МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОК ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ  
ВЫСШЕГО ОБРАЗОВАНИЯ

«НОВОСИБИРСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

Кафедра Автоматики

**

Лабораторная работа № 6  
по дисциплине «Компьютерная графика»

**«Работа с графикой на Си++»**

Факультет: АВТФ. Преподаватель:

Группа: АА-96. Веретельникова Е.Л.

Студент: Вдовин В.А.

Дата выполнения: 02.11.2020

Отметка о защите:

Новосибирск 2020

# Цель работы

Освоить необходимые средства для написания программ на языке С++, работающих в графическом режиме.

# Задание 1. Российский флаг

# Условие задачи

Написать программу, которая рисует на экране Государственный флаг России.

# Текст программы

#include <QApplication>

#include <QPushButton>

#include <QGraphicsView>

#include <QGraphicsScene>

#include <QGraphicsItem>

#include <QLayout>

#include <QBrush>

#include <QPen>

#include <QFont>

int main(int argc, char \*argv[])

{

QApplication a(*argc*, argv);

// Создаём объекты

QWidget \*window = new QWidget;

QGraphicsScene \*scene = new QGraphicsScene(0, 0, 500, 400);

QGraphicsView \*view = new QGraphicsView(scene);

QVBoxLayout \*layout = new QVBoxLayout;

QFont \*font = new QFont();

QGraphicsTextItem \*label = new QGraphicsTextItem;

// Делаем предварительную настройку

font->setPointSize(24);

label->setPlainText("Россия");

label->setFont(\*font);

// Размещаем надпись внизу по центру

label->setPos((scene->width() - label->*boundingRect*().width()) / 2,

scene->height() - label->*boundingRect*().height());

layout->setMargin(0); // Убираем отступы

layout->addWidget(view);

// Высота графической сцены минус высота надписи

int height = (scene->height() - label->*boundingRect*().height()) / 3;

//Рисуем все объекты в окне

scene->addRect(0, 0, scene->width(), height, QPen(), QBrush(Qt::white));

scene->addRect(0, height, scene->width(), height, QPen(), QBrush(Qt::blue));

scene->addRect(0, height \* 2, scene->width(), height, QPen(), QBrush(Qt::red));

scene->addItem(label);

window->setLayout(layout);

window->show();

return a.exec();

}

# Тест - результат работы программы

# 

* 1. Внесенные изменения в программу (другой флаг по вариантам - Франция)

#include <QApplication>

#include <QPushButton>

#include <QGraphicsView>

#include <QGraphicsScene>

#include <QGraphicsItem>

#include <QLayout>

#include <QBrush>

#include <QPen>

#include <QFont>

int main(int argc, char \*argv[])

{

QApplication a(*argc*, argv);

// Создаём объекты

QWidget \*window = new QWidget;

QGraphicsScene \*scene = new QGraphicsScene(0, 0, 500, 400);

QGraphicsView \*view = new QGraphicsView(scene);

QVBoxLayout \*layout = new QVBoxLayout;

QFont \*font = new QFont();

QGraphicsTextItem \*label = new QGraphicsTextItem;

// Делаем предварительную настройку

font->setPointSize(24);

label->setPlainText("Франция");

label->setFont(\*font);

// Размещаем надпись внизу по центру

label->setPos((scene->width() - label->*boundingRect*().width()) / 2,

scene->height() - label->*boundingRect*().height());

layout->setMargin(0); // Убираем отступы

layout->addWidget(view);

// Высота графической сцены минус высота надписи

int height = scene->height() - label->*boundingRect*().height();

//Рисуем все объекты в окне

scene->addRect(0, 0, scene->width() / 3, height, QPen(), QBrush(Qt::darkBlue));

scene->addRect(scene->width() / 3, 0, scene->width() / 3, height, QPen(), QBrush(Qt::white));

scene->addRect(scene->width() / 3 \* 2, 0, scene->width() / 3, height, QPen(), QBrush(Qt::darkRed));

scene->addItem(label);

window->setLayout(layout);

window->show();

return a.exec();

}

# Тест - результат работы программы



# Задание 2. Анимация

# Условие задачи

Написать программу, которая рисует на экране анимированный смайлик.

