Segunda prova

Características de Linguagens de Programação I

G. C. de Carvalho

ATENÇÃO: Todas as questões devem ser resolvidadas em um único projeto Netbeans nomeado "Prova2<NomeProfessor>" (por exemplo, o professor Carlos Silva criará um arquivo chamado "Prova2CarlosSilva").

ATENÇÃO: **NÃO COLOQUE COMENTÁRIOS NEM FAÇA DOCUMENTAÇÃO. NÃO COLOQUE CASOS DE TESTE.**

TEMA

O projeto consiste na implementação de um programa que controla os horários aula de professores na UERJ. Detalhes são fornecidos nas questões.

QUESTÕES:

1. Crie a classe **Professor** no pacote "backend" com as seguintes características:
   (a) (0,5 pontos) Objetos desta classe devem conter os atributos privados **matrícula**, **nome**, **sobrenome**, **cor** (classe **Color** do pacote **swing** do JDK) e **área de concentração**.
   (b) (0,25 pontos) Todos os atributos devem possuir **getters**, e cor deve possuir um **setters** público.
   (c) (0,5 pontos) O único construtor da classe recebe a matrícula, o nome do objeto e área de concentração. Caso o nome seja uma String com mais de uma palavra, a primeira se torna o atributo **nome** e o restante se torna o **sobrenome**.
   (d) (0,5 pontos) Duas instâncias de **Professor** são consideradas iguais se possuem a mesma **matrícula** ou o mesmo **nome + sobrenome + área de concentração**.

2. Crie a classe **BDProfessores** no pacote "backend" com as seguintes características:
   (a) (0,25 pontos) Objetos desta classe devem possuir uma lista (**List**) de instâncias da classe Professor e um objeto de arquivo (File) como atributos privados.
   (b) (0,25 pontos) Há um construtor que recebe uma **Collection<Professor>** e cria sua lista a partir dela, com um arquivo vazio.
   (c) (1,5 pontos) Há um construtor que recebe o nome de um arquivo e cria sua lista e seu atributo File a partir dele. Este construtor gera **erros específicos** (Exceções personalizadas) em cada um dos seguintes casos:
      i. O arquivo não existe ou não pode ser aberto.
      ii. O arquivo não está no formato correto.
      iii. Há professores repetidos no arquivo.
   (d) (1,5 pontos) Há um método que salva a lista de professores em um arquivo. Caso o atributo arquivo seja vazio, a lista é salva em um arquivo chamado "professores.csv" na pasta "bd". Caso não haja modificações em relação ao arquivo original, nada é feito. Caso contrário, um novo arquivo com mesmo nome do anterior e sufixo "_novo" é criado e a lista salva.
   (e) (0,5 pontos) Há um método que **insere** um novo Professor na lista. Este método gera um erro se o professor já pertence à lista.

(f) (0,5 pontos) Há um método que **atualiza** um Professor da lista. Este método gera um erro se o professor não pertence à lista ou se a atualização não modifica o Professor.

(g) (0,5 pontos) Há um método que **remove** um Professor da lista. Este método gera um erro se o professor não pertence à lista.

3. Crie a classe **TelaArquivoBD**, classe filha de **JFrame**, no pacote "frontend" com as seguintes características:

   (a) (0,5 pontos) Um objeto desta classe é composto por uma simples tela com um campo de texto (devidamente rotulado), um botão e um rótulo abaixo dele.

   (b) (0,5 pontos) Quando o botão é acionado, o texto que está na caixa de texto é usado para instanciar um **BDProfessores** (ou seja, a caixa de texto representa o nome do arquivo recebido).

   (c) (0,5 pontos) Caso o objeto seja criado corretamente, a tela deve ficar invisível (setVisible(false)) e uma **TelaManipulaBD** deve ser instanciada. Caso haja algum erro, uma mensagem descrevendo-o deve ser mostrado no JLabel abaixo do botão.

4. (**EXTRA - até 2,0 pontos**) Crie a classe **TelaManipulaBD**, classe filha de **JFrame**, no pacote "frontend" com as seguintes características:

   (a) Um objeto desta classe é uma tela de cadastro. Devem existir caixas de texto, devidamente rotuladas, para cada campo de professor. Deve haver um rótulo na parte inferior da tela que indica se a operação foi bem sucedida ou não. Caso não tenha sido, deve informar o motivo em vermelho. Caso contrário a mensagem de sucesso deve aparecer em verde.

   (b) Deve haver um botão que, quando pressionado, insere o professor na lista. Caso a ação seja bem sucedida, todo o conteúdo de todas as caixas de texto devem ser removidas. Caso não seja, um JLabel na parte mais baixa da tela deve indicar o motivo.

   (c) Deve haver uma **ComboBox** representando as opções disponíveis de manipulação (inserir, atualizar e remover). Pressionar o botão deve executar a operação selecionada.

5. Crie uma classe para conter o método **main** do projeto, no pacote "main". Nesta classe, deve haver um tratamento dos argumentos da main, isto é, deve ser feita a checagem sobre o vetor de Strings **args** para verificar se está de acordo com as restrições abaixo. Caso alguma das restrições seja quebrada, uma mensagem de erro conveniente deve ser mostrada ao usuário e o programa fechado:

   (a) (0,25 pontos) Caso o argumento $-$**gui** seja passado (i.e., a String "$-$gui" pertence á **args**), todas as outras restrições devem ser ignoradas.

   (b) (0,25 pontos) Caso o argumento $-$**h** ou $--$**h** ou $-$**help** ou $--$**help** seja passado, todos os outros argumentos devem ser ignorados, e uma lista com um pequena descrição dos argumentos disponíveis deve ser mostrada na tela e o programa fechado.

   (c) (0,25 pontos) O primeiro elemento de **args** é, obrigatoriamente, o nome do arquivo onde está o banco de dados, devendo ser da extensão **.csv**. Logo, é obrigatório que o tamanho de **args** seja pelo menos 1.

   (d) (0,25 pontos) Caso o argumento $-$**u** esteja na posição i de **args**, a posição $i+1$ deve conter a matrícula de um professor, a posição $i+2$ deve conter o nome de um campo da classe Professor e a posição $i+3$ o novo valor deste campo.

   (e) (0,25 pontos) Caso o argumento $-$**i** esteja na posição i de **args**, as posições seguintes devem conter a matrícula, o nome e a área de conhecimento, do novo professor.

   (f) (0,25 pontos) Caso o argumento $-$**d** esteja na posição i de **args**, a posição $i+1$ deve conter a matrícula do professor a ser

   (g) (0,25 pontos) Após o tratamento de argumentos, estando tudo correto, a operação escolhida deve ser executada e o programa fechado em seguida. Caso ocorra algum erro, este deve ser mostrado na tela para o usuário de forma coerente (sem mostrar Exceções para o usuário).removido.