**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ**

**Федеральное государственное бюджетное образовательное учреждение высшего**

**профессионального образования**

**«Казанский национальный исследовательский технический университет**

**им. А.Н. Туполева-КАИ»**

Институт компьютерных технологий и защиты информации

Кафедра Прикладной математики и Информатики имени Ю.В. Кожевникова

Лабораторная работа № 3

по дисциплине «Объектно-ориентированное программирование»

**Выполнил**

студент группы 4210

Нигамадянов Фанис Магефурович

Казань

2021

**Лабораторная работа №3. Реализация взаимодействия классов на основе** **агрегации.**

Описание классов:

class circle

{

int lx\_edge = 430;

int ly\_edge = 38;

int rx\_edge = 1100;

int ry\_edge = 750;

Random rnd = new Random();

Point circle\_center = new Point(0, 0);

public int R { get; set; }

public bool circle\_is\_visible { get; set; }

public circle()

{}

public circle( int x1, int y1, int R, bool circle\_is\_visible)

{}

public void show(Graphics gc, Color color)

{}

public void Move(int x, int y)

{}

}

class frame

{

int lx\_edge = 430;

int ly\_edge = 38;

int rx\_edge = 1100;

int ry\_edge = 750;

rectangle internal\_rect = new rectangle();

rectangle external\_rect = new rectangle();

Point center\_of\_frame = new Point(0,0);

Random rnd = new Random();

public bool frame\_is\_visible { get; set; }

private int out\_more\_than = 10;

public int OUT\_MORE\_THAN

{

get

{

return out\_more\_than;

}

set

{

out\_more\_than = value;

}

}

public frame()

{}

public frame(int x, int y, int a, int b)

{}

public void show(Graphics gc, Color color)

{}

public void Move(int x, int y)

{}

}

class Point

{

public int x { get; set; }

public int y { get; set; }

public Point(int x, int y)

{

this.x = x;

this.y = y;

}

}

class rectangle

{

int lx\_edge = 430;

int ly\_edge = 38;

int rx\_edge = 1100;

int ry\_edge = 750;

Random rnd = new Random();

Point rect\_center = new Point(0, 0);

public int a { get; set; }

public int b { get; set; }

public bool rectangle\_is\_visible { get; set; }

public rectangle()

{}

public rectangle(int x1, int y1, int A, int B, bool rectangle\_is\_visible)

{}

public void show(Graphics gc, Color color)

{}

public void Move(int x, int y)

{}

}

class ring

{

int lx\_edge = 430;

int ly\_edge = 38;

int rx\_edge = 1100;

int ry\_edge = 750;

Random rnd = new Random();

circle internal\_circle = new circle();

circle external\_circle = new circle();

Point center\_of\_ring = new Point(0,0);

public bool ring\_is\_visible { get; set; }

public ring()

{

}

public ring(int x, int y, int R, int a)

{

}

public void show(Graphics gc, Color color)

{

}

public void Move(int x, int y)

{

}

}

class square

{

int lx\_edge = 430;

int ly\_edge = 38;

int rx\_edge = 1100;

int ry\_edge = 750;

Random rnd = new Random();

Point square\_center = new Point(0, 0);

public int x1 { get; set; }

public int y1 { get; set; }

public int x2 { get; set; }

public int y2 { get; set; }

public int a { get; set; }

public bool square\_is\_visible { get; set; }

public square()

{

}

public square(int x1, int y1, int A, bool square\_is\_visible)

{

}

public void show(Graphics gc, Color color)

{

}

public void to\_double()

{

}

public void Move(int x, int y)

}

Описание свойств, полей и методов:

Для круга:

1)Ограничение поля случайности

int lx\_edge = 430;

int ly\_edge = 38;

int rx\_edge = 1100;

int ry\_edge = 750;

2) Объект класса Random

Random rnd = new Random();

3) Объект класса Point

Point circle\_center = new Point(0, 0);

4) Радиус окружности

public int R { get; set; }

5) Поля для отслеживания видимости

public bool circle\_is\_visible { get; set; }

6) Конструктор по умолчанию

public circle()

{

circle\_center.x = rnd.Next(lx\_edge, rx\_edge - 100);

circle\_center.y = rnd.Next(ly\_edge, ry\_edge + 100);

this.R = rnd.Next(10, 200);

this.circle\_is\_visible = true;