# Текст программы

#include <QApplication>

#include <QPushButton>

#include <QGraphicsView>

#include <QGraphicsScene>

#include <QGraphicsItem>

#include <QLayout>

#include <QBrush>

#include <QPen>

#include <QFont>

#include <QThread>

#include <QTimer>

#include <QObject>

// Создаём объекты

QWidget \*window;

QGraphicsScene \*scene;

QGraphicsView \*view;

QVBoxLayout \*layout;

QPen \*pen;

QPen \*pen2;

QTimer \*timer;

int size, smileEyes,

smileX, smileY,

smileEyeX, smileEyeY,

smileLeftEyeCenter, smileRightEyeCenter,

smileMouthWidth, smileMouthHeigth;

void **nextFrame**() {

static int frame = 0;

scene->clear();

scene->addEllipse(smileX, smileY, size, size, QPen(Qt::yellow), QBrush(Qt::yellow));

switch(frame) {

case 0:

//Add happy mouth

scene->addEllipse(smileX + (size - smileMouthWidth) / 2, smileY + size / 32 \* 19, smileMouthWidth, smileMouthHeigth, \*pen2);

scene->addRect(smileX + (size - smileMouthWidth - pen2->width()) / 2, smileY + size / 32 \* 19 - pen2->width(),

smileMouthWidth + pen2->width() \* 2, smileMouthHeigth / 2 + pen2->width(), QPen(Qt::yellow), QBrush(Qt::yellow));

//Add circle eyes

scene->addEllipse(smileLeftEyeCenter - smileEyeX / 2, smileY + (size - smileEyeY) / 3, smileEyeX, smileEyeY, \*pen);

scene->addEllipse(smileRightEyeCenter - smileEyeX / 2, smileY + (size - smileEyeY) / 3, smileEyeX, smileEyeY, \*pen);

break;

case 1:

//Add happy mouth

scene->addEllipse(smileX + (size - smileMouthWidth) / 2, smileY + size / 32 \* 19, smileMouthWidth, smileMouthHeigth, \*pen2);

scene->addRect(smileX + (size - smileMouthWidth - pen2->width()) / 2, smileY + size / 32 \* 19 - pen2->width(),

smileMouthWidth + pen2->width() \* 2, smileMouthHeigth / 2 + pen2->width(), QPen(Qt::yellow), QBrush(Qt::yellow));

//Add left rect eye and right circle eye

scene->addRect(smileLeftEyeCenter - smileEyeX / 2, smileY + size / 3, smileEyeX, pen->width(), \*pen);

scene->addEllipse(smileRightEyeCenter - smileEyeX / 2, smileY + (size - smileEyeY) / 3, smileEyeX, smileEyeY, \*pen);

break;

case 2:

//Add happy mouth

scene->addEllipse(smileX + (size - smileMouthWidth) / 2, smileY + size / 32 \* 19, smileMouthWidth, smileMouthHeigth, \*pen2);

scene->addRect(smileX + (size - smileMouthWidth - pen2->width()) / 2, smileY + size / 32 \* 19 - pen2->width(),

smileMouthWidth + pen2->width() \* 2, smileMouthHeigth / 2 + pen2->width(), QPen(Qt::yellow), QBrush(Qt::yellow));

//Add left circle eye and right rect eye

scene->addEllipse(smileLeftEyeCenter - smileEyeX / 2, smileY + (size - smileEyeY) / 3, smileEyeX, smileEyeY, \*pen);

scene->addRect(smileRightEyeCenter - smileEyeX / 2, smileY + size / 3, smileEyeX, pen->width(), \*pen);

break;

case 3:

