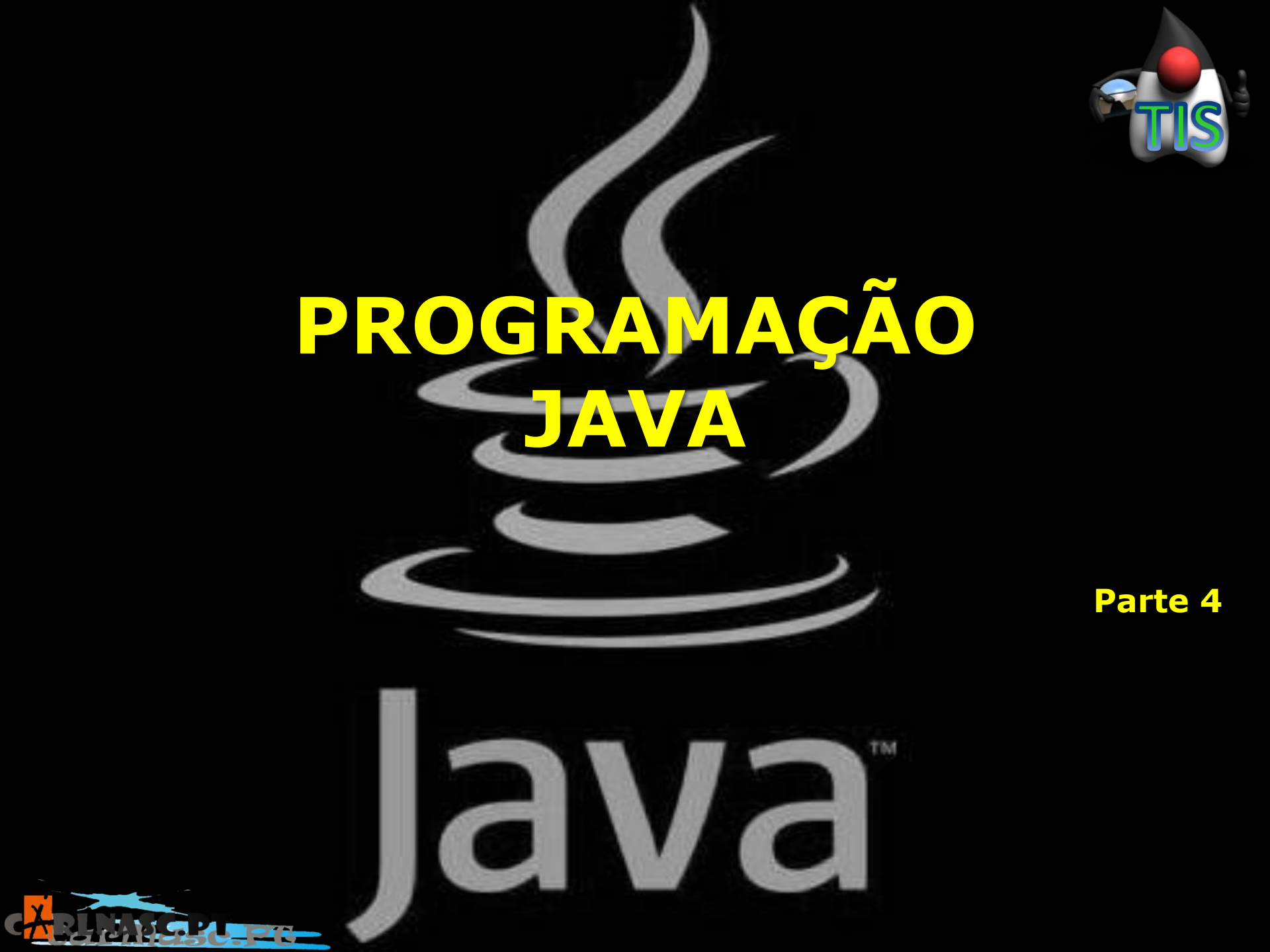
TIS
PROGRAMAÇÃO
JAVA
Parte 4
Java™

A instrução condicional IF

```
if (x < 10) x = 10;
```

O sistema avalia uma expressão e, se for verdadeira, é executada a ação especificada :