}

7) Конструктор с заданными параметрами

public circle( int x1, int y1, int R, bool circle\_is\_visible)

{

circle\_center.x = x1;

circle\_center.y = y1;

this.R = R;

this.circle\_is\_visible = circle\_is\_visible;

}

8) Показ фигуры

public void show(Graphics gc, Color color)

{

if (circle\_is\_visible == true)

{

Pen pen = new Pen(color, 5);

gc.DrawEllipse(pen, circle\_center.x-R, circle\_center.y-R, 2\*R, 2\*R);

}

}

9) Перемещение объекта

public void Move(int x, int y)

{

moving\_to moving = moving\_to.Right;

if (x > 0 && y == 0)

{

moving = moving\_to.Right;

}

else if (x < 0 && y == 0)

{

moving = moving\_to.Left;

}

else if (x == 0 && y > 0)

{

moving = moving\_to.Down;

}

else if (x == 0 && y < 0)

{

moving = moving\_to.Up;

}

switch (moving)

{

case moving\_to.Right:

circle\_center.x += x;

break;

case moving\_to.Left:

circle\_center.x += x;

break;

case moving\_to.Up:

circle\_center.y += y;

break;

case moving\_to.Down:

circle\_center.y += y;

break;

}

}

Для рамки:

1. Создаем два объекта класса rectangle

rectangle internal\_rect = new rectangle();

rectangle external\_rect = new rectangle();

1. Определяем расстояние между прямоугольниками

private int out\_more\_than = 10;

public int OUT\_MORE\_THAN

{

get

{

return out\_more\_than;

}

set

{

out\_more\_than = value;

}

}

1. Конструктор по умолчанию

public frame()

{

center\_of\_frame.x = rnd.Next(lx\_edge, rx\_edge);

center\_of\_frame.y = rnd.Next(ly\_edge, ry\_edge);

internal\_rect.a = rnd.Next(16, 400);

internal\_rect.b = rnd.Next(16, 200);

frame\_is\_visible = true;

}

1. Конструктор с заданными параметрами

public frame(int x, int y, int a, int b)

{

center\_of\_frame.x = x;

center\_of\_frame.y = y;

internal\_rect.a = a;

internal\_rect.b = b;

frame\_is\_visible = true;

}

1. Показ фигур

public void show(Graphics gc, Color color)

{

if (frame\_is\_visible == true)

{

Pen pen = new Pen(color, 5);

gc.DrawRectangle(pen, center\_of\_frame.x - internal\_rect.a / 2, center\_of\_frame.y - internal\_rect.b / 2, internal\_rect.a,

internal\_rect.b);

gc.DrawRectangle(pen, center\_of\_frame.x - internal\_rect.a / 2 + OUT\_MORE\_THAN, center\_of\_frame.y - internal\_rect.b / 2 + OUT\_MORE\_THAN,

internal\_rect.a - 2 \* OUT\_MORE\_THAN, internal\_rect.b - 2 \* OUT\_MORE\_THAN);

}

}

Для класса-точки:

1. Поля координаты:

public int x { get; set; }

public int y { get; set; }

1. Конструктор создания точки:

public Point(int x, int y)

{

this.x = x;

this.y = y;

}

Для класса кольца:

1. Два объекта класса circle

circle internal\_circle = new circle();

circle external\_circle = new circle();

Point center\_of\_ring = new Point(0,0);

1. Конструктор для создания кольца

public ring()

{

center\_of\_ring.x = rnd.Next(lx\_edge, rx\_edge);

center\_of\_ring.y = rnd.Next(ly\_edge, ry\_edge);

external\_circle.R = rnd.Next(50, 200);

internal\_circle.R = rnd.Next(10, 49);

ring\_is\_visible = true;

}

1. Конструктор для создания кольца по заданным параметрам

public ring(int x, int y, int R, int a)

{

center\_of\_ring.x = x;

center\_of\_ring.y = y;

external\_circle.R = R;

internal\_circle.R = R - a;

ring\_is\_visible = true;

}

1. Метод для показа фигуры

public void show(Graphics gc, Color color)

{

if (ring\_is\_visible == true)

