PCS 2190 - VAR: Variáveis e expressões

Ricardo Nakamura e Romero Tori

2015

1 Introdução

Até o momento, nossos programas foram feitos como simples sequências de comandos que recebem informações adicionais – também conhecidos como parâmetros. Se você fez a última atividade da apostila anterior, deve ter notado que em programas complexos, os parâmetros de um comando podem depender de outros comandos. Por exemplo, o tamanho da porta da casa e a posição da janela dependem do tamanho da parede.

Nesta apostila introduzimos o conceito de variáveis, que ajudam a trabalhar com as informações de um programa. Vamos começar com um programa que desenha um retângulo da largura da janela gráfica.

```
size(300, 300);
background(0);
noStroke();
fill(0, 200, 0);
rect(0, 0, 300, 200);
```

O que acontece, agora, se quisermos desenhar a mesma figura em uma janela maior, por exemplo 500x500 pixels? Aumentar o tamanho da janela é simples:

```
size(500, 500);
```

Faça essa alteração e rode o programa. Como você deve ter notado, o retângulo não tem mais a largura da janela. Isso acontece porque as instruções de desenho utilizam números fixos, como por exemplo, 300.

Precisamos mudar também o comando **rect**. O problema de se fazer isso é que, a cada vez que mudarmos o tamanho da janela, teremos que mudar manualmente o comando que desenha o retângulo, também. No caso de um desenho mais complexo, a chance de cometermos um erro vai aumentando...

Seria muito conveniente se pudéssemos usar nomes – como **largura** – no lugar de valores para nossos comandos. Assim, poderíamos escrever um programa como mostrado a seguir e nosso retângulo sempre ficaria da largura da janela:

```
size(400, 400);
background(0);
noStroke();
fill(0, 200, 0);
rect(0, 0, largura, 200);
```

Se você tentar executar este programa no Processing, receberá uma mensagem de erro. O Processing aceita a idéia de associarmos um nome a um valor -– isso é chamado de variável — mas usar variáveis no Processing envolve alguns outros detalhes.

2 Variáveis

A forma mais simples de se entender uma variável em um programa é como um nome dado para um valor. Um exemplo mais concreto é o de uma gaveta com uma etiqueta colada. Na etiqueta, podemos escrever um nome qualquer que nos ajuda a lembrar o que está guardado na gaveta.

Em algumas linguagens de programação podemos guardar virtualmente qualquer tipo de informação em uma gaveta (ou variável). Em outras, precisamos especificar o tipo de informação que será guardada – esse é o caso do Processing.

Para criar uma variável que armazena números inteiros – um dos tipos de informação mais comuns e úteis nos programas – devemos escrever uma linha como a seguinte:

```
int <nome da variável>;
```

Onde **<nome da variável>** é um nome qualquer que pode ser formado por letras, números e o caractere “sublinhado” (_). O nome não

pode **começar** com um número nem pode ter espaços. Exemplos de nomes válidos: *largura, posicaoX, tamanhoDaLetra*.

Outra informação importante é que maiúsculas e minúsculas **são** diferentes em nomes de variáveis. Portanto o Processing entende *Largura, largura, LaRgUrA e LARGURA* como quatro variáveis diferentes.

Quando escrevemos algo como **int x;** estamos, efetivamente, dizendo que a variável **x** existe – ou seja, criamos um nome ou "gaveta" para uso no programa. No entanto, precisamos associar um valor a essa variável. Quando cientistas de computação escrevem programas de forma abstrata (os chamados *algoritmos*), isso é escrito da seguinte forma:

variável ← **valor**

Infelizmente, não se utiliza esse tipo de notação nas linguagens de programação. Ao invés disso, a maioria das linguagens usam o caractere "=" para indicar essa associação de nome a valor. Ou seja, a linha abaixo associa o nome "abc" ao valor 4:

```
int abc;
abc = 4;
```

Com essas informações, podemos completar o programa anterior, criando uma variável chamada **largura** e associando o valor 400 a ela.

```
int largura;
largura = 400;
size(largura, 400);
background(0);
noStroke();
fill(0, 200, 0);
rect(0, 0, largura, 200);
```

Experimente modificar o valor associado a **largura** e executar novamente o programa. Note que o retângulo permanece sempre com largura igual à janela.

Atividade 1

Modifique o programa anterior para que o retângulo seja desenhado com a largura **e também** a altura da janela.

