INSTITUTO FEDERAL SUL-RIO-GRANDENSE – CÂMPUS CHARQUEADAS
CURSO SUPERIOR DE TECNOLOGIA EM SISTEMAS PARA INTERNET
Disciplina Programação Orientada a Objetos – Professor Carlos Souza

LISTA II - Data de Entrega: 29/12/2020 22:00:00

INSTRUÇÕES: A resolução da lista deve ser individual com consulta apenas ao material pessoal. Para a entrega, o aluno deve criar uma pasta para cada questão, e nesta pasta colocar apenas as classes da respectiva questão e algum arquivo de texto, caso haja necessidade de alguma explicação adicional. A entrega deverá ser feita pelo e-mail carlossouza@acad.charqueadas.ifsul.edu.br até as 22 horas do dia 29/12/2020, impreterivelmente, utilizando seu e-mail institucional. Após a entrega, o aluno deverá receber uma confirmação de recebimento enviada pelo professor, caso não receba até as 23 horas o aluno deve entrar em contato até as 23 horas e 59 minutos do mesmo dia, reportando que enviou e não recebeu confirmação. Trabalhos não recebidos até esses horários, definidos aqui como limite, não serão considerados. Atenção: soluções com quaisquer vestígios de cópia serão encaminhadas diretamente para o colegiado do curso para deliberação.

(Questão 01) Crie uma classe Lampada, que possua os atributos voltagem (int) e cor (String). Encapsule estes campos. Esta classe apenas deve ser instanciada se todos os parâmetros forem informados. Crie uma classe Abajour, que possua os atributos privados voltagem (int), lampada (Lampada) e estaLigado (boolean), que apenas pode ser instanciada se for informada a voltagem, e inicialmente sem lampada e estaLigado sendo false, e os métodos públicos trocarLampada, que recebe uma instancia de Lampada, estaLigado() que retorna verdadeiro ou falso caso o abajour esteja ou não ligado, removerLampada() que troca o atributo lampada para null, ligar() e desligar() que altera o atributo estaLigado, de acordo com o estado do abajour. Atente para os seguintes detalhes: A troca de lâmpada apenas pode ser realizada com o abajour desligado e por uma lampada de mesma voltagem. O abajour só pode ser ligado se tiver alguma lampada e, ao liga-lo, deve mostrar a cor da lâmpada.

Agora, crie uma classe para testar as estruturas criadas através da criação de diferentes objetos. Crie cinco objetos lampadas, sendo três com voltagem 110 e dois 220, e com cores diferentes. Crie dois objetos abajour, sendo um com voltagem 110 e outro com 220. Tente chamar o método ligar sem passar alguma lampada. Tente passar uma lampada com voltagem diferente e tente ligar. Tente passar a lampada de voltagem correta e tente novamente. Tente trocar lampada com o abajour ligado.

(Questão 02) Crie a classe Midia, que não deve permitir sem instanciada, com os atributos artista (String) e titulo (String), que devem ser encapsulados, e as classes CD, DVD e BluRay, que estendem Midia, e devem permitir serem criados de duas formas: 1) informando todos os parâmetros, e; 2) sem informar nenhum parâmetro. Para esta questão, objetos de Midia que não possuírem os atributos preenchidos, devem ser considerados “graváveis”. Crie as interfaces ReproduzCD, ReproduzDVD, ReproduzBluRay, GravaCD, GravaDVD e GravaBluRay, cada uma com um método reproduzir() ou gravar(), respectivamente, que recebe a respectiva mídia como parâmetro. Crie a classe Tocador, que não deve permitir ser instanciada e que deve ser estendida pelas classes TocadorCD, TocadorDVD e TocadorBluRay. Crie as classes GravadorCD, que deve estender TocadorCD, GravadorDVD que deve estender TocadorDVD, e, GravadorBluRay, que deve estender TocadorBluRay. A classe TocadorCD deve implementar a interface ReproduzCD, a classe GravadorCD deve implementar as interfaces GravaCD. A classe TocadorDVD deve implementar as interfaces ReproduzDVD e ReproduzCD, e a classe GravadorDVD deve implementar as interfaces GravaDVD e GravaCD. A classe TocadorBluRay deve implementar as interfaces ReproduzBluRay, ReproduzDVD e ReproduzCD, e, a classe GravadorBluRay deve implementar as interfaces GravaBluRay, GravaDVD e GravaCD. Crie, em cada uma das classes (TocadorCD, TocadorDVD, TocadorBluRay) um método reproduzir(), que deve receber como parâmetro um objeto do tipo Midia, ou melhor, filho de mídia, já que esta não pode ser instanciada, e exibir uma mensagem informando se o dispositivo pode ou não reproduzir o tipo de mídia passada, e em caso positivo, deve exibir o artista e o titulo da mídia passada. Os métodos reproduzir() apenas devem reproduzir mídias que sejam compatíveis e que não sejam “graváveis”. Já os métodos gravar() apenas devem aceitar mídias que seja “graváveis”.