{

Pen pen = new Pen(color, 5);

gc.DrawEllipse(pen, center\_of\_ring.x - (internal\_circle.R), center\_of\_ring.y - (internal\_circle.R), 2 \* (internal\_circle.R), 2 \* (internal\_circle.R));

gc.DrawEllipse(pen, center\_of\_ring.x - external\_circle.R, center\_of\_ring.y - external\_circle.R, 2 \* external\_circle.R, 2 \* external\_circle.R);

}

}

1. Метод для перемещения кольца

public void Move(int x, int y)

{

moving\_to moving = moving\_to.Right;

if (x > 0 && y == 0)

{

moving = moving\_to.Right;

}

else if (x < 0 && y == 0)

{

moving = moving\_to.Left;

}

else if (x == 0 && y > 0)

{

moving = moving\_to.Down;

}

else if (x == 0 && y < 0)

{

moving = moving\_to.Up;

}

switch (moving)

{

case moving\_to.Right:

center\_of\_ring.x += x;

break;

case moving\_to.Left:

center\_of\_ring.x += x;

break;

case moving\_to.Up:

center\_of\_ring.y += y;

break;

case moving\_to.Down:

center\_of\_ring.y += y;

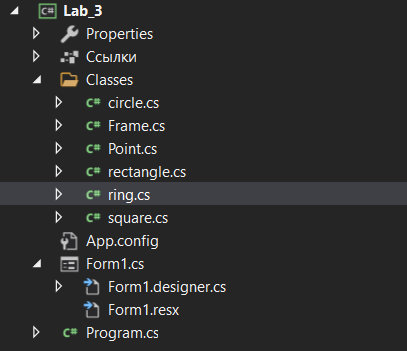
break;

}

}

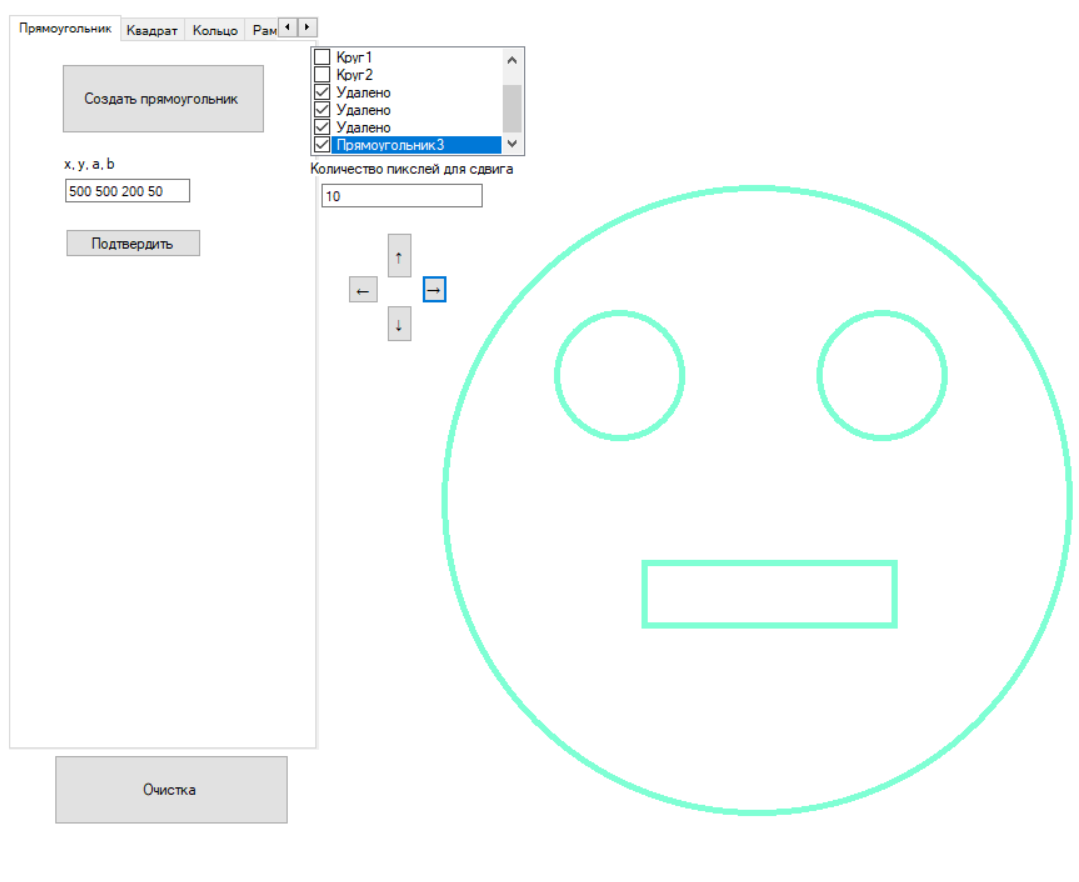
}

Структура программы:



