**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИИ**  
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ  
ВЫСШЕГО ОБРАЗОВАНИЯ  
**«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ**  
**ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ им. В.Г.ШУХОВА»**  
**(БГТУ им. В.Г.Шухова)**

Кафедра программного обеспечения вычислительной техники и автоматизированных систем

Лабораторная работа №1

дисциплина “Компьютерная графика”

по теме “Графические примитивы”

Выполнил: ст. группы ПВ-31  
Ковалев Павел Александрович

Проверил: Осипов О.В.

**Белгород 2020**

**Вариант 8**

**Цель работы:** Изучить графические 2d-примитивов с использованием GDI в среде Qt.

Заданние: *a –* сторона треугольника, *n –* количество секторов в окружности. *a* и *n* вводятся с клавиатуры. Реализовать вращение окружности по часовой стрелке. Раскрасить каждый элемент сектора своим цветом.

*a*

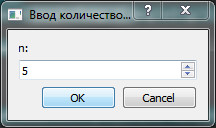
*a*

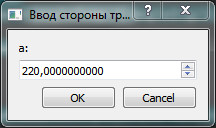
*a*

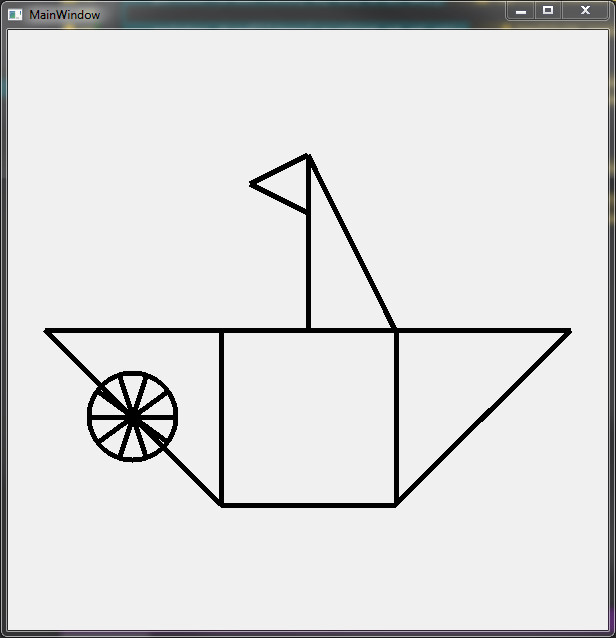
Ход работы

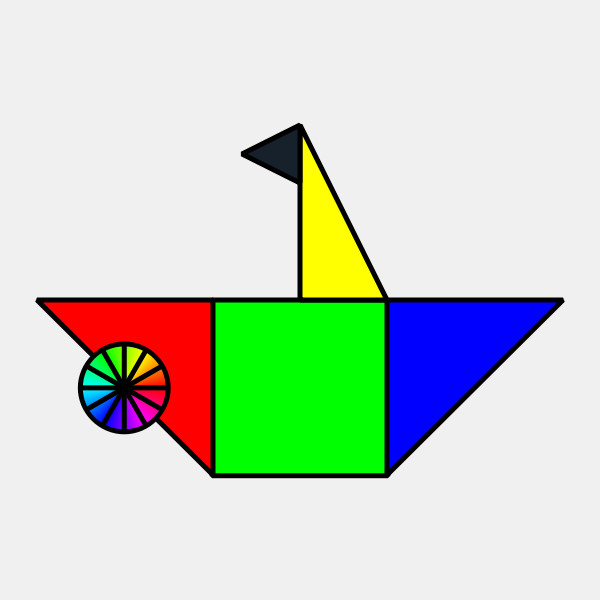
Используемые формулы в построении фигуры.

Не использовал.









Приложения

Содержимое файла mainwindow.h

#ifndef MAINWINDOW\_H

#define MAINWINDOW\_H

#include <QMainWindow>

#include <QDesktopWidget>

#include <math.h>

#include <QPainter>

#include <QWheelEvent>

#include <QInputDialog>

#include <QLabel>

#include <QWidget>

#include <QTimer>

*namespace* **Ui** {

*class* **MainWindow**;

}

*class* **MainWindow** : *public* QMainWindow

{

Q\_OBJECT

*public*:

double a;

int n;

*explicit* **MainWindow**(QWidget \*parent = 0);

~***MainWindow***();

void **ReadDate**();

*private*:

Ui::MainWindow \*ui;

int angle=0;

void ***paintEvent***(QPaintEvent\* event);

void ***keyPressEvent***(QKeyEvent\*);

};

#endif *//* *MAINWINDOW\_H*

Содержимое файла mainwindow.cpp

#include "mainwindow.h"

#include "ui\_mainwindow.h"

#include <iostream>

#include "mainwindow.h"

#include "ui\_mainwindow.h"

#include <iostream>

MainWindow::**MainWindow**(QWidget \*parent) :

QMainWindow(parent),

ui(*new* Ui::MainWindow)

{

ui->setupUi(*this*);

}

MainWindow::~***MainWindow***()

{

*delete* ui;

}

void MainWindow::**ReadDate**()

{

bool Ok;

n = QInputDialog::getInt(0, tr("Ввод количество секторов в окружности:"),tr("n:"), 1, 1, 12, 1, &Ok);

*if* (!Ok) {

}

bool Ok1;

a = QInputDialog::getDouble(0, tr("Ввод стороны треугольника:"),tr("a:"), 100, 100, 300, 10, &Ok1);

*if* (!Ok1) {

}

}

int **max\_RGB**(int r, int g, int b){

int max;

