

Aula 6



Objetivos

- **Operadores de atribuição;**
- **Operadores de incremento e decremento;**
- **Tipos de dados primitivos;**
- **Princípios básicos da repetição controlada por contador;**

Aula 6



Operadores de atribuição;

variável = *variável* **operador** *expressão*

+

-

*

/

%

Aula 6



Operadores de atribuição;

```
//Programa de Atribuicao
//Pacotes de extensão Java
import javax.swing.JOptionPane;

public class Atribuicao {

    // Método main início da aplicação Java
    public static void main( String args[] )
    {
        // inicializando variável
        int c = 0;
        c = c + 3;

        JOptionPane.showMessageDialog( null, c,
            "RESULTADO",
            JOptionPane.INFORMATION_MESSAGE );

        System.exit( 0 ); // término da aplicação

    } // Fim do método main
} // Fim da classe Atribuicao
```

Aula 6



Operadores de atribuição;

variável **operador** = *expressão*

+=

-=

***=**

/=

%=

Aula 6



```
//Programa de Atribuicao
//Pacotes de extensão Java
import javax.swing.JOptionPane;

public class Atribuicao2 {

    // Método main início da aplicação Java
    public static void main( String args[] )
    {
        // inicializando variável
        int c = 3, d = 5, e = 4, f = 6, g = 12;

        JOptionPane.showMessageDialog( null, "c = 3, d = 5, e = 4, f = 6, g = 12 ",
            "Exemplo : ",
            JOptionPane.INFORMATION_MESSAGE );

        JOptionPane.showMessageDialog( null, "c += 7 -> c = c + 7 = " + (c += 7),
            "Adição : ",
            JOptionPane.INFORMATION_MESSAGE );

        System.exit( 0 ); // término da aplicação

    } // Fim do método main
} // Fim da classe Atribuicao
```

Aula 6



Operadores de incremento e decremento;

Java fornece o operador de incremento unário, **++**, e o operador de decremento unário, **--**.

O programa pode incrementar:

c = c + 1 na forma **C++**

O programa pode decrementar:

c = c - 1 na forma **C--**

Se o operador de incremento ou decremento é colocado antes de uma variável, ele passa a ser chamado de operador de pré-incremento ou pré-decremento.

Aula 6



Operadores de incremento e decremento;

Se o operador de incremento ou decremento é colocado depois de uma variável, ele passa a ser chamado de operador de pós-incremento ou pós-decremento.

Ex:

| | | |
|-----------|----------------|------------|
| ++ | pré-incremento | ++a |
| ++ | pós-incremento | a++ |
| -- | pré-decremento | --b |
| -- | pós-decremento | b-- |

Aula 6



Operadores de incremento e decremento;

Java fornece o operador de incremento unário, **++**, e o operador de decremento unário, **--**.

O programa pode incrementar:

c = c + 1 na forma **C++**

O programa pode incrementar:

c = c - 1 na forma **C--**

Aula 6

//Programa de Atribuicao

//Pacotes de extensão Java

```
import javax.swing.JOptionPane;
```

```
public class Incremento {
```

```
// Método main início da aplicação Java
```

```
public static void main( String args[] )
```

```
{
```

```
// inicializando variável
```

```
int c;
```

```
c = 5;
```

```
JOptionPane.showMessageDialog( null, "c = 5 ",
```

```
"Incremento : ",
```

```
JOptionPane.INFORMATION_MESSAGE );
```

```
JOptionPane.showMessageDialog( null, "c ++ -> " + (c ++),
```

```
"Incremento : ",
```

```
JOptionPane.INFORMATION_MESSAGE );
```

```
JOptionPane.showMessageDialog( null, "c -> " + c ,
```

```
"Incremento : ",
```

```
JOptionPane.INFORMATION_MESSAGE );
```

```
System.exit( 0 ); // término da aplicação
```

```
} // Fim do método main
```

```
} // Fim da classe Incremento
```



Aula 6



```
//Programa de Atribuicao  
//Pacotes de extensao Java  
import javax.swing.JOptionPane;
```

```
public class Incremento2 {
```

```
// Método main início da aplicação Java
```

```
public static void main( String args[] )
```

```
{
```

```
// inicializando variável
```

```
int c;
```

```
c = 5;
```

```
JOptionPane.showMessageDialog( null, "c = 5 ",  
    "Incremento : ",  
    JOptionPane.INFORMATION_MESSAGE );
```

```
JOptionPane.showMessageDialog( null, "++c -> " + (++c),  
    "Incremento : ",  
    JOptionPane.INFORMATION_MESSAGE );
```

```
JOptionPane.showMessageDialog( null, "c -> " + c ,  
    "Incremento : ",  
    JOptionPane.INFORMATION_MESSAGE );
```

```
System.exit( 0 ); // término da aplicação
```

```
} // Fim do método main
```

```
} // Fim da classe Incremento2
```