Agora, crie uma classe para testar as estruturas criadas através da criação de diferentes objetos. Nela, crie dois objetos de cada uma das mídias (CD, DVD e BluRay) sendo um com os atributos preenchidos e outro com atributos vazios (graváveis). Crie também um objeto para cada Tocador (TocadorCD, TocadorDVD e TocadorBluRay) e um objeto para cada Gravador (GravadorCD, GravadorDVD e GravadorBluRay). Experimente passar cada mídia para cada método reproduzir() ou gravar() e exiba mensagens conforme cada situação.

INSTITUTO FEDERAL SUL-RIO-GRANDENSE – CÂMPUS CHARQUEADAS
CURSO SUPERIOR DE TECNOLOGIA EM SISTEMAS PARA INTERNET
Disciplina Programação Orientada a Objetos – Professor Carlos Souza

LISTA II - Data de Entrega: 29/12/2020 22:00:00

(Questão 03) Crie uma classe ContaSimples, com um atributo saldo (double) e com os métodos deposito(double valor) e saque(double valor) que devem retornar verdadeiro ou falso caso as operações venham a ser concretizadas. Essas operações devem somar ou subtrair o valor do saldo atual da conta, respectivamente. Os objetos da classe Conta podem ser criados da seguinte forma: 1) sem passar parâmetros, onde fica subentendido que o saldo inicial é 0, e; 2) passando algum valor inicial, que seria como um depósito inicial. Importante destacar que: a operação de saque apenas pode ser realizada caso o saldo seja maior do que o valor desejado. Crie objetos para testar as estruturas criadas.

(Questão 04) Crie uma classe Conta com os atributos numero (int) e saldo (double), que devem ser encapsulados. Crie uma classe Banco que possua um atributo chamado contas que seja uma coleção de objetos da classe Conta, utilizando alguma das estruturas de Collections estudadas. Nessa classe Banco, coloque os métodos abrirConta() e encerrarConta(), que respectivamente adicionam e remover instancias de Conta à lista de contas do banco. O método abrirConta() deve poder ser chamado informando um deposito inicial ou ainda nenhum parâmetro. Coloque também os métodos saque(int numeroDaConta, double valor), depósito(int numeroDaConta, double valor) e transferência(int numeroContaOrigem, int numeroContaDestino, double valor) que deverão ser chamados informando o número da conta e o valor da operação. Ao serem chamados, você deve primeiramente verificar se a conta com o respectivo número consta na lista de contas do Banco, e, em caso positivo, verificar se a conta possui saldo suficiente para realizar a operação, caso seja de saque ou transferência. Caso seja depósito, verificar apenas se a conta está na lista de contas do banco. Crie objetos para testar as estruturas criadas.

(Questão 05) Incrementando a questão anterior, na classe Conta, crie um atributo operacoes (List<String>), que seja privado, e só possa ser acessado pelos métodos inserirOperacao(String operacao) e listarOperacoes(), que permitam inserir operações que foram realizadas, e; retornem a lista das operações que foram realizadas. Ao realizar qualquer operação na classe Banco (saque, deposito ou transferência) um texto deve ser inserido na movimentação do respectivo objeto conta. Crie na classe Banco, o método extrato(int numeroDaConta), que ao ser chamado, deve buscar a respectiva conta e, se esta existir no banco, deve exibir na tela a lista de operações realizadas nesta conta. Crie objetos para testar as estruturas criadas.

(Questão 06) Incrementando a questão anterior, crie três classes SaldoInsuficienteException, LimiteValorExcedidoException e DepositoSemValorException, todas herdam de java.lang.Exception, e que indicam "Saldo Insuficiente para Saque" para caso de valor do saque ser maior do que o saldo da conta; "Limite de Valor Excedido", para saques e transferências maiores que 500 reais, e; "Não foi possível identificar o valor do depósito" (para depósitos com valor menor ou igual a 0. Em cada uma das classes, sobrescreva o método String getMessage(), colocando em seu conteúdo apenas return "mensagem específica"; onde a mensagem deverá ser referente ao tipo da exceção. Agora, informe na classe Banco, que o métodos podem gerar as seguintes exceções: deposito() e transferência() podem gerar exceções do tipo SaldoInsuficienteException e LimiteValorExcedidoException; deposito() pode gerar exceção do tipo DepositoSemValorException.

Agora, ao chamar esses métodos, exceções podem ser geradas e, portanto, devem ser tratadas. Crie objetos para testar as estruturas criadas.

(Questão 07) Crie uma classe Urna, que possua um Map<String, int> de nome de candidatos e quantidade de votos, a ser inseridos pelo método adicionarCandidato ou removidos pelo método removerCandidato(), ambos recebem um nome de candidato como parâmetro, e; e votos que devem ser incrementados pela chamada do método votar(String candidato). Crie também um método que imprima na tela o nome dos candidatos e a sua quantidade de votos.

(Questão 08) Crie classes e objetos que demonstre e explique a diferença, através de comentários nas classes e mensagens na tela, o funcionamento de alocação dinâmica (*dynamic binding*) e alocação estática (*static binding*).

(Questão 09) Crie um projeto simples, que utilize pelo menos os conceitos de herança, sobrecarga, sobrescrita, classes abstratas e interfaces. Fique a vontade para utilizar os demais conceitos e estruturas. Escreva um parágrafo explicando o seu projeto e informando onde e porque utilizou cada conceito e estrutura.