

Site: <https://sigs.ufrpe.br/sigaa/ava/index.jsf>

Disciplina: Modelagem e Programação Orientada a Objetos (MPOO)

Profº: Richarlyson D'Emery

LISTA DE EXERCÍCIOS II

Leia atentamente as instruções gerais:

- Ao responder as perguntas desta lista informe, em cada questão, se você baseou sua resposta em alguma pesquisa ou se você respondeu a partir de seus próprios conhecimentos. Sendo assim use: "REFERÊNCIA: Elaboração própria" ou "REFERÊNCIA: citar local da pesquisa".

Você sabe como referenciar uma fonte? Saiba mais sobre as normas da ABNT em

- o <https://www.normasabnt.org/referencias-bibliograficas/>
- o ABNT NBR 6023 (<https://www.ufpe.br/documents/40070/1837975/ABNT+NBR+6023+2018+%281%29.pdf/3021f721-5be8-4e6d-951b-fa354dc490ed>)

- No Eclipse crie um novo **projeto chamado br.edu.mpoo.listall.SeuNomeSobrenome**, o qual deverá ter **pastas de pacotes** (Menu File -> Submenu New -> Opção Source Folder) para cada questão: questao1, questao2, e assim sucessivamente, contendo todas as respostas da lista.
- A lista envolve questões práticas e conceituais.
- Quando a questão envolver uma discussão teórica utilize um arquivo .txt (Menu File -> Submenu New -> Opção File), por exemplo, questao1.txt
- Quando se tratar de questão prática, na pasta de pacote utilize um arquivo .java (Menu File -> Submenu New -> Opção Class), por exemplo, Classe.java

Responda:

- 1) O que são ambientes de programação?
- 2) Quais as principais características encontradas em um IDE?
- 3) De que é constituído o J2SE?
- 4) Qual a diferença entre JRE e JDK?
- 5) Qual a função da JVM?
- 6) Em Java, por que uma classe principal não pode ser private?
- 7) Para que serve os argumentos do main?
- 8) Sabendo-se da existência de argumentos no método main:
 - 8.1) Ilustre uma aplicação Main.java que utiliza os argumentos do main. A codificação deverá ser apresentada.
 - 8.2) Ilustre um diagrama de *use case* em que um SistemaA se comunica com o SistemaB
 - 8.3) Implemente uma aplicação Java para o diagrama de 8.2)
- 9) O que é a assinatura de um método?
- 10) A partir da codificação abaixo:

```

1 package util;
2 import javax.swing.JOptionPane;
3
4 public class Mensagem {
5
6     public static final String MENSAGEM_FALHA = "Falha no sistema";
7     protected static final String MENSAGEM_SUCESSO= "Operação realizada com sucesso";
8     private static final String MENSAGEM_ERRO = "O sistema será finalizado";
9     static final String MENSAGEM = "Bem vindo ao sistema";
10
11     public static String exibirMensagemFalha(){
12         return "Falha";
13     }
14
15     public static void exibirMensagem(String mensagem){
16         JOptionPane.showMessageDialog(null, mensagem);
17     }
18 }

```

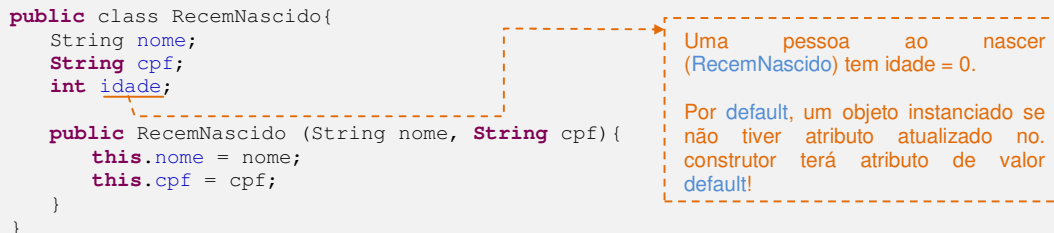
- 10.1) Apresente uma aplicação Java que utiliza `exibirMensagemFalha()`, `exibirMensagem(String mensagem)` e as diferentes mensagens `MENSAGEM_FALHA`, `MENSAGEM_SUCESSO`, `MENSAGEM_ERRO` e `MENSAGEM`.
- 10.2) Apresente o diagrama de classes de 10.1)

Fique atento!