//Add happy mouth

scene->addEllipse(smileX + (size - smileMouthWidth) / 2, smileY + size / 32 \* 19, smileMouthWidth, smileMouthHeigth, \*pen2);

scene->addRect(smileX + (size - smileMouthWidth - pen2->width()) / 2, smileY + size / 32 \* 19 - pen2->width(),

smileMouthWidth + pen2->width() \* 2, smileMouthHeigth / 2 + pen2->width(), QPen(Qt::yellow), QBrush(Qt::yellow));

//Add rect eyes

scene->addRect(smileLeftEyeCenter - smileEyeX / 2, smileY + size / 3, smileEyeX, pen->width(), \*pen);

scene->addRect(smileRightEyeCenter - smileEyeX / 2, smileY + size / 3, smileEyeX, pen->width(), \*pen);

break;

case 4:

//Add circle mouth

scene->addEllipse(smileX + (size - smileMouthWidth) / 2, smileY + size / 8 \* 5, smileMouthWidth, smileMouthHeigth, \*pen2);

//Add rect eyes

scene->addRect(smileLeftEyeCenter - smileEyeX / 2, smileY + size / 3, smileEyeX, pen->width(), \*pen);

scene->addRect(smileRightEyeCenter - smileEyeX / 2, smileY + size / 3, smileEyeX, pen->width(), \*pen);

break;

case 5:

//Add circle mouth

scene->addEllipse(smileX + (size - smileMouthWidth) / 2, smileY + size / 8 \* 5, smileMouthWidth, smileMouthHeigth, \*pen2);

//Add circle eyes

scene->addEllipse(smileLeftEyeCenter - smileEyeX / 2, smileY + (size - smileEyeY) / 3, smileEyeX, smileEyeY, \*pen);

scene->addEllipse(smileRightEyeCenter - smileEyeX / 2, smileY + (size - smileEyeY) / 3, smileEyeX, smileEyeY, \*pen);

break;

}

scene->addText("Frame: " + QString::number(frame));

if (frame >= 5) frame = 0;

else frame++;

scene->update(0, 0, view->width(), view->height());

}

int main(int argc, char \*argv[])

{

QApplication a(*argc*, argv);

// Создаём объекты

window = new QWidget;

scene = new QGraphicsScene(0, 0, 700, 800);

view = new QGraphicsView(scene);

layout = new QVBoxLayout;

pen = new QPen();

pen2 = new QPen();

timer = new QTimer();

layout->setMargin(0); // Убираем отступы

layout->addWidget(view);

window->setLayout(layout);

timer->setInterval(3000);

// Подключаем таймер к функции nextFrame()

QObject::connect(timer, &QTimer::timeout, nextFrame);

if (scene->width() > scene->height())

size = scene->height();

else

size = scene->width();

pen->setColor(Qt::darkBlue);

pen->setWidth(size / 40);

pen2->setColor(Qt::darkRed);

pen2->setWidth(size / 40);

smileX = (scene->width() - size) / 2,

smileY = (scene->height() - size) / 2,

smileEyeX = size / 4,

smileEyeY = size / 4,

smileEyes = size / 8,

smileLeftEyeCenter = smileX + size / 2 - smileEyeX / 2 - smileEyes / 2,

smileRightEyeCenter = smileX + size / 2 + smileEyeX / 2 + smileEyes / 2,

smileMouthWidth = size / 2,

smileMouthHeigth = size / 4;