1. Папка классов, где находятся 4 фигуры и один класс, определяющий точку-центр
2. Файл Form1.cs, содержащий файл Form1.designer.cs, определяющий кнопки и их обработчики, файл Form1.res.
3. Файл Program.cs, запускающий форму

Интерфейс программы:

****

Руководство пользования:

Имеются пять вкладок для разных фигур. При помощи кнопки «Создать (соответствующая фигура)» вы можете создать фигуру со случайными координатами и размерами. Если в текстовом поле ввести соответствующие координаты и размеры, можно получить фигуру с заданными координатами. Для перемещения и удаления фигур воспользуйтесь окном CheckBox, выберите интересующую фигуру и нажимайте на соответствующие кнопки (стрелки для перемещения, кнопку очистки и т.д.)

Листинг программы:

public partial class Form1 : Form // пытаюсь все испортить

{

string row\_name; // для имени элемента в CheckBox

int cb\_for\_circ = 0;

int cb\_for\_rect = 0; // индекс для фигур в Checkbox

int cb\_for\_square = 0;

int cb\_for\_ring = 0;

int cb\_for\_frame = 0;

int move\_pixels = 10;

int for\_circ = 0;

int for\_rect = 0; // индексация по массиву

int for\_square = 0;

int for\_ring = 0;

int for\_frame = 0;

string[] circ\_data;

int user\_circ\_x; // координаты, если пользователь сам захочет их ввести

int user\_circ\_y;

int user\_circ\_R;

string[] rect\_data;

int user\_rect\_x;

int user\_rect\_y;

int user\_rect\_a;

int user\_rect\_b;

string[] square\_data;

int user\_square\_x;

int user\_square\_y;

int user\_square\_a;

string[] ring\_data;

int user\_ring\_x;

int user\_ring\_y;

int user\_ring\_R;

int user\_ring\_a;

string[] frame\_data;

int user\_frame\_x;

int user\_frame\_y;

int user\_frame\_a;

int user\_frame\_b;

Graphics gr;

static int num\_of\_circles = 20;

static int num\_of\_rectangles = 20;

static int num\_of\_squares = 20;

static int num\_of\_rings= 20;

static int num\_of\_frames = 20;

circle[] m\_circles = new circle[num\_of\_circles];

rectangle[] m\_rectangles = new rectangle[num\_of\_rectangles];

square[] m\_squares= new square[num\_of\_squares];

ring[] m\_rings = new ring[num\_of\_rings];

frame[] m\_frames = new frame[num\_of\_frames];

public Form1()

{

InitializeComponent();

this.WindowState = FormWindowState.Maximized;

this.BackColor = System.Drawing.Color.White;

}

private void button1\_Click(object sender, EventArgs e)

{

if (textBox2.Text != "")

{

m\_circles[for\_circ] =

new circle(user\_circ\_x, user\_circ\_y, user\_circ\_R, true);

}

else

{

m\_circles[for\_circ] = new circle();

}

m\_circles[for\_circ].show(gr = CreateGraphics(), Color.Aquamarine); //заполнение массива объектов одной группы

for\_circ++;

MessageBox.Show("Круг успешно создан!");

row\_name = "Круг";

checkedListBox1.Items.Add(row\_name + cb\_for\_circ);

row\_name = "";

cb\_for\_circ++;

}// создание круга

private void button2\_Click(object sender, EventArgs e) // создание прямоугольника

{

if (textBox3.Text != "")

{

m\_rectangles[for\_rect] =

new rectangle(user\_rect\_x - user\_rect\_a/2, user\_rect\_y - user\_rect\_b/2,

user\_rect\_a, user\_rect\_b, true);

}

else

{

m\_rectangles[for\_rect] = new rectangle();

}

m\_rectangles[for\_rect].show(gr = CreateGraphics(), Color.Aquamarine); //заполнение массива объектов одной группы

for\_rect++;

