1ª Lista de Exercícios de Linguagem de Programação I - Modelagem

1. Modele um restaurante que oferece a seus clientes comida a Kg, sobremesa, refrigerante e cerveja. O valor do Kg de comida é R$ 30,00, a unidade da sobremesa fica por R$ 6,00, o valor do refrigerante é R$ 5,00 e a cerveja custa R$ 10,00. A conta é contabilizada através do número da mesa, sendo de responsabilidade do estabelecimento informar o total geral e o valor por pessoa, que é calculado através da divisão do valor total pelo número de integrantes da mesa.
   As principais ações do modelo são referentes a abertura do pedido, a adição de novas pessoas na mesa caso cheguem mais tarde, a adição de comida, sobremesa, refrigerante e cerveja a qualquer instante que seja necessário, o fechamento da conta e a emissão da nota fiscal com o valor total e com o valor por pessoa.

2. Escreva um modelo para representar uma lâmpada que está à venda em um supermercado. Que dados devem ser representados por este modelo?

3. Imagine uma lâmpada que possa ter três estados: apagada, acesa e meia-luz. Usando o modelo "Lâmpada" como base, escreva o modelo "LampadaTresEstados".

4. Inclua, no modelo "Lâmpada", uma operação "estáLigada" que retorne verdadeiro se a lâmpada estiver ligada e falso, caso contrário.

5. Crie um modelo Livro que represente os dados básicos de um livro, sem se preocupar com a sua finalidade.

6. Usando o resultado do exercício anterior como base, crie um modelo "LivroDeLivraria" que represente os dados básicos de um livro que está à venda em uma livraria.

7. Usando o resultado do modelo "Livro" como base, crie um modelo "LivroDeBiblioteca" que represente os dados básicos de um livro de uma biblioteca, que pode ser emprestado a leitores.

8. Escreva um modelo para representar uma conta bancária simplificada com os seguintes dados: nome do titular, número da conta e valor na conta corrente. Crie uma operação para abrir a conta inicializando os dados. Além disso, especifique as operações para mostrar os dados da conta (emitir saldo), e fazer movimentações de depósitos e retiradas

2ª Lista de Exercícios de Linguagem de Programação I – Classes em Java

1. Escreva uma classe "Livro" que represente o modelo desenvolvido na lista 1.

2. Escreva uma classe "LivroLivraria" que represente o modelo desenvolvido na lista 1.

3. Escreva uma classe "LivroBiblioteca" que represente o modelo desenvolvido na lista 1.

4. Escreva uma aplicação em Java que demonstre o uso de instâncias das classes "Livro", "LivroLivraria" e "LivroBiblioteca" .

3ª Lista de Exercícios de Linguagem de Programação I – Construtores e Sobrecarga

1. Escreva um construtor para a classe “Livro” (use a classe desenvolvida em exercícios anteriores).

2. Escreva dois construtores usando o conceito de sobrecarga e referência “this” para a classe “LivroLivraria” (use a classe desenvolvida em exercícios anteriores).

3. Defina uma classe Professor com os dados: nome do professor, nome do departamento, data de admissão, número de registro. Inclua na classe um construtor para setar os dados e um método para imprimir o conteúdo.

4. Defina uma classe Data com os dados: dia, mês e ano.

5. Reescreva a classe Professor de forma que a data de admissão seja um objeto da classe Data.

6. Modifique a classe Data criada anteriormente para conter um construtor capaz de setar o dia, mês e ano e um método capaz de imprimir a data.

7. Crie uma classe com um método principal capaz de ler informações do teclado para instânciar três objetos da classe professor e utilizar o método imprime_dados.

4ª Lista de Exercícios de Linguagem de Programação I – Estruturas de Decisão e Repetição

1. Escreva uma classe chamada classe “Comparavel” que tem como atributo um valor inteiro, um construtor para inicializar o atributo e os métodos éMaiorOuIgual, éMenorOuIgual e éDiferenteDe que recebem um valor do tipo int como parâmetro e retornam true se o valor encapsulado for, respectivamente, maior ou igual, menor ou igual ou diferente do passado como parâmetro.

2. Escreva uma classe em Java que simule uma calculadora bem simples. Essa classe deve ter como atributos duas variáveis double e um char. Deve possuir um construtor que recebe como parâmetro dois números e um caracter, correspondente a uma das operações básicas (+, -, *, /). Deve ter um método para calcular a operação desejada e um para imprimir o resultado. O programa deve considerar divisões por zero como sendo erros, e imprimir uma mensagem adequada.

