PCS 2190 - REP: Repetição

Ricardo Nakamura e Romero Tori

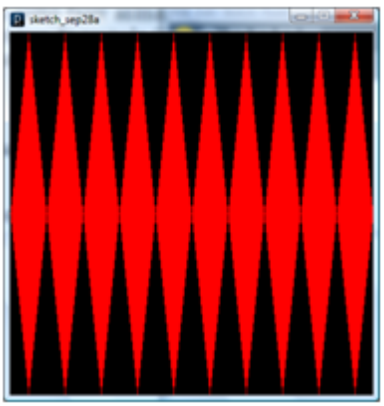
2015

1 Introdução

Nesta apostila, vamos estudar uma forma de realizar a repetição de comandos sem utilizar o bloco **draw**, permitindo maior liberdade na construção de programas.

2 Criando padrões

Vamos retornar um pouco aos programas para criar desenhos estáticos. Imagine que queremos desenhar o seguinte padrão na tela:



Podemos começar pensando em como desenhar um dos losangos que formam o padrão. O seguinte programa faz exatamente isso:

```
size(400, 400);
background(0);
noStroke();
fill(255, 0, 0);
quad(20, 0, 40, 200, 20, 400, 0, 200);
```

Agora que sabemos como desenhar um losango, basta copiar mais nove vezes o comando e mudar as coordenadas para obtermos o padrão. No entanto, esta é uma solução ruim porque teremos um programa longo, com a repetição do mesmo comando várias vezes – e é muito fácil errar na digitação de uma coordenada de um dos losangos.

Podemos solucionar o problema com o bloco **draw**, como já fizemos outras vezes. O programa fica da seguinte forma:

```
int i;

void setup() {
  size(400, 400);
  background(0);
  noStroke();
  fill(255, 0, 0);
}

void draw() {
  quad(i+20, 0, i+40, 200,
       i+20, 400, i, 200);
  i = i + 40;
  if (i > 400) {
        noLoop();
  }
}
```

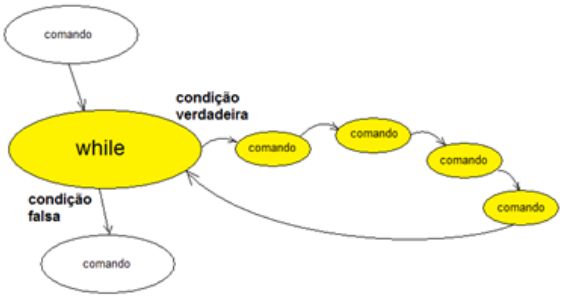
No entanto, essa solução tem dois inconvenientes: o primeiro é que tivemos que acrescentar muitas linhas só para conseguir a repetição do comando **quad**, tornando o programa mais complexo. Observe que tivemos que acrescentar uma condição (i > 400) para fazer com que o bloco **draw** pare de ser executado quando tivermos preenchido a tela.

Outro problema é que conseguimos ver o padrão sendo desenhado progressivamente na tela. Não é possível mostrar o padrão completo logo de início, usando apenas o bloco draw.

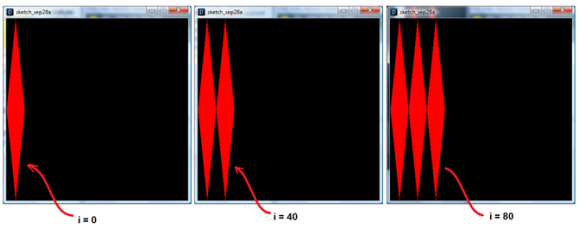
O comando **while** pode ser usado nessas situações para repetir um conjunto de comandos várias vezes, de forma parecida com o bloco **draw**. Experimente executar o seguinte programa.

```
size(400, 400);
background(0);
noStroke();
fill(255, 0, 0);
int i = 0;
while(i < 400) {
  quad(i+20, 0, i+40, 200,
       i+20, 400, i, 200);
  i = i + 40;
}
```

O efeito do comando **while** é repetir tudo o que estiver dentro do bloco entre chaves enquanto a condição indicada for verdadeira. No programa acima, definimos uma nova variável (int i) que começa com valor zero. Enquanto ela tiver valor menor do que 400 (a largura da janela gráfica), os dois comandos seguintes serão repetidos. Este tipo de estrutura é conhecido como laço de repetição ou "loop", porque pode-se imaginar que os comandos formam uma espécie de laço, conforme a ilustração a seguir.



A cada vez que o bloco do while é executado no programa acima, o valor de i é aumentado em 40. Assim, depois de executar algumas vezes (quantas?) a condição "i < 400" não será mais verdadeira e o programa seguirá em frente. Como o desenho do losango (comando quad) depende dos valores de i, o resultado deste laço de repetição é o desenho do padrão completo da janela (sem ter de repetir a mesma linha de programa dez vezes). A figura a seguir mostra a sequência de desenho para os três primeiros valores de i. Como o Processing só mostra a figura completa, não conseguimos ver estes passos acontecendo durante a execução do programa – ao contrário do que acontecia quando usávamos o bloco **draw** para repetição.



