|  |  |
| --- | --- |
| **Gerb-BMSTU_01** | **Министерство науки и высшего образования Российской Федерации**  Калужский филиал  федерального государственного бюджетного  образовательного учреждения высшего образования  ***«Московский государственный технический университет имени Н.Э. Баумана (национальный исследовательский университет)»***  ***(КФ МГТУ им. Н.Э. Баумана)*** |

**ФАКУЛЬТЕТ** \_***ИУК «Информатика и управление»*\_\_**\_\_\_\_\_\_\_\_\_\_\_\_

**КАФЕДРА** \_\_***ИУК4 «Программное обеспечение ЭВМ, информационные технологии»***

**ДОМАШНЯЯ РАБОТА №1**

**«Библиотека SFML»**

**ДИСЦИПЛИНА: «Высокоуровневое программирование»**

|  |  |  |
| --- | --- | --- |
| Выполнил: студент гр. ИУК4-22Б | | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ ( Карельский М.К. )  (Подпись) (Ф.И.О.) |
| Проверил: | | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ ( Козина А.В. )  (Подпись) (Ф.И.О.) |
| Дата сдачи (защиты):  Результаты сдачи (защиты): | | |
|  | - Балльная оценка:  - Оценка: | |
| Калуга , 2021 | | |

**Цель:** формирование практических навыков реализации графических программ с использованием библиотеки SFML.

**Задачи:**

1. Познакомиться с разработкой графических программ на языке программирования C++
2. Изучить основные процедуры и функции библиотеки SFML

**Вариант 1**

**Задание:**

**Задание 1.**

Напишите программу вычерчивания дуги окружности по заданному радиусу и координатам центра окружности. Параметры дуги могут вводиться в форме координат начальной и конечной точек или углов, соответствующих началу и конку дуги.

**Задание 2.**

Преобразовать дугу в сектор и закрасить. Сектор должен вращаться по часовой стрелке и перемещаться вдоль экрана.

**Блок-схема:**



**Рис. 1.1.** Блок-схема



**Рис. 2.2.** Блок-схема



**Рис. 3.3.** Блок-схема

**Листинг:**

#include <SFML/Graphics.hpp>

#define \_USE\_MATH\_DEFINES

#include <math.h>

#include <iostream>

using namespace sf;

const unsigned short FACES = 20;

const unsigned int CENTER\_RADIUS = 5;

const float WINDOW\_WIDTH = 1000;

const float WINDOW\_HEIGTH = 500;

const float ROTATE\_SPEED = 1000;

const float MOVE\_SPEED = 0.1;

int main()

{

setlocale(LC\_ALL, "Russian");

std::cout << "Введите радиус: ";

float radius;

std::cin >> radius;

std::cout << "Введите координаты центра: ";

float centerX;

float centerY;

std::cin >> centerX >> centerY;

std::cout << "Как вы хотите задать дугу?\n";

std::cout << "0. Через координаты\n";

std::cout << "1. Через углы\n";

std::cout << ">>> ";

int mode{};

std::cin >> mode;

while (mode != 0 && mode != 1)

{

std::cout << "Ошибка ввода, попробуйте еще раз\n";

std::cout << ">>> ";

std::cin >> mode;

}

float startAngle{};

float endAngle{};

if (mode == 0)

{

std::cout << "Введите координаты начальной точки: ";

float startPointX;

float startPointY;

std::cin >> startPointX >> startPointY;

std::cout << "Введите координаты конечной точки: ";

float endPointX;

float endPointY;

std::cin >> endPointX >> endPointY;

startAngle = atan((centerY - startPointY) / (startPointX - centerX));

endAngle = atan((centerY - endPointY) / (endPointX - centerX));

if (startPointX < centerX)

{

startAngle += M\_PI;

}

if (endPointX < centerX)

{

endAngle += M\_PI;

}

}

else if (mode == 1)

{

std::cout << "Введите угол начальной точки (град): ";

std::cin >> startAngle;

startAngle \*= M\_PI / 180;

std::cout << "Введите угол конечной точки (град): ";

std::cin >> endAngle;

endAngle \*= M\_PI / 180;

}

float angularLength = abs(endAngle - startAngle);

std::cout << "Чтобы перейти к сектору, нажмите enter\n";

CircleShape center(CENTER\_RADIUS);

center.setPosition(centerX - CENTER\_RADIUS, centerY - CENTER\_RADIUS);

center.setFillColor(Color(0, 0, 0));

VertexArray arc(LineStrip, FACES + 1);

VertexArray sector(TriangleFan, FACES + 2);

float angle = startAngle;

for (unsigned short i{}; i < FACES + 1; ++i)

{

arc[i].color = Color(0, 0, 0);

sector[i].color = Color(0, 0, 0);

arc[i].position = Vector2f(centerX + radius \* cos(angle), centerY - radius \* sin(angle));

angle += angularLength / FACES;

}

sector[FACES + 1].color = Color(0, 0, 0);

RenderWindow window(VideoMode(WINDOW\_WIDTH, WINDOW\_HEIGTH), "Arc and sector");

float rotateAngle = 0;

float offsetX = 0;

float directionMultiplier = 1;

bool enterWasPressed = false;

while (window.isOpen())

{

Event event;

while (window.pollEvent(event))

{

if (Keyboard::isKeyPressed(Keyboard::Enter) && enterWasPressed == false)

{

enterWasPressed = true;

center.setFillColor(Color(255, 0, 0));

}

if (event.type == Event::Closed)

window.close();

