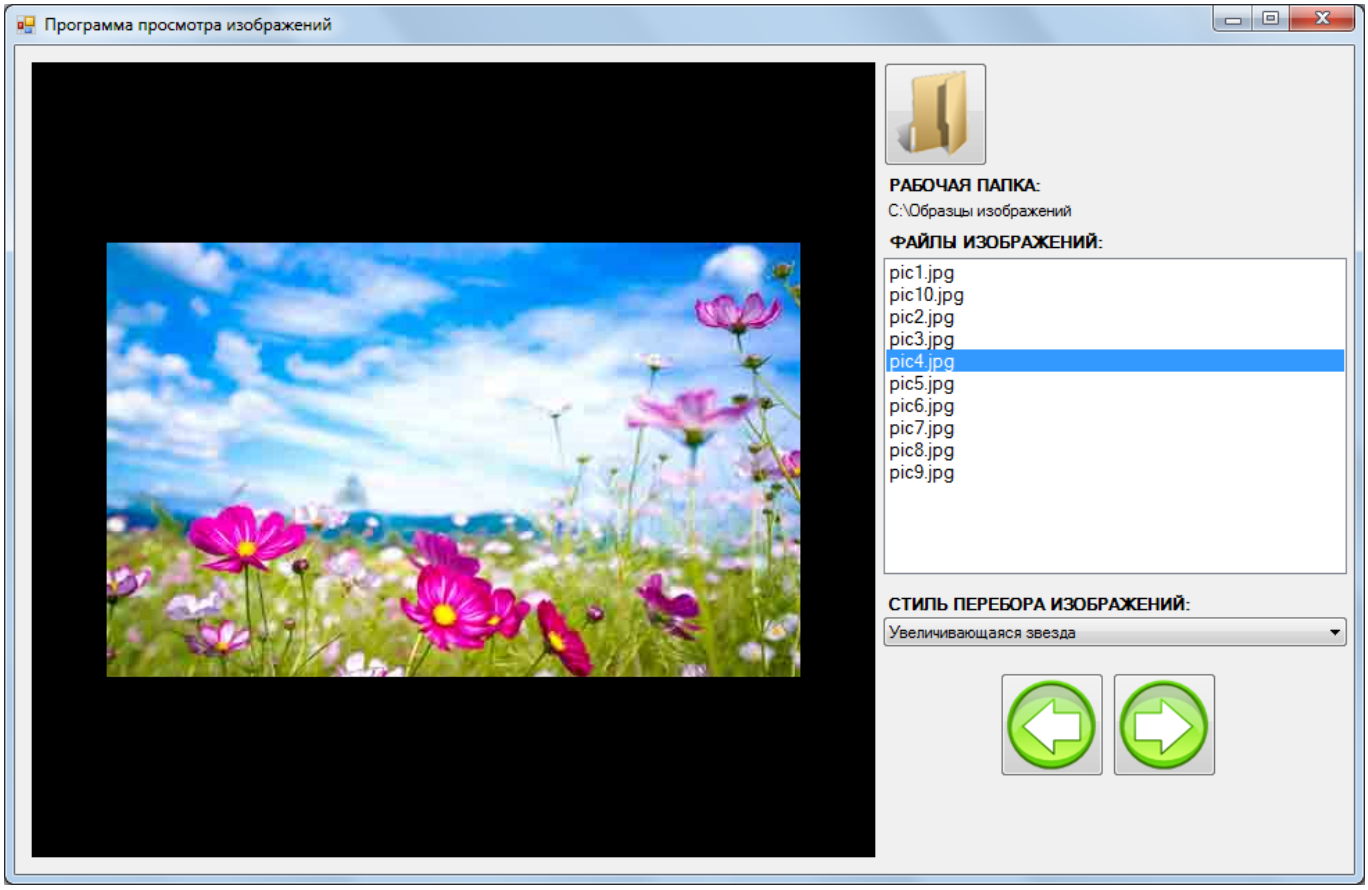
Лабораторная работа №3
«Создание демонстратора изображений с помощью нескольких стилей на языке C#»

Задание

- Создать программу демонстрации на экране файлов изображений, содержащихся в выбранной папке диска.
- При отображении в окне программы каждого изображения реализовать несколько базовых визуальных переходов.
- Реализовать собственный визуальный переход.

