# Laboratório de Programação 2
# 2015.2

Lista de exercícios

25 de Setembro de 2015

## Exercícios de introdução à Java

### Exercício 1

Quais dos identificadores abaixo podem ser usados como nomes de classes, atributos, métodos e variáveis em Java? Quais não podem, e por quê?

**A.** quatro

**B.** for

**C.** from

**D.** 4

**E.** FOR

### Exercício 2

Quais dos identificadores abaixo podem ser usados como nomes de classes, atributos, métodos e variáveis em Java? Quais não podem, e por quê?

**A.** dia&noite

**B.** diaENoite

**C.** dia & noite

**D.** dia E noite

**E.** dia_e_noite

### Exercício 3

Quais dos identificadores abaixo podem ser usados como nomes de classes, atributos, métodos e variáveis em Java? Quais não podem, e por quê?

**A.** contador

**B.** 1contador

**C.** contador de linhas

**D.** Contador

**E.** count

## Exercício 4

Identifique e explique o(s) erro(s) na classe abaixo.

```
1 class DemoImpressao
2 {
3   int a, b = 0.0;
4   main(String args)
5   {
6     a = 7.0;
7     b = 2.0;
8
9     if (a > b)
10      System.out.println("a-b="-(a-b));
11    else
12      System.out.println("a+b="+(a+b));
13  }
14 }
```

## Exercício 5

Escreva um programa que lê do usuário o raio de um círculo como um inteiro e imprime o diâmetro do círculo, circunferência e área, utilizando o valor do ponto flutuante para PI. **Dica**: Utilize a constante *Math.PI* e as seguintes fórmulas:

- diâmetro = 2r
- circunferência = $2\pi r$
- área = $\pi r^2$

## Exercício 6

Escreva um programa que atribua valores aleatórios a um array inteiro de 100 elementos e exibe a média de todos os valores, os números menores e maiores.

## Exercício 7

Escreva um programa que calcule os quadrados e cubos dos números de 0 a 10 e imprime os valores resultantes em um formato de tabela.

## Exercício 8

Escreva um aplicativo que calcule os fatorias de 1 a 20. **Dica**: utilize o tipo **long**.

## Exercício 9

Escreva um programa que calcule o valor de $\pi$ a partir de uma série infinita:

$$\pi = 4 - \frac{4}{3} + \frac{4}{5} - \frac{4}{7} + \frac{4}{9} - \frac{4}{11} + \cdots$$

Imprima uma tabela que mostra o valor de $\pi$ aproximado calculando os 200.000 primeiros termos da série. Quantos termos você tem de utilizar antes de obter um valor que comece com 3.14159?

## Exercício 10

Escreva um programa que exibe os equivalentes inteiros de algumas letras maiúsculas, minúsculas, dígitos e símbolos especiais. Exiba os equivalentes inteiros do seguinte: A B C a b c 0 1 2 $ * + / e o caracter em branco. **Dica**: Utilize a coerção de caracter para inteiro.

## Exercício 11

Escreva o programa do **jogo da velha**, de forma que o usuário possa jogar contra o computador. O tabuleiro deverá ser representado como uma matriz quadrada $N \times N$ e apresentado em forma textual no console. O computador somente realizará jogadas aleatórias. Ao término de cada partida, o programa deverá perguntar ao jogador se ele deseja jogar mais uma vez.

### Exercício 12

Crie um aplicativo de calculadora de IMC que lê o peso do usuário e a altura e, então, calcula e exibe o índice de massa corporal do usuário. Além disso, o aplicativo deverá avaliar o IMC do usuário segundo os critérios abaixo:

- Abaixo de 18.5: abaixo do peso normal
- Entre 18.5 e 24.9: peso normal
- Entre 25 e 29.9: acima do peso normal
- Acima de 30: obeso

### Exercício 13

Escreva um programa que leia o nome completo do usuário e apresente as primeiras letras do nome em maiúsculo.

