Etec
Escola Técnica Estadual

Técnicas de Orientação a Objeto

A03 - Variáveis e tipos

Sumário

- Sumário
  - Tipos Primitivos em Java
  - Variáveis
  - Casting de tipos primitivos
  - Vetores
  - Ponteiros

Tipos Primitivos

- Tipos Primitivos em Java

Tipo	*Contém*	*Default*	*Tamanho*	*Faixa*
`boolean`	***true ou false***	`false`	***1 bit***	***NA***
`char`	***Caracter Unicode***	`\u0000`	***16 bits***	***\u0000 até \uFFFF***
`byte`	***Inteiro c/ Sinal***	`0`	***8 bits***	***-128 até 127***
`short`	***Inteiro c/ Sinal***	`0`	***16 bits***	***-32.768 até 32.767***
`int`	***Inteiro c/ Sinal***	`0`	***32 bits***	***-21bilhões até 21 bilhões***
`long`	***Inteiro c/ Sinal***	`0`	***64 bits***	***quatrilhões***
`float`	***Ponto Flutuante***	`0.0`	***32 bits***	***+/-1.4 E-45 até +/- 3.4E+38***
`double`	***Ponto Flutuante***	`0.0`	***64 bits***	***+/- 4.9 E-324 até +/- 1.79 E+308***

Variáveis em Java

- Uma **variável** é:
  - Um **apelido** pelo qual é acessado o conteúdo de um endereço de memória;
  - Sintaxe genérica:

```
<tipo><nome_variavel> [ = <valor>];
```

  - Em Java temos 2 tipos de variáveis:
    - Primitivas;
    - Objetos.

Variáveis em Java

- Exemplo de **variável primitiva:**
  - Só declaração : `int idade;`
  - Declaração e inicialização : `int idade = 10;`
- Exemplo de **variável objeto:**
  - Só declaração: `Pessoa p;`
  - Declaração e inicialização:

```
Pessoa p = new Pessoa();
```

Variáveis em Java

- A única variável objeto que pode ser inicializada sem o operador `new` é a do tipo `String`;
- `String` permite uma atribuição direta:

```
String nome = “Marcia”;
```

Dicas de Qualidade de Código

- Nomeação de Variáveis
  - Sempre descreva o que a variável realmente significa, ou seja, não use `x` e sim `somaSalarios`.
  - Utilize, em média, entre 8 a 10 caracteres:
    - Mais que isso é ilegível;
    - Menos que isso é pouco significativo.

Dicas de Qualidade de Código

- Nomes muito curtos geralmente estão dentro de um escopo pequeno, relacionados a questões sintáticas:
  - `i`, `j` e `k` para laços;
- Evite nomes como `aux` ou `temp`; no geral existe um significado melhor;
- Evite nomes invertidos: `formatoArquivo` e `arquivoFormato`

Variáveis em Java - Exercícios

Crie classes em Java para:

- Declarar uma variável do tipo `short`, atribuir o valor da soma 100+100 e imprimir seu resultado.
- Calcular os 10 primeiros termos da Sequência Fibonacci:
  - `1 1 2 3 5 8 13 21…`
  - $f_0 = 1$
  - $f_1 = 1$
  - $f_n = f_{n-1} + f_{n-2}$
  - `For` em Java : `for (i=1;i<1;i++){}`
  - Lembre-se de declarar `i`

Casting de Tipos Primitivos

- **Down-Casting** e **Up-Casting** são processos de conversão de tipos em alguma atribuições;
- **Up-Casting** (ocorre implicitamente):
  - `short` para `int`;
  - `int` para `long`;
  - `float` para `double`.