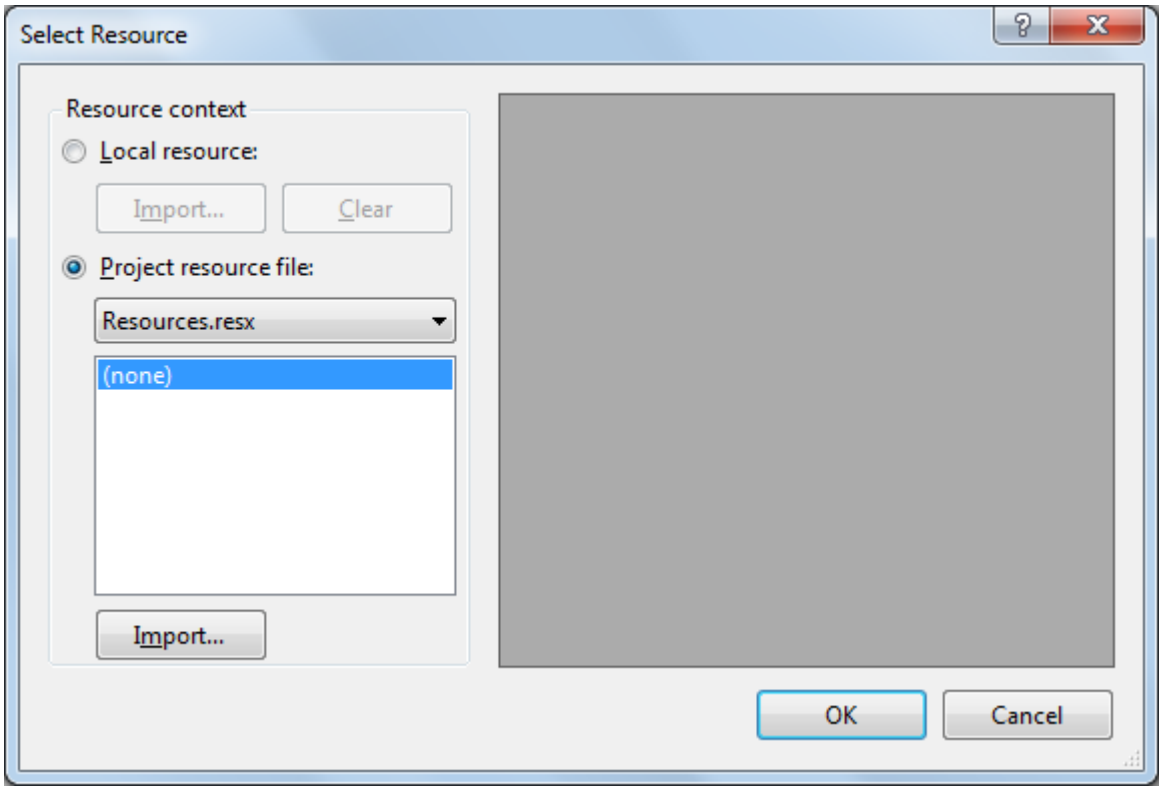
Результат



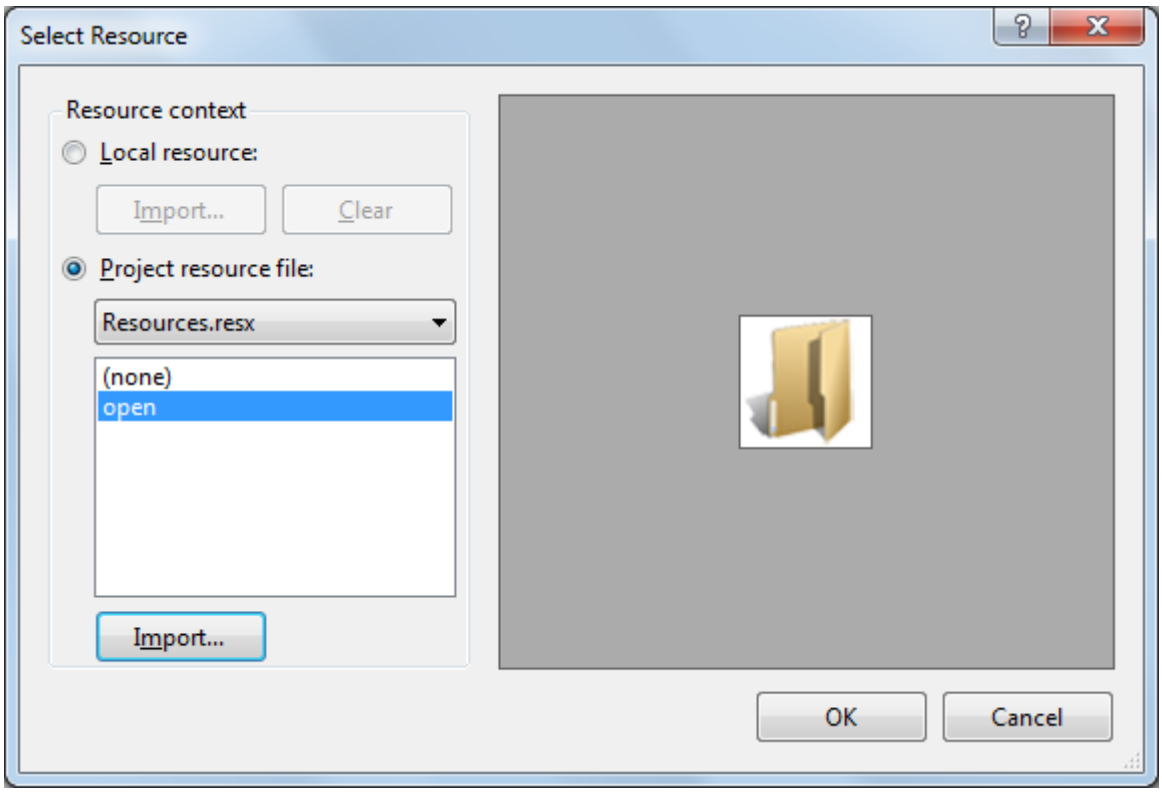
Реализация

Создание формы приложения

- Создать проект приложения.
- Настроить форму приложения (появление формы по центру экрана, название формы «Программа просмотра изображений»)
- На форму установить элемент PictureBox, дав ему имя picMainPicture.
- Установить его соответствующее свойство, чтобы отображаемое им изображение отображалось по центру элемента.
- На форме в правой части оставить пространство для размещения управляющих элементов. При этом пикчебокс настроить так, чтобы при изменении размеров формы он всегда был «прибит» к верхней, левой и нижней частям формы. А расстояние от его правой границы до правого края формы всегда было одинаковым.
- В правую часть формы установить кнопку, дать ей имя butFolder.
- Найти свойство Image кнопки. В этом поле кликнуть справа и нажать появившуюся кнопку с точками. На экран будет выведено следующее окно.

