

**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: Толок А.В.

Оценка Подпись Дата

Студент:

Баканов П.О.

Группа ИДБ-20-11

Подпись Дата

Москва, 2023

# Задание

# Сделать изображение разных функций на кнопки(sin, cos, x^2 и т.д.).

# Код

Для выполнения лабораторной работы за основу взят проект второй лабораторной.

sf::VertexArray drawSin(sf::RenderWindow &window, float amplitude, float frequency, float phase, float left,

float right)

{

sf::VertexArray lines(sf::LinesStrip, 800);

for (int x = left; x < right; x++)

{

lines[x].position = sf::Vector2f(x, window.getSize().y / 2 - amplitude \* sin(frequency \* x + phase));

lines[x].color = sf::Color::Red;

}

return lines;

}

sf::VertexArray drawCos(sf::RenderWindow &window, float amplitude, float frequency, float phase, float left, float right)

{

sf::VertexArray lines(sf::LinesStrip, 800);

for (int x = left; x < right; x++)

{

lines[x].position = sf::Vector2f(x, window.getSize().y / 2 - amplitude \* cos(frequency \* x + phase));

lines[x].color = sf::Color::Green;

}

return lines;

}

sf::VertexArray drawSquare(sf::RenderWindow &window, float scale, float left, float right)

{

sf::VertexArray lines(sf::LinesStrip, 800);

for (int x = left; x < right; x++)

{

lines[x].position = sf::Vector2f(x, window.getSize().y / 2 - scale \* x \* x);

lines[x].color = sf::Color::Blue;

}

return lines;

}

void drawAxes(sf::RenderWindow &window)

{

sf::VertexArray axes(sf::Lines, 4);

axes[0].position = sf::Vector2f(0, window.getSize().y / 2);

axes[1].position = sf::Vector2f(window.getSize().x, window.getSize().y / 2);

axes[0].color = axes[1].color = sf::Color::Black;

axes[2].position = sf::Vector2f(window.getSize().x / 2, 0);

axes[3].position = sf::Vector2f(window.getSize().x / 2, window.getSize().y);

axes[2].color = axes[3].color = sf::Color::Black;

window.draw(axes);

}

void Render(sf::RenderWindow& window)

{

drawAxes(window);

if (drawFunction)

{

window.draw(currentFunction);

}

}

void RenderGui(sf::RenderWindow& window)

{

ImGui::Begin("Functions");

if (ImGui::Button("sin"))

{

currentFunction = drawSin(window, 100, 0.01, 0, 0, 800);

drawFunction = true;

}

if (ImGui::Button("cos"))

{

currentFunction = drawCos(window, 100, 0.01, 0, 0, 800);

drawFunction = true;

}

if (ImGui::Button("x^2"))

{

currentFunction = drawSquare(window, 0.01, 0, 800);

drawFunction = true;

}

ImGui::End();

}

int main()

{

sf::RenderWindow window(sf::VideoMode(800, 800), "Lab1");

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);

RenderGui(window);

Render(window);

ImGui::SFML::Render(window);

window.display();

}

ImGui::SFML::Shutdown();

return 0;

}

# Результат работы программы

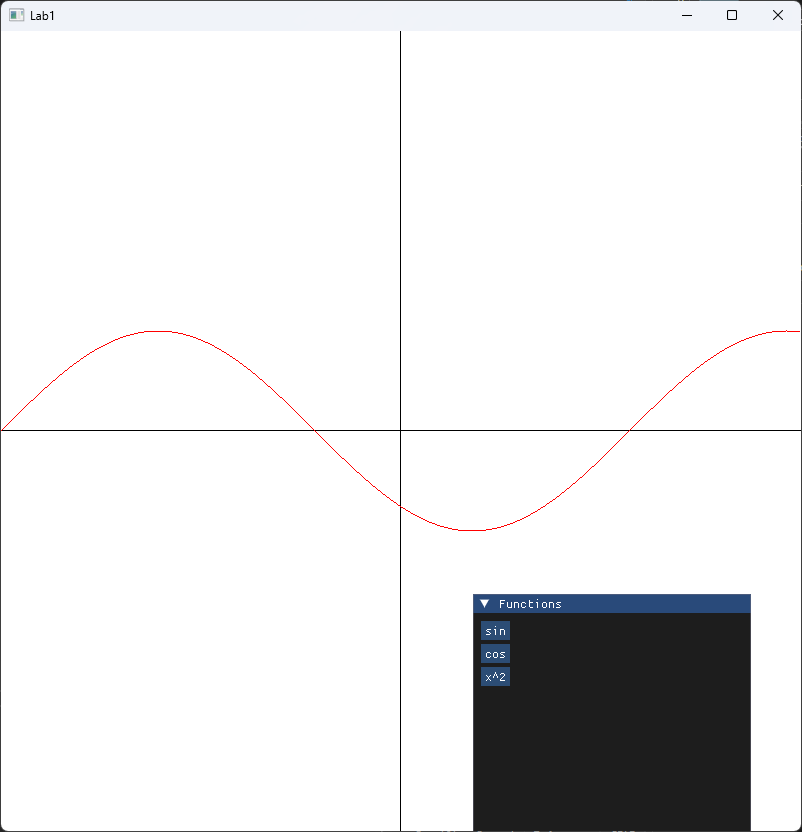


Рис1.

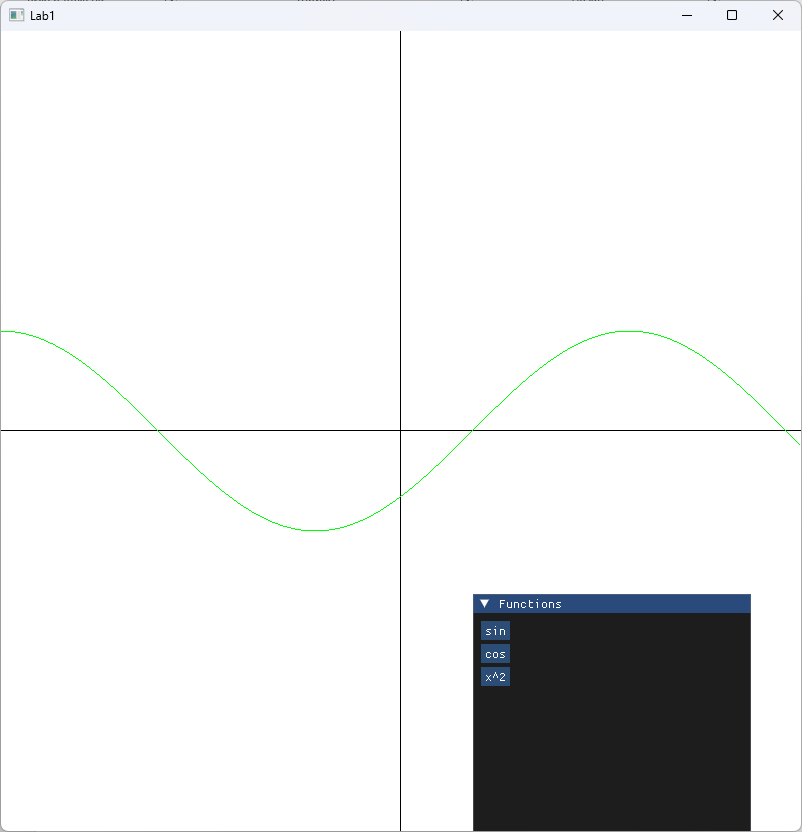


Рис2.