MessageBox.Show("Прямоугольник успешно создан!");

row\_name = "Прямоугольник";

checkedListBox1.Items.Add(row\_name + cb\_for\_rect);

row\_name = "";

cb\_for\_rect++;

}

private void button3\_Click(object sender, EventArgs e) // создание квадрата

{

if (textBox4.Text != "")

{

m\_squares[for\_square] =

new square(user\_square\_x, user\_square\_y,

user\_square\_a, true);

}

else

{

m\_squares[for\_square] = new square();

}

m\_squares[for\_square].show(gr = CreateGraphics(), Color.Aquamarine); //заполнение массива объектов одной группы

for\_square++;

MessageBox.Show("Квадрат успешно создан!");

row\_name = "Квадрат";

checkedListBox1.Items.Add(row\_name + cb\_for\_square);

row\_name = "";

cb\_for\_square++;

}

private void button13\_Click(object sender, EventArgs e)

{ if (textBox5.Text != "")

{

m\_rings[for\_ring] = new ring(user\_ring\_x, user\_ring\_y, user\_ring\_R, user\_ring\_a);

}

else

{

m\_rings[for\_ring] = new ring();

}

m\_rings[for\_ring].show(gr = CreateGraphics(), Color.Aquamarine); //заполнение массива объектов одной группы

for\_ring++;

MessageBox.Show("Кольцо успешно создано!");

row\_name = "Кольцо";

checkedListBox1.Items.Add(row\_name + cb\_for\_ring);

row\_name = "";

cb\_for\_ring++;

}// создание кольца

private void button15\_Click(object sender, EventArgs e)// создание рамки

{

if (textBox6.Text != "")

{

m\_frames[for\_frame] = new frame(user\_frame\_x, user\_frame\_y, user\_frame\_a, user\_frame\_b);

}

else

{

m\_frames[for\_frame] = new frame();

}

m\_frames[for\_frame].show(gr = CreateGraphics(), Color.Aquamarine); //заполнение массива объектов одной группы

for\_frame++;

MessageBox.Show("Рама успешно создана!");

row\_name = "Рама";

checkedListBox1.Items.Add(row\_name + cb\_for\_frame);

row\_name = "";

cb\_for\_frame++;

}

private void button4\_Click(object sender, EventArgs e)

{

for (int k = 0; k < checkedListBox1.Items.Count; k++) // проверка, что есть отмеченные, чтобы не стереть все

{

if (checkedListBox1.GetItemChecked(k))

{

gr.Clear(Color.White);

break;

}

}

int i = 0;

int counter = 0;

bool square\_exist = false;

foreach (var item in checkedListBox1.Items.OfType<string>().ToList())

{

if (item.Contains("Квадрат"))

{

square\_exist = true;

if (checkedListBox1.GetItemChecked(i))

{

m\_squares[int.Parse(item.Replace("Квадрат", ""))].to\_double();

counter++;

}

m\_squares[int.Parse(item.Replace("Квадрат", ""))].show(gr, Color.Aquamarine);

}

else if (item.Contains("Круг"))

{

m\_circles[int.Parse(item.Replace("Круг", ""))].show(gr, Color.Aquamarine);

}

else if (item.Contains("Прямоугольник"))

{

m\_rectangles[int.Parse(item.Replace("Прямоугольник", ""))].show(gr, Color.Aquamarine);

}

else if (item.Contains("Кольцо"))

{

m\_rings[int.Parse(item.Replace("Кольцо", ""))].show(gr, Color.Aquamarine);

}

else if (item.Contains("Рама"))

{

m\_frames[int.Parse(item.Replace("Рама", ""))].show(gr, Color.Aquamarine);

}

i++;

}

if (counter == 0)

{

MessageBox.Show("Не выбрано ни одного квадрата");

}

if (!square\_exist)

{

MessageBox.Show("Квадраты не созданы!");

}

} // удвоение длины квадрата

private void button5\_Click(object sender, EventArgs e)

{

for (int k = 0; k < checkedListBox1.Items.Count; k++) // проверка, что есть отмеченные, чтобы не стереть все

{

if (checkedListBox1.GetItemChecked(k))

{

gr.Clear(Color.White);

break;

}

}

int i = 0;

foreach (var item in checkedListBox1.Items.OfType<string>().ToList())