### Exercício 14

Escreva um programa que leia um texto qualquer (parágrafo, frase, etc.) e "quebra" o texto segundo os espaços em branco existentes entre os caracteres. Ou seja, o programa deverá imprimir cada palavra do texto original em uma linha nova.

### Exercício 15

Escreva um programa que leia o nome de 10 pessoas, ordena-os em ordem crescente e imprima o resultado no console

### Exercício 16

Escreva um programa que leia uma frase qualquer e apresenta-na invertida no console.

### Exercício 17

Escreva um programa que leia uma frase qualquer e apresenta-na criptografada no console. A criptografia a ser aplicada é bastante simples: percorra a frase caracter a caracter, pegando o valor inteiro do caracter corrente. Some um valor $x$ qualquer ao valor do caracter em questão. Em seguida, converta o valor de volta para caracter e o insira novamente na mesma posição da frase.

### Exercício 18

Escreva um programa que calcule o determinante de uma matriz $3 \times 3$ informada pelo usuário.

### Exercício 19

Um número primo é qualquer inteiro maior que 1 que é igualmente divisível apenas por si mesmo e por 1. O **crivo de Eratóstenes** é um método de encontrar números primos. Ele opera como segue:

**a)** Crie um array boolean com todos os elementos inicializados como *true*. Os elementos do array com índices primos permanecerão *true*. Todos os outros elementos do array por fim são configurados como *false*.

**b)** A partir do índice de array 2, determine se um dado elemento é *true*. Se for, faça um loop pelo restante do array e configure como *false* cada elemento cujo índice é um múltiplo do índice para o elemento com valor *true*. Então continue o processo com o próximo elemento com valor *true*. Para o índice de array 2, todos os elementos além do elemento 2 no array que tiverem índices que são múltiplos de 2 (índices 4, 6, 8, 10, etc.) serão configurados como *false*; para o índice de array 3, todos os elementos além do elemento 3 no array que tiverem índices que são múltiplos de 3 (índices 6, 9, 12, 15, etc.) serão configurados como *false*; e assim por diante.

Quando esse processo for concluído, os elementos de array que ainda forem *true* indicam que o índice é um número primo. Esses índices podem ser exibidos. Escreva um aplicativo que utiliza um array de 1000 elementos para determinar e exibir os números primos entre 2 e 999. Ignore elementos de array 0 e 1.

### Exercício 20

Utilize um array unidimensional para resolver o seguinte problema: escreva um programa que lê cinco números. À medida que cada número é lido, só o exiba se ele não for uma duplicata de um número já lido. Cuide de tratar o "pior caso", em que todos os cinco números são diferentes. Utilize o menor array possível para resolver esse problema. Exiba o conjunto completo de valores únicos inseridos depois que o usuário inserir cada valor novo.

## Exercícios de Programação Orientada a Objetos

### Exercício 1

Todos os nomes de classe em Java devem obrigatoriamente iniciar com uma letra maiúscula? Justifique.

### Exercício 2

Todos as classes em Java devem implementar métodos set & get para cada um de seus atributos? Justifique.

### Exercício 3

Identifique e explique o(s) erro(s) na classe abaixo.

```
1 class DemoImpressao
2 {
3   int a, b = 0.0;
4   main(String args)
5   {
6     a = 7.0;
7     b = 2.0;
8
9     if (a > b)
10      System.out.println("a-b="-(a-b));
11    else
12      System.out.println("a+b="+(a+b));
13  }
14 }
```

## Exercício 4

Identifique e explique o(s) erro(s) na classe abaixo.

```
1 class DoisValores
2 {
3   int valor1, valor2;
4
5   int maior()
6   {
7     if(valor1 > valor2)
8       return true;
9     else
10      return false;
11  }
12
13  void menor()
14  {
15    if(valor1 < valor2)
16      return valor1;
17    else
18      return valor2;
19  }
20 }
```