3. Escreva para a classe “Livro” o método “éIgual” que aceite o título do livro como parâmetro e retorne true se o valor encapsulado for igual ao valor passado como parâmetro.

4. Escreva uma classe “EntradaDeCinema” com a seguinte estrutura (dados):
   - Data dataDoFilme
   - float horário
   - int sala
   - float valor

   E com as seguintes ações (métodos):
   - Construtor: com a finalidade de inicializar todos os atributos.

- CalculaDesconto: que deve receber como parâmetro a data de nascimento do cliente (do tipo Data) e caso seja menor de 12 anos, deve ser dado um desconto de 50% no valor normal.
- CalculaDesconto: que deve receber como parâmetro a data de nascimento do cliente (do tipo Data) e o número de sua carteira de estudante (do tipo int). Se o estudante tiver idade entre 12 e 15 anos, deve ser dado um desconto de 40%, de 16 a 20 um desconto de 30% e mais que 20 anos um desconto de 20% no valor normal.
- CalculaDescontoHorário: esse método deve dar um desconto de 10% sobre o valor aferido após todas as outras opções de desconto, caso o horário do filme seja antes das 16 horas.
- toString(): que deve imprimir todos os dados do ingresso.

5. Desenvolva uma aplicação que leia os dados necessários para instanciar e imprimir o ingresso para clientes normais, menores de 12 anos e estudantes.

6. Acrescente a classe Data (ver exercícios anteriores) para que essa possua um método chamado "retornaMes", que ao ser passado uma determinada data (do tipo Data) retorne uma String indicando o mês do ano da data passada como parâmetro. Use o comando switch para resolver o exercício.

7. O valor de x^y pode ser calculado como sendo x multiplicado por si mesmo y vezes (se y for inteiro). Escreva uma classe chamada "SeriesMatemáticas" que contenha um construtor para inicializar x e y, um método chamado "elevadoA" que calcule e retorne o resultado de x^y , e um método chamado "imprimeResultado" que mostre o resultado obtido. Obs: Use o comando while.

9. Acrescente a classe "SeriesMatematicas" o método "piQuadradoSobre8" que calcule a série $(1/1^2) + (1/3^2) + (1/5^2) + (1/7^2) + (1/9^2) + ...$ Evidentemente a série não poderá ser calculada infinitamente, devendo parar depois de N termos, sendo que o valor de N deve ser fornecido como parâmetro ao método. Obs: Use o comando do-while.

10. Acrescente a classe "SeriesMatematicas" o método "calculaPi" que calcule a série 2 x (2/1) x (2/3) x (4/3) x (4/5) x (6/5) x (6/7)... Evidentemente a série não poderá ser calculada infinitamente, devendo parar depois de N termos, sendo que o valor de N deve ser fornecido como parâmetro ao método. O resultado da série deve retornar um valor aproximado a PI.

11. Escreva uma classe chamada "Fibonacci" que tenha um método que receba como parâmetro um número inteiro para indicar a quantidade de termos que se quer calcular e imprimir a série.
    Série: 1 + 1 + 2 + 3 + 5 + 8 + 13 + ...

12. Acrescente a classe "Fibonacci" um método que receba como parâmetro um número inteiro e retorne verdadeiro se o número pertence a série e falso em caso contrário.

13. Escreva uma classe chamada "Primos" que tenha um método para identificar se um determinado número é primo.

14. Acrescente a classe "Primo" um método que receba dois números como parâmetros e possa mostrar todos os primos no intervalo indicado pelos dois números. Envie uma mensagem de intervalo errado caso o primeiro número seja menor que o segundo.

5ª Lista de Exercícios de Linguagem de Programação I – Vetores e Matrizes

1. Fazer um programa Java que leia um vetor com 8 elementos, e imprima a soma dos elementos lidos.

2. Fazer um programa Java que leia dois vetores A e B de 8 elementos cada um e calcula um terceiro vetor C, cujos elementos são resultado da soma dos elementos correspondentes de A e B.

3. Leia dois vetores VET1 e VET2, ambos com 10 elementos cada, sendo que só devem ser aceitos valores em ordem crescente. Após gere e imprima o vetor VET3, resultado da intercalação de VET1 e VET2.