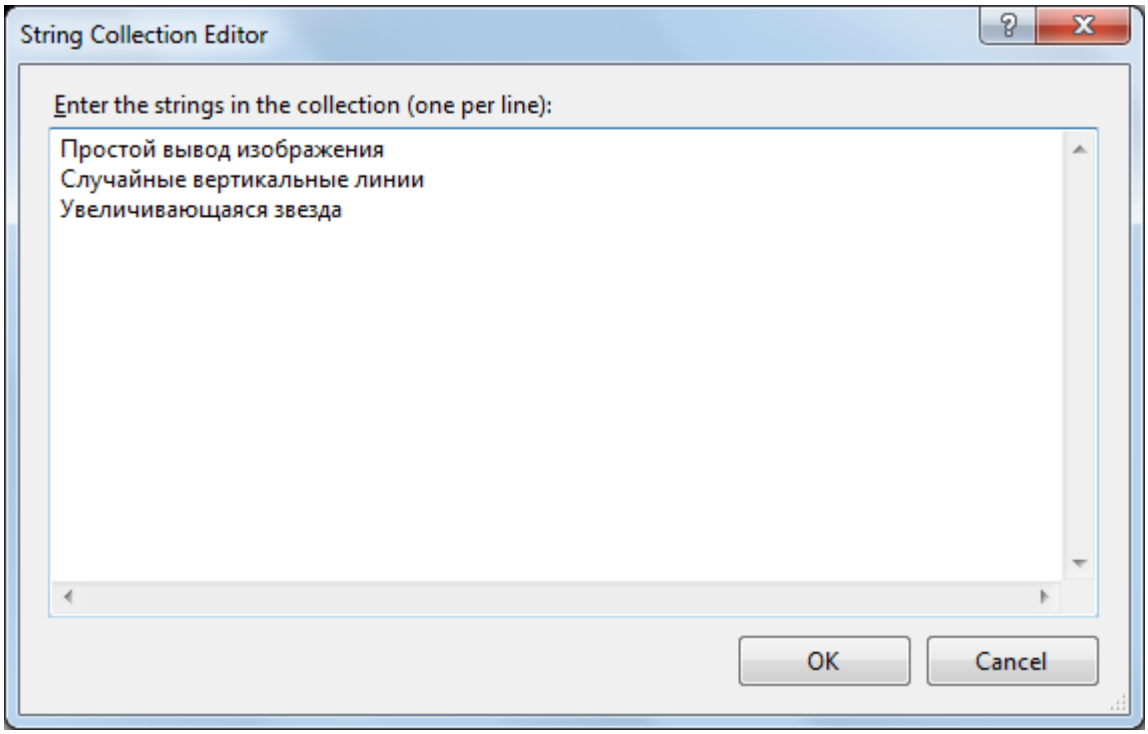


- Нажать кнопку «Import».
- В окне выбора изображения, которое нужно «приклеить» на кнопку, найти и выбрать соответствующее изображение (образцы изображений расположены на кафедральном диске).