Note que é possível criar uma condição que permanece verdadeira para sempre (por exemplo, "i >= 0" no programa anterior). Isso é normalmente indesejável porque faz com que o programa fique repetindo para sempre aquelas instruções – o que é conhecido como laço infinito ou loop infinito.

Atividade 1

Modifique o programa acima para desenhar elipses no lugar dos losangos.

Atividade 2

Modifique o programa acima para desenhar cinco losangos, ao invés de dez – mas ainda ocupando todo o espaço da janela gráfica.

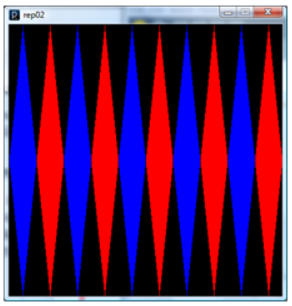
Atividade 3

Modifique o programa acima para desenhar quatro losangos, cada um com 40 pixels de largura. Ou seja, neste caso os losangos não irão preencher toda a janela. Dica: modifique a condição de término do **while**.

3 Combinando repetições e condições

É possível combinar comandos de repetição (while) e execução condicional (if) para se obter resultados mais interessantes. Por exemplo, considere o programa a seguir e a imagem gerada por ele:

```
size(400, 400);
background(0);
noStroke();
int i = 0;
boolean impar = false;
while(i < 400) {
  if (impar == true) {
    fill(255, 0, 0);
    impar = false;
  }
  else {
    fill(0, 0, 255);
    impar = true;
  }
  quad(i+20, 0, i+40, 200,
       i+20, 400, i, 200);
  i = i + 40;
}
```



A primeira novidade que temos neste programa é a definição da variável impar, que recebe o tipo **boolean**. Este tipo indica uma variável que só pode ter dois valores: verdadeiro (true) ou falso (false) – como expressões lógicas.

O comando **if** adicionado dentro do laço de repetição verifica o valor da variável **impar** e seleciona a cor vermelha se o valor for verdadeiro, ou a cor azul se o valor for falso. Note que, nos dois casos, o valor da variável é invertido (passa de verdadeiro para falso ou de falso para ver-

dadeiro). Assim, na próxima execução dos comandos dentro do **while**, a outra cor será selecionada.

Assim, combinando repetição e condições, conseguimos criar padrões que não se repetem de forma exatamente igual sempre, mas dependem de outras variáveis.

Atividade 4

Modifique o programa anterior para desenhar somente o primeiro losango com a cor azul e os demais com a cor vermelha.

Atividade 5

Modifique o programa anterior para desenhar os losangos na metade esquerda da tela de uma cor e os losangos na metade direita da tela de outra cor.

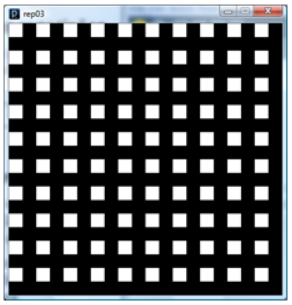
Atividade 6

Modifique o programa anterior para desenhar, alternadamente, duas formas geométricas diferentes. Elas podem ser desenhadas com a mesma cor ou com cores diferentes – como você preferir.

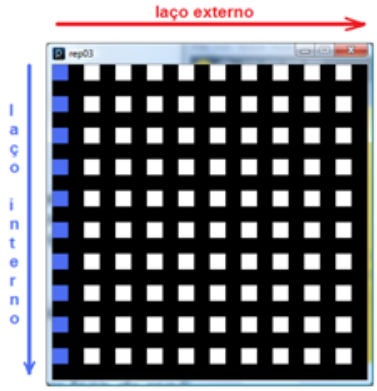
4 Combinando múltiplas repetições

Da mesma forma que fizemos a combinação de **while** e **if**, podemos combinar vários laços de repetição. Imagine que você tem um padrão que se repete dentro de outro padrão – esse é um exemplo em que podemos aplicar este recurso. Considere, por exemplo, o seguinte programa:

```
size(400, 400);
background(0);
noStroke();
int i = 0;
int j = 0;
while(i < width) {
  j = 0;
  while(j < height) {
    rect(i, j, 20, 20);
    j = j + 40;
  }
  i = i + 40;
}
```



Observe que o laço "interno" (o segundo **while**) desenha uma sequência de retângulos na vertical (veja que a variável j é usada para a coordenada vertical do retângulo). Ao mesmo tempo, o laço "externo" (o primeiro **while**) faz com que essa sequência de retângulos seja repetida horizontalmente na tela, conforme ilustrado na figura a seguir. Note que, para desenhar cada coluna, precisamos inicialmente retornar o valor da variável j para zero. O que aconteceria se não fizéssemos isso?