4. Escreva um programa que leia o tamanho de um vetor, seus elementos e proceda a ordenação, apresentando o vetor ordenado no final.

5. Escreva uma classe chamada "VetorDeChar" que tenha como atributo um vetor de char e um construtor que receba como parâmetro uma frase. No construtor deve ser passada a frase para o vetor de char.
   Dica: Use o método "toCharArray" da classe String para colocar os elementos da frase no vetor.
   Exemplo: char vetor[] = frase.toCharArray();
   Acrescente os seguintes métodos a classe:
   a. Um método que retorne o número de vogais existentes na frase.
   b. Um método que retorne o número palavras iguais na frase.

6. Escreva um programa em Java que leia a ordem de uma matriz, a matriz e encontre o maior número.

7. Escreva um programa para ler a ordem de uma matriz quadrada, os elementos da matriz e some a diagonal secundária.

8. Escreva uma classe chamada "MatrizDeInteiros" que tenha como atributo uma matriz de inteiros e um construtor que receba como parâmetro a ordem da matriz, a instancie e inicialize com zeros. Acrescente a classe os seguintes métodos:
   a. um método que receba como parâmetro três números inteiros indicando respectivamente linha, coluna e o valor que deve ser armazenado na linha e coluna indicada.
      Obs: Caso a linha ou a coluna passadas como parâmetro estejam fora da ordem da matriz indique com uma mensagem o erro.
   b. um método "éQuadrada", que retorna true se a matriz for quadrada (isto é, tem o mesmo número de linhas e colunas).
   c. um método total que some todos os valores da matriz retornando o resultado.
   d. um método que receba como parâmetro um determinado valor e retorne a linha onde o elemento foi encontrado na matriz ou –1 caso contrário.

9. Escreva uma classe "Funcionário" com os atributos matricula (int), nome (String), departamento (int), salário (float) e função (String). Adicione na classe um construtor que receba todos os parâmetros para inicializar os dados de um funcionário.

10. Escreva uma classe "SetorPessoal" que tenha como atributo um vetor da classe "Funcionário" e uma variável inteira para ser usada como índice do vetor. Crie um construtor que receba como parâmetro o número de funcionários de uma empresa para instanciar o vetor e inicialize o índice do vetor com zero. Acrescente os seguintes métodos a classe:

a. um método para adicionar funcionários no vetor definido na classe;
b. um método que possa imprimir a folha de pagamento informando o nome dos funcionários e o seus respectivos salários.
c. um método que possa retornar o valor total da folha de pagamento.
d. um método que possa retornar o nome do funcionário que recebe o maior salário.
e. um método que possa receber como parâmetro o número de um determinado departamento e mostrar o nome e a função de todos os funcionários daquele departamento.
f. um método que possa receber como parâmetro o nome de uma determinada função e posteriormente imprimir o nome de todas as pessoas que exercem essa função.
g. um método que possa imprimir a folha de pagamento informando o nome dos funcionários e o seus respectivos salários em ordem crescente de salário.
   Dica: É necessário criar um outro vetor que conterá o vetor original e posteriormente ordena-lo.

11. Escreva uma classe “Aplicação” que receba a quantidade de funcionários de uma empresa, possa instanciá-los e colocá-los em um vetor da classe “SetorPessoal”. Após execute todos os métodos da classe “SetorPessoal” e apresente os resultados.

6ª Lista de Exercícios de Linguagem de Programação I – Reutilização de Classes

1. Escreva as classes “LivroLivraria” e “LivroBiblioteca” que herdam da classe Livro.

   Dica: Os campos em comum devem ser preferencialmente representados pela classe ancestral.

2. Observe o cenário a seguir. Identifique os objetos, seus atributos, métodos e implemente as classes necessárias: Uma conta corrente possui um número, um saldo, um status que informa se ela é especial ou não, um limite e um conjunto de movimentações. Uma movimentação possui uma descrição, um valor e uma informação se ela é uma movimentação de crédito ou débito. Crie uma classe banco que armazene um conjunto de contas e forneça métodos que permitam que sejam feitos criações de conta, exclusão de contas, saques (uma conta corrente só pode fazer saques desde que o valor não exceda o limite de saque -limite + saldo-), depósitos, emissão de saldo e extrato e transferência entre contas.