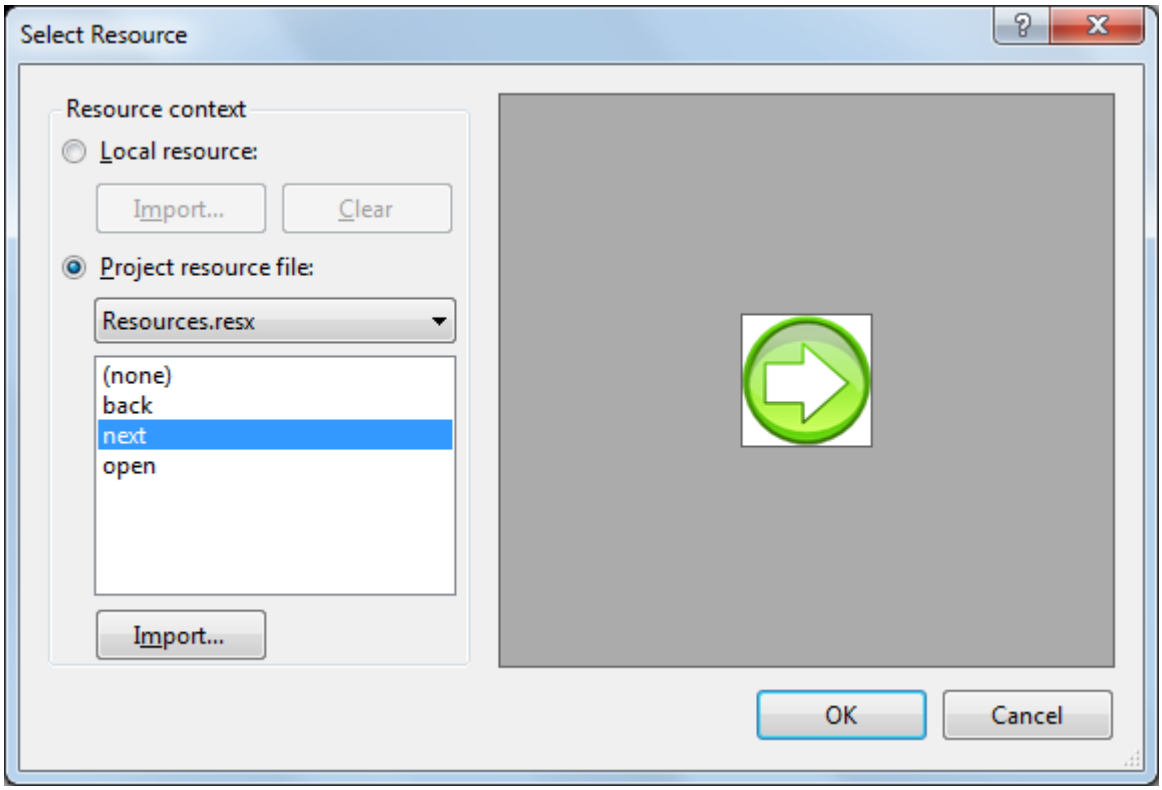


- Нажать кнопку «Ок».
- В результате на кнопку будет наклеено выбранное изображение.
- Для данной кнопки стереть отображаемый ею текст.

- Установить размеры кнопки, равными 75x75 пикселей.
- Чуть ниже кнопки установить на форме элемент Label, в котором написать «РАБОЧАЯ ПАПКА:» (см. Рисунок итоговой формы).
- Чуть ниже установить еще один элемент Label, дав ему имя labFolder. В этот элемент будем записывать имя папки, которую выбрал пользователь.
- Еще ниже установить на форме элемент Label, в котором написать «ФАЙЛЫ ИЗОБРАЖЕНИЙ:» (см. Рисунок итоговой формы).
- Еще ниже установить элемент ListBox. Задача этого элемента – отображать список графических файлов выбранной папки с возможностью выбор одного из них для отображаения в окне формы. Дать ему имя `listFiles.`
- Чуть ниже списка установить на форме элемент Label, в котором написать «СТИЛЬ ПЕРЕБОРА ИЗОБРАЖЕНИЙ:» (см. Рисунок итоговой формы).
- Чуть ниже расположить элемент ComboBox, дав ему имя cmbDrawMode. Его задача отображать комбинированный список стилей смены изображений в окне формы (когда выбираем новое изображение, оно отображается с анимацией выбранного стиля).
- Для его свойства DropDownStyle установить значение DropDownList. Исследуйте самостоятельно назначение различных стилей DropDownStyle списка.
- Найти свойство Items, кликнуть по его правому полю. После появления кнопки с точками нажать кнопку. В появившемся окне ввести пункты (наши стили отображения картинок из файлов), которые будет отображать этот элемент при своей работе. По окончании нажать кнопку «Ок».



- Чуть ниже элемента со списком установить на форму две кнопки, дав им имена butPrev и butNext. Задачей этих кнопок будет перебор для демонстрации в окне формы очередного графического файла из списка.
- Стереть с кнопок отображаемый ими текст.
- Установить их размеры 75x75 пикселей.
- Также как и для кнопки открытиф папки с афйлами (см. выше) – выбрать для каждой из кнопок свойство Image, кликнуть по кнопке с точками. В появившемся окне «приляпывания» к кнопке изображения выбрать и установить нужное изображение (см. следующий рисунок).



- Из панели инструментов выбрать и «кинуть» на форму элемент Timer. Задача этого элемента – организация процесса отображения изображения в окне формы с помощью выбранного стиля.
- Для свойства Interval установить значение 1. Это означает, что таймер раз в 0.001 секунды будет вызывать функцию вывода с заданным стилем изображени в окно программы.
- Скомпилировать программу. Запустить её.