## Exercício 5

Identifique e explique o(s) erro(s) na classe abaixo.

```
1 class FaceDoDado {
2   int 1,2,3,4,5,6;
3
4   void 1() {
5     System.out.println(1);
6   }
7
8   void 2() {
9     System.out.println(2);
10  }
11
12  void 3() {
13    System.out.println(3);
14  }
15
16  void 4() {
17    System.out.println(4);
18  }
19
20  void 5() {
21    System.out.println(5);
22  }
23
24  void 6() {
25    System.out.println(6);
26  }
27 }
```

## Exercício 6

Identifique e explique o(s) erro(s) na classe abaixo.

```
1 class NumeroComplexo {
2   float real, imaginario;
3
4   float valor()
5   {
6     return real, imaginario;
7   }
8 }
```

## Exercício 7

Identifique e explique o(s) erro(s) na classe abaixo.

```
1 class Amplitude {
2   double val1, val2, val3;
3
4   double amplitude()
5   {
6     double amplitude2()
7     {
8       return val1 - val2;
9     }
10    return amplitude2() - val3;
11  }
12 }
```

## Exercício 8

Identifique e explique o(s) erro(s) na classe abaixo.

```
1 class Registro De Eleitor {
2   int tituloDeEleitor;
3   String nome;
4   short zonaEleitoral;
5 }
```

## Exercício 9

Escreva uma classe *Contador* que encapsule um valor usado para contagem de itens ou eventos. Essa classe deve esconder o valor encapsulado de programadores-usuários, fazendo com que o acesso ao valor seja feito somente através de métodos que devem zerar, incrementar e imprimir o valor do contador.

## Exercício 10

Crie uma classe *Ponto3D* para representar um ponto no espaço cartesiano de três dimensões. Que atributos e métodos essa classe deve ter?

## Exercício 11

Crie uma classe Linha3D para representar uma linha, unida por dois pontos no espaço cartesiano de três dimensões, usando duas instâncias da classe Ponto3D, criada no exercício anterior.

## Exercício 12

Escreva dois construtores para a classe *Contador* (exercício 9), um que não receba argumentos e considere que o contador começa a contar a partir do zero, e outro que aceita um valor inicial para contagem.

## Exercício 13

Identifique e explique o(s) erro(s) na classe abaixo.

```
1 public class Data {
2   private byte dia,mes;
3   private short ano;
4
5   private Data(byte dd, byte mm, short aa) {
6     dia = dd; mes = mm; ano = aa;
7   }
8 }
```

## Exercício 14

Identifique e explique o(s) erro(s) na classe abaixo.

```
1 public class Point2D{
2   private double x,y;
3
4   public Ponto2D(double _x, double _y) {
5     x = _x; y = _y;
6   }
7
8   public Ponto2D(double coord1, double coord2)  {
9     x = coord1; y = coord2;
10  }
11 }
```

## Exercício 15

Identifique e explique o(s) erro(s) na classe abaixo.

```
1 public class Media{
2
3   public int Media(int a, int b)
4     return (a+b)/2;
5   }
6
7   public double Media(int a, int b)
8     return (a+b)/2;
9   }
10
11 }
```

## Exercício 16

Escreva a classe *ConversaoDeUnidadesDeVolume* com métodos estáticos para conversão das unidades de volume segundo a lista abaixo:

- 1 litro = 1000 centímetros cúbicos
- 1 metro cúbico = 1000 litros
- 1 metro cúbico = 35.32 pés cúbicos
- 1 galão americano = 231 polegadas cúbicas
- 1 galão americano = 3785 litros

## Exercício 17

O volume de uma piscina olímpica é de 1890 metros cúbicos. Usando a classe criada no exercício anterior, escreva uma programa que mostre qual é o volume de uma piscina olímpica em litros, pés cúbicos e centímetros cúbicos. Escreva métodos adicionais para a classe *ConversaoDeUnidadesDeVolume*, se necessário.