Se o valor de X for menor que 10, então o X ficará igual a 10

Poderia ter sido escrita assim:

```
If (X < 10)
X = 10;
```

ou

```
if (X<10) {X = 10;}
```

A instrução condicional IF... ELSE

```
if (x  !=  xantes)
        {
        System.out.print("o X foi alterado");
        }
else
        {
         System.out.print("o X não mudou");
        }
```

O sistema avalia uma expressão e, se for verdadeira, é executada a ação especificada , SENÂO é executada uma alternativa.

Instruções IF… ELSE IF

```
if ( n == 1 )  {
        // executa o Bloco 1
}
else  if ( j == 2 ) {
        // executa o Bloco 2
}
else    {
        // se todos os testes anteriores falharam,
                executa o Bloco 3
}
```

Instruções IF… ELSE encadeadas

```
if (meuX  > 100)  {
        if (resto == true) {        //resto? sim ou não?
                meuX = meuX % 100;
        }
        else  {
                meuX = meuX / 100.0; }
        }
else
        {
         System.out.print("o meuX está correto");
        }
```

Mas... Atenção!

MAL

```
if ( i == j )
        if ( j == k )
                System.out.print("i igual a j");
        else
                System.out.print("i diferente de j");
```

BEM

```
if ( i == j ) {
        if ( j == k )
                System.out.print("i igual a j"); }
        else
                System.out.print("i diferente de j");
```

O ciclo FOR – Iteração

O ciclo FOR – Iteração

O loop FOR permite a repetição dum bloco um certo número de vezes, conforme a variável de controlo , a sua inicialização e o seu incremento

No início de cada ciclo é testada a condição

```
for (var1 = 0;  var1 <50; var1 = var1+1)
        {
                Bloco de instruções que será executado
                50 vezes ;
        }
```

Ou var1++

Sequência: atribuição, teste, incremento, bloco

O ciclo SWITCH … CASE n

A Instrução SWITCH

```
switch ( n )  {
    case 1:
        // executa o Bloco 1
        break;
    case 2:
        // executa o Bloco 2
        break;
        default:
        // se todos os testes anteriores falharam,
                executa o Bloco 4
        break;  }
```

O ciclo FOR

Loop de n vezes:

```
for ( i = 0; i < n; i++)   {
      // executa o Bloco  n vezes,
      // desde 0 até n-1
      }
```

Loop encadeado:

```
for ( j = 0; j < 10;  j++)   {
       for ( i = 0; i < 20;  i++)  {
      // este Bloco será executado 200 vezes
      }
      }
```

O ciclo WHILE … faz

O ciclo WHILE … faz

```
while ( resposta == 1)  {
        System.out.print( “ID = “ + userID[n]);
        n++;
        resposta = readInt( “Enter”);
        }
```

Qual é o menor número de vezes que o ciclo é executado?
e
Qual é o maior número de vezes que o ciclo é executado?

O ciclo DO … WHILE

O ciclo DO … WHILE

```
do    {
      System.out.print( “ID = “ + userID[n]);
      n++;
      resposta = readInt( “Enter”);
      }
      while (response == 1);
```

Qual é o menor número de vezes que o ciclo é executado?
e
Qual é o maior número de vezes que o ciclo é executado?

BREAK

```
for ( int 1 = 0;  i < maxID,  i++)   {
        if (userID[i] == targetID )  {
        index = i;
        break;
        }
}   // o programa salta para aqui a seguir ao break
```

A instrução break provoca uma saída da parte interior de um ciclo while, do, for ou switch.

CONTINUE

```
for ( int 1 = 0;  i < maxID,  i++)   {
        if (userID[i] !=  .-1) continue;
        System.out.print ( “UserID “ + i + “:” userID);
        }
```

A instrução continue só pode ser udada com os ciclo while, do, for .
Provoca o início da próxima iteração.