Universidade do Estado de Santa Catarina
Ciência da Computação
Programação Orientada a Objetos - Prova III – 2018 / 2

Questão 1$_{(1,5)}$ – O que é polimorfismo *adhoc* e polimorfismo universal? Qual a diferença entre eles? Cite uma subcategoria e apresente um exemplo para cada um destes dois tipos de polimorfismo.

Questão 2$_{(4,0)}$ – Usando C++, faça o que se pede:

a) $_{(0,3)}$ Crie uma classe denominada Complex, que será utilizada para representar números complexos, com dois atributos ponto flutuantes e privados: *real* (a) e *imaginário* (b) (representando: a+bi, onde i = $\sqrt{-1}$).

b) $_{(0,3)}$ Crie um método construtor público que recebe dois parâmetros (n : float) e (img : float), os quais representam o valor real (a) do número complexo e a parte imaginária (b), respectivamente.

c) $_{(0,2)}$ Crie outro método construtor público que recebe apenas um parâmetro (n : float), o qual representa a parte real (a) do número complexo e define a parte imaginária como zero.

d) $_{(0,3)}$ Crie dois métodos *getters* e dois métodos *setters* (públicos), cada qual retornam ou definem os valores da parte real e imaginária do número complexo, respectivamente.

e) $_{(0,2)}$ Crie um método público denominado *show* que não recebe parâmetros e mostra em tela a fração no seguinte formato tradicional de números complexos: "(a ± bi)", onde a letra '*a'* deve ser substituída pelo valor da parte real do número imaginário e a letra '*b*' deve ser substituída pelo componente imaginário.

f) $_{(0,6)}$ Crie uma implementação para o **operador binário *** que multiplica dois números complexos retornando um novo número complexo. A multiplicação entre números complexos é dada por:

$$(a + bi) * (c + di) = ((ac - bd), (ad + bc)i)$$

g) $_{(0,6)}$ Crie uma implementação para o **operador binário %** que realiza uma operação de divisão real entre um número complexo e um ponto flutuante. Esta operação simplesmente ignora o componente imaginário fazendo a divisão entre o termo real e o ponto flutuante, retorna um novo ponto flutuante:

$$(a + bi)\ \%\ c\ =\ a/c$$

h) $_{(0,6)}$ Crie uma implementação para o **operador unário -** que retorna um novo número complexo que possui os mesmos valores do número complexo *this*, mas com os sinais dos termos invertidos.

i) $_{(0,6)}$ Crie uma implementação para o **operador unário !** que inverte o componente real com o componente imaginário do objeto (*this*), alterando o valor do próprio objeto.

j) $_{(0,3)}$ Utilize todos os métodos e operadores implementados no método *main*.

Questão 3$_{(4,5)}$ – Usando Java, escreva uma parte de um sistema de gestão de estoque:

a) $_{(0,3)}$ Crie uma classe para modelar uma entidade de Produto, sendo que todo produto possui os seguintes atributos principais: código, nome, descrição, quantidade, tipo e valor. O tipo representa se o produto é um alimento ou uma bebida.

b) $_{(1,0)}$ Crie uma classe denominada DAO que possui como único atributo uma coleção de Produtos (escolha a coleção que desejar). Esta classe deve ser implementada com o padrão *Singleton*. Adicione ainda dois métodos públicos a esta classe: `adicionarProduto( Produto p ) : boolean` e `consultarProduto( int cod ) : Produto`. O primeiro método recebe um produto e tenta adicioná-lo na coleção, verificando primeiramente se já não existe um produto com o mesmo código. Caso um produto de mesmo código exista (não é necessário validar se os outros campos são iguais), deve-se incrementar apenas a quantidade de produtos do item já existente, caso contrário, o novo registro deve ser adicionado na coleção. Já o segundo método deve varrer a coleção tentando encontrar um registro de produto com o mesmo código passado por parâmetro. Caso encontre deve-se criar uma cópia desse objeto e retorná-lo, caso contrário, deve-se retornar *null*.

c) (0,2) Crie uma janela de menu principal com as opções de ação: Adicionar Produto e Consultar Produto.

d) (0,3) Crie uma classe principal denominada Main que inicializa o sistema mostrando a janela do menu principal. Essa classe também deverá ter um atributo público da classe DAO, permitindo a centralização e acesso aos dados que serão cadastrados e utilizados no sistema.

e) (1,2) Crie uma janela de formulário para realizar a adição de um novo produto. Garanta que todos os campos sejam preenchidos corretamente e que o usuário só possa selecionar "Alimento" ou "Bebida" como tipo de produto. Caso os valores numéricos código, quantidade e valor não sejam numéricos ou caso algum campo não seja preenchido, gere uma mensagem de erro. O formulário deve ter também um botão de cancelar operação e adicionar produto. O botão de cancelar operação deverá levar o usuário até o menu principal, ao passo que o botão de adicionar produto deverá interagir com o objeto DAO da classe Main a fim de adicionar o novo produto.

f) (1,2) Crie uma janela de formulário para realizar a consulta de um determinado produto. Tal formulário deve permitir que o usuário entre com um número de código de produto e após pressionar um botão "Consultar" o formulário deverá exibir os dados referentes a esse produto, caso ele esteja cadastrado, ou uma mensagem de erro informando que um produto com tal código não está cadastrado. OPCIONALMENTE, pode-se criar um formulário estilo tabela, no qual são listados todos os produtos. Neste caso, pode-se criar um novo método na classe DAO para retornar uma cópia da coleção contendo todos os produtos cadastrados no sistema.

g) (0,3) Garanta que o sistema tenha um layout consistente e organizado, seja possível transitar entre as três janelas criadas e que o programa seja finalizado ao pressionar o botão fechar na tela de menu principal, isto é, o botão fechar dos formulários de consulta e inclusão de produtos NÃO devem fechar o programa – no máximo voltar ao menu principal.