3　グラフィックスの基礎つづき

3.1　プログラム Moyou

前回のプログラム Sen を改造して模様を描く Moyou を作ります。

3.1.1　ファイル名変更

［ファイル｜プロジェクトを開く］で SenP を開く。
［ファイル｜名前をつけて保存］で SenU を MoyouU に名前を変えて保存する。
［ファイル｜プロジェクトに名前をつけて保存］で SenP を MoyouP に名前を変えて保存する。

3.1.2　メインイメージを移動

ImageMain のプロパティを変更する。

Align	alNone
Height	100
Width	100

［編集｜切り取り］して ImageMain をクリップボードに切り取る。
メインパネル PanelMain をクリックして,［編集｜貼り付け］して ImageMain を貼り付ける。
色パネルの下側に位置を調整する。

3.1.3　フォームを修正

Caption	グラフィックスの基礎　－模様－

OnCreate ハンドラーを新しく記入する。

```
procedure TFormMain.FormCreate(Sender: TObject);
begin
  WindowState := wsMaximized;                      // 最大化する
  ImageMain.Canvas.Rectangle(Rect(0,0,100,100));  // メインイメージを初期化する
end; {FormCreate}
```

3.1.4　模様イメージ

[Additional—Image] をフォームの中，左側に置く。

Align	alClient
Name	ImageMoyou

3.1.5　コピーボタン

Button をメインパネルの中，メインイメージの下に置く。（スペースが足りないときは，フォームを縦に拡大してください。）

Name	ButtonCopy	
Caption	コピー (&C)	コピー (C)

3.1.6 OnClick ハンドラー

```
procedure TFormMain.ButtonCopyClick(Sender: TObject);
  CopyMainKaraMoyou;
end; {ButtonCopyClick}
```

3.1.7 メソッド CopyMainKaraMoyou

メインイメージに描いた図を模様イメージにコピーします。縦，横に繰り返しコピーし模様イメージ全体を敷き詰めるようにします。

TFormMain の private 部にヘッダーを宣言する。

```
  private
    { Private 宣言 }
    DownFlag : Boolean;          // マウスボタンを押した状態かどうか
    procedure CopyMainKaraMoyou;
```

本体（実現部）を，フォームのメソッドの部分に記入する。

注意：　今後メソッドを追加するとき，いちいち断りませんが，ヘッダーを private 部に書いてから実現部をフォームのメソッドの部分に書いてください。

```
(******************** フォームのメソッド ********************)
procedure TFormMain.CopyMainKaraMoyou;
var
  X,Y : Integer;
begin
  X := 0;
  repeat
    Y := 0;
    repeat
      ImageMoyou.Canvas.CopyRect(Rect(X,Y,X+100,Y+100),ImageMain.Canvas,Rect(0,0,100,100));
      Y := Y+100;
    until Y > ImageMoyou.Height;
    X := X+100;
  until X > ImageMoyou.Width;
end; {CopyMainKaraMoyou}
```

内側の repeat 〜 until で縦に並べられるだけ並べています。外側の repeat 〜 until で横に並べられるだけ並べています。

プロシージャ CopyRect について

コピー先キャンバス.CopyRect(コピー先長方形, コピー元キャンバス, コピー元長方形)

例

`Image1.Canvas.CopyRect(Rect(100,50,150,250),Image2.Canvas,Rect(0,0,100,100))`
`Image2` の $0 \leqq x < 100,\ 0 \leqq y < 100$ の長方形を，`Image1` の $100 \leqq x < 150,\ 50 \leqq y < 250$ の長方形にコピーします。この場合，横は `0.5` 倍に縮小され，縦は `2` 倍に拡大されます。

3.1.8　実行

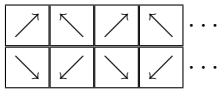
メインイメージに何かを描いてコピーボタンを押すとどうなるでしょうか。

3.2　改良

このように同じ図形を縦横に並べるといちおう模様らしく見えますが，対称性がないとあまり美しく感じません。

CopyRect で，コピー先長方形 $\mathrm{Rect}(x_1, y_1, x_2, y_2)$ を $x_1 > x_2$ とすると，左右反転してコピーします。同様に，$y_1 > y_2$ とすると，上下反転してコピーします。

このことを利用して，メインイメージ ↗ を，下記のようにコピーするように変更します。



```
procedure TFormMain.CopyMainKaraMoyou;
          { メインイメージを模様イメージにコピーする }
var
  X,Y : Integer;
begin
  X := 0;
  repeat
    Y := 0;
    repeat
      ImageMoyou.Canvas.CopyRect(Rect(X,Y,X+100,Y+100),ImageMain.Canvas,Rect(0,0,100,100));
      ImageMoyou.Canvas.CopyRect(Rect(X+200,Y,X+100,Y+100),ImageMain.Canvas,Rect(0,0,100,100));
      ImageMoyou.Canvas.CopyRect(Rect(X,Y+200,X+100,Y+100),ImageMain.Canvas,Rect(0,0,100,100));
      ImageMoyou.Canvas.CopyRect(Rect(X+200,Y+200,X+100,Y+100),ImageMain.Canvas,Rect(0,0,100,100));
      Y := Y+200;
    until Y > ImageMoyou.Height;
    X := X+200;
  until X > ImageMoyou.Width;
end; {CopyMainKaraMoyou}
```

3.2.1　実行

前より模様らしくなりましたか。

ちょっと変ですね。四角と四角が重なったり隙間が空いたりしています。これは，コピー先長方形が

$$\begin{array}{lll} \texttt{Rect}(0,0,100,100) & \text{のとき} & 0 \leqq x < 100 \\ \texttt{Rect}(200,0,100,100) & \text{のとき} & 200 \geqq x > 100 \\ \texttt{Rect}(200,0,300,100) & \text{のとき} & 200 \leqq x < 300 \end{array}$$

