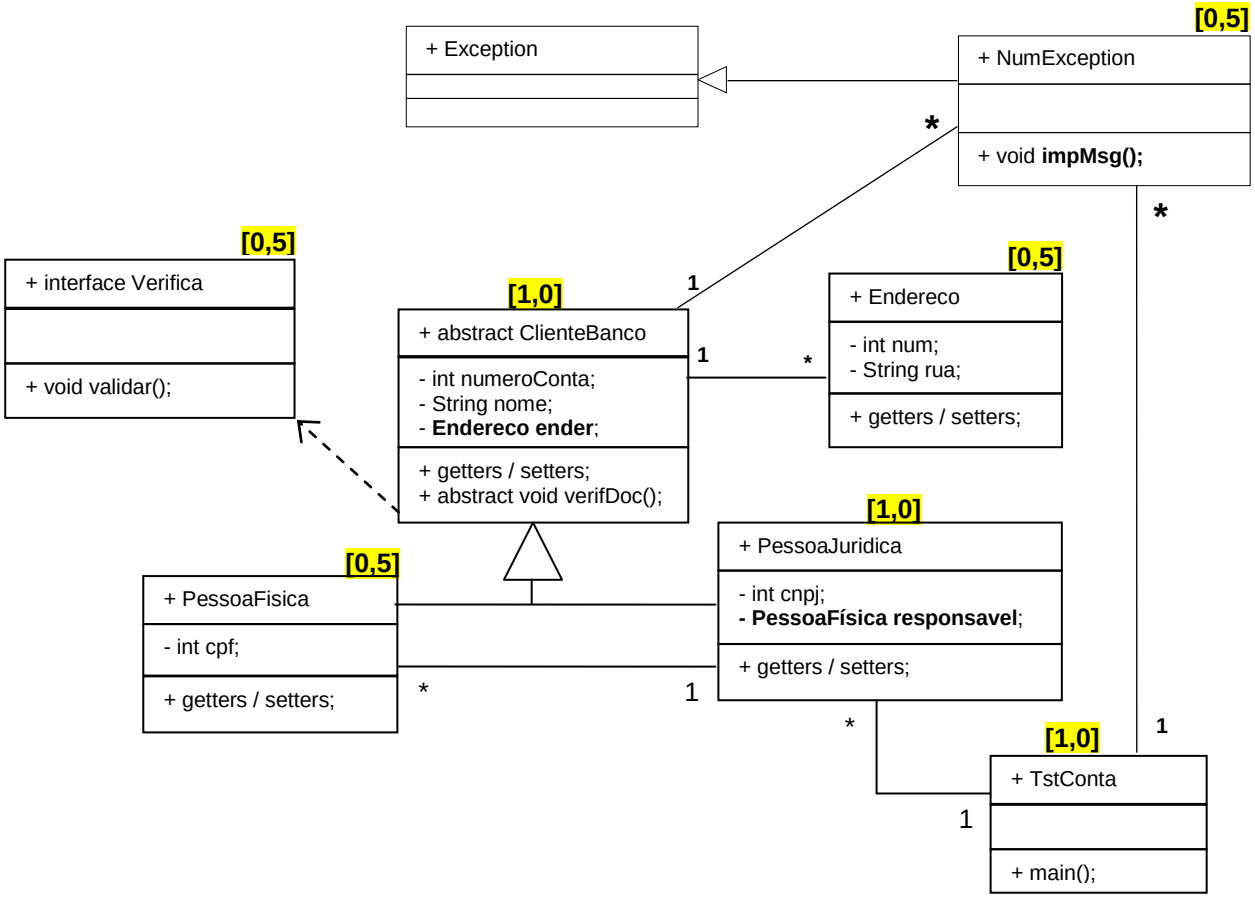
	MINISTÉRIO DA EDUCAÇÃO UNIVERSIDADE TECNOLÓGICA FEDEREAL DO PARANÁ CAMPUS CORNÉLIO PROCÓPIO		
	Curso.....: Especialização em Java **Disciplina.:** Java I	**Professor ..:** José Antonio Gonçalves	**Valor** **5,0**

1) O diagrama de classes a seguir trata de um sistema bancário, no qual encontramos como correntistas (clientes do banco) tanto pessoas físicas (cidadãos comuns) quanto pessoas jurídicas (empresas), porém, **toda empresa terá uma pessoa física responsável (um funcionário) por sua conta bancária**:



Importante:

I) No diagrama há sinais que indicam se os membros das classes são:
" - " privados** ou **"+" públicos;

