**Федеральное государственное образовательное бюджетное учреждение высшего профессионального образования**

**Московский технический университет связи и информатики**

**КАФЕДРА ИНФОРМАТИКИ**

**Лабораторная работа по теме 1  
Компьютерное моделирование графа, заданного в виде квадратной сети.**

**Выполнила:**

**Студентка группы БВТ1701**

**Сенькова Ольга**

**Вариант 22**

**Москва 2019**

**Цель работы:** ознакомится с тем, как хранить и рисовать квадратную сеть, заданной размерности.

**1 Формулировка задания на лабораторную работу**

Реализовать с помощью выбранного языка программирования генерацию, от рисовку и хранение графа в виде квадратной сети заданной размерности n\*m.Интерфейс программы реализовать в виде оконного приложения с областью графического вывода сети, полями для задания размерности и кнопкой для от рисовки.

**2 Техническое задание**

-Параметры ввода: n-количество ребер по Y, m-количество ребер по X

-Параметры вывода: сеть (узлы и соединяющие их линии)

-Вспомогательные переменные: e-холст, X-координаты по оси абсцисс, Y-координаты по оси ординат, Width-размерность области по X, Height- размерность области по Y, Length-длина ребра, DotsX-массив с координатами узла по X, DotsY- массив с координатами узла по Y.

-Класс Node: класс узлов, состоящий из i- номер узла по вертикали, j- номер узла по горизонтали, X- координаты узла по оси X, Y- координаты узла по осиY

**3 Функции программы**

-void Draw():

Осуществляется рисовку на оконном приложении за счет метода CreateGraphics(), при помощи класса рисовки линий Pen(), класса кисти закрашивания SolidBrush(), вырисовки окружности FillEllipse()

-button2\_Click():

Основная функция программы, которая включает в себя ввод, вывод программы, а так нахождения координат узлов, соседних координат узлов для функции Draw()

**4 Код программы**

void Draw()

{

var e = CreateGraphics();

e.Clear(Color.White);

Pen greenPen = new Pen(Color.Green, 3);

for (int i = 0; i < Nodes.Count - 1; i++)

{

if (Nodes[i].i == Nodes[i + 1].i)

e.DrawLine(greenPen, Nodes[i].X, Nodes[i].Y, Nodes[i + 1].X, Nodes[i + 1].Y);

}

for (int i = 0; i < Nodes.Count - m; i++)

e.DrawLine(greenPen, Nodes[i].X, Nodes[i].Y, Nodes[i + m].X, Nodes[i + m].Y);

greenPen.Dispose();

SolidBrush mySolidBrush = new SolidBrush(Color.Green);

for (var i = 0; i < Nodes.Count; i++)

e.FillEllipse(mySolidBrush, (int)Nodes[i].X-50/m, (int)Nodes[i].Y-50/m, (50/m)\*2, (50 / m) \* 2);

mySolidBrush.Dispose();

}

public void button2\_Click(object sender, EventArgs a)

{

var e = CreateGraphics();

int X, Y, Width, Height, Length;

Width = 600;

Height = 600;

m = Convert.ToInt32(textBox1.Text);

n = Convert.ToInt32(textBox2.Text);

if (Height / (m + 1) > Width / (n + 1))

Length = Width / (n + 1);

else

Length = Height / (m + 1);

int[] DotsX = new int[m];

int[] DotsY = new int[n];

for (var i = 0; i < m; i++)

DotsX[i] = Length / 2 + Length \* i;

for (var i = 0; i < n; i++)

DotsY[i] = Length / 2 + Length \* i;

Nodes.Clear();

for (var i = 0; i < n; i++)

for (var j = 0; j < m; j++)

Nodes.Add(new Node(i, j, DotsX[j], DotsY[i]));

for (int i = 0; i < Nodes.Count - 1; i++)

{

if (Nodes[i].i == Nodes[i + 1].i)

{

Nodes[i].Near.Add(Nodes[i + 1]);

Nodes[i + 1].Near.Add(Nodes[i]);

}

}

for (int i = 0; i < Nodes.Count - m; i++)

{

Nodes[i].Near.Add(Nodes[i + m]);

Nodes[i + m].Near.Add(Nodes[i]);

}

Draw();

}

**5 Результаты программы**

На рисунке 1 и 2 изображен результат программы квадратной и прямоугольной сети соответственно

