

Curso de Java

Comandos de Controle

Prof. Fernando Vanini

IC-Unicamp

- Comando condicional
- Comandos repetitivos
- Comando de seleção múltipla
- Comando break
- Comando for

- Como acontece com a maioria das linguagens de programação, os comandos em Java são executados sequencialmente, na ordem em que aparecem no programa fonte.
- A seqüência de execução pode ser alterada através de comandos de controle do fluxo de execução.

- Comando condicional:

`if (condição) comando ;`

ou

`if (condição) comando ; else comando ;`

onde

- *condição* é uma expressão cujo resultado é booleano
- *comando* é qualquer comando da linguagem

- Exemplos:

```
if((a && b) || c) m += 10;
```

```
if(i > j) max = i; else max = j;
```

```
if(a > b)
    if(a > c) max = a;
    else max = c;
else if(b > c) max = b;
    else max = c;
```

- forma geral:

```
switch ( expressão ) {  
    case valor1 : comandos  
    case valor2 : comandos  
    ...  
    default: comandos  
}
```

- O valor de *expressão* é calculado e comparado sequencialmente com cada uma das opções *valor1*, *valor2*, etc.
- Se uma das opções for igual ao valor da *expressão*, a execução continua a partir dos comandos associados a esse valor.
- Se nenhuma das opções for igual ao valor da *expressão*, a opção *default* será selecionada e a execução irá continuar a partir dos comandos associados à mesma.

- O comando **break** interrompe a seqüência de execução de um bloco de comandos.
- Normalmente ele é necessário ao final de cada opção no comando de seleção **switch-case** porque a interrupção da seqüência ao final de cada opção não é automática.

```
switch(dia % 7) {  
    case 6:  /*** sábado ***/  
            tarifa = 8;  
            break;  
    case 0:  /*** domingo ***/  
            tarifa = 5;  
            break;  
    default: /*** demais dias ***/  
            tarifa = 10;  
            break;  
}
```

```
switch (dia % 7) {  
    case 1:  
    case 2:  
    case 3:  
    case 4:  
    case 5:    tarifa = 10;    break;  
    case 6:    tarifa = 8;    break;  
    case 0:    tarifa = 5;    break;  
}
```

- Comando while:

while (*condição*) *comando* ;

- Enquanto *condição* for verdadeira, *comando* será executado repetidamente. O teste de condição é feito antes de cada execução de comando.

- Exemplos:

```
int s = 0, i = 1;  
while(i <= 10) s +=(i++);
```

```
int i = 0;  
while(k > 0) { k/=2; i++; }
```

O exemplo acima usa o *comando composto*, formado por uma seqüência de comandos entre { e }. Ele pode ser usado em qualquer lugar onde é usado um *comando*.

```
/* MDC entre 2 inteiros */  
public class MDC {  
    public static void main(String[] args) {  
        int a = 100;  
        int b = 333;  
        while(a != b)  
            if(a > b) a -= b;  
            else    b -= a;  
        System.out.println("MDC(100,333) == "+a);  
    }  
}
```

- comando do-while

```
do { comandos } while ( condição );
```

- a seqüência de *comandos* irá se repetir enquanto *condição* for verdadeira. O teste de condição é feito depois da execução de *comandos*.

- Exemplos:

```
int s = 0, i = 1;  
do {  
    s += i++;  
} while ( i < n );
```

- Forma geral:

for (*inicialização* ; *condição* ; *incremento*)
comando ;

1. Inicialmente *inicialização* é executada.
2. Se *condição* for verdadeira *comando* é executado senão a execução do comando **for** é encerrada.
3. *incremento* é executado e o comando continua a partir do passo anterior.

- O comando
for (*inicialização ; condição ; incremento*)
comando ;
- É equivalente a um comando **while** da forma

inicialização ;

```
while ( condição ) {  
    comando ; incremento ;  
}
```

```
for(j = 0; k > 0; j++) k /= 2;  
  
s = 0;  
for(int i = 0; i <= 10; i++) s += i;
```

No exemplo acima, a variável `i` é declarada 'dentro' do comando `for`. Isso é possível em Java e nesse caso, ela só pode ser usada no próprio comando.

```
/* Exemplo: for */
public class Exemplo{
    public static void main(String[] args){
        int n = 10;
        for(int i = 1; i <= n; i++) {
            int s = 1; System.out.print(s);
            for(int j = 2; j <= i; j++) {
                s+=j; System.out.print("+"+j);
            }
            System.out.println("="+s);
        }
    }
}
```