Активация элементов управления на форме

- В класс формы ввести целочисленное поле `m_Mode`. Задача его – хранить номер стиля отображения (рисования) в пикчебоксе картинки из файла. Это поле получает свое новое значение при выборе элемента комбобокса cmbDrawMode.
- Добавить в класс формы также следующие переменные-члены:

```
// режим показа слайда
int m_Mode;

// основное рабочее изображение (содержимое файла)
Bitmap m_MainBmp;
// рабочий битмап с теми же размерами, что и исходное изображение
Bitmap m_ProcessBmp;
// дескриптор графики для работы с копией основного изображения
Graphics m_ProcessGraphics;

// координаты и размеры прямоугольников фрагментов кадра
int m_pw, m_ph, m_px, m_py;
// выполненное колическтво итераций показа фрагмента кадра
int m_NumIterations=0;
// радиус окружности звезды
double m_R = 5
// имя папки с изображениями
```

```
String m_baseFolder;
```

- Чтобы иметь возможность присвоить значение полю `m_Mode` нужно создать функцию-обработчик события SelectedIndexChanged комбобокса cmbDrawMode. Дать имя `OnDrawModeChanged` функции-обработчику.
- В этой функции полю `m_Mode` присвоить значение свойства `SelectedIndex` комбобокса cmbDrawMode.
- В конструктор формы добавить строку

```
cmbDrawMode.SelectedIndex = 0;
```

- Создать функцию-обработчик события Click для кнопки butFolder.
- В этой функции создать объект стандартного диалога выбора пользователем какой-либо папки (вспомнить. лаб. раб. №2).
- По нажатию кнопки Ок в диалоге выбора папки – прочитать директорию, выбранную пользователем (вспомнить лаб. раб. №2).
- В цикле foreach перебирать все файлы, находящиеся в данной папке (вспомнить лаб. раб. №2).
- Для каждого текущего перебираемого файла проверять, графическое ли у него расширение (png, PNG, jpg, JPG, gif, GIF…). Для этого вспомнить строки кода из лаб. раб. №2.
- Если рассматриваемый файл удовлетворяет данным условиям, добавить имя этого файла в список файлов listFiles. Для этого вставить строку:

```
listFiles.Items.Add(fi.Name);
```

- Если количество добавленных файлов в список равно нулю, то сообщить пользователю, что отображать в форме нечего, файлов нет. Для этого вставить в программу следующее условие и вывести на экран сообщение с помощью класса MessageBox и функции его Show. См. предыдущие лаб. раб. для напоминания. В функции Show «грамотно» выбрать соответсвующие типу сообщения иконки и кнопки.

```
if (listFiles.Items.Count == 0)
```

- Если же ссписок содержит файлы, то в переменную-член `m_baseFolder` класса формы скопировать имя выбранной папки и отобразить его в лэйбле `labFolder.`
- Создать функцию UpdateImage. Задача этой функции – отображать в данный момент времени какой-то фрагмент текущего изображения. Многократный последовательный вызов этой функции должен привести к сбору всех частных фрагментов изображения в изображение итоговое. Это как сборка из кирпичей целой стены дома.
- Функция ничего не возвращает и ничего не принимает.
- С помощью оператора switch в данной функции проводить анализ значения переменной класса `m_Mode`. То есть, текущего стиля вывода изображения в окно.
- В ветку для значения `m_Mode,` равного 0 ввести следующий программный код.

```
m_px = 0;
m_py = 0;
m_pw = m_MainBmp.Width;
m_ph = m_MainBmp.Height;
m_ProcessGraphics.DrawImage(m_MainBmp,
                            new Rectangle(m_px, m_py, m_pw, m_ph),
                            new Rectangle(m_px, m_py, m_pw, m_ph),
                            GraphicsUnit.Pixel);
picMainPicture.Image = m_ProcessBmp;
```

```
    timer1.Enabled = false;

    listFiles.Enabled = true;
    cmbDrawMode.Enabled = true;
    butPrev.Enabled = true;
    butNext.Enabled = true;
```

- Здесь определяются ширина и высота изображения из файла. А координаты левого верхнего начала вывода изображения берутся, равными 0.
- С помощью дескриптора графики (Graphics) `m_ProcessGraphics` берется исходное изображение, взятое из файла, и копируется в рабочий битмап таких же размеров. Затем этот битмап копируется в пикчебокс на форме.
- Таким образом, задача нулевого стиля перебора изображений (`m_Mode = 0`) заключается просто в копировании целиком содержимого из файла в пикчебокс на форме.

