Estruturas Fundamentais

Técnicas de Programção I

José Antonio F. de Macêdo jose.macedo@lia.ufc.br

Comentários

```
// Comentário de uma linha
/* Comentário de várias linhas */
/** Comentário Documentado */
```

- Blocos, ponto-e-vírgula e espaço
 - toda declaração termina com ";"
 - "{" e "}" delimitam um bloco de declarações

```
{
  int x;
  x = 10 + 30;
}
```

- Identificadores
 - Olniciam com: uma letra, " " ou "\$"
 - puc
 - ApucRIO
 - Puc_Rio
 - Puc Rio
 - A\$Puc\$
 - Não tem tamanho máximo

Palavras-Chave

abstract, boolean, break, byte, case, catch, char, class, continue, default, do, double, else, extends, false, final, finally, float, for, if, implements, import, instanceof, int, interface, long, native, new, null, package, private, protected, public, return, short, static, super, switch, syncronized, this, throw, throws, transient, true, try, void, volatile, while ...

Tipos Básicos

○ boolean	true ou false
○ char	caracter UNICODE (16 bits)
byte	número inteiro com sinal (8 bits)
○ short	número inteiro com sinal (16 bits)
○int	número inteiro com sinal (32 bits)
○ long	número inteiro com sinal (64 bits)
○ float	número em ponto-flutuante (32 bits)
○ double	número em ponto-flutuante (64 bits)

char: representa um caracter Unicode de 16 bits

\b retrocesso

● \t tab

■ \n avanço de linha

\r retorno de carro

\" aspas

\' apóstrofo

\\ barra invertida

\u001B indica o caracter unicode em hexadecimal

- Tipos Inteiros
 - Obyte 8 bits $-2^7...2^7-1$
 - Oshort 16 bits -2¹⁵..2¹⁵-1
 - oint 32 bits -2³¹..2³¹-1
 - Olong 64 bits -2⁶³..2⁶³-1
 - Representações:
 - 2 (decimal)
 - 077 (octal)
 - OxBA (hexadecimal)

- Tipos Ponto Flutuante
 - Ofloat 32 bits
 - Odouble 64 bits
 - Representações:
 - 3.14
 - 6.02E23
 - 2.718F
 - 123.4E+306D

Declarações de variáveis

```
int x, y; // variáveis inteiras
float z; // variável float
double w; // variável double
boolean verdade; // variável booleana
char c; // variável caracter
```

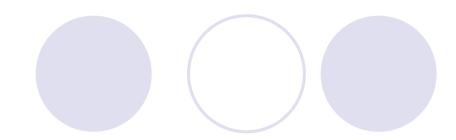
Atribuições e inicializações

```
// declaração
int x, y;
float z = 3.144f; // declaração com
                         // atribuição
double w = 3.1415;
boolean verdade = true;
char c, d;
                       // atribuição
c = 'A';
d = 'u0013';
x = 6;
y = 1000;
```

Variáveis

- Inicialização default do Java
 - Ovariáveis numéricas com zero
 - Ovariáveis boolean com false
 - Ovariáveis com null

Operadores



Lógicos

- ○Comparação de valor: == , != , > e <
- OJunção de expressões: &, &&, |, ||
- Comparação de objetos: instanceof()

Operadores

- Aritméticos
 - Soma (+)
 - Subtração ()
 - OMultiplicação (*)
 - ODivisão (/)
 - OResto (%)

Operadores



- Considere um operador @ qualquer
- Expressões do tipo X = X @ Z podem ser substituídas por X @= Z

x = x + 3	x += 3
x = x * (9 + y)	x *= (9 + y)

if, else

```
if (expressão booleana)
{ instrução ou bloco de comandos }
else
{ instrução ou bloco de comandos }
```

```
if (cont >= 0)
{
   System.out.Println("Erro !!!");
}
else
{
   System.out.println("Ok !");
}
```

Switch

```
switch (expressão short,int,byte ou char)
{
    case expressão2:
        comandos;
        break;
    case expressão3:
        comandos;
        break
    default:
        comandos;
        break;
}
```

```
switch (cor)
      case 0:
        setBackground(Color.black);
        break;
      case 2:
        setBackground(Color.red);
        break;
      default:
        setBackground(Color.white);
        break;
```

- for
 - ofor (expr_inicial; expr_booleana; expr_increm)
 { bloco de comandos }

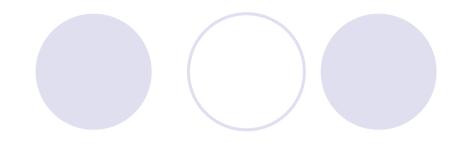
```
for (int x = 0, int y=0; x <10; x++, y--)
{
    System.out.println("Valor do X : " + x);
}</pre>
```

while
while (expr_booleana)
{ bloco de comandos }

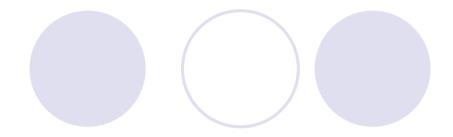
```
int cont = 0
while (cont < 100)
{
   System.out.println("contando "+
   Integer.toString(cont);
   cont++;
}</pre>
```

dodo{ bloco de comandos }while

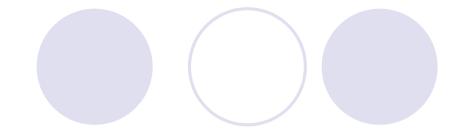
```
int x = 0;
do
{
x++;
} while (x <10);</pre>
```



Declaração

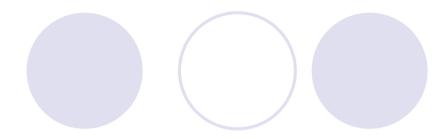


Declarando, criando e iniciando um array



Multi-dimensionais

```
int duasDim [ ] [ ] = new int [4] [ ];
duasDim [0] = new int [5];
duasDim [1] = new int [4];
duasDim [2] = new int [3];
duasDim [3] = new int [2];
duasDim [0][0] = 300;
duasDim [1][3] = 600;
```



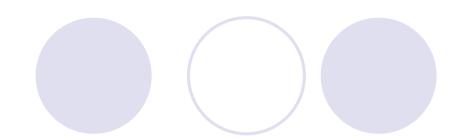
Multi-dimensionais, outra forma

```
int matriz [][] = new int [4][5];
matriz [0][0] = 300;
matriz [1][3] = 600;
```

- Todo array é um objeto
- Para determinarmos o seu tamanho podemos usar o método length:

```
int lista [] = new int [10];
for (int j = 0; j < lista.length; j+
    +)
{
    System.out.println(lista[j]);
}</pre>
```





Método arraycopy

```
System.arraycopy(ArrayOrigem,
PosiçãoInicial,
ArrayDestino,
PosiçãoDestino,
nrDeElementosParaCopiar)
```