# Exercícios de Herança e Polimorfismo

## Exercício 1

Identifique e explique o(s) erro(s) na classe abaixo.

```
1 public class Ponto2D_V2{
2   private double x, y;
3
4   public Ponto2D_V2(double _x, double _y) {
5     x = _x;
6     y = _y;
7   }
8 }
9
10 public class Ponto3D_V2 extends Ponto2D_V2
11 {
12   private double z;
13
14   public Ponto3D_V2(double _x, double _y, double _z){
15     x = _x;
16     y = _y;
17     z = _z;
18   }
19 }
```

## Exercício 2

Identifique e explique o(s) erro(s) na classe abaixo.

```
1 public class DataParaAgenda{
2   private byte dia,mes;
3   private short ano;
4
5   public DataParaAgenda(byte d, byte m, short a) {
6     dia = d;
7     mes = m;
8     ano = a;
9   }
10
11  public String toString() {
12    return dia + "/" + mes + "/" + ano;
13  }
14 }
15
16 public class Hora{
17   private byte hor, min, seg;
18
19   public Hora(byte h, byte m, byte s) {
20     hor = h;
21     min = m;
22     seg = s;
23   }
24
25   public String toString() {
26     return hora + ":" + min + ":" + seg;
27   }
28 }
29
30 public class DataHoraParaAgenda extends DataParaAgenda {
31   private Hora hora;
32   public DataHora(byte d, byte m, short a, byte hor, byte min, byte seg) {
33     super(d,m,a);
34     hora = new Hora(hor,min,seg);
35   }
36
37   public String toString() {
38     return super.toString() + " " + hora.toString();
39   }
40 }
```

## Exercício 3

Identifique e explique o(s) erro(s) na classe abaixo.

```
1 public class DataHora extends Data,Hora{
2
3   public DataHora(byte d, byte m, short a, byte hor, byte min, byte seg) {
4     super(d,m,a);
5     super(hor,min,seg);
6   }
7
8   public String toString() {
9     return super.toString() + " " + super.toString();
10  }
11
12 }
```

## Exercício 4

Explique com suas palavras por que uma classe abstrata não pode ser instanciada.

## Exercício 5

Identifique e explique o(s) erro(s) na classe abaixo.

```
1 public class Produto{
2   private String identificacao;
3   private double custoDeFabricacao;
4
5   Produto(String i, double c) {
6     identificacao = i;
7     custoDeFabricacao = c;
8   }
9
10  abstract public String toString();
11  abstract public void novoCusto(double nc);
12 }
```