- Для таймера создать функцию-обработчик события Tick, дав ей имя `OnTimer`. Задача этой функции – вызываться каждую 1 мс и перерисовывать фрагмент отображаемого изображения с помощью выбранного пользователем стиля.
- В этой функции вызвать метод `UpdateImage()`.

- Для списка файлов listFiles создать функцию OnImageListClick как обработчик события Click по списку.
- В эту функцию вставить и изучить следующий программный код.

```
String file;

// если работает таймер, отключить его
if (timer1.Enabled) timer1.Enabled = false;

// задать имя графического файла
file = m_baseFolder + "//" +
       listFiles.Items[listFiles.SelectedIndex].ToString();

// удалить прежнее всё
if (m_MainBmp != null) m_MainBmp.Dispose();
if (m_ProcessBmp != null) m_ProcessBmp.Dispose();
if (m_ProcessGraphics != null) m_ProcessGraphics.Dispose();

// битмап рассматриваемого графического файла
m_MainBmp = new Bitmap(file);
// создать рабочий битмап с теми же размерами,
// что и исходное изображение
m_ProcessBmp = new Bitmap(m_MainBmp.Width, m_MainBmp.Height);
// инициализировать объект Graphics с помощью m_ProcessBmp
// для последующего рисования прямо по m_ProcessBmp
m_ProcessGraphics = Graphics.FromImage(m_ProcessBmp);
// заполнить битмап черным цветом
m_ProcessGraphics.Clear(Color.Black);

// сбросить некоторые рабочие параметры
m_NumIterations = 0;
m_R = 0;
```

```
m_px = m_py = 0;
m_pw = m_MainBmp.Width / 10;
m_ph = m_MainBmp.Height / 10;

// запустить таймер перерисовки фрагмента изображения
timer1.Enabled = true;

listFiles.Enabled = false;
cmbDrawMode.Enabled = false;
butPrev.Enabled = false;
butNext.Enabled = false;
```

- Для кнопок butPrev и butNext создать функции-обработчики кликов по кнопкам.
- В каждой их этих функций менять текущий выбранный элемент в списке файлов на другой (предыдущий или последующий). При этом контролировать его возможный выход на правильные границы значений. Управлять значением текущего выбранного элемента можно с помощью свойства listFiles.SelectedIndex.
- Для отрисовки нового изображения в конце каждой функции вызывать:

```
OnImageListClick(null, null);
```

- В начало программы добавить подключение пространства имен

```
using System.Drawing.Drawing2D;
```

- В класс формы объявить генератор случайных чисел. Он потребуется, чтобы создать стиль построения изображения из вертикальных линий, горизонтальные координаты которых выбираются случайным образом – с помощью генератора случайных чисел.

```
Random m_random;
```

- В конструкторе формы создать генератор случайных чисел

```
m_random = new Random();
```