{

if (item.Contains("Круг"))

{

if (checkedListBox1.GetItemChecked(i))

{

m\_circles[int.Parse(item.Replace("Круг", ""))].circle\_is\_visible = false;

checkedListBox1.Items[i] = "Удалено";

}

m\_circles[int.Parse(item.Replace("Круг", ""))].show(gr, Color.Aquamarine);

}

else if (item.Contains("Прямоугольник"))

{

if (checkedListBox1.GetItemChecked(i))

{

m\_rectangles[int.Parse(item.Replace("Прямоугольник", ""))].rectangle\_is\_visible = false;

checkedListBox1.Items[i] = "Удалено";

}

m\_rectangles[int.Parse(item.Replace("Прямоугольник", ""))].show(gr, Color.Aquamarine);

}

else if (item.Contains("Квадрат"))

{

if (checkedListBox1.GetItemChecked(i))

{

m\_squares[int.Parse(item.Replace("Квадрат", ""))].square\_is\_visible = false;

checkedListBox1.Items[i] = "Удалено";

}

m\_squares[int.Parse(item.Replace("Квадрат", ""))].show(gr, Color.Aquamarine);

}

else if (item.Contains("Кольцо"))

{

if (checkedListBox1.GetItemChecked(i))

{

m\_rings[int.Parse(item.Replace("Кольцо", ""))].ring\_is\_visible = false;

checkedListBox1.Items[i] = "Удалено";

}

m\_rings[int.Parse(item.Replace("Кольцо", ""))].show(gr, Color.Aquamarine);

}

else if (item.Contains("Рама"))

{

if (checkedListBox1.GetItemChecked(i))

{

m\_frames[int.Parse(item.Replace("Рама", ""))].frame\_is\_visible= false;

checkedListBox1.Items[i] = "Удалено";

}

m\_frames[int.Parse(item.Replace("Рама", ""))].show(gr, Color.Aquamarine);

}

i++;

}

} // очистка

private void button8\_Click(object sender, EventArgs e)

{

for (int k = 0; k < checkedListBox1.Items.Count; k++) // проверка, что есть отмеченные, чтобы не стереть все

{

if (checkedListBox1.GetItemChecked(k))

{

gr.Clear(Color.White);

break;

}

}

int i = 0;

foreach (var item in checkedListBox1.Items.OfType<string>().ToList())

{

if (item.Contains("Круг"))

{

if (checkedListBox1.GetItemChecked(i))

{

m\_circles[int.Parse(item.Replace("Круг", ""))].Move(move\_pixels, 0);

}

m\_circles[int.Parse(item.Replace("Круг", ""))].show(gr, Color.Aquamarine);

}

else if (item.Contains("Прямоугольник"))

{

if (checkedListBox1.GetItemChecked(i))

{

m\_rectangles[int.Parse(item.Replace("Прямоугольник", ""))].Move(move\_pixels, 0);

}

m\_rectangles[int.Parse(item.Replace("Прямоугольник", ""))].show(gr, Color.Aquamarine);

}

else if (item.Contains("Квадрат"))

{

if (checkedListBox1.GetItemChecked(i))

{

m\_squares[int.Parse(item.Replace("Квадрат", ""))].MoveToRight(move\_pixels);

}

m\_squares[int.Parse(item.Replace("Квадрат", ""))].show(gr, Color.Aquamarine);

}

else if (item.Contains("Кольцо"))

{

if (checkedListBox1.GetItemChecked(i))

{

m\_rings[int.Parse(item.Replace("Кольцо", ""))].Move(move\_pixels, 0);

}

m\_rings[int.Parse(item.Replace("Кольцо", ""))].show(gr, Color.Aquamarine);

}

else if (item.Contains("Рама"))

{

if (checkedListBox1.GetItemChecked(i))

{

m\_frames[int.Parse(item.Replace("Рама", ""))].Move(move\_pixels, 0);

}

m\_frames[int.Parse(item.Replace("Рама", ""))].show(gr, Color.Aquamarine);

}

i++;

}

} // сдвиг вправо

private void button7\_Click(object sender, EventArgs e)

{

for (int k = 0; k < checkedListBox1.Items.Count; k++) // проверка, что есть отмеченные, чтобы не стереть все

{

if (checkedListBox1.GetItemChecked(k))

gr.Clear(Color.White);

