Unidade 0 - Nivelamento - Ponteiros (C/C++) e Referências (Java)

Instituto de Ciências Exatas e Informática
Departamento de Ciência da Computação

Exercício Resolvido (1)

- O que significa cada um dos trechos de código abaixo?

```
int [ ] vet
```

R: Criação do ponteiro vet

```
= new int [5]
```

R: Alocação de 5 espaços de memória para armazenar números inteiros

```
int [ ]  vet = new  int [5];
```

R: Direcionamento do ponteiro vet para a área de memória alocada

Exercício Resolvido (2)

- Explique o que o programa abaixo imprime na tela

```
class Ponteiro01Array {

    public static void main (String[] args) {
        int[] vet = new int [5];
        escrever(vet);
        R: Impressão do endereço da 1a posição do array acima
        vet = new int [5];
        escrever(vet);
    }
        R: Endereço da 1a posição do “novo” array

}
```

Passagem de Parâmetros em Java e C

- Java e C têm somente a passagem de parâmetros por valor

- O valor de uma variável passada por parâmetro nunca é modificada quando passada por parâmetro

- Na verdade, o método chamado cria uma nova variável cujo valor inicial é o passado como parâmetro

Exercício (1)

- Faça o quadro de memória do programa abaixo

```
class Ponteiro02PassagemTipoPrimitivo {
    public static void passagemDeTipoPrimitivo(int a){
        escrever("a: " + a);
        a = 10;
        escrever("a: " + a);
    }
    public static void main(String[] args) {
        int x = 5;
        escrever("x: " + x);
        passagemDeTipoPrimitivo(x);
        escrever("x: " + x);
    }
}
```

Passagem de *Arrays* como Parâmetros em Java

- Um *array* é um ponteiro, ponteiro é uma variável que armazena endereço de memória

- Passando um *array* como parâmetro, passamos o endereço de memória da área referenciada (na verdade, o da 1ª posição)

- Tal endereço é o mesmo antes/depois do método chamado, contudo, esse pode alterar o conteúdo daquela área

Exercício Resolvido (3)

- Faça o quadro de memória e mostre a saída na tela

```
class Ponteiro03PassagemArray {
      public static void passagemDeArray(int[ ] b){
            for (int i = 0; i < 5; i++){
                  b[i] *= 5;        escrever("b[" + i + "]: " + b[i]);
            }
            b = new int [5];
            for (int i = 0; i < 5; i++){
                  b[i] = i;         escrever("b[" + i + "]: " + b[i]);
      }     }
      public static void main(String[] args) {
            int [] y = new int [5];
            for (int i = 0; i < 5; i++){
                  y[i] = i;         escrever("y[" + i + "]: " + y[i]);
            }
            passagemDeArray(y);
            for (int i = 0; i < 5; i++){
                  escrever("y[" + i + "]: " + y[i]);
}     }     }
```

Memória		
Y:	7Eh	75h
		76h
0		77h
1		78h
2		79h
3		7Ah
4		7Bh
		7Ch
		7Dh
0		7Eh
5		7Fh
10		80h
15		81h
20		82h
b:	77h	83h

Tela
y[0] : 0
y[1] : 5
y[2] : 10
y[3] : 15
y[4] : 20

Mais Ponteiro

- Seja a classe abaixo ...

```
class Cliente {
      private int codigo;
      private String nome;
      public Cliente (){
            this.codigo = 0;                this.nome = “”;
      }
      public Cliente (int codigo, String nome){
            this.codigo = codigo;    this.nome = nome;
      }
      public int getCodigo(){                    return codigo;          }
      public void setCodigo(int codigo){     this.codigo = codigo;   }
      public String getNome(){                   return nome;            }
      public void setNome(String nome){    this.nome = nome;       }
}
```