とコピーされるからです。

ですから，反転する場合は 1 ドット小さい値にしないといけません。

```
procedure TFormMain.CopyMainKaraMoyou;
          { メインイメージを模様イメージにコピーする }
var
  X,Y : Integer;
begin
  X := 0;
  repeat
    Y := 0;
    repeat
      ImageMoyou.Canvas.CopyRect(Rect(X,Y,X+100,Y+100),ImageMain.Canvas,Rect(0,0,100,100));
      ImageMoyou.Canvas.CopyRect(Rect(X+199,Y,X+99,Y+100),ImageMain.Canvas,Rect(0,0,100,100));
      ImageMoyou.Canvas.CopyRect(Rect(X,Y+199,X+100,Y+99),ImageMain.Canvas,Rect(0,0,100,100));
      ImageMoyou.Canvas.CopyRect(Rect(X+199,Y+199,X+99,Y+99),ImageMain.Canvas,Rect(0,0,100,100));
      Y := Y+200;
    until Y > ImageMoyou.Height;
    X := X+200;
  until X > ImageMoyou.Width;
end; {CopyMainKaraMoyou}
```

3.2.2　実行

こんどは美しくなりましたね。

3.3　ランダムに描く

メインイメージに図を描くのが結構大変です。ランダムに線を描いて模様を作りましょう。

3.3.1　ランダムボタン

Button をコピーボタンの下に置く。

Name	ButtonRandom	
Caption	ランダム (&R)	ランダム (R)

OnClick ハンドラー

```
procedure TFormMain.ButtonRandomClick(Sender: TObject);
begin
  RandomByouga;
end; {ButtonRandomClick}
```

3.3.2　メソッド RandomByouga

```
procedure TFormMain.RandomByouga;
          { メインイメージにランダムに描画してコピーする }
var
  N : Integer;
  X1,Y1,X2,Y2 : Integer;
  Iro : TColor;
begin
  for N := 1 to 100 do
    begin
      X1 := Random(100);
      Y1 := Random(100);
      X2 := Random(100);
      Y2 := Random(100);
      Iro := Random($1000000);
      with ImageMain.Canvas do
        begin
          Pen.Color := Iro;
          MoveTo(X1,Y1);
          LineTo(X2,Y2);
        end;
    end;
  CopyMainKaraMoyou;
end; {RandomByouga}
```

3.3.3　実行

Alt-R を何回も押してみましょう。

3.4　自動化

いちいち，Alt-R を押さなくてもよいように自動的に描くように改良します。

3.4.1　模様タイマー

[System—Timer] をフォームの中に置く（どこでもよい)。

Name	TimerMoyou
Enabled	False

OnTimer ハンドラー

```
procedure TFormMain.TimerMoyouTimer(Sender: TObject);
begin
  RandomByouga;
end; {TimerMoyouTimer}
```

3.4.2　自動ボタン

Button をランダムボタンの下に置く。

Name	ButtonAuto	
Caption	自動 (&A)	自動 (A)

OnClick ハンドラー

```
procedure TFormMain.ButtonAutoClick(Sender: TObject);
begin
  TimerMoyou.Enabled := not TimerMoyou.Enabled;
end; {ButtonRandomClick}
```

タイマーの Enabled が True になると，一定時間（タイマーの Interval ミリ秒）ごとに OnTimer イベントが発生します。Enabled が False のときはイベントは発生しません。

自動ボタンを押すたびに，タイマーの Enabled を否定するので，True と False が切り替わります。すなわち，自動描画と停止をひとつのボタンでコントロールします。

3.4.3　実行

自動的に描画したり停止したりしてみましょう。停止しているときに手動で線を書き足すことができます。

3.5　保存

気に入った模様ができたらファイルに保存することもできます。

3.5.1　保存ボタン

Button を自動ボタンの下に置く。

Name	ButtonSave	
Caption	保存 (&S)	保存 (*S*)

OnClick ハンドラー

```
procedure TFormMain.ButtonSaveClick(Sender: TObject);
begin
  ImageMoyou.Picture.SaveToFile('模様.bmp');
end; {ButtonSaveClick}
```

保存ボタンをクリックすると，模様が “模様.bmp” というファイルに保存されます。保存した画像は Paint などの画像処理プログラムで加工したり，壁紙に設定したりすることができます。ただし，実行画面を最大化しているため，非常に大きなファイルになるので注意が必要です。

3.6　課題

3.6.1　対称性を増加

現在，対称軸は水平と垂直の 2 方向があります。これでも模様らしいのですが，さらに斜めの軸に関しても対称性をもたせると，ずっと模様らしくなります。＼に関して対称にすれば（左右対称なので）／に関しても対称になります。すなわち，ランダムに線分を描いている箇所で，それと $y = x$ に関して対称な線分も描けばよいのです。そのように追加しなさい。

3.6.2　線の太さ

描く線の太さは Pen.Width で設定します。デフォルト（初期設定値）は `Pen.Width = 1` です。ランダムに線分を描くとき，線の太さも 1〜5 の範囲でランダムに選ぶようにしなさい。

太い線もいれると，細い線だけの模様と雰囲気がかなり変わります。あなたはどちらが好きですか。

3.6.3　枠を消す

メインイメージに黒い枠があるので，模様に格子縞ができてしまいます。FormMain の OnCreate ハンドラーでメインイメージを描くとき，枠の色（すなわち Pen.Color）を白にして描くようにしなさい。