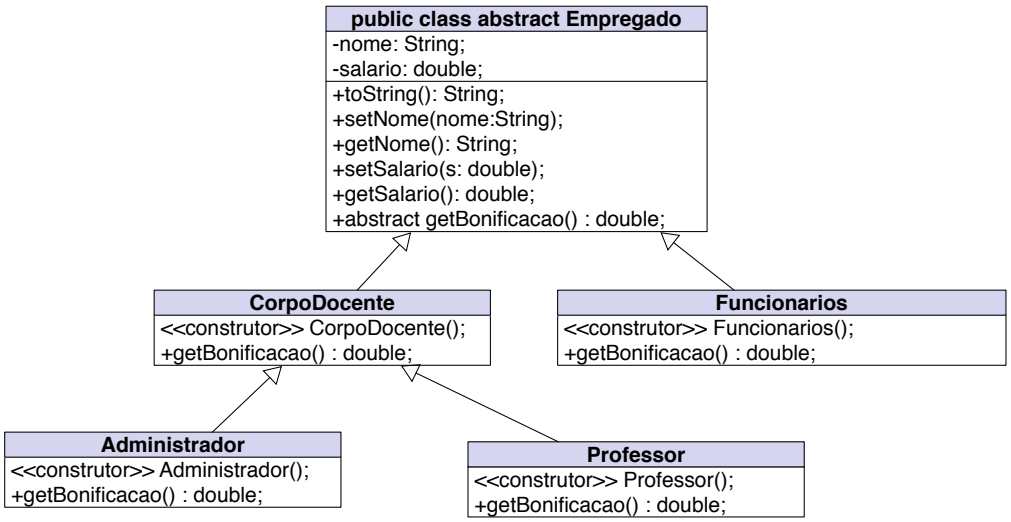
## Exercício 6

Identifique e explique o(s) erro(s) na classe abaixo.

```
1 public abstract class Dispositivo{
2   private String nome;
3   private long capacidadeEmBytes;
4
5   public Dispositivo(String n, long c){
6     nome = n;
7     capacidadeEmBytes = c;
8   }
9
10  abstract public String toString();
11  abstract public double capacidadeEmMegaBytes();
12 }
13
14 public class DiscoOtico extends Dispositivo{
15   public DiscoOtico(long c) {
16     super("Disco Ótico", 241172480L);
17   }
18
19   public String toString() {
20     return "Dispositivo:" + nome + " Capacidade:" + c;
21   }
22 }
```

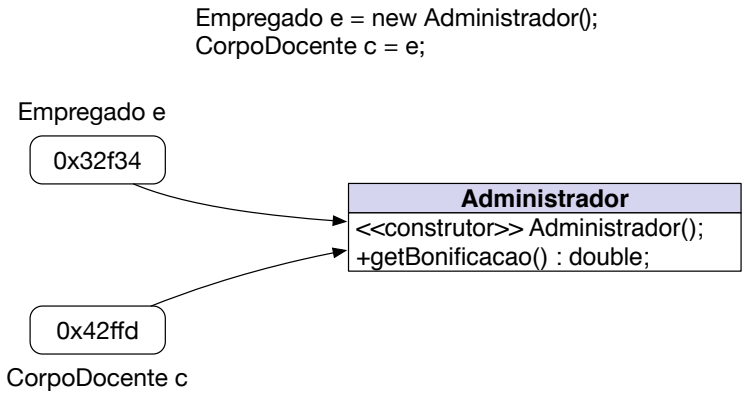
### Exercício 7

Explique com suas palavras a figura abaixo:




### Exercício 8

Usando a figura da questão anterior, explique com suas palavras a figura abaixo:




## Desafios (não necessariamente difíceis)

### Exercício 1

Escreva uma classe cujos atributos e métodos implementem a estrutura de dados de uma *Pilha* com capacidade para empilhar $N$ objetos quaisquer. A classe deverá implementar as operações básicas de uma pilha: inserir dados (add), remover dados da pilha (pop), visualizar topo da pilha (top), verificar se a pilha está cheia e verificar se a pilha está vazia. Em seguida, escreva uma programa para testar sua classe. Um dos atributos da classe *Pilha* deverá ser um *array* de até $N$ referências a objetos, onde $N$ deverá ser indicado no construtor.

## Exercício 2

Escreva uma classe Data que represente uma data no formato Dia/Mês/Ano. A classe deverá implementar as operações que permitam configurar uma data qualquer, respeitando os limites de dias de cada mês. Para isso, a classe deverá também verificar se o ano informado é bissexto. A classe Data deverá também implementar um método "vemAntesDe" que receba como argumento outra instância da classe Data e implemente um algoritmo para verificar qual ocorre antes da outra, retornando *true* se a data encapsulada vier antes da passada como argumento e *false* caso contrário. Se as datas forem exatamente iguais, o método deve retorna *true*.