3. Escreva uma classe que represente um país. Um país tem como atributos o seu nome, capital. a sua dimensão em Km^2, uma lista de países com os quais ele faz fronteira e um contador de vizinhos. Represente a classe e forneça os seus membros a seguir:
   a) Construtor que inicialize o nome, a capital, a dimensão, o vetor de fronteiras (considere que um país tem no máximo 40 outros países com os quais faz fronteira) e o contador de vizinhos com zero;
   b) Um método para adicionar países vizinhos;
   c) Métodos get e set para acesso aos seus atributos;
   d) Um método que permita verificar se dois países são iguais. Dois países são iguais se tiverem o mesmo nome e a mesma capital. A assinatura deste método deve ser:
      · public boolean equals(Pais outro);
   e) Um método que informe se um outro país é seu limítrofe (faz fronteira);
   f) Um método que receba um outro país como parâmetro e retorne uma lista de vizinhos comuns aos dois países.

4. Escreva uma classe continente. Um continente possui um nome e um conjunto de países, que dele fazem parte. Escreva um construtor para inicializar os atributos e um método para adicionar países ao continente. Desenvolva um método que forneça a dimensão total do continente.

5. Um avião é representado pelo seu prefixo, pelo nome do seu modelo, pelo nome do seu fabricante, pela quantidade de assentos e pela quantidade de assentos ocupados.
   Escreva uma classe que represente um avião e desenvolva um método que informe qual o prefixo da aeronave.

   Um aeroporto é representado pelo seu código e seu nome (ex.: SSA: Salvador, BSB: Brasília. etc), por um status que informa se ele é um aeroporto internacional ou não, uma lista de outros aeroportos para os quais partam voos diretos a partir dele e uma lista de outros aeroportos de onde venham voos para ele. Além disto possui uma lista de aviões que estão atualmente no pátio.
   Escreva uma classe que contemple os atributos apresentados e mais os seguintes membros:
   a) Um construtor que inicialize o código do aeroporto, seu nome e o seu status.
   b) Métodos que verifiquem e alterem a situação relativa ao fato do aeroporto ser internacional ou não.
   c) Um método que verifique e informe se um aeroporto é igual a ele:
      - public boolean equals (Aeroporto aeroporto)
      - Dois aeroportos são iguais se o seu código for coincidente.
   d) Um método que receba um prefixo de avião como parâmetro e informe se a aeronave está pousada nele.

e) Um método que receba um outro aeroporto como parâmetro e verifique se existe voos diretos entre eles.

Obs: A quantidade máxima de aeronaves que um aeroporto pode ter em terra é igual a 40 e a quantidade máxima de outros aeroportos com os quais o aeroporto pode ter linhas saindo e chegando é 40.

6. Forneça o esquema de classes em Java para um programa funcional para a seguinte situação:
Uma biblioteca possui um nome, um endereço e armazena um conjunto de publicações. Atualmente, as publicações disponíveis são os artigos e os livros. Todas as publicações possuem data de publicação, um título, uma lista de outras publicações a que elas fazem referência e o conjunto de autores da publicação. São atributos relevantes para um autor o seu nome e sua titulação. São atributos relevantes para artigos o seu título, a sua data de publicação, os seus autores, as outras publicações que ele faz referência, a data de sua publicação e o seu resumo. Para os livros é relevante armazenar seu título, sua data de publicação, as outras publicações que ele referencia, seus autores, o número da edição, o nome da editora e o seu ISBN. As publicações são formas genéricas de se referir a uma grande categoria de classes, mas na biblioteca não existem objetos publicações; o que ela vai efetivamente armazenar são livros ou artigos. Futuramente a biblioteca pretende operar com outros tipos de publicações como manuais, teses e etc.
Obs: É necessário definir apenas os atributos das classes

7. Implemente uma classe chamada "Arquivo" que possua o controle de acesso RWX dos Sistemas Unix:
→ O primeiro bit indica que o objeto é um objeto de leitura apenas.
→ O segundo bit indica que o objeto é um objeto de escrita apenas.
→ O terceiro bit indica que o objeto é um objeto executável apenas.

