3](#_Toc152108709)

[Код 3](#_Toc152108710)

[Результаты работы программы 6](#_Toc152108711)

# Задание

## №4. Сделать изображение разных функций на кнопки(sin, cos, x^2 и т.д.) с использованием библиотек SFML и ImGui.

# Код

#include <cmath>

#include <iostream>

#include "SFML/Graphics/RenderWindow.hpp"

#include "SFML/Graphics/VertexArray.hpp"

#include "SFML/System/Clock.hpp"

#include "SFML/Window/Event.hpp"

#include "imgui-SFML.h"

#include "imgui.h"

// Прототипы функций

float SinFunction(float x);

float CosFunction(float x);

float SquareFunction(float x);

float SqrtFunction(float x);

float ExpFunction(float x);

// Структура для хранения информации о каждой кнопке

struct FunctionButton

{

std::string name;

float (\*function)(float); // Указатель на функцию

};

FunctionButton functionButtons[] = {{"sin(x)", SinFunction},

{"cos(x)", CosFunction},

{"x^2", SquareFunction},

{"sqrt(x)", SqrtFunction},

{"exp(x)", ExpFunction}};

float (\*selectedFunction)(float) = nullptr; // Выбранная функция

// Реализация функций

float SinFunction(float x) { return std::sin(x); }

float CosFunction(float x) { return std::cos(x); }

float SquareFunction(float x) { return x \* x; }

float SqrtFunction(float x) { return std::sqrt(x); }

float ExpFunction(float x) { return std::exp(x); }

void HandleUserInput(sf::RenderWindow &window, const sf::Event &event)

{

switch (event.type)

{

case sf::Event::Closed:

window.close();

break;

default:

break;

}

}

void Update(sf::RenderWindow &window, const sf::Time &deltaClock)

{

// Make some time-dependent updates, like: physics, gameplay logic, animations, etc.

}

void RenderGui(sf::RenderWindow &window)

{

ImGui::Begin("Function Buttons");

for (const auto &button : functionButtons)

{

if (ImGui::Button(button.name.c\_str()))

{

// Выбираем функцию для отображения

selectedFunction = button.function;

}

}

ImGui::End();

}

void Render(sf::RenderWindow &window)

{

if (selectedFunction != nullptr)

{

sf::VertexArray graph(sf::LineStrip, 800);

for (int i = 0; i < 800; ++i)

{

float x = static\_cast<float>(i) / 100.0f - 4.0f; // Устанавливаем диапазон x от -4 до 4

float y = selectedFunction(x);

graph[i].position = sf::Vector2f(100.0f \* x + window.getSize().x / 2, -50.0f \* y + window.getSize().y / 2);

graph[i].color = sf::Color::Black; // Цвет графика

}

window.draw(graph);

}

}

int main()

{

sf::RenderWindow window(sf::VideoMode(800, 600), "Function Buttons");

window.setFramerateLimit(60);

if (!ImGui::SFML::Init(window))

{

std::cout << "ImGui initialization failed\n";

return -1;

}

sf::Clock deltaClock;

while (window.isOpen())

{

sf::Event event;

while (window.pollEvent(event))

{

ImGui::SFML::ProcessEvent(window, event);

HandleUserInput(window, event);

}

sf::Time deltaTime = deltaClock.restart();

ImGui::SFML::Update(window, deltaTime);

Update(window, deltaTime);

window.clear(sf::Color::White); // Установка цвета очистки на белый

RenderGui(window);

Render(window);

ImGui::SFML::Render(window);

window.display();

}

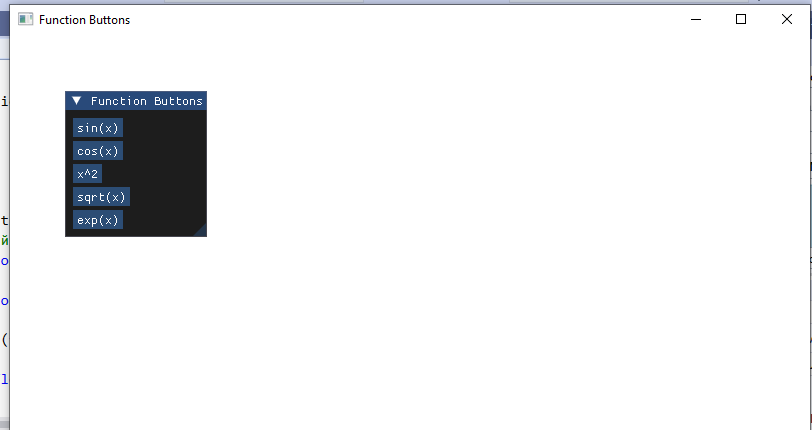
ImGui::SFML::Shutdown();

return 0;

}

# Результаты работы программы

## После запуска программы получим следующее окно:



## Если поочередно нажимать на кнопки, можно наблюдать отрисовку графиков:

## 