Casting de Tipos Primitivos

- Exemplos de **Up-Casting**:

```
short s = 1;
int   i = 10;
long  l = 100;
l = i; // int para long
i = s; // short para int
```

- E se fosse o contrário?

```
s = i; // int para short
i = l; // long para int
```

Casting de Tipos Primitivos

- **Down-Casting**: converter um tipo "maior"para um menor".
- Exemplos:

```
short s  = 1;
int   i  = 10;
long  l  = 100;
double d = 1000.1;
s = (short)i;
i = (int)l;
s = (short)d; //perde a parte decimal
```

Casting de Tipos Primitivos

- Pegadinhas:

```
float                    pi = 3.1416f;
float          piEngenheiro = 3;  //Ok up
double      pareceInteiro = 3.0; //Ok

i = piEngenheiro;            //Problemas
i = pareceInteiro;           //Problemas
```

Casting de Tipos Primitivos - Exercícios

- Sem testar compilando diga o que não vai e o que vai compilar.

```
short  s = 40000;
long   l = 100;
int    i = (short) l;
float  f = 1f;
double d = f+s;

i = s + (int)f;
```

Casting de Tipos Primitivos - Exercícios

- Converta `40000` para `short` teste novamente:

```
short  s = (short)40000;
long   l = 100;
int    i = (short) l;
float  f = 1f;
double d = f+s;

i = s + (int)f;
```

- O que aconteceu era o esperado? Onde apareceu o número negativo?

Vetores (Array) em Java

- Um **array** ou **vetor** é uma **lista de itens similares**, acessíveis via um **índice** que representa a posição do item que queremos obter.
- Um array deve sempre ter:
  - Um **nome**
  - Um **tipo** de dado
  - Um **tamanho**

Vetores (Array) em Java

- O tamanho de um **array** não pode ser alterado depois de sua criação, ou seja, sempre tem o **tamanho fixo**;
- Em Java, a primeira posição do vetor é sempre no índice de valor **0**. Assim, se criarmos um array de 5 posições, os respectivos índices de acesso são 0, 1, 2, 3 e 4.

Vetores (Array) em Java

- Como **declarar** um array:

```
int myArray[];
int[] myArray;  //também funciona
```

- Como **criar** um array:

```
myArray = new int[3];
```

- Como **declarar** e **criar** um array:

```
int myArray[] = new int[3];
int[] myArray = new int[3]; //também funciona
```

Vetores (Array) em Java

- Como **declarar**, **criar** e **inicializar** um array:

```
int myArray[] = new int[]{1,2,3};
int[] myArray = new int[]{1,2,3};
int[] myArray = {1,2,3};
```

- Como acessar uma **posição** em um array:

```
int myValue = myArray[2];
```

- Como obter o **tamanho** de um array:

```
myArray.length;
```

Vetores (Array) em Java

- Exemplo simples de manipulação de Array:

```
1  public class ArrayPessoas {
2    public static void main (String[] args) {
3      String[] pessoas = new String[]{"Luke", "Lea", "Hans", "Lando"};
4      for(int i=0; i< pessoas.length ; i++){
5        System.out.println(pessoas[i]);
6      }
7    }
8  }
```

- De que outras maneiras o array pessoa poderia ter sido inicializado?

Vetores (Array) em Java - Exercícios

- Para que serve o vetor de `String` chamado `args` passado ao método `public static void main` de toda classe principal em Java?
- Altere a classe **Soma** para que:
  - Some todos os números inteiros passados como parâmetro na linha de comando;
  - Deixe o tipo da variável `soma` como `short` e execute com valores que caibam dentro do limite da variável;
  - Teste agora valores que estourem o limite da variável `soma`;
- Dica: para converter uma `String` para um `short`, use `Short.parseShort("10");`

Vetores (Array) em Java - Exercícios

- Altere a Classe **Fibonacci** para guardar a sequência em um vetor.

Ponteiros em Java

- Ponteiros são variáveis que guardam **referências** a uma **posição de memória**;
- Tipos em Java:
  - Primitivos (`short`, `int`, `long`, `char`, `float`...);
  - Referência (`class`, `interface`, `array`...).
- Tipos de Referência só podem ser instanciados através de **ponteiros implícitos**.

Ponteiros em Java

- Tipos de Referência só podem ser criados com o comando `new`;
- Todo gerenciamento de memória é feito **automaticamente** em Java, ou seja, não é possível:
  - Realizar aritmética de ponteiros;
  - Liberar memória.

Ponteiros em Java

- A liberação de memória é feita pelo **Coletor de Lixo** (Garbage Collector):
  - A linguagem torna-se mais **segura**;
  - Consome **recursos**;
  - Porém o Coletor de Lixo é **não-determinístico**. É difícil prever ou forçar o momento de sua atuação.