МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение

высшего образования

УЛЬЯНОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

кафедра «Вычислительная техника».

**Лабораторная работа №2**

по дисциплине: «Автоматизация проектирования технических средств»

Выполнил:

студент 4 курса, гр. ИВТВМбд-41

Захарычев Никита Алексеевич.

Проверил:

кандидат технических наук,

доцент кафедры ВТ

Войт Николай Николаевич.

г. Ульяновск, 2018

**Задание**

Реализовать программу, выполняющую визуализацию логической схемы алгоритма (ЛСА).

Входные данные: ЛСА

Выходные данные: визуальное представление ЛСА.

**Реализация**

Данная работа выполнена на основе кода из предыдущей лабораторной работы.

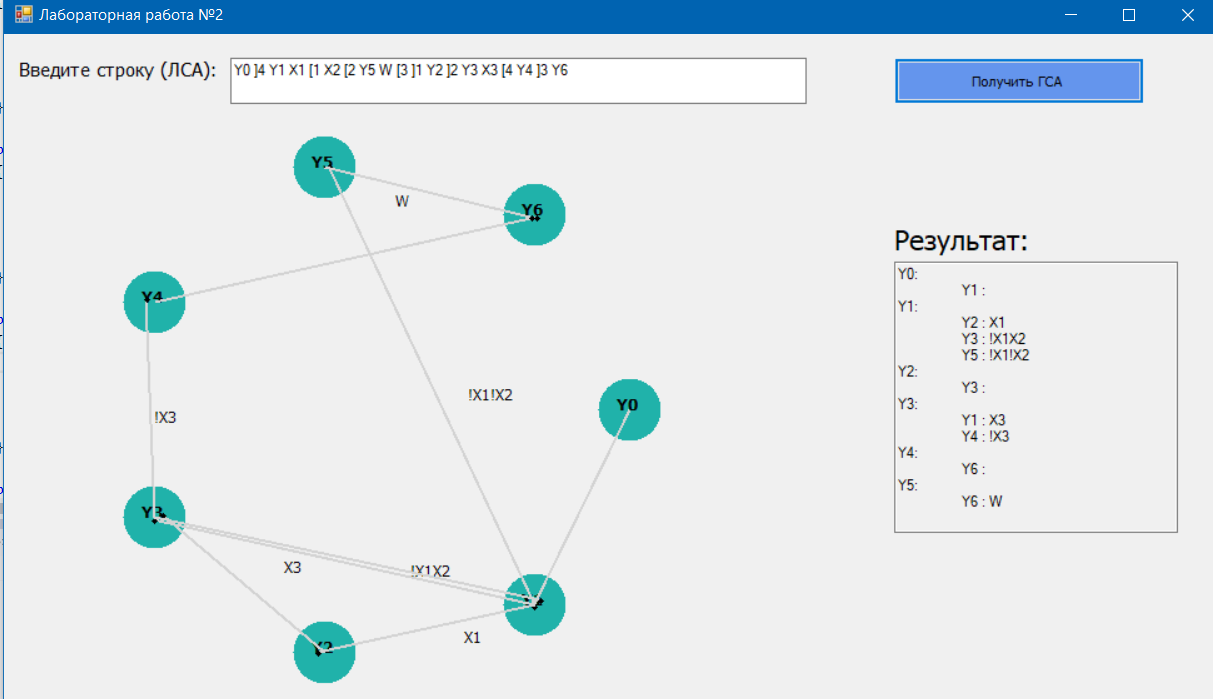
Для визуализации автомата были реализованы следующие функции:

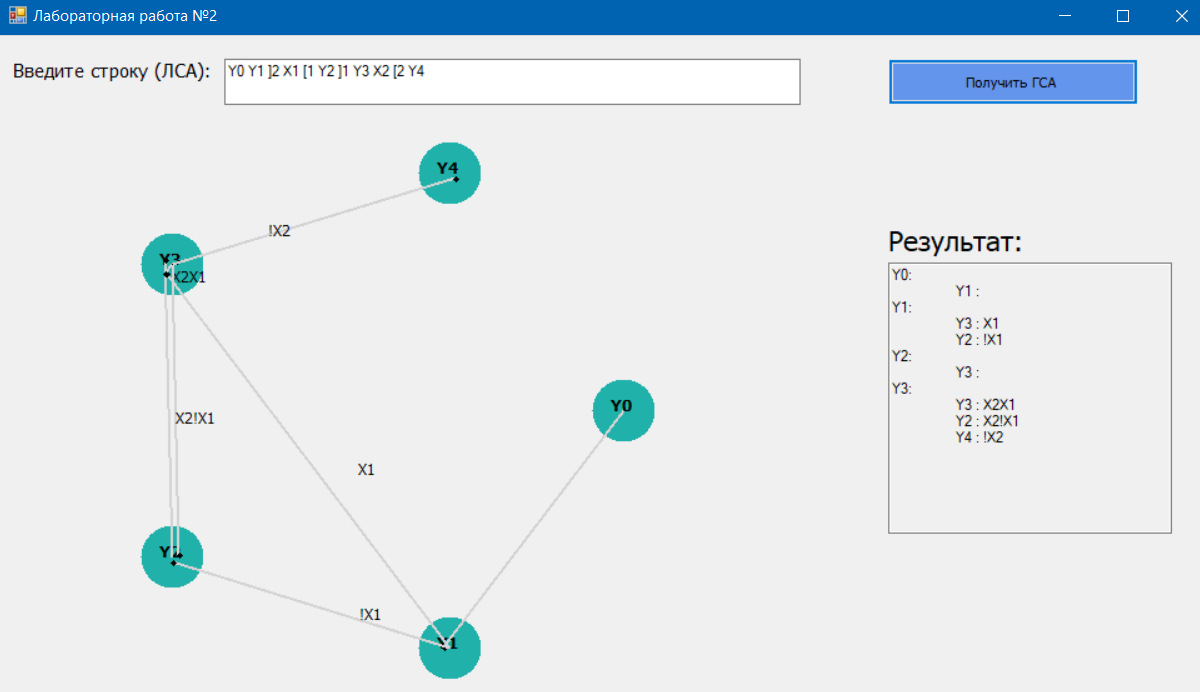
* protected override void OnPaint(PaintEventArgs e) – функция отрисовки состояний и переходов
* void getState() – функция создания состояния
* void getGSA() – функция, реализующая получение ГСА

Также реализовано 2 класса (State и Line), которые представляют из себя создание и отрисовки состояний(кругов) и переходов (линий).