}

if (enterWasPressed == true)

{

sector[0].position = Vector2f(offsetX + centerX, centerY);

center.setPosition(offsetX - CENTER\_RADIUS + centerX, centerY - CENTER\_RADIUS);

angle = startAngle - rotateAngle;

for (unsigned short i = 1; i < FACES + 2; ++i)

{

sector[i].position = Vector2f(offsetX + centerX + radius \* cos(angle), centerY - radius \* sin(angle));

angle += angularLength / FACES;

}

}

window.clear(Color(255, 255, 255));

if (enterWasPressed == false)

{

window.draw(arc);

}

else

{

window.draw(sector);

rotateAngle -= M\_PI / ROTATE\_SPEED;

offsetX += directionMultiplier \* MOVE\_SPEED;

if (offsetX >= WINDOW\_WIDTH - centerX - radius)

{

directionMultiplier = -1;

}

else if (offsetX <= radius - centerX)

{

directionMultiplier = 1;

}

}

window.draw(center);

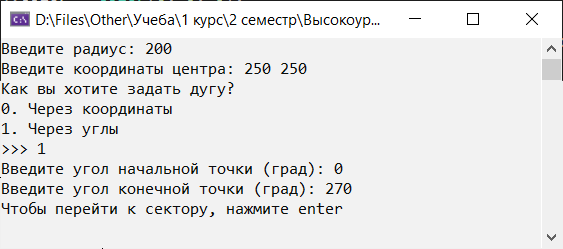
window.display();

}

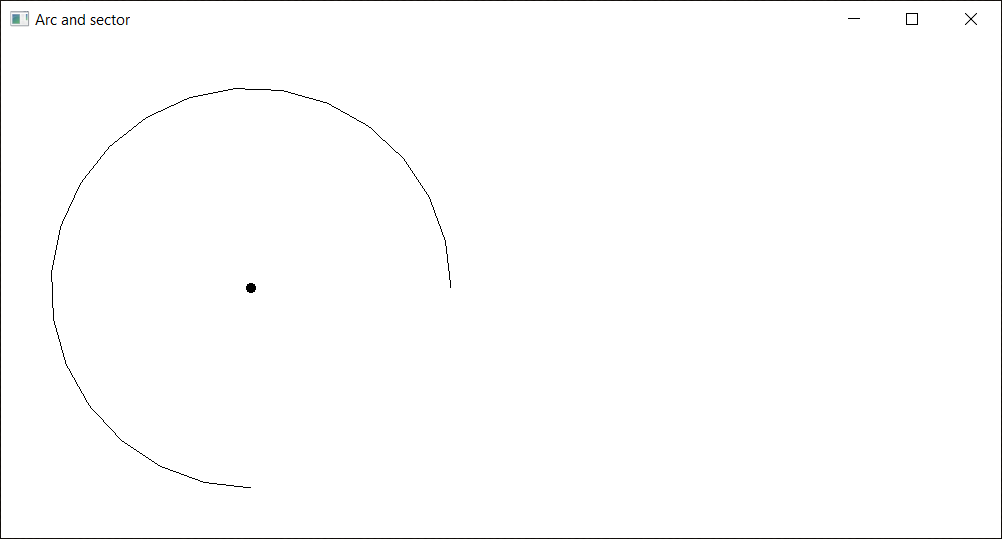
return 0;

}

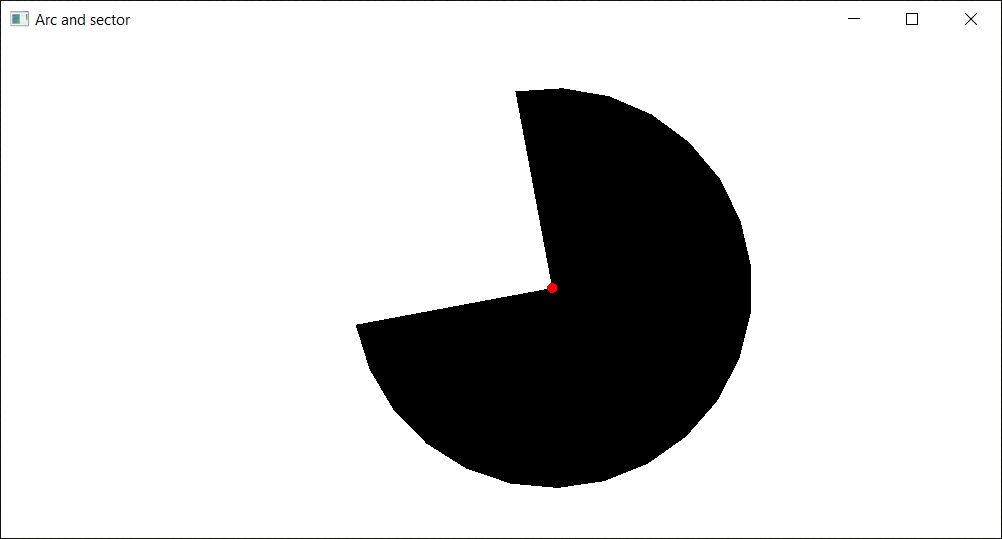
**Демонстрация:**



**Рис. 2.** Консоль



**Рис. 3.** Окно, дуга



**Рис. 4.** Окно, сектор

**Вывод:** в ходе работы были получены практические навыки использования библиотеки SFML, создания окна, ломанных линий, выпуклой фигуры, окружности, их отображения на экране и анимирования, отслеживания нажатий клавиш.