}

int i = 0;

foreach (var item in checkedListBox1.Items.OfType<string>().ToList())

{

if (item.Contains("Круг"))

{

if (checkedListBox1.GetItemChecked(i))

{

m\_circles[int.Parse(item.Replace("Круг", ""))].Move(-move\_pixels, 0);

}

m\_circles[int.Parse(item.Replace("Круг", ""))].show(gr, Color.Aquamarine);

}

else if (item.Contains("Прямоугольник"))

{

if (checkedListBox1.GetItemChecked(i))

{

m\_rectangles[int.Parse(item.Replace("Прямоугольник", ""))].Move(- move\_pixels,0);

}

m\_rectangles[int.Parse(item.Replace("Прямоугольник", ""))].show(gr, Color.Aquamarine);

}

else if (item.Contains("Квадрат"))

{

if (checkedListBox1.GetItemChecked(i))

{

m\_squares[int.Parse(item.Replace("Квадрат", ""))].MoveToLeft(move\_pixels);

}

m\_squares[int.Parse(item.Replace("Квадрат", ""))].show(gr, Color.Aquamarine);

}

else if (item.Contains("Кольцо"))

{

if (checkedListBox1.GetItemChecked(i))

{

m\_rings[int.Parse(item.Replace("Кольцо", ""))].Move(-move\_pixels, 0);

}

m\_rings[int.Parse(item.Replace("Кольцо", ""))].show(gr, Color.Aquamarine);

}

else if (item.Contains("Рама"))

{

if (checkedListBox1.GetItemChecked(i))

{

m\_frames[int.Parse(item.Replace("Рама", ""))].Move(-move\_pixels, 0);

}

m\_frames[int.Parse(item.Replace("Рама", ""))].show(gr, Color.Aquamarine);

}

i++;

}

} // сдвиг влево

private void button9\_Click(object sender, EventArgs e)

{

for (int k = 0; k < checkedListBox1.Items.Count; k++) // проверка, что есть отмеченные, чтобы не стереть все

{

if (checkedListBox1.GetItemChecked(k))

gr.Clear(Color.White);

}

int i = 0;

foreach (var item in checkedListBox1.Items.OfType<string>().ToList())

{

if (item.Contains("Круг"))

{

if (checkedListBox1.GetItemChecked(i))

{

m\_circles[int.Parse(item.Replace("Круг", ""))].Move(0, move\_pixels);

}

m\_circles[int.Parse(item.Replace("Круг", ""))].show(gr, Color.Aquamarine);

}

else if (item.Contains("Прямоугольник"))

{

if (checkedListBox1.GetItemChecked(i))

{

m\_rectangles[int.Parse(item.Replace("Прямоугольник", ""))].Move(0, move\_pixels);

}

m\_rectangles[int.Parse(item.Replace("Прямоугольник", ""))].show(gr, Color.Aquamarine);

}

else if (item.Contains("Квадрат"))

{

if (checkedListBox1.GetItemChecked(i))

{

m\_squares[int.Parse(item.Replace("Квадрат", ""))].MoveToDown(move\_pixels);

}

m\_squares[int.Parse(item.Replace("Квадрат", ""))].show(gr, Color.Aquamarine);

}

else if (item.Contains("Кольцо"))

{

if (checkedListBox1.GetItemChecked(i))

{

m\_rings[int.Parse(item.Replace("Кольцо", ""))].Move(0, move\_pixels);

}

m\_rings[int.Parse(item.Replace("Кольцо", ""))].show(gr, Color.Aquamarine);

}

else if (item.Contains("Рама"))

{

if (checkedListBox1.GetItemChecked(i))

{

m\_frames[int.Parse(item.Replace("Рама", ""))].Move(0, move\_pixels);

}

m\_frames[int.Parse(item.Replace("Рама", ""))].show(gr, Color.Aquamarine);

}

i++;

}

} // сдвиг вниз

private void button6\_Click(object sender, EventArgs e)

{

for (int k = 0; k < checkedListBox1.Items.Count; k++) // проверка, что есть отмеченные, чтобы не стереть все

{

if (checkedListBox1.GetItemChecked(k))

gr.Clear(Color.White);

}

int i = 0;

foreach (var item in checkedListBox1.Items.OfType<string>().ToList())

