Centro Universitário de Brasília

P1 – PROVA 1

MENÇÃO

Disciplina: Linguagens e técnicas de programação I / Laboratório de programação
Prof.: Caio César de Melo e Silva

Ciências da Computação **Data: 29/04/2014**

Aluno(a):

Assinatura:

Informações Importantes:

1. É permitido o uso de materiais de consulta como livros, cadernos, folhas à parte ou internet.

2. Os celulares deverão estar desligados dentro das bolsas ou mochilas.

3. As mochilas ou bolsas devem ser colocadas na frente da sala.

4. No caso de o aluno tentar copiar informações ou respostas da prova de um outro aluno, ambos serão desclassificados da avaliação, sendo-lhes atribuída menção SR. Regra aplicada tanto para meios analógicos quanto digitais.

5. A ausência do aluno, enquanto a prova estiver em andamento, só irá ocorrer mediante autorização do fiscal de prova.

6. Deverá ser mantido silêncio total e absoluto durante a realização da prova.

7. A interpretação das questões é parte integrante da avaliação.

8. As questões devem ser respondidas em forma de código.

9. Ao finalizar a prova, chame o fiscal de prova para recolher suas respostas.

BOA SORTE!

Instruções para realização da prova:

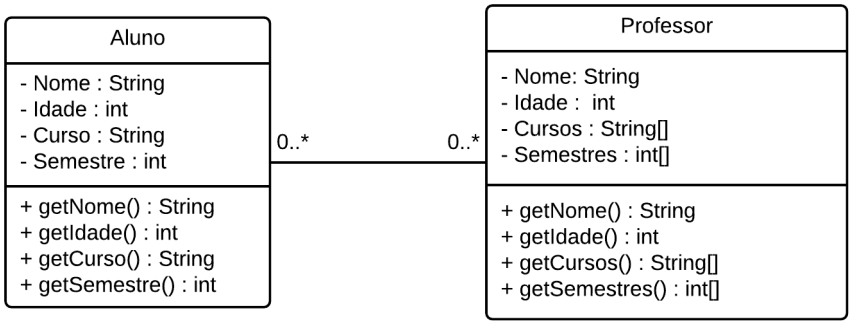
1. Na ferramenta eclipse, crie um **projeto** chamado **Prova1**. O seu código deverá ser implementado nesse projeto.

2. Crie os **pacotes** *model*, *control*, *view*, *database* e *main*.

3. Ao finalizar a avaliação, exporte o projeto para um arquivo *.zip*. O nome do arquivo deve ser "<seu_nome>.zip".

Parte 1. Conceitos Básicos (cobre 20% dos objetivos desta avaliação)

Questão 1.a Diagramas UML são muito utéis para descrever classes, independentemente de uma linguagem de programação específica. Sabendo disso, crie classes Java ou PHP ou C# baseando-se no diagrama de classes fornecido abaixo. Os métodos devem ser implementados. **(10% dos objetivos)**



Questão 1.b Apesar de não aparecerem no diagrama, os métodos para atualizar valores de atributos são fundamentais em programas concebidos no paradigma de orientação a objetos. Implemente tais métodos para todos os atributos das classes **Aluno** e **Professor**. **(5% dos objetivos)**

Questão 1.c Através dos métodos construtores é possível determinar a forma de construção de um determinado objeto. Sabendo que um aluno, para ser cadastrado **deve** possuir um nome, idade, um curso, e um semestre, e que um professor **deve** possuir um nome, idade, um conjunto de cursos, e um conjunto de semestres. Implemente os métodos construtores para as classes supracitadas. **(5% dos objetivos)**

Parte 2. Conceitos Aplicados (cobre 40% dos objetivos desta avaliação)

Para a Parte 2 da prova, construa classes denomindas ControleAluno e ControleProfessor. Além disso, implemente o código apresentado no anexo 1, esse código está disponível no espaço aluno.