Aula 6



Resumo dos operadores:

Operadores

()

++ --

++ -- + -

* / %

+ -

< <= > >=

== !=

= += -= *= /= %=

Tipo

parênteses

unário pós-fixado

unário

multiplicação

de adição

relacional

de igualdade

de atribuição

Aula 6



Tipos de dados primitivos;

Ao contrário das linguagens de programação *C* e *C++*, os tipos primitivos em Java são portáveis entre todas as plataformas de computador que suportam Java. Este e muitos outros recursos de portabilidade de Java permitem que os programadores escrevam programas uma só vez, sem saber qual a plataforma de computador em que o programa será executado. Esse atributo é às vezes conhecido como “*WORA (Write Once, Run Anywhere – escreva uma vez, rode em qualquer lugar)*”.

Aula 6



Tipos de dados primitivos;

| Tipo | Tamanho em bits | Valores | Padrão |
|----------------|-----------------|---|--|
| <u>boolean</u> | 8 | <u>True</u> ou <u>false</u> | |
| <u>char</u> | 16 | '\u0000' a '\uFFFF' (0 a 65535) | (conj. De caracteres <u>Unicode ISO</u>) |
| <u>byte</u> | 8 | -128 a +127 | |
| <u>short</u> | 16 | -32768 a +32767 | |
| <u>int</u> | 32 | -2.147.483.648 a +2.147.483.647 -2^{31} a $2^{31}-1$ | |
| <u>long</u> | 64 | -9.223.372.036.854.775.808 a +9.223.372.036.854.775.807 -2^{63} a $2^{63}-1$ | |

Aula 6



Tipos de dados primitivos;

| Tipo | Tamanho em bits | Valores | Padrão |
|---------------|-----------------|---|-------------------------------|
| <u>float</u> | 32 | <i>Intervalo negativo</i> $-3,40282346638552886E+38$ a $-1,40129846432481707E-45$ <i>Intervalo positivo</i> $1,40129846432481707E-45$ a $3,40282346638552886E+38$ | (Ponto flutuante IEEE 754) |
| <u>double</u> | 64 | <i>Intervalo negativo:</i> $-1,7976931348623157E+308$ a $-4,94065645841246544E-324$ <i>Intervalo positivo:</i> $4,94065645841246544E-324$ a $1,7976931348623157E+308$ | (Ponto flutuante IEEE 754) |

Aula 6



Princípios básicos da repetição controlada por contador;

- o nome de uma variável de controle (ou contador de laço);
- o valor inicial da variável de controle;
- o incremento(ou o decremento) pelo qual a variável de controle é modificada a cada passagem pelo laço(também conhecido como cada iteração do laço); e
- a condição que testa o valor final da variável de controle (isto é, se o laço deve continuar).



```
// Repetição controlada

// Pacote do Java
import java.awt.Graphics;

// Pacote extensão Java
import javax.swing.JApplet;

public class WhileConte extends JApplet {

    // desenha linhas no fundo do applet
    public void paint( Graphics g )
    {
        // chamada hierarquica de versão do método pintar
        super.paint( g );

        int counter = 1;           // inicialização

        while ( counter <= 10 ) { // condição de repetição
            g.drawLine( 10, 10, 250, counter * 10 );
            ++counter;           // incremento
        } //fim da estrutura while

    } // fim do método pintar

} // fim da classe WhileConte
```

Aula 6



Princípios básicos da repetição controlada por contador.

A estrutura *while* utiliza a referência *Graphics.g*, que faz referência ao objeto *Graphics* do *applet*, para enviara a mensagem *drawLine* para o objeto *Graphics*, pedindo-lhe para desenhar uma linha.

Estrutura *while* incrementa a variável de controle por 1 a cada iteração do laço.

Aula 6



Princípios básicos da repetição controlada por contador.

- Os programas devem controlar a contagem dos laços com valores inteiros;
- Como os valores em ponto flutuante podem ser aproximados, controlar a contagem de laços com variáveis em ponto flutuante pode resultar em valores imprecisos do contador e teste de terminação não-exatos;
- Recue as instruções no corpo de cada estrutura de controle;
- Coloque uma linha em branco antes e depois de cada estrutura de controle importante para destacá-la no programa.