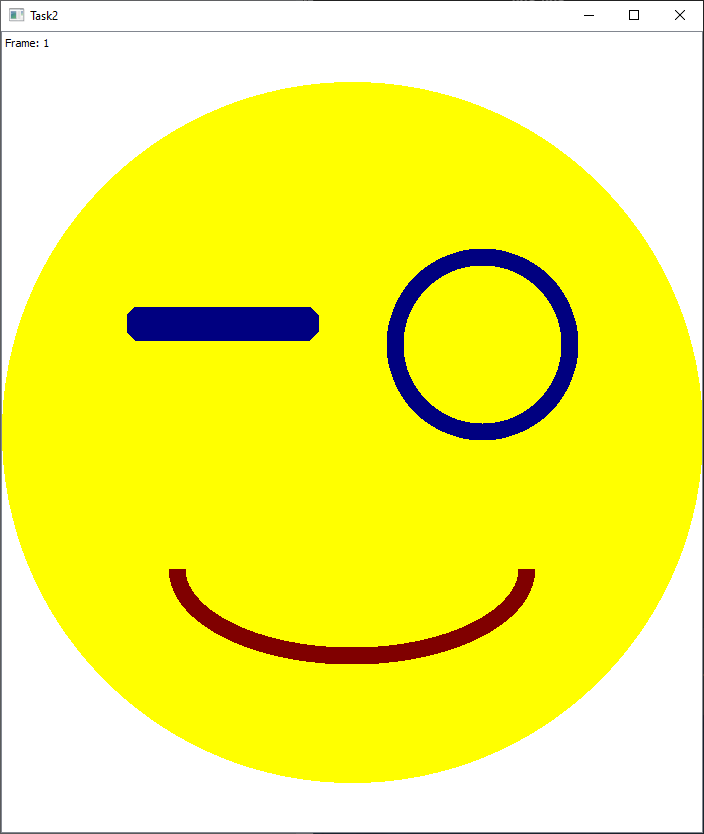
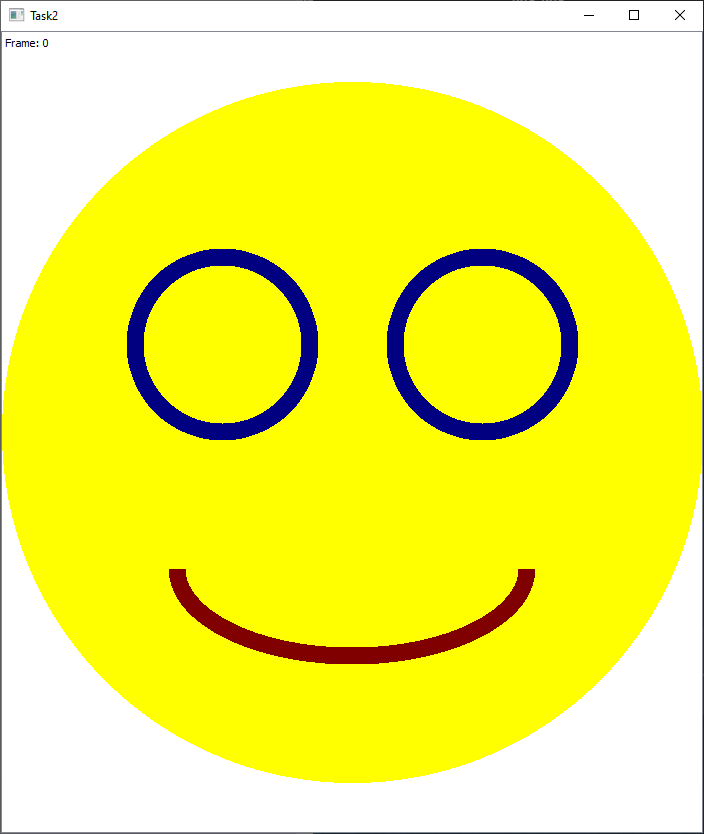
window->show();