Read	Write	Execute
1	0	1

Indica um arquivo que permite Leitura e Execução

Forneça os seguintes métodos para os objetos instanciados nessa classe:

→ Construtor da classe:
```
public Arquivo(byte r, byte w, byte e)
```

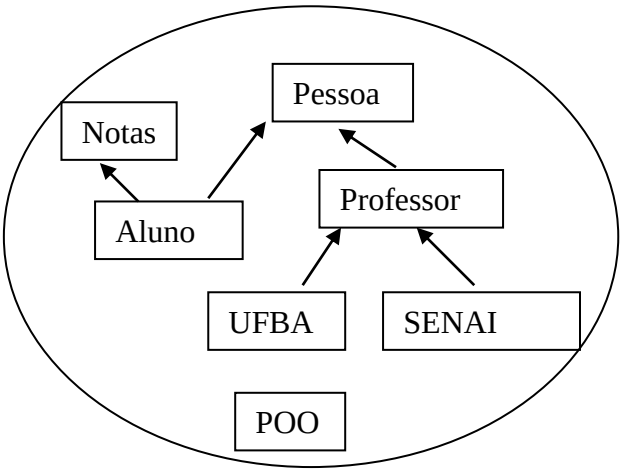
→ Métodos que verificam se o arquivo fornece permissão de leitura, escrita e execução:
```
public boolean isReadable();
public boolean isWriteable();
public boolean isExecuteable();
```

→ Métodos que concedem a troca do atributo de leitura, escrita e execução, ou seja, se o arquivo tinha permissão de leitura ao ser chamado o método "trocaRead" ele passa a não ter mais a permissão de leitura, e caso ele não tenha a permissão passa a ter.
```
public byte trocaRead();
public byte trocaWrite();
public byte trocaExecute();
```

→ Um método que mostra as permissões do arquivo.
```
Public byte mostraEstado();
```

8. Desenvolver um sistema de cadastro de Pessoas, Alunos e Professores.

Classes envolvidas



Classe Pessoa:
Atributos: protected String nome, telefone, endereco, cpf;
protected int identidade, idade;
Métodos: public pessoa(String n, String t, String e, String c, int id, int i)
public pessoa(String n, int i)
public String getNome()
public String getTelefone()
public String getEndereco()
public String getCpf()
public int getIdentidade()
public int getIdade()
public int getIdade(int anoAtual, int anoFuturo)
public void setNome(String nome)
public void setTelefone(String telefone)
public void setEndereco(String endereço)
public void setCpf(String cpf)
public void setIdentidade(int identidade)
public void setIdade(int idade)

Classe Notas:
Atributos: float nota1, nota2, nota3, media;
Métodos: public notas(float n1, float n2, float n3)
public void calculaMedia()
public float getNota1()
public float getNota2()
public float getNota3()
public float getMedia()
public void setNota1(float nota1)
public void setNota2(float nota2)
public void setNota3(float nota3)
public void setMedia(float media)

Classe Aluno extends Pessoa:
Atributos: int matricula;
Notas notaAluno = new notas();
Métodos: public Aluno(String nome, String telefone, String endereco, String cpf, int identidade, int idade, int matricula, float nota1, float nota2, float nota3)
public void setMatricula(int matricula)
public int getMatricula()

Abstract classe Professor extends Pessoa:
Atributos: float salario;
int horas;
Métodos: public Professor(String nome, String telefone, String endereco, String cpf, int identidade, int idade, int horas)
abstract void calculoSalario();
public float getSalario()
public int getHoras()
public void setSalario(float salario)
public void getHoras(int horas)
public void mostraSalario()

Classe UFBA extends professor:
Atributos: int horasAtendimento;
Métodos: public UFBA(String nome, String telefone, String endereco, String cpf, int identidade, int idade, int horas, int horasAtendimento)
public void calculoSalario()
public int getHorasAtendimento()
public void setHorasAtendimento(int horas)

Classe SENAI extends Professor:
Atributos: int horasPesquisa;
Métodos: public SENAI(String nome, String telefone, String endereco, String cpf, int identidade, int idade, int horas, int horasPesquisa)
public void calculoSalario()
public int getHorasPesquisa()
public void setHorasPesquisa(int horas)

Classe POO:
a) Responsável em ler todas as informações para o cadastramento de Pessoas, Alunos e Professores da SENAI e da UFBA e colocar no vetor adequado.
b) Responsável pela leitura de um nome e a pesquisa no vetor adequado.
c) Responsável em mostrar as informações existentes no objeto com o nome solicitado.