{

if (item.Contains("Круг"))

{

if (checkedListBox1.GetItemChecked(i))

{

m\_circles[int.Parse(item.Replace("Круг", ""))].Move(0, - move\_pixels);

}

m\_circles[int.Parse(item.Replace("Круг", ""))].show(gr, Color.Aquamarine);

}

else if (item.Contains("Прямоугольник"))

{

if (checkedListBox1.GetItemChecked(i))

{

m\_rectangles[int.Parse(item.Replace("Прямоугольник", ""))].Move(0, -move\_pixels);

}

m\_rectangles[int.Parse(item.Replace("Прямоугольник", ""))].show(gr, Color.Aquamarine);

}

else if (item.Contains("Квадрат"))

{

if (checkedListBox1.GetItemChecked(i))

{

m\_squares[int.Parse(item.Replace("Квадрат", ""))].MoveToUp(move\_pixels);

}

m\_squares[int.Parse(item.Replace("Квадрат", ""))].show(gr, Color.Aquamarine);

}

else if (item.Contains("Кольцо"))

{

if (checkedListBox1.GetItemChecked(i))

{

m\_rings[int.Parse(item.Replace("Кольцо", ""))].Move(0, - move\_pixels);

}

m\_rings[int.Parse(item.Replace("Кольцо", ""))].show(gr, Color.Aquamarine);

}

else if (item.Contains("Рама"))

{

if (checkedListBox1.GetItemChecked(i))

{

m\_frames[int.Parse(item.Replace("Рама", ""))].Move(0, - move\_pixels);

}

m\_frames[int.Parse(item.Replace("Рама", ""))].show(gr, Color.Aquamarine);

}

i++;

}

} // сдвиг вверх

private void textBox1\_TextChanged(object sender, EventArgs e) // сдвиг по пикселям

{

if (textBox1.Text == "")

move\_pixels = 10;

else

{

move\_pixels = int.Parse(textBox1.Text);

}

}

private void button10\_Click(object sender, EventArgs e)

{

try

{

circ\_data = textBox2.Text.Split(' ');

user\_circ\_x = int.Parse(circ\_data[0]);

user\_circ\_y = int.Parse(circ\_data[1]);

user\_circ\_R = int.Parse(circ\_data[2]);

}

catch

{

MessageBox.Show("Возникла ошибка");

}

} // задание значений кругу

private void button11\_Click(object sender, EventArgs e) // задание значений для прямо-ка

{

try

{

rect\_data = textBox3.Text.Split(' ');

user\_rect\_x = int.Parse(rect\_data[0]);

user\_rect\_y = int.Parse(rect\_data[1]);

user\_rect\_a = int.Parse(rect\_data[2]);

user\_rect\_b = int.Parse(rect\_data[3]);

}

catch

{

MessageBox.Show("Возникла ошибка");

}

}

private void button12\_Click(object sender, EventArgs e)

{

try

{

square\_data = textBox4.Text.Split(' ');

user\_square\_x = int.Parse(square\_data[0]);

user\_square\_y = int.Parse(square\_data[1]);

user\_square\_a = int.Parse(square\_data[2]);

}

catch

{

MessageBox.Show("Возникла ошибка");

}

} // задание значений квадрата

private void button14\_Click(object sender, EventArgs e)

{

try

{

ring\_data = textBox5.Text.Split(' ');

user\_ring\_x = int.Parse(ring\_data[0]);

user\_ring\_y = int.Parse(ring\_data[1]);

user\_ring\_R = int.Parse(ring\_data[2]);

user\_ring\_a = int.Parse(ring\_data[3]);

if (user\_ring\_a <= 0 | user\_ring\_a >= user\_ring\_R)

{

throw new Exception();

}

}

catch

{

MessageBox.Show("Возникла ошибка");

}

} // задание значений кольца

private void button16\_Click(object sender, EventArgs e)

{

try

{

frame\_data = textBox6.Text.Split(' ');

user\_frame\_x = int.Parse(frame\_data[0]);

user\_frame\_y = int.Parse(frame\_data[1]);

user\_frame\_a = int.Parse(frame\_data[2]);

user\_frame\_b = int.Parse(frame\_data[3]);

//if ()

//{

// throw new Exception();

//}

}

catch

{

MessageBox.Show("Возникла ошибка");

}

}