Mais Ponteiro

- ... o que significa cada um dos trechos de código abaixo?

```
Cliente c;
```

R: Criação de um ponteiro chamado c

```
= new Cliente ();
```

R: Criação um objeto do tipo Cliente

```
Cliente c = new Cliente ();
```

R: Direcionamento do ponteiro c para o objeto criado

Exercício Resolvido (4)

- Faça o quadro de memória do programa abaixo

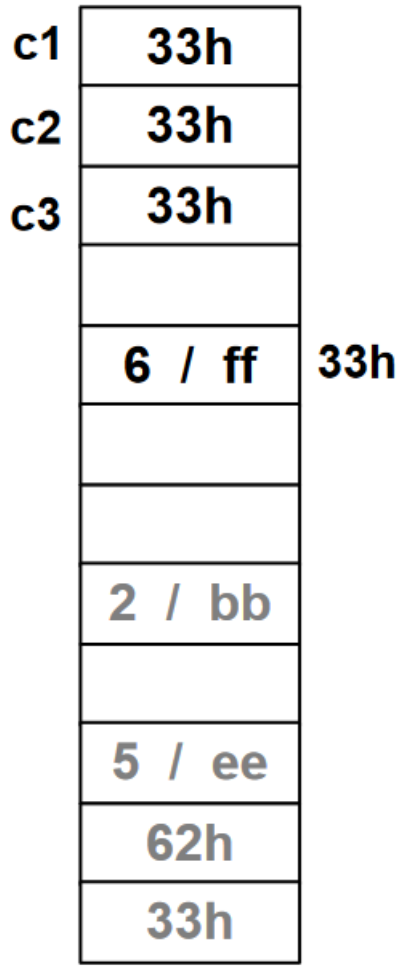
```
class Ponteiro04Objeto {
    public static void main (String[] args){
        Cliente c1 = null, c2 = null, c3 = null;
        escrever("ADDRs:\nc1(" + c1 + ")\nc2(" + c2 + ")\nc3(" + c3 + ")");
        c1 = new Cliente(1, "aa");          c2 = c1;     c3 = new Cliente(2, "bb");
        escrever("ADDRs:\nc1(" + c1 + ")\nc2(" + c2 + ")\nc3(" + c3 + ")");
        c2.setCodigo(3);
        escrever("ATRIBUTOs:");
        escrever("c1(" + c1.getCodigo() + " / " + c1.getNome() + ")");
        escrever("c2(" + c2.getCodigo() + " / " + c2.getNome() + ")");
        escrever("c3(" + c3.getCodigo() + " / " + c3.getNome() + ")");
}     }
```

Tela
ADDRs:
c1(null)
c2(null)
c3(null)
ADDRs:
c1(7Ah)
c2(7Ah)
c3(A5h)
ATRIBUTOs:
c1(3/aa)
c2(3/aa)
c3(2/bb)

Exercício Resolvido (5)

- Faça o quadro de memória do programa abaixo

```
class Ponteiro05PassagemObjeto {
    public static Cliente setar2(Cliente y){
        y.setCodigo(6);    y.setNome("ff");
        return y;
    }
    public static void setar1(Cliente x){
        x.setCodigo(4);    x.setNome("dd");    x = new Cliente (5, "ee");
    }
    public static void main (String[] args){
        Cliente c1 = new Cliente(1, "aa"), c2 = null; c3 = new Cliente(2, "bb");
       c2 = c1;
        setar1(c1);
        c3 = setar2(c2);
}     }
```



Exercício Resolvido (6)

- Na verdade, no comando c2 = c1 do exercício anterior, o programador gostaria que os atributos do objeto apontado por c2 fossem iguais aos do objeto apontado por c1, contudo, apontando para objetos distintos. Como podemos ajudá-lo?