## Exercício 3

Escreva a classe NumeroComplexo que represente um número complexo. A classe deverá ter os seguintes métodos:

**inicializaNumero** , que recebe dois valores como argumentos para inicializar os campos da classe (parte real e imaginária);

**imprimirNumero** , que deve imprimir o número complexo encapsulado usando a notação $a + bi$, onde $a$ é a parte real e $b$ a imaginária;

**igual** , que recebe outra instância da classe NumeroComplexo e retorna *true* se os valores dos campos encapsulados forem iguais aos da instância passada como argumento;

**soma** , que recebe outra instância da classe NumeroComplexo e soma este número complexo com o encapsulado usando a fórmula $(a + bi) + (c + di) = (a + c) + (b + d)i$;

**subtrai** , que recebe outra instância da classe NumeroComplexo e subtrai este número complexo com o encapsulado usando a fórmula $(a + bi) - (c + di) = (a - c) + (b - d)i$;

**multiplica** , que recebe outra instância da classe NumeroComplexo e multiplica este número complexo com o encapsulado usando a fórmula $(a + bi) * (c + di) = (ac - bd) + (ad + bc)i$;

**divide** , que recebe outra instância da classe NumeroComplexo e divide este número complexo com o encapsulado usando a fórmula $\frac{(a+bi)}{(c+di)} = \frac{ac+bd}{c^2+d^2} + \frac{bc-ad}{c^2+d^2}i$;

## Exercício 4

Escreva o programa do **jogo da velha**, usando todos os artifícios da Programação Orientada a Objetos. Crie classes para representar o tabuleiro, os jogadores (humano ou artificial), o estado do jogo, etc.

## Exercício 5

Escreva o programa do **crivo de Eratóstenes** (exercício 19, Introdução), usando Programação Orientada a Objetos. Crie uma classe *Crivo* cujo construtor receba como parâmetro a faixa de valores a ser pesquisada. A classe deverá ter os seguintes métodos:

**proximo** , que retorna o próximo número primo identificado na faixa indicada no ato da instanciação do objeto;

**reset** , que reinicia o indicador de número primo;

**ehPrimo** , que recebe um número inteiro e retorna *true* caso o número seja primo ou *false* caso contrário;

**imprimirTabela** , que imprime o crivo segundo a faixa de valores, no formato de tabela;

## Exercício 6

Crie a classe *Ponto2D* para representar um ponto no espaço cartesiano de duas dimensões. Crie uma classe *Retangulo* para representar um retângulo cujos pontos opostos seja duas instâncias de uma classe *Ponto2D*. A classe *Retangulo* deve conter um método *calculaIntersecao*, que recebe como argumento uma outra instância da própria classe *Retangulo* e que calcula as coordenadas de um retângulo que é a interseção do retângulo encapsulado com aquele que é passado como argumento, retornando uma **nova** instância da classe *Retângulo* correspondente à interseção. **Dica**: os pontos do novo retângulo podem ser calculados com regras simples, implementadas através de *ifs* encadeados. Nem sempre existe interseção entre dois retângulos. No caso de não existir interseção, o método deve retornar *null*.

## Exercício 7

Modifique a classe *Retangulo* para que esta contenha um método para verificar se um ponto (*Ponto2D*), passado como argumento, está localizado dentro de um retângulo. O método deverá retornar *true* se o ponto estiver contido no retângulo, e *false* se não estiver. **Dica**: para verificar se o ponto está dentro do retângulo, verifique se as coordenadas do ponto estão dentro das coordenadas do retângulo.

## Exercício 8

Crie a classe *Linha2D* para representar um ponto no espaço cartesiano de duas dimensões, unida por dois pontos no espaço cartesiano de duas dimensões, usando duas instâncias da classe Ponto2D, criada no Exercício 6. Modifique a classe *Retangulo* para que esta contenha um método para verificar se uma linha (*Linha2D*), passado como argumento, está localizado dentro de um retângulo. O método deverá retornar *true* se a linha estiver contida no retângulo, e *false* se não estiver. **Dica**: para verificar se a linha está dentro do retângulo, verifique se as coordenadas dos dois ponto que definem a linha estão dentro das coordenadas do retângulo.