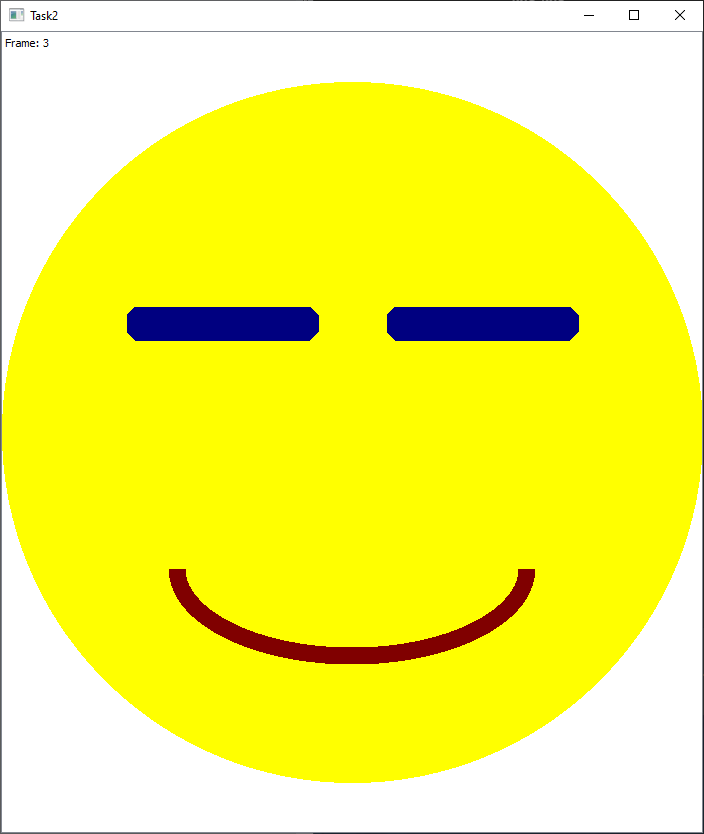
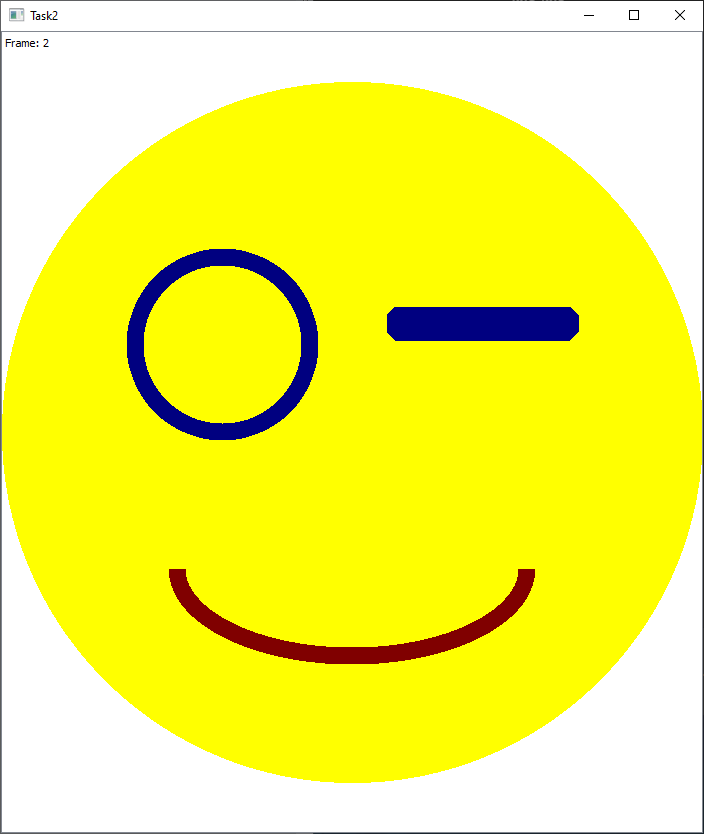
timer->start();