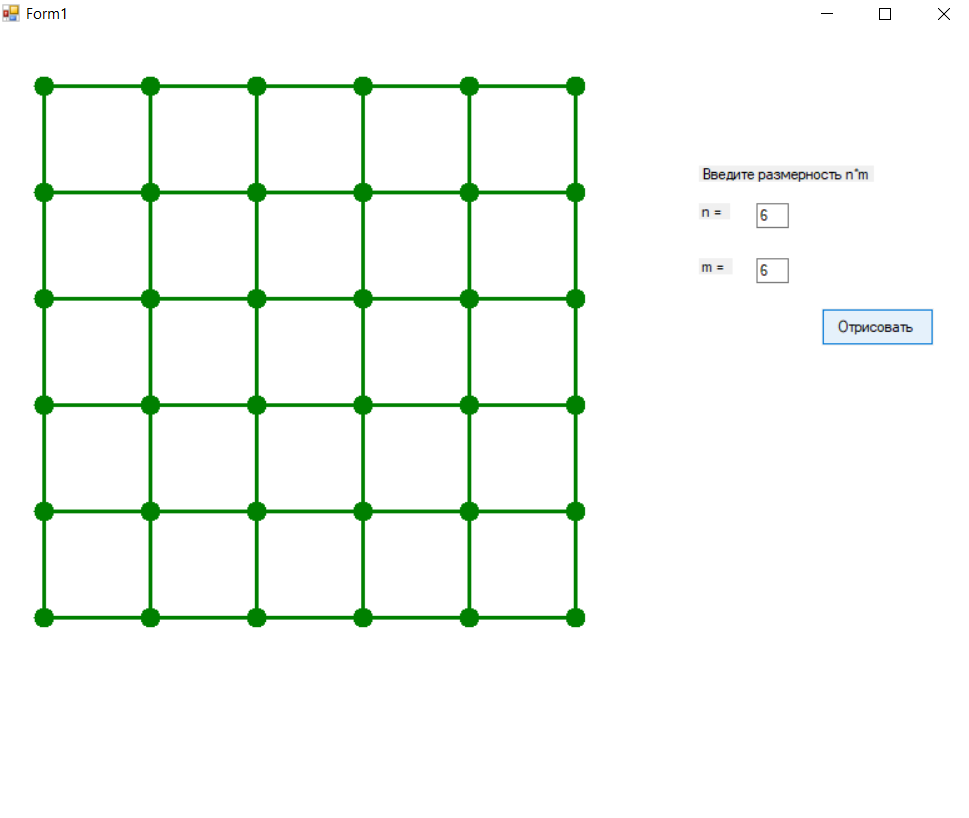


Рисунок 1 - Квадратная сеть.

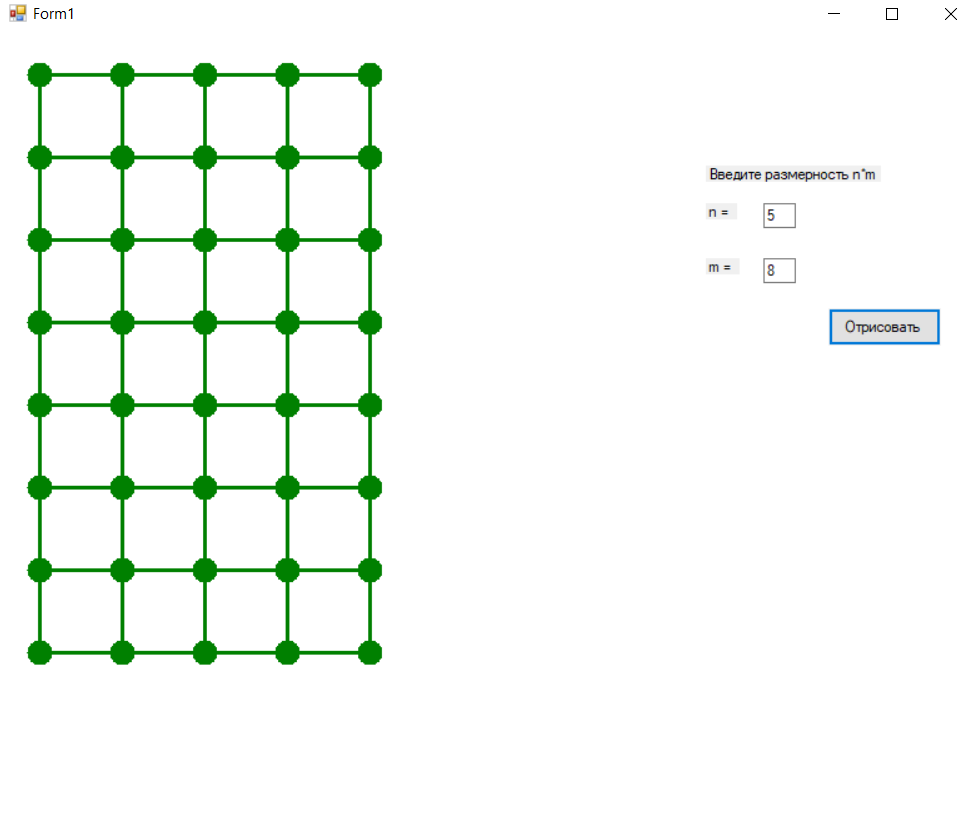


Рисунок 2 - Прямоугольная сеть.

**Вывод:** научились строить граф в виде квадратной или прямоугольной сети.