Принцип действия визуализации автомата заключается в том, что в первую очередь, происходит рисование всех вершин графа, которые соответствуют состояниям автомата, затем выполняется обход всех состояний. В матрице переходов определяется наличие перехода из текущего состояния в какое-либо другое, если переход существует, то происходит рисование ребра из вершины соответствующей текущему состоянию в новую вершину.

**Тестирование**





**Вывод:** В результате выполнения данной лабораторной работы были получены навыки в реализации графа визуализации автоматов.

**Исходный код**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using apts\_lab1;

namespace apts\_lab2

{

public partial class Form1 : Form

{

// Список смежности

List<List<apts\_lab1.StateLSA>> graph;

// Список состояний

List<State> states = new List<State>();

// Список линий

List<Line> lines = new List<Line>();

Random rand = new Random();

// Инициализация формы

public Form1()

{

InitializeComponent();

}

//Отрисовка состояний и линий

protected override void OnPaint(PaintEventArgs e)

{

base.OnPaint(e);

Graphics g = e.Graphics;

for (int i = 0; i < states.Count; i++)

states[i].draw(g);

for (int i = 0; i < lines.Count; i++)

lines[i].draw(g, rand);

}

// Получения состояния У

void getState()

{

int R = 200;

int centerX = 300;

int centerY = 300;

double alp = 2 \* Math.PI / states.Count;

for(int i = 0; i < states.Count; i++)

{

states[i].x = centerX + (int)(R \* Math.Cos(i \* alp));

states[i].y = centerY + (int)(R \* Math.Sin(i \* alp));

}

}

// Получение ГСА

void getGSA()

{

//Состояние

for(int i = 0; i < graph.Count; i++)

{

states.Add(new State(0, 0, "Y" + i));

}

getState();

// Линии

for(int i = 0; i < graph.Count; i++)

{

if (graph[i] == null)

continue;

State s1 = states[i];

for(int j = 0; j < graph[i].Count; j++)

{

State s2 = State.getState(states, graph[i][j]);

lines.Add(new Line(s1, s2, graph[i][j].cond\_if));

}

}

}

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

{

states.Clear();

lines.Clear();

string input = textBox1.Text;

Apts\_lab1 lab1 = new Apts\_lab1(input);

lab1.getLSA();

textBox2.Text = lab1.output.Replace("\n", "\r\n");

if (lab1.stackLeft.Count > 0)

{

textBox3.Text = string.Join(" ", lab1.stackLeft);

textBox4.Text = string.Join(" ", lab1.stackRight);

}

else

{

textBox3.Text = lab1.stckLeftStr;

textBox4.Text = lab1.stckRightStr;

}

graph = lab1.graph;

getGSA();

Refresh();

}

}

// Класс для отрисовки состояния

class State

{

Pen p;

SolidBrush fp = new SolidBrush(Color.LightSeaGreen);

SolidBrush p\_str = new SolidBrush(Color.Black);

public int x, y;

public int r = 50;

string text;

Font font = new Font("Tahoma", 10, FontStyle.Bold);

public State(int \_x = 0, int \_y = 0, string \_text = "Y")

{

x = \_x;

y = \_y;

text = \_text;

p = new Pen(Color.White);

}

// Отрисовка круга

public void draw(Graphics g)

{

int rectX = x - r / 2;

int rectY = y - r / 2;

g.FillEllipse(fp, rectX, rectY, r, r);

g.DrawString(text, font, p\_str, x - r / 4, y - r / 4);

}

// Отрисовка точек к линиям

public Point randomPoint(Random rand)

{

int range = r / 6;

int randX = rand.Next(x - range, x + range);

int randY = rand.Next(y - range, y + range);

return new Point(randX, randY);

}

// Получение состояния из списка смежностей

public static State getState(List<State> states, apts\_lab1.StateLSA state)

{

int stateNum = int.Parse(state.state);

return states[stateNum];

}

}

// Класс для отрисовки линий

class Line

{

Pen p;

State s1;

State s2;

string cond\_if;

Font font = new Font("Tahoma", 9, FontStyle.Regular);

// Инициализация

public Line(State \_s1, State \_s2, string \_cond\_if)

{

s1 = \_s1;

s2 = \_s2;

cond\_if = \_cond\_if;

p = new Pen(Color.LightGray, 2.0f);

}

// Отрисовка линии

public void draw(Graphics g, Random rand)

{

int cursorSize = 2;

Point s2Point = s2.randomPoint(rand);

g.DrawLine(p, s1.x, s1.y, s2Point.X, s2Point.Y);

g.DrawEllipse(new Pen(Color.Black, 2), s2Point.X - cursorSize/4, s2Point.Y - cursorSize/4, cursorSize, cursorSize);

Point cond\_ifPoint = new Point(s1.x + (s2Point.X - s1.x) / 3, s1.y + (s2Point.Y - s1.y) / 2);

g.DrawString(cond\_if, font, Brushes.Black, cond\_ifPoint);

}

}

}