9. Desenvolva o sistema especificado abaixo utilizando o conceito de interface.

Desenvolvimento da classe aluno

a) Escreva uma classe chamada aluno com os atributos: matrícula, nome, nota1, nota2, nota3, média e total de faltas.
b) Crie um construtor para a classe aluno para possibilitar a inicialização de todos os atributos.
c) Crie um método que retorne a média do aluno;
d) Crie outro método que retorne à quantidade de faltas do aluno.

Desenvolvimento da interface Define Disciplina

a) Desenvolva uma interface (DefineDisciplina) que represente as ações que uma disciplina de um curso universitário deve ter. Essa interface deve definir dois métodos: um para identificar se um aluno está aprovado ou reprovado por falta e outro para identificar se um aluno está aprovado ou reprovado por nota. Ainda deve ser definida uma constante com o valor de 25% indicando o máximo de faltas que um aluno pode ter e outra com um valor igual a 5,0 indicando a média para ser aprovado.

Desenvolvimento da classe Disciplinas

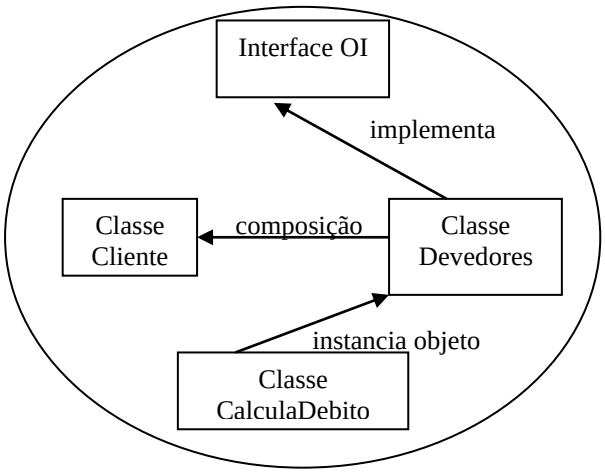
a) Desenvolva uma classe Disciplinas que implemente a interface anterior e tenha os seguintes atributos: código, nome, carga horária, um vetor de alunos que a está cursando e um contador para ser usado como índice no vector de alunos.
b) Um construtor para inicializar os atributos: código, nome e carga horária, instanciar o vector de alunos e zerar o contador que será usado como índice.
c) Acrescente um método a classe Disciplinas que receba como parâmetro um objeto de aluno e possa inseri-lo no vector de alunos cursando a disciplina.
d) Acrescente um método a classe Disciplinas que receba um objeto de aluno e possa identificar se esse aluno está aprovado ou reprovado, baseando-se na sua média e total de faltas. Esse método deve retornar TRUE se o aluno está aprovado ou FALSE caso contrário.

Desenvolva a classe Aplicação

a) Desenvolva uma classe com um método principal para ler os dados para criar uma disciplina, ler os dados para criar e inserir alunos nessa disciplina e criar uma listagem dos alunos aprovados na disciplina.

10. Desenvolva o sistema especificado abaixo utilizando o conceito de interface. A figura e a classe Cliente servem de auxílio para a resolução do problema.

```
class Cliente
{   private int codigo;
    private float divida;
    private float totalAPagar;

    public Cliente(int codigo, float divida)
    {  this.codigo = codigo;
       this.divida = divida;
    }

    public float getPagamento()
    {   return totaAPagar;
    }

     public int getCodigo()
    {   return codigo;
    }

     public float getDivida()
    {   return divida;
    }

     public void setPagamento(float valor)
     {  totalApagar = valor;
     }
}
```



Considere os seguintes Cálculos:
Dívida Inferior a R$ 200,00 = 10% de Desconto
Dívida Entre R$ 200,00 e R$ 500,00 = 20% de Desconto
Dívida Maior do que R$ 500,00 = 30% de Desconto

A. Desenvolva uma interface chamada OI que determine o desconto que devedores terão no momento do pagamento das contas atrasadas. Essa interface deve declarar o método defineDesconto sem parâmetros de retorno e com um parâmetro da classe Cliente de entrada. Além disso, deve ter a declaração das seguintes constantes: DI = 0.10, DE = 0.20 e DM = 0.30.

B. Desenvolva uma classe chamada Devedores que deve ter:
   i. Um atributo Vetor inadimplentes, o qual deve armazenar objetos da classe Cliente.
   ii. Implementar a interface OI.
   iii. Ter um método que possa adicionar devedores ao Vetor inadimplentes.

Obs: Os cálculos para a construção do método defineDesconto são os definidos no começo do exercício.