Orientações iniciais:

- A prova é individual e sem consulta;
- A prova contém 5 questões práticas valendo **12 pontos**;
- A prova tem duração de mínima de **40 minutos** e duração máxima de **3 horas**;
- Utilize a IDE Eclipse ou um editor de texto comum (Sublime, Kate, etc.) para implementar a prova;
- Crie uma pasta com o seu nome dentro da pasta **Documentos** e armazene o seus arquivos dentro dela.
- Caso você tenha alguma dúvida referente ao enunciado das questões durante a prova, pergunte em voz alta. O professor não irá atendê-lo individualmente;

BOA PROVA!

Prova 2

Linguagem de Programação Orientada a Objetos — 2017

1. [**40 pontos**] Joãozinho é um menino religioso e gosta muito de realizar orações. Após cursar um mini-curso de Programação Orientada a Objetos, Joãozinho teve a ideia de construir um software orientado a objetos que registra as orações realizadas pelas pessoas. Joãozinho deseja registrar as orações de três categorias de pessoas: políticos, esportistas e professores. Apesar de não terem condições de fazer orações que façam sentido para os humanos, Joãozinho acredita que os latidos e miados dos cães e gatos também chegam aos céus.

   Pessoas possuem nome e cpf, enquanto que para os políticos é necessário armazenar o valor aproximado que o político já recebeu de propina em sua vida, para esportistas é necessário armazenar o nome do seu esporte e para professores é necessário armazenar qual o valor que o professor está devendo para os bancos.

   Animais possuem raça e idade, enquanto que para cachorros é necessário armazenar quantas mordidas o cachorro já deu até hoje, e para gatos é necessário armazenar o nome da sua ração favorita.

   Quando faz uma oração, um político perde R$ 100,00 de dinheiro de sua propina, e quando recebe uma oração, ele emite uma mensagem de gratidão e pede voto. Quando recebe uma oração, um esportista emite uma mensagem de gratidão e pede para que a pessoa torça para ele nas próximas olimpíadas, enquanto que quando faz uma oração, um esportista apenas diz: "no pain, no gain". Quando faz uma oração, um professor têm 10% da sua dívida perdoada, enquanto que quando recebe uma oração, um professor recebe recebe um desconto de R$250,00 na sua dívida.

   Quando faz uma oração, um cachorro emite a mensagem "Au au au au", e quando recebe uma oração, o cachorro tem uma de suas mordidas perdoadas. Quando faz uma oração, um gato faz "Mi mi mi mi", e quando recebe uma oração, um gato fica em silêncio, pois tem dificuldade de agradecer.

   (a) Crie um conjunto de classes que modelem o sistema de Joãozinho.

   (b) Crie uma classe *SistemaDeOrações* com um método *main*. Nesse método, deve existir uma instância de cada classe criada no item a), e você deve fornecer uma interação com o usuário que permita com que cada um dos objetos orem e recebam orações. Um pequeno detalhe é que, para este item, todas as variáveis declaradas devem ser abstratas/interfaces. Ao final do método *main*, imprima cada um dos objetos.

2. **[30 pontos]**

   Crie uma classe que modele uma *Pessoa* com *nome*, *cpf* e *rg*. Sua classe deve validar os atributos da seguinte maneira: o atributo *nome* não pode possuir mais que 50 caracteres e o atributo *cpf* deve seguir o formato básico de um cpf (YYY.YYY.YYY-YY), onde Y corresponde a um número entre 0 e 9. Caso o *nome* não se enquadre na restrição exigida, lance uma exceção verificada chamada *HugeNameException*, e caso o *cpf* não se encaixe na restrição exigida, lance uma exceção não verificada chamada *InvalidCpfFormatException*.

   Crie uma classe *CadastroDePessoas*, que permite a inclusão e a listagem de pessoas. Caso o usuário informe pessoas cujos dados não respeitam às restrições dos atributos de uma pessoa, você deve pedir para que o usuário digite novamente o atributo que foi considerado inválido. Para isso, você DEVE utilizar tratamento de exceções.

3. **[15 pontos]** Para que serve o bloco *finally* do comando *try catch*? Dê um exemplo em que este bloco deveria ser usado.

4. **[15 pontos]** Classes abstratas permitem construtores? Justifique e exemplifique.

5. **[20 pontos][EXTRA]** Implemente a classe *CadastroDePessoas* do exercício 2 utilizando interface gráfica.