- В функцию `UpdateImage` вставить ветки для значений `m_Mode`, равных 1 и 2.

```
case 1: // случайные вертикальные полосы

    for (i = 1; i < 5; i++)
    {
        // случайная горизонтальная координата
        m_px = m_random.Next(0, m_MainBmp.Width);
        m_py = 0;
        // ширина линии - 1 пиксель
        m_pw = 1;
        // высота линии - вся высота изображения
        m_ph = m_MainBmp.Height;
        // скопировать часть изображения
        m_ProcessGraphics.DrawImage(m_MainBmp, new Rectangle(m_px, m_py, m_pw, m_ph),
                                    new Rectangle(m_px, m_py, m_pw, m_ph),
                                    GraphicsUnit.Pixel);

        // переписать текущее рабочее изображение в пикчебокс формы
        picMainPicture.Image = m_ProcessBmp;

        // подсчитать выполненное количество итераций построения изображения
        m_NumIterations++;
        // если уже много итераций, закончить процесс
        if (m_NumIterations > 2.0 * m_MainBmp.Width)
        {
            // остановить таймер
```

```
            timer1.Enabled = false;

            listFiles.Enabled = true;
            cmbDrawMode.Enabled = true;
            butPrev.Enabled = true;
            butNext.Enabled = true;

            picMainPicture.Image = m_MainBmp;
            break;
        }
      }
      break;

case 2: // звезда

      double r;   // малый радиус звезды
      Region reg; // трафарет обрезки звезды
      GraphicsPath gp = new GraphicsPath(); // контур звезды

      m_R+=5;     // большой радиус
      r = m_R / 2;

      int n = 50;              // число вершин
      double alpha = 0;        // угол поворота
      double x0 = m_MainBmp.Width / 2, y0 = m_MainBmp.Height / 2; // центр звезды

      // массив вершин звезды
      PointF[] points = new PointF[2 * n + 1];
      double a = alpha, da = Math.PI / n, l;
      for (int k = 0; k < 2 * n + 1; k++)
      {
          if (k % 2 == 0)
              l = r;
          else
              l = m_R;
          // заполнить массив вершин звезды
          points[k] = new PointF((float)(x0 + l * Math.Cos(a)),
                                 (float)(y0 + l * Math.Sin(a)));
          a += da;
      }
      // сформировать контур звезды
      gp.AddLines(points);
      // сформировать трафарет звезды,по которому обрезать всё,что не попало в звезду
      reg = new Region(gp);
      m_ProcessGraphics.Clip = reg;

      m_px = 0;
      m_py = 0;
      m_pw = m_MainBmp.Width;
      m_ph = m_MainBmp.Height;
      m_ProcessGraphics.DrawImage(m_MainBmp, new Rectangle(m_px, m_py, m_pw,m_ph),
                                  new Rectangle(m_px, m_py, m_pw, m_ph),
                                  GraphicsUnit.Pixel);
      // скопировать результат в пикчебокс формы
      picMainPicture.Image = m_ProcessBmp;

      // если звезда уже вылезла за пределы пикчебокса, - заканчивать
      if (m_R > 1.2 * m_MainBmp.Width)
      {
          // остановить таймер
          timer1.Enabled = false;

           listFiles.Enabled = true;
           cmbDrawMode.Enabled = true;
           butPrev.Enabled = true;
           butNext.Enabled = true;
```

```
            picMainPicture.Image = m_MainBmp;
            break;
        }
        break;
```

- Скомпилировать и проверять программу после ввода каждой ветки.

Задания для самостоятельной работы

Реализовать следующие стили показа изображений из файлов.

- Заполнение изображения рядами сверху вниз. В каждом ряду - квадратами слева направо.
- Заполнение изображения случайными горизонтальными линиями во всю высоту изображения.
- Заполнение изображения случайными квадратами.
- Заполнение изображения случайными треугольниками.
- Заполнение изображения столбцами слева направо. В каждом столбце - квадратами сверху вниз.
- Заполнение изображения случайными кругами.
- Заполнение изображения случайными горизонтальными линиями, горизонтальные координаты которых выбираются случайно.
- Заполнение изображения случайными фигурами из списка: круг, квадрат, треугольник.
- Формирование изображения, путем схлопывания (уменьшения) по высоте и ширине черного прямоугольника, который в первый момент размерами совпадает с размерами исходного изображения, а затем начинает умешьшаться по ширине и высоте к центру, показывая исходное изображение.
- Заполнение изображения линиями, координаты которых лежат на сторонах прямоугольника с изображеним и выбираются случайно.
- Заполнение изображения случайными прямоугольниками со случайными значениями их ширин и высот.
- Заполнение изображения случайными вертикальными линиями, вертикальные координаты которых выбираются случайно.