*if*(r>g){

max=r;

} *else* {

max=g;

}

*if*(max<b){

max=b;

}

*return* max;

}

int **min\_RGB**(int r, int g, int b){

int min;

*if*(r<g){

min = r;

} *else* {

min = g;

}

*if*(min>b){

min = b;

}

*return* min;

}

qreal **calcH**(int r, int g, int b, int min, int max){

qreal temp = 0;

*if*(r == max){

*if*(g>=b){

temp = 60\*((g-b)/(max-min))+0;

} *else* {

temp = 60\*((g-b)/(max-min))+360;

}

}

*if*(g == max){

temp = 60\*((b-r)/(max-min))+120;

}

*if*(b == max){

temp = 60\*((r-g)/(max-min))+240;

}

*return* temp;

}

qreal **calcS**(int min, int max){

*if*(max==0){

*return* 0;

} *else* {

*return* (1-min/max);

}

}

*inline* qreal **calcV**(int max){

*return* max;

}

void MainWindow::***paintEvent***(QPaintEvent\* event)

{

QPainter painter(*this*); *//* *Создаём* *объект-живописец*

painter.translate(width() / 2, height() / 2);

*if* (width() < 50 || height() < 50)

*return*;

*if* (width() > height()) *//* *Если* *ширина* *окна* *больше* *высоты*

a = (height() - sqrt(2)\*a) / 2;

*else* a = (width() - sqrt(2)\*a) / 2;

qreal a2=a/2;

qreal a3=a/3;

qreal a4=a/4;

qreal a6=a/6;

painter.setRenderHint(QPainter::*Antialiasing*);

QRectF rect = QRectF(-a2, 0, a, a);

painter.setPen(QPen(Qt::*black*, 5, Qt::*SolidLine*, Qt::*FlatCap*));

painter.setBrush(Qt::*green*);

painter.drawRect(rect);

*//Левый* *треугольник*

QPolygon Triangle;

Triangle << QPoint(-a2,0) << QPoint(-a-a2,0) << QPoint(-a2,a);

painter.setBrush(Qt::*red*);

painter.drawPolygon(Triangle);

*//Правый* *треугольник*

QPolygon Triangle2;

Triangle2 << QPoint(a2,0) << QPoint(a+a2,0) << QPoint(a2,a);

painter.setBrush(Qt::*blue*);

painter.drawPolygon(Triangle2);

*//Вверхний* *треугольник(Правый)*

QPolygon Triangle3;

Triangle3 << QPoint(0,0) << QPoint(0,-a) << QPoint(a2,0);

painter.setBrush(Qt::*yellow*);

painter.drawPolygon(Triangle3);

*//Вверхний* *треугольник(Левый)*

QPolygon Triangle4;

Triangle4 << QPoint(0,-a+a3) << QPoint(0,-a) << QPoint(-a3,-a+a3-a6);

painter.setBrush(QColor(22,33,44));

painter.drawPolygon(Triangle4);

QRectF ellipseRect(-a-a2+a4,a4,a2,a2);

QConicalGradient conicalGradient(-a,a2,0);

conicalGradient.setColorAt(0.0, Qt::*red*);

conicalGradient.setColorAt(60.0/360.0, Qt::*yellow*);

conicalGradient.setColorAt(120.0/360.0, Qt::*green*);

conicalGradient.setColorAt(180.0/360.0, Qt::*cyan*);

conicalGradient.setColorAt(240.0/360.0, Qt::*blue*);

conicalGradient.setColorAt(300.0/360.0, Qt::*magenta*);

conicalGradient.setColorAt(1.0, Qt::*red*);

*/\** *QColor* *colorGradien;*

*int* *r,g,b,min,max;*

*qreal* *tempSector* *=* *0.000;*

*qreal* *dtemp* *=* *360/n;*

*for* *(int* *i* *=* *0;* *i* *<* *n+1;* *i++)* *{*

*r* *=* *rand()* *%* *255;*

*g* *=* *rand()* *%* *255;*

*b* *=* *rand()* *%* *255;*

*min* *=* *min\_RGB(r,g,b);*

*max* *=* *max\_RGB(r,b,g);*

*colorGradien.setHsv(* *(int)* *calcH(r,g,b,min,max),255.0,255.0);*

*if(i* *!=* *n){*

*conicalGradient.setColorAt(tempSector/360.0,colorGradien);*

*}* *else* *{*

*conicalGradient.setColorAt(1.0,colorGradien);*

*}*

*tempSector* *=* *tempSector* *+* *dtemp;*

*}*

*\*/*

QBrush brush(conicalGradient);

painter.setBrush(brush);

painter.drawEllipse(ellipseRect);

QPointF centerCircl;

centerCircl.setX(-a);

centerCircl.setY(a2);

double angleSig = 360/n;

double angleTemp = 0;

QLineF line;

QPointF bpos[n];

*for* (int i = 0; i < n; i++) {

bpos[i].setX(-a+a4);

bpos[i].setY(a2);

line = QLineF(centerCircl,bpos[i]);

line.setAngle(-(angleTemp+angle));

painter.drawLine(line);

line = line.normalVector();

line = line.normalVector();

painter.drawLine(line);

*if*(n % 2 == 0)

angleTemp = angleTemp + angleSig/2;

*else*

angleTemp = angleTemp + angleSig;

}

}

void MainWindow::***keyPressEvent***(QKeyEvent \*event)

{

*if*(event->key() == Qt::*Key\_0*){

angle=angle+4;

}

repaint(); *//* *Обновляем* *окно*

}