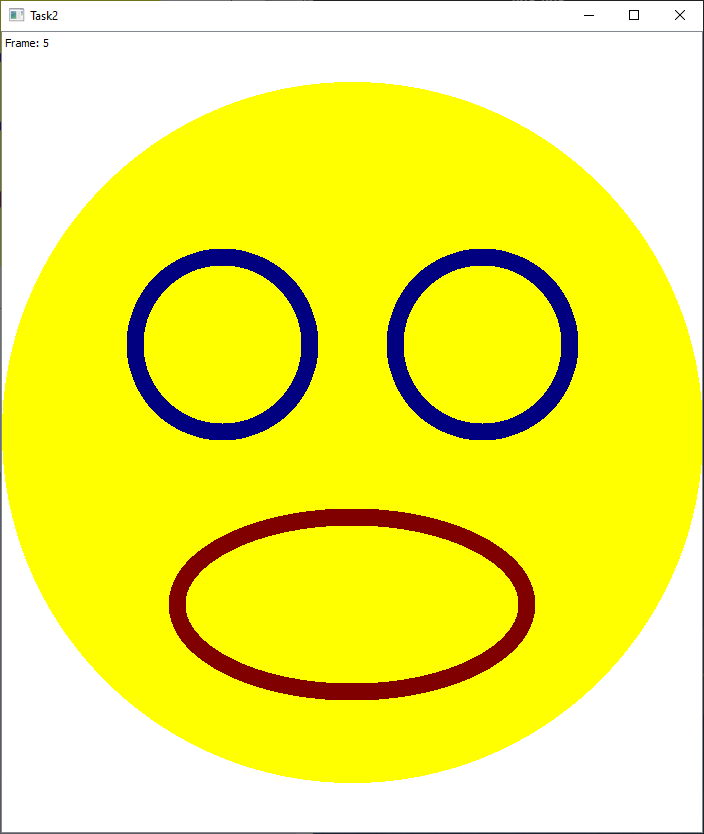
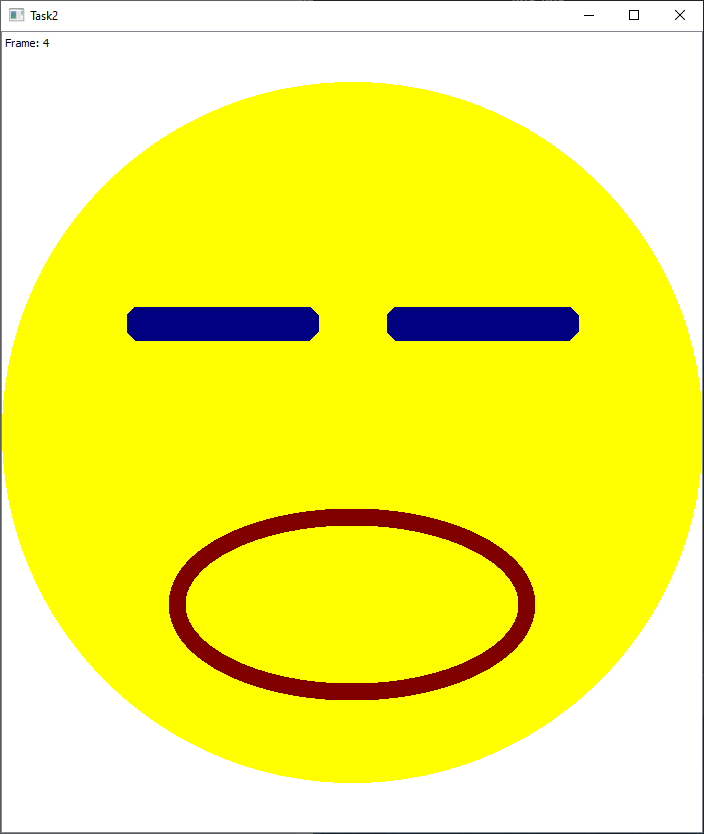
return a.exec();

}

# 3.3. Тест - результат работы программы







**Общие выводы по работе:** работа интересная, потому что у меня уже есть опыт использования графики в приложениях, написанных на Си++. В связи с устареванием библиотеки graphics.h для рисования был использован открытый фреймворк Qt, что не помешало выполнению лабораторной работы.