3 Modificando o valor de uma variável

Como o próprio nome indica, variáveis podem ter o seu valor modificado. Para isso, basta fazer a associação de um novo valor para a mesma variável. O programa a seguir ilustra este recurso para desenhar duas linhas em alturas diferentes da tela:

```
int altura;
size(400, 400);
altura = 100;
line(0, altura, 400, altura);
altura = 200;
// veja que o comando é o mesmo, mas agora o valor de altura é 200
line(0, altura, 400, altura);
```

Observe como duas linhas em posições diferentes são desenhadas, porque modificamos o valor da variável **altura**. Lembre-se que os comandos de um programa são executados em ordem, portanto o Processing desenhará uma linha, a variavel terá seu valor alterado e uma nova linha será desenhada. Qual o valor da variável **altura** no final do programa?

Atividade 2

Modifique o programa anterior para desenhar quatro linhas verticais, usando variáveis.

4 Expressões numéricas

Além de utilizar diretamente os valores associados às variáveis, podemos também escrever expressões matemáticas envolvendo essas variáveis. Por exemplo, o seguinte programa usa uma expressão (largura/2) para desenhar um círculo sempre centralizado horizontalmente na janela.

```
int largura;
largura = 400;
size(largura, 400);
background(0);
ellipse(largura/2, 200, 50, 50);
```

Note que a expressão matemática "largura/2" **não** modifica o valor associado à variável largura. Se quiséssemos comandar o computador a associar um novo valor, poderíamos escrever:

```
largura = largura / 2;
```

Além da operação de divisão, podemos também usar outros operadores para construir as expressões, bem como combiná-los para criar expressões mais complexas:

- adição (+)
- subtração (-)
- multiplicação (*)
- resto da divisão inteira (%)

Atividade 3

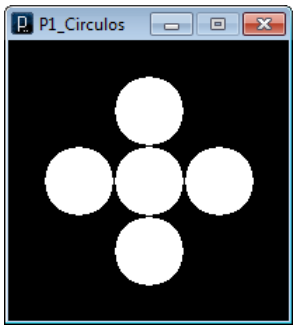
Modifique o programa anterior para desenhar o círculo centralizado na tela, tanto horizontalmente quanto verticalmente.

O programa a seguir desenha cinco retângulos dispostos em uma diagonal na janela. Esse programa mostra também que podemos criar mais de uma variável ao mesmo tempo, desde que sejam do mesmo tipo. Para isso, basta separar os nomes por vírgulas (**int x, y;**).

```
size(400, 400);
int lado;
lado = width/5;
int x, y;
x = 0;
y = 0;
background(0);
rect(x, y, lado, lado);
x = x + lado;
y = y + lado;
rect(x, y, lado, lado);
x = x + lado;
y = y + lado;
rect(x, y, lado, lado);
x = x + lado;
y = y + lado;
rect(x, y, lado, lado);
x = x + lado;
y = y + lado;
rect(x, y, lado, lado);
```

Atividade 4

Faça um programa que desenha o seguinte padrão de círculos, utilizando variáveis.



5 Variáveis automáticas

O Processing já possui algumas variáveis automáticas que representam valores usados com frequência nos programas. Duas dessas variáveis são **width** e **height**, que armazenam as dimensões da janela gráfica. Ou seja, para desenhar uma elipse no centro da tela, podemos escrever o seguinte programa, sem precisar criar novas variáveis:

```
size(400, 400);
background(0);
ellipse(width/2, height/2, 50, 50);
```

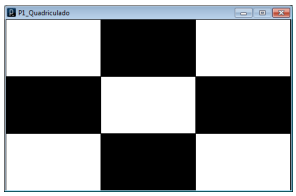
Ao escrever o programa, note que as variáveis **width** e **height** aparecem desenhadas em azul, para indicar que já são reconhecidas como variáveis automáticas.

6 Resumo

Nesta apostila, estudamos como utilizar variáveis para representar informações em nossos programas e aprendemos como criar variáveis numéricas no Processing. Vimos também como escrever expressões numéricas para combinar diferentes variáveis.

Atividade 5

Escreva um programa que desenha um padrão quadriculado com três linhas e três colunas. Esse quadriculado deve cobrir a janela gráfica inteira, não importa o seu tamanho. Use variáveis na solução.



Atividade 6

Escreva um programa que desenha cinco retângulos, um ao lado do outro. Todos os retângulos devem ter a mesma largura, sendo que os cinco, juntos, devem ter a largura da janela. Cada retângulo deve ter o dobro da altura do retângulo à sua esquerda, sendo que o quinto retângulo deve ter a altura igual à altura da janela. Use variáveis na solução.