Questão 2.a O padrão de projeto MVC é utilizado em larga escala para o desenvolvimento de *softwares*, tanto no âmbito da indústria quanto no meio acadêmico. Essa padrão define camadas que possuem responsabilidades específicas no que diz respeito a manipulação das informações

trafegadas em um determinado programa. Dado esse contexto, crie na classe ControleAluno um método que retorne todas as informações de um Aluno. (Sugestão de assinatura do método: “public String getTodasInformações()”) **(10% dos objetivos)**

Questão 2.b Crie um método, na classe ControleAluno, que receba como parâmetro o nome de um curso e retorne a quantidade de alunos que cursam aquele curso. (Sugestão de assinatura do método: “public int getNumeroEstudantes(String nomeCurso)”) **(10% dos objetivos)**

Questão 2.c Crie um método, na classe ControleProfessor, que receba um semestre como parâmetro e retorne uma lista com o nome de todos os professores que lecionam naquele semestre. (Sugestão de assinatura do método: “public ArrayList<String> getProfessoresPorSemestre(int semestre)”) **(10% dos objetivos)**

Questão 2.d Crie um método, na classe ControleProfessor, que receba uma idade e retorne o nome de todos os professores que possuem uma idade superior a idade fornecida. (Sugestão de assinatura do método: “public ArrayList<String> showProfessoresPorIdade(int idade)”)

Parte 3 – Extrapolação dos conceitos (cobre 40% dos objetivos desta avaliação, pode adicionar até 20% aos objetivos dessa avaliação)

Questão 3.a O diagrama UML apresentado na questão 1.a, apresenta a multiplicidade (0...*), que usualmente significa uma relação de zero ou muitos. Mais especificamente, para situação modelada, uma possível interpretação seria que um professor pode dar aula para zero ou mais alunos, e um aluno pode ter zero ou mais professores. Altere as classes **Aluno** e **Professor** para suportar essa multiplicidade indicada no diagrama. **(10% dos objetivos)**

Questão 3.b.1 Crie um **outro método construtor** na classe professor, para que seja possível criar um professor passando como parâmetro um conjunto de alunos. **(5% dos objetivos)**

Questão 3.b.2 Adicione dois novos professores, prof4 e prof5, na SimulatedDatabase que recebam, respectivamente, o conjunto de alunos {aluno1, aluno2} e {aluno3, aluno4, aluno5}. **(5% dos objetivos)**

Questão 3.c Crie classes de *view* para as operações implementadas na parte 2 dessa avaliação. As janelas devem captar informações do usuário e apresentar as respostas dos métodos para o mesmo. (Dica: as janelas e os controles devem “conversar” para trocar as informações desejadas) **(de 20% - 40% dos objetivos)**

ANEXO I

```
package database;

import model.Aluno;
import model.Professor;

public class SimulatedDatabase {

public Aluno aluno1 = new Aluno("Zé Goiaba", 18, "Jogos Digitais", 1);
public Aluno aluno2 = new Aluno("Zé Macaxeira", 23, "Ciências da Computação", 2);
public Aluno aluno3 = new Aluno("Zé Farofa", 31, "Direito", 6);
public Aluno aluno4 = new Aluno("Zéfa Cristina", 20, "Ciências da Computação", 6);
public Aluno aluno5 = new Aluno("Zéfa Marta", 21, "Ciências da Computação", 3);
public Aluno aluno6 = new Aluno("Zéfa Miriam", 35, "Relações Internacionais", 9);


public String[] cursosProf1 = {"Jogos Digitais", "Ciências da Computação"};
public String[] cursosProf2 = {"Relações Internacionais", "Direito"};
public String[] cursosProf3 = {"Ciências da Computação"};
public int[] semestresProf1 = {1,2,3};
public int[] semestresProf2 = {9,6};
public int[] semestresProf3 = {6};



public Professor prof1 = new Professor("Prof. Tomate", 56, cursosProf1, semestresProf1);
public Professor prof2 = new Professor("Prof. Cebola", 37, cursosProf2, semestresProf2);
public Professor prof3 = new Professor("Prof. Alfafa", 26, cursosProf3, semestresProf3);

}
```