Método construtor é um método que inicializa os atributos da classe. O nome do método construtor deverá ser o mesmo nome da classe. Mas nem sempre o construtor precisa receber por parâmetro dados para todos os atributos. Por exemplo:

```
public class RecemNascido{
    String nome;
    String cpf;
    int idade;

    public RecemNascido (String nome, String cpf){
        this.nome = nome;
        this.cpf = cpf;
    }
}
```



Uma pessoa ao nascer (RecemNascido) tem idade = 0.

Por default, um objeto instanciado se não tiver atributo atualizado no construtor terá atributo de valor default!

Toda classe Java possui um construtor **default**, mas uma vez definido um método construtor para a existência do default é necessário explicitá-lo. Por exemplo:

```
public class RecemNascido{
    String nome;
    String cpf;
    int idade;

    public RecemNascido (){}

    public RecemNascido (String nome, String cpf){
        this.nome = nome;
        this.cpf = cpf;
    }
}
```

- 11) Para que serve o **this**?
- 12) Em Java o “método construtor” realiza a alocação de uma área de memória de uma instância de uma classe. Então:
 - 12.1) Devem-se colocar parâmetros para atualizar atributos **static**?
 - 12.2) Devem-se colocar parâmetros para atualizar atributos **final**?
- 13) Quais os valores default dos atributos de um objeto?
- 14) Qual a diferença entre objeto e objeto anônimo?
- 15) A partir da codificação disponibiliza no projeto br.edu.mpoo.listall.SeuNomeSobrenome demonstre:
 - 15.1) Como gerar um atributo *auto-increment*? (usar o atributo id)
 - 15.2) A garantia da validação de um cpf para um usuário, caso contrário o usuário terá **cpf = null**;
 - 15.3) A criação de um objeto concreto para o usuário: login: “maria”, senha: “mariamaria”, cpf: “166.728.840-76”, id = 1;
 - 15.4) A criação de um objeto concreto para o usuário: login: “jose”, senha: “JoSe”, cpf: null, id = 2;
 - 15.5) A criação de um objeto anônimo para o usuário: login: “joao”, senha: “JoAo123”, cpf: “123.456.789-00”, id = 3;
 - 15.6) Exibir no console os cpfs dos usuários.

Desafio

Você, aluno de MPOO, está experienciando situações-problemas do universo de desenvolvimento de software e começará a ser desafiado a solucionar problemas a partir de conhecimentos de Programação e Orientação a Objetos.



16) O que significa a expressão “depurar um programa”?

17) A partir da codificação abaixo

```
public class Classe {  
    int valor1;  
  
    public static void metodo(){  
        int valor2;  
        ///?  
    }  
  
    public static void main(String[] args) {  
        int valor3;  
        Classe instancia1;  
        Classe instancia2 = new Classe();  
        metodo();  
        ///?  
    }  
}
```

Explique o resultado para `///?` e `///??`:

Instrução <code>///?</code>	Saída
<code>System.out.println(valor1);</code>	
<code>System.out.println(valor2);</code>	
<code>System.out.println(valor3);</code>	

Instrução <code>///??</code>	Saída
<code>System.out.println(valor1);</code>	
<code>System.out.println(valor2);</code>	
<code>System.out.println(valor3);</code>	
<code>System.out.println(instancia1);</code>	
<code>System.out.println(instancia2);</code>	
<code>System.out.println(instancia1.valor1);</code>	
<code>System.out.println(instancia2.valor1);</code>	

18) A partir das descrições abaixo, proponha uma solução representada em diagrama de classes e codificada em Java:

A classe Robot possui atributos para nome, posição e direção. Possui métodos para inicializar um robô com:

- um nome indicado e supondo que esteja na posição (0,0) é direcionado para o Norte; e
- indicação de um nome, posição e direção.

Possui métodos para:

- fazer um robô andar 1 passo por vez;
- fazer um robô andar vários passos;
- transportar um robô para uma nova posição;
- mudar a direção do robô;
- retornar a configuração do robô (*status* atual); e
- retornaPosZero() que leva o robô a posição inicial: posição (0,0) e direcionado para o Norte

Crie uma aplicação para o Robô chamado robot que utiliza as informações da questão.

19) Na questão anterior e ilustre a partir de capturas de telas (*print screen*) as alterações de dados do objeto robot durante a execução da aplicação. *Observação: não colocar toda a tela, apenas recortes para a(s) linha(s) que está(ão) sendo executada(s) e para o(s) valor(es) do(s) dado(s).*