Na classe Cliente, é possível de se elaborar o seguinte método clone:

```
class Ponteiro06PassagemObjetoClone {
public static Cliente setar2(Cliente y){
y.setCodigo(6); y.setNome(“ff”);
return y;
}
public static void setar1(Cliente x){
x.setCodigo(4); x.setNome(“dd”); x = new Cliente (5, “ee”);
}
public static void main (String[] args){
Cliente c1 = new Cliente(1, “aa”), c2 = null; c3 = new Cliente(2, “bb”);
c2 = c1.clone();
setar1(c1);
c3 = setar2(c2);
} }
```

Exercício Resolvido (8)

- Mostre a alteração anterior na classe Ponteiro04Objeto

```
class Ponteiro04Objeto {
public static void main (String[] args){
Cliente c1 = null, c2 = null, c3 = null;
escrever("ADDRs:\nc1(" + c1 + ")\nc2(" + c2 + ")\nc3(" + c3 + ")");
c1 = new Cliente(1, "aa"); c2 = c1; c3 = new Cliente(2, "bb");
escrever("ADDRs:\nc1(" + c1 + ")\nc2(" + c2 + ")\nc3(" + c3 + ")");
c2.setCodigo(3);
escrever(“ATRIBUTOs:”);
escrever(“c1(” + c1.getCodigo() + “ / ” + c1.getNome() + “)”);
escrever(“c2(” + c2.getCodigo() + “ / ” + c2.getNome() + “)”);
escrever(“c3(” + c3.getCodigo() + “ / ” + c3.getNome() + “)”);
} }
```

Exercício Resolvido (8)

- Mostre a alteração anterior na classe Ponteiro04Objeto

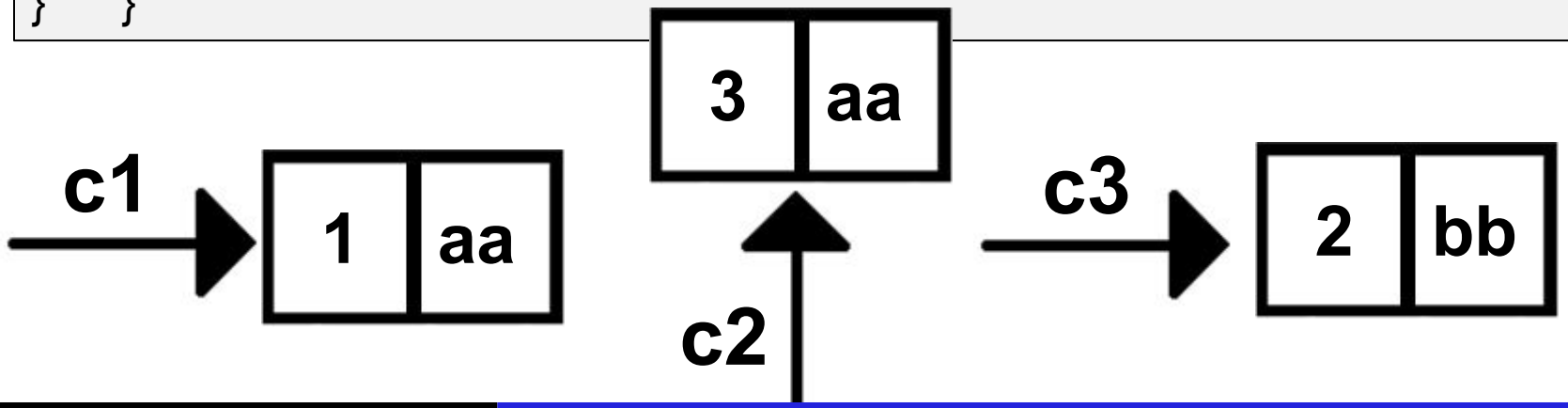
```
class Ponteiro04Objeto {
    public static void main (String[] args){
        Cliente c1 = null, c2 = null, c3 = null;
        escrever("ADDRs:\nc1(" + c1 + ")\nc2(" + c2 + ")\nc3(" + c3 + ")");
        c1 = new Cliente(1, "aa");          c2 = c1;     c3 = new Cliente(2, "bb");
        escrever("ADDRs:\nc1(" + c1 + ")\nc2(" + c2 + ")\nc3(" + c3 + ")");
        c2.setCodigo(3);
        escrever("ATRIBUTOs:");
        escrever("c1(" + c1.getCodigo() + " / " + c1.getNome() + ")");
        escrever("c2(" + c2.getCodigo() + " / " + c2.getNome() + ")");
        escrever("c3(" + c3.getCodigo() + " / " + c3.getNome() + ")");
}     }
```