Atividade 7

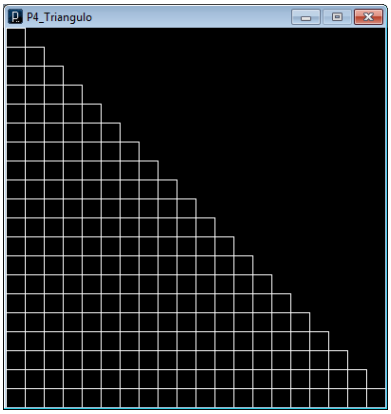
Modifique o programa anterior para desenhar círculos no lugar de retângulos.

Atividade 8

Modifique o programa para preencher somente a metade esquerda da janela com o padrão de quadrados ou círculos.

Uma coisa importante a observar é que podemos modificar a condição de um comando **while** dentro de outro. Por exemplo, o programa a seguir usa este recurso para criar um padrão triangular:

```
size(400, 400);
background(0);
noFill();
stroke(255);
int i = 0;
int j = 0;
int largura = 20;
while(i < height) {
  j = 0;
  while(j < largura) {
    rect(j, i, 20, 20);
    j = j + 20;
  }
  i = i + 20;
  largura = largura + 20;
}
```



Note que o efeito é obtido através de uma nova variável, **largura**. Esta variável começa associada ao valor 20, fazendo com que somente um quadrado seja desenhado na primeira linha – afinal, a condição de repetição horizontal é (j < largura). Em seguida, o valor de **largura** aumenta para 40, permitindo que dois quadrados sejam desenhados na segunda linha e assim por diante.

Atividade 9

Modifique o programa anterior para fazer com que o triângulo seja desenhado de forma invertida, isto é, começando com a linha completa e terminando com somente um quadrado, na última linha.

Atividade 10

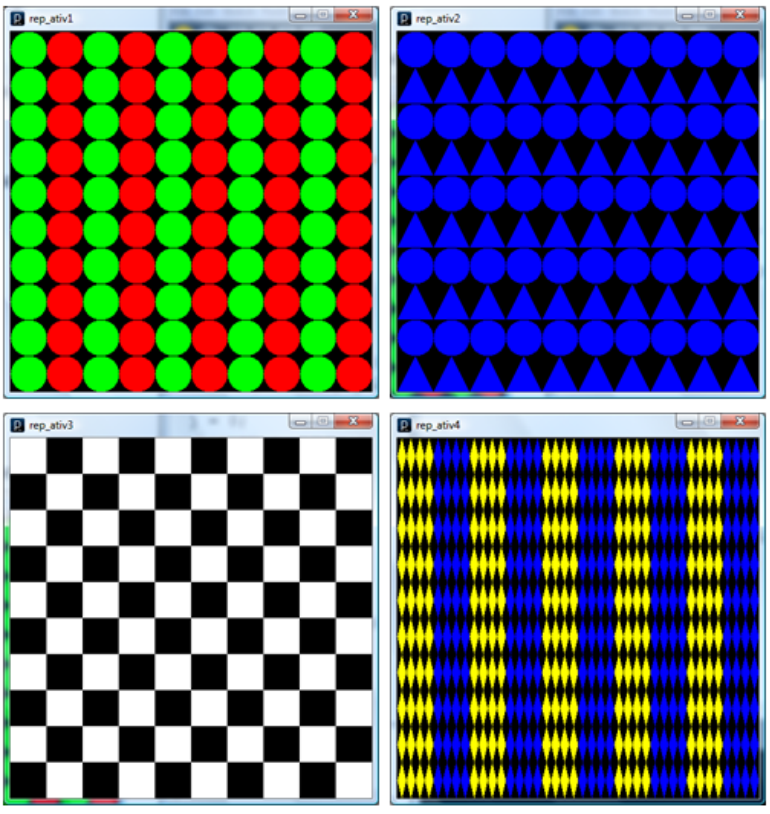
Modifique o programa anterior para fazer com que o triângulo seja desenhado da seguinte forma: a primeira coluna contém um quadrado, a segunda coluna contém três, a terceira coluna contém cinco e assim sucessivamente.

5 Resumo

Nesta atividade, aprendemos a utilizar laços de repetição para obter diferentes efeitos em nossos programas. Vimos também um novo tipo de variável (boolean), usado para armazenar valores lógicos.

Atividade 11

Crie um programa para desenhar cada um dos seguintes padrões. Utilize o comando de repetição **while** em todos eles, e o comando condicional **if** quando for necessário.



Atividade 12

Desafio: Utilizando laços e condicionais, faça um programa que desenha o padrão abaixo: