Linguagens de Programação II

Aula 02 – Orientação à Objetos com Java

Prof. Marc Antonio Vieira de Queiroz

Centro Universitário Filadélfia
Ciência da Computação/Sistemas de Informação

Conteúdo

- Exercícios de Revisão
- Variáveis
  - Primitivas
  - Referências

- Exemplos
- Exercícios

Exercício de revisão 01

- Codifique cada um dos programas JAVA, verificando se podem ser executados. Caso não possam, identifique os possíveis erros e os corrijam.

```java
public class TapeDeck {
    boolean canRecord = false;

    void playTape() {
        System.out.println("tape playing");
    }

    void recordTape() {
        System.out.println("tape recording");
    }
}
```

```java
public class TapeDeckTestDrive {

    public static void main(String []args) {

        t.canRecord = true;
        t.playTape();

        if(t.cannotRecord == true){
            t.recordTape();
        }
    }

}
```

Exercício de revisão 02

- Idem ao ex 01.

```
class DVDPlayer{

    boolean canRecord = false;

    void recordDVD(){
        System.out.println("DVD recording");
    }
}
```

```
class DVDPlayerTestDrive{

    public static void main (String [] args){

        DVDPlayer d = new DVDPlayer();
        d.canRecord = true;
        d.playDVD();

        if(d.canRecord() == true){
            d.recordDVD();
        }
    }
}
```

Variáveis

- Já as usamos:
  - Estado de objeto (variáveis de instância)
  - Variáveis locais (dentro de um método)

- Iremos usá-las:
  - Argumentos
  - Tipos de retorno

Declaração de Variáveis

- Fortemente **tipada**

Forma esperada

recebe

Valor passado

Não funciona.

Declaração de Variáveis

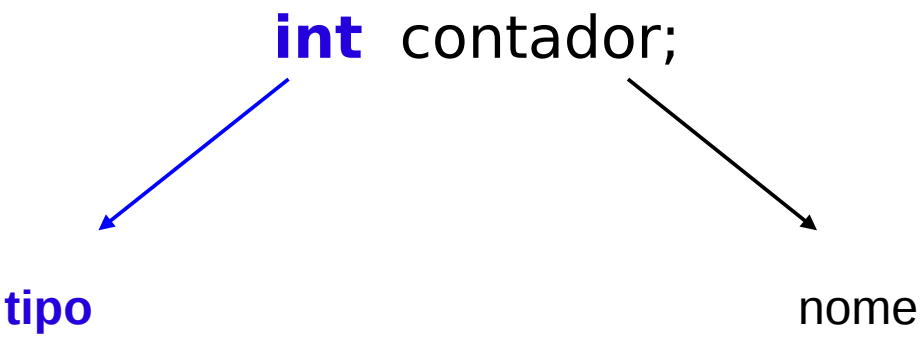
- Em termos de código ...

```
Coelho saltador = new Girafa();
```
→ funciona?

```
int contador = 245.768;
```
→ funciona?

Declaração de Variáveis

- Forma ...



Declaração de Variáveis

- tamanho

long 64 bits | **int** 32 bits | **short** 16 bits | **byte** 8 bits

double 64 bits | **float** 32 bits

Tipos Básicos

Tipo	Valor Mínimo	Valor Máximo	Tamanho
byte	-128	127	1 byte
short	-32.768	32.767	2 bytes
int	-2.147.483.648	2.147.483.647	4 bytes
long	-9.223.372.036.854.775.808	9.223.372.036.854.775.808	8 bytes
float	-3,402823E+38	3,402823E+38	4 bytes
double	-1,79769313486232E+308	1,79769313486232E+308	8 bytes
boolean	**false**	**true**	
char	caracteres individuais, letras, números, pontuação e símbolos		

Exemplo 01

- Declarações primitivas com atribuições

```
1   int x;
2   x = 234;
3   byte b = 89;
4   boolean isFun = true;
5   double d = 3456.78;
6   char c = 'f';
7   int z = x;
8   boolean isPunkRock;
9   isPunkRock = false;
10  boolean powerOn = isFun;
11  long big = 3456789;
12  float f = 32.5f;
```

Porque uso **.f**?

Exercício 01

- Quais instruções são válidas?

```
1   int x = 34.5;
2   boolean boo = x;
3   int g = 17;
4   int y = g;
5   y = y + 10;
6   short s;
7   s = y;
8   byte b = 3;
9   byte v = b;
10  short n = 12;
11  v = n;
12  byte k = 128;
```

Objetos

- E a manipulação de objetos?
- Como é feita?

Referências

Valores
(bits)

long 64 bits

int 32 bits

short 16 bits

byte 8 bits

Valores
(outro valor)
Mas que é um "**controle**"

Uma maneira de chegar ao objeto

Referências

```
Dog d = new Dog();
d.bark( );
```

Considere isso,

como se fosse isso

“ use o que está **antes** do ponto para me trazer o que está **depois** do ponto”

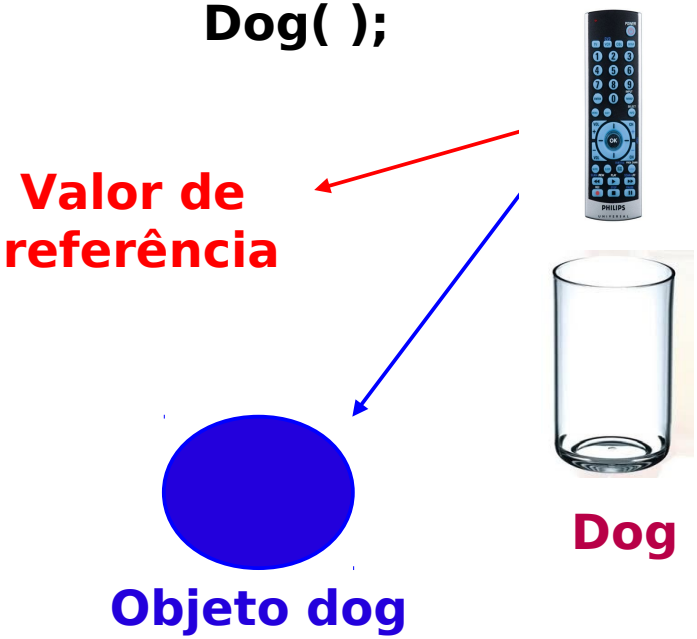
Diferença

Variável primitiva

```
byte x = 7;
```

Variável de referência

```
Dog myDog = new Dog( );
```



Etapas de declaração

1 → “Declare uma variável de referência”

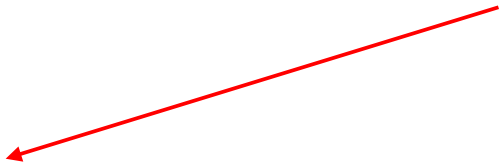
Dog myDog = new Dog();

- Solicita à JVM espaço para uma variável de referência
- Nomeia a variável como myDog
- O tipo da variável é Dog

Etapas de declaração

2 → “Crie um objeto”

Dog myDog = new Dog();

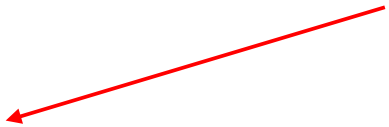


- Solicita à JVM alocar espaço para um novo objeto Dog no acervo (heap)

Etapas de declaração

3 → “Vincule o objeto e a referência”

Dog myDog = new Dog();

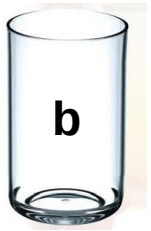


- Atribui o novo objeto à variável de referência myDog
- Em outras palavras, “***programa o controle remoto***”

Exemplo 02

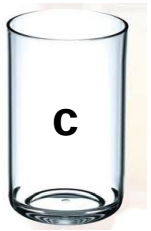
- O que ocorre?

```
1  Book b = new Book();
2  Book c = new Book();
3
4  Book d = c;
5
6  c = b;
```



book

?

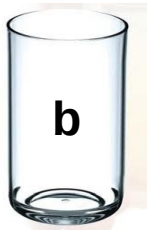


book

Exemplo 03

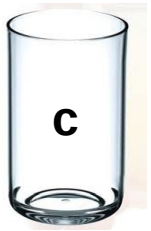
- O que ocorre?

```
1  Book b = new Book();
2  Book c = new Book();
3
4  b = c;
5
6  c = null;
```



book

?



book

Exercício 02

- Qual o valor atual de pets[2] ?
- Que código faria pets[3] referenciar um dos dois objetos Dog existentes?

```
1   Dog[] pets;
2   pets = new Dog[7];
3
4   pets[0] = new Dog();
5   pets[1] = new Dog();
```

Exercício 03

- Existe erro de compilação? Se houver, como corrigir para que o programa execute?

```
1  class Books{
2      String title;
3      String author;
4  }
```

```
1  class BookTestDrive{
2
3      public static void main(String [] args){
4          Books[] myBooks = new Books[3];
5          int  = 0;
6
7          myBooks[0].title = "The Grapes of Java";
8          myBooks[1].title = "The Java Gatsby";
9          myBooks[2].title = "The Java Cookbook";
10         myBooks[0].author = "bob";
11         myBooks[1].author = "sue";
12         myBooks[2].author = "ian";
13
14         while( x < 3){
15             System.out.print(myBooks[x].title);
16             System.out.print(" by ");
17             System.out.println(myBoooks[x].author);
18         }
19     }
20 }
```

Exercício 04

- Idem ex 03.

```
1  class Hobbits{
2
3      String name;
4
5      public static void main(String [] args){
6
7          Hobbits[] h = new Hobbits[3];
8          int z = 0;
9
10         while(z < 4){
11             z = z + 1;
12             h[z] = new Hobbits();
13             h[z].name = "bilbo";
14
15             if(z == 1){
16                 h[z].name = "frodo";
17             }
18             if(z == 2){
19                 h[z].name = "sam";
20             }
21
22             System.out.print(h[z].name + "is a ");
23             System.out.print("good Hobbit name");
24         }
25     }
26 }
```

Continua …