Exercício Resolvido (8)

- Mostre a alteração anterior na classe Ponteiro04Objeto

```
class Ponteiro07ObjetoClone {
    public static void main (String[] args){
        Cliente c1 = null, c2 = null, c3 = null;
        escrever("ADDRs:\nc1(" + c1 + ")\nc2(" + c2 + ")\nc3(" + c3 + ")");
        c1 = new Cliente(1, "aa"); c2 = c1.clone(); c3 = new Cliente(2, "bb");
        escrever("ADDRs:\nc1(" + c1 + ")\nc2(" + c2 + ")\nc3(" + c3 + ")");
        c2.setCodigo(3);
        escrever("ATRIBUTOs:");
        escrever("c1(" + c1.getCodigo() + " / " + c1.getNome() + ")");
        escrever("c2(" + c2.getCodigo() + " / " + c2.getNome() + ")");
        escrever("c3(" + c3.getCodigo() + " / " + c3.getNome() + ")");
}    }
```

Tela
ADDRs:
c1(null)
c2(null)
c3(null)
ADDRs:
c1(7Ah)
c2(9Ah)
c3(A5h)
ATRIBUTOs:
c1(1/aa)
c2(3/aa)
c3(2/bb)



Exercício (2)

- Mostre o quadro de memória para o programa abaixo

```
class  Ponteiro08Objeto {
    public static void main (String[] args){
      Cliente c1 = new Cliente(1, "aa");
      Cliente vet[] = new Cliente [5];
      sop(c1 + "/" + c1.getCodigo() + "/" + c1.getNome());
      for (int i = 0; i < vet.length; i++){
         vet[i] = c1.clone();
         System.out.println(vet[i] + "/" + vet[i].getCodigo() + "/" + vet[i].getNome());
      }
   }
}
```

Exercício Resolvido (9)

- Um estudante de Algoritmos e Estruturas de Dados (em JAVA) implementou uma classe Hora, cujo construtor recebe e armazena uma hora, minuto e segundo. O que acontece se a classe X abaixo for colocada na mesma pasta que a classe Hora?

```
class X {
   public static void main (String[] args){
      Hora h1 = new Hora(12, 30, 30);
      Hora h2 = new Hora(12, 30, 30);
      if (h1 == h2)
         System.out.println("Identicos!");
      else
         System.out.println("Diferentes!");
   }
}
```

A) Escreve na tela "Identicos!".

B) Escreve na tela "Diferentes".

C) Erro de compilação.

D) Erro de execução na linha do if.

E) Erro de execução na declaração objetos.

Exercício Resolvido (10)

- Seja a classe X abaixo e a Animal implementada e não mostrada, avalie as afirmações listadas a seguir.

```
class X {
  public static void main (String[] args){
     Animal a = new Animal ("Cao", 32, 'a');
     Animal b = new Animal ("Cao", 'x');
     Animal c = b;
     c.nome = "Gato";
     System.out.println(b.nome);
     c.setIdade(45);
} }
```

I – Possivelmente, a Classe Animal tem três ou mais atributos. Além disso, no construtor com três parâmetros, o atributo que recebe valor do primeiro parâmetro pode ser do tipo String e os que recebem os outros dois podem ser do tipo int.

II - O comando System.out.println(b.nome) imprime a palavra "Gato".

III - A classe Animal deve ter um atributo idade e esse será obrigatoriamente privado.

IV - Na classe animal o atributo nome tem que ser estático.

É correto apenas o que se afirma em: A) I e II. B) II e III. C) III e IV. D) I, II e III.