*II) Perceba que **só há associações** da classe TstConta com apenas 2 outras classes: PessoaJuridica e NumException. Sendo assim, na classe TstConta, só haverá estes 2 tipos de objetos;*

*III) **Métodos Construtores:** excepcionalmente nesta prova **não serão desenvolvidos os métodos construtores.** Desta forma, a instanciação de cada atributo será feita (obrigatoriamente) na mesma linha de sua declaração e da seguinte maneira:*
- Os de tipos numerais com zeros;
- Os de tipos literais com espaço em branco;
*- E, **quando forem objetos,** instancie com o seu respectivo tipo;*

VI) O diagrama de classes descreve as únicas classes que deverá **construir** para resolução da prova;

V) **Não utilizará** interface gráfica nesta prova.

Subsídios (A, B, C, D e F)

A) As classes Endereco, PessoaFisica e PessoaJuridica **NÃO** poderão ser herdadas (**será descontado 0,15 ponto de cada classe que não atender esta colocação**).

B) A **classe NumException**, trata-se de uma classe de exceção do **tipo verificada.** Esta classe contém (apenas) o **método** chamado **impMsg()** que, ao ser chamado, imprimirá a mensagem: "**ERRO: Não pode haver Número Negativo para conta!**". *O método impMsg() será utilizado na classe TstConta.*

C) Interface **Verifica**: contém um método chamado validar(), que não tem retorno e não recebe parâmetros. Quando implementado deverá verificar o Número da Conta **e imprimir na tela** se este é par ou é ímpar;

D) A classe **ClienteBanco é abstrata** e contém:

D1) um **método abstrato** chamado "verifDoc" o qual verificará:

a) Em **PessoaFisica:** se o valor informado para o atributo CPF está entre 10 e 20. Caso o valor esteja fora desta escala, informará na tela a mensagem "**CPF inválido**", se não, informará na tela a mensagem "**CPF válido**";

b) Em **PessoaJuridica:** se a quantidade de letras do nome do "*responsavel*" é maior que 30, caso seja deverá informar na tela a mensagem "**Nome inválido para Responsável**", se não, informará na tela a mensagem "**Nome válido para Responsável**";

D2) O seu método **setNumeroConta**: se valor informado for positivo, atribuirá este valor ao atributo numeroConta, **se não**, deverá disparar uma exceção do tipo **NumException**;

D3) Contém um atributo chamado **ender**, que define os dados cadastrais dos clientes do banco, seja esta pessoa física ou jurídica;

E) A classe **PessoaJuridica** contém um atributo denominado **responsavel,** este indica uma pessoa física da empresa incumbida de gerenciar a conta.

Observação importante:

Considerar o valor do CPF como de um "CPF fictício", respeitando o limite do tipo primitivo.

Considerar o valor do CNPJ como de um "CNPJ fictício", respeitando o limite do tipo primitivo.

F) Considerando todo enunciado, a classe **TstConta** será construída de forma a testar a estrutura do sistema como um todo por meio de suas funcionalidades e **de acordo com a ordem pré determinada** nos itens F1 e F2. Para isso tratará dos dados de uma empresa (pessoa jurídica):

F1) **Entradas de dados**: os valores serão passados como parâmetros por meio dos métodos setters**,** **obrigatoriamente na ordem a seguir e apenas para os seguintes atributos**:

ENTRADA DE DADOS	
Classes	Atributos a serem instanciados
Pessoa Jurídica	**numeroConta** **IMPORTANTE:** *Na classe TstConta, ao tentar utilizar o método set para o numero da conta, se este disparar a exceção NumException, seu catch deverá chamar o método impMsg().*
	cnpj
	rua
Pessoa Física *(trata-se do atributo "responsavel" que está na classe PessoaJuridica especificada acima)*	**cpf**
	nome

F2) **Saídas de dados:** de acordo com as **entradas**, as saídas serão por meio de **impressões na tela, obrigatoriamente na ordem a seguir e apenas dos seguintes dados**:

SAÍDA DE DADOS	
Classes	Dados a serem impressos na tela
Pessoa Jurídica	**Número da Conta**
	se o número da conta é **par** ou é **ímpar**
	cnpj
	rua
Pessoa Física *(trata-se do atributo "responsavel" que está na classe PessoaJuridica especificada acima)*	**cpf** da pessoa física, **Responsável pela Conta;**
	se o **CPF** do **Responsável pela Conta** é válido ou inválido
	Nome da pessoa física, **Responsável pela Conta**
	se o **NOME do RESPONSAVEL** pela conta é válido ou inválido.