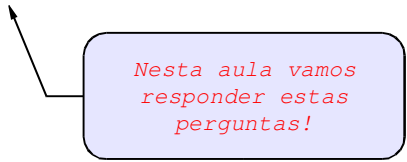


Tipos, Nomes e Expressões

Dalton Serey
© 2004 DSC/UFCG

Questões de hoje...

- Que coisas programas podem tratar?
- O que se pode fazer com elas?
- Como um programa as manipula e mantém?



*Nesta aula vamos
responder estas
perguntas!*

Coisas a manipular

- Todo programa manipula coisas
- Algumas são bastante simples
 - números
 - caracteres
 - palavras
- Mas há outras bastante complexas
 - livros de uma biblioteca
 - correntistas de um banco
 - clientes de uma empresa

Objetos são coisas

- Objetos são coisas em LPs OO
- São usados para representar coisas...
 - de natureza complexa
 - e do domínio do problema
- Todas as **coisas** que o programador **precise** criar devem ser definidas como **objetos**
 - exemplos: livros, contas, clientes, etc

Coisas simples...

- Coisas simples são providas pela LP
- Cabe ao projeto da LP decidir se são
 - valores primitivos (não objetos)
 - ou objetos
- Em Java, temos coisas simples e complexas
 - números, caracteres, valores verdade;
 - palavras, números de precisão, etc.

Tipos de coisas

- Tipos determinam **o que podemos fazer com as coisas** em nosso programa
- ... com números podemos
 - fazer aritmética e outras operações matemáticas
- ... com palavras podemos
 - fazer concatenações, identificar subpalavras, mudar trechos, identificar o tamanho, etc
- ...com livros podemos
 - determinar o autor, o número de páginas, tomá-lo emprestado de uma biblioteca, etc

Tipos de coisas

- Atentem: **tipos são contratos!**
- Tipos determinam quais **interações** são válidas **com coisas daquele tipo**
- Em Java, o **cumprimento do contrato** definido pelo tipo é “**vigiado**” pela linguagem
 - dizemos que a LP é fortemente tipada
- Por isso, qualquer coisa em Java tem um tipo!

© 2004 Dalton Serey DSC/UFCG

Tipos em Java

- Quatro tipos de inteiros
 - **int** (4 bytes)
 - de -2.147.483.648 a 2.147.483.647
 - **short** (2 bytes)
 - de -32.768 a 32.767
 - **long** (8 bytes)
 - de -9.223.372.036.854.775.808L a 9.223.372.036.854.775.807L
 - sufixo L obrigatório
 - **byte** (1 byte)
 - de -128 a 127

© 2004 Dalton Serey DSC/UFCG

Tipos em Java

- Dois tipos de ponto flutuante
 - **float** (4 bytes)
 - aproximadamente $\pm 3.40282347E+38F$
 - 6-7 dígitos significativos
 - sufixo F obrigatório
 - **double** (8 bytes)
 - aproximadamente $\pm 1.79769313486231570E+308$
 - 15 dígitos significativos
 - sufixo D opcional
 - seguem a especificação IEEE 754

© 2004 Dalton Serey DSC/UFCG

Tipos em Java

- Tipo caractere
 - **char**
 - Exemplos de literais: 'a', 'b', 'c', etc.
 - baseados em Unicode (literais: '\u0000' a '\uFFFF')
 - seqüências de controle estilo C
 - \b para backspace
 - \t para tabulação
 - \n para alimentação de linha (newline)
 - \r para retorno de carro (carriage return)
 - \" para aspas duplas
 - \' para apóstrofes
 - \\ para barra invertida

© 2004 Dalton Serey DSC/UFCG

Tipos em Java

- Tipo valor verdade
 - **boolean**
 - valores: **false** e **true**
 - utilizado em cláusulas com testes lógicos
 - manipulados com operadores lógicos
- Atenção programadores C
 - não há conversão para inteiros
 - nem valores inteiros têm valor lógico!

© 2004 Dalton Serey DSC/UFCG

Tipos em Java

- Tipo palavra
 - **String**
 - não é um tipo primitivo
 - palavras são objetos em Java
 - vêm pré-definidos
 - Exemplo de literais: “ooprog”, “ufcg”, etc

© 2004 Dalton Serey DSC/UFCG

Tipos em Java

- Há uma infinidade de outros tipos prontos
- Ao conjunto de tipos oferecidos para suportar a programação chamamos de API
 - Application Programming Interface
 - Há APIs específicas para vários domínios
 - Podemos desenvolver novas APIs
- Java oferece uma forma própria de documentação para as APIs
 - <http://java.sun.com/j2se/1.4.2/docs/api/>

© 2004 Dalton Serey DSC/UFG

Nomes

- Para identificar coisas em nossos programas, precisamos dar-lhes *nomes*
- Em tempo de interpretação, **um nome se refere a alguma coisa** de interesse
 - ao longo do tempo, a várias coisas
 - em um dado instante, a uma única coisa
- Exemplos
 - a, b, c, delta
 - idade, nome, correntista, livroAEmprestar
 - cliente, pessoaDeContato, etc.

© 2004 Dalton Serey DSC/UFG

Nomes

- Em Java, um nome só pode se referir a objetos de um único tipo
- Por isso, todo nome deve ser **declarado**
 - é isto que permite ao sistema verificar tipos
 - sintaxe: "nome_do_tipo nome"

```
int a, b, c;  
double delta;  
double x1, x2;  
String nome, mensagem;
```

© 2004 Dalton Serey DSC/UFG

Atribuições

- Já vimos como declarar nomes e como introduzir valores como literais...
- Questão: como *ligamos* os dois?
 - instrução de atribuição
 - notação: "**nome = valor**"

```
a = 5; b = 2; c = -16;  
nome = "José";
```

© 2004 Dalton Serey DSC/UFG

Expressões

- Expressões são a forma pela qual manipulamos as *coisas* em um programa
- Expressões são usadas para calcular uma *coisa* a partir de outras
 - $5 + 2 * 3$
 - $4 * a * c - (a + b) / 2$
 - "Oi " + nome " , como vai?"
- Para preservar os resultados combinamos

© 2004 Dalton Serey DSC/UFG

Expressões

- Podemos usar...
 - `System.out.println(a+b)`
 - `System.out.println("Oi " + nome)`
 - `if ( a+b > 0 ) return`
- ... ou guardar o resultado de uma expressão
 - `média = (a+b)/2`
 - `mensagem = "Oi " + nome`

© 2004 Dalton Serey DSC/UFG

Expressões com Objetos

- Para aplicar operações sobre objetos usamos a seguinte sintaxe:

- `nome.operação (parâmetros)`

```
char c;  
nome = "José";  
c = nome.charAt(0);
```

- Lembre: é o tipo do objeto que determina as possíveis operações
 - veja, por exemplo, a documentação de [String](#)

Expressões

- Algumas observações
 - toda expressão calcula (ou produz) coisas de um determinado **tipo**
 - diferente de nomes, cujo tipo é declarado, o tipo das expressões é **automaticamente inferido**...
- Mais adiante no curso, voltaremos ao tema.