**Universidade Federal de Campina Grande – UFCG**

**Centro de Engenharia Elétrica e Informática – CEEI**

**Unidade Acadêmica de Sistemas e Computação – UASC**

Disciplina: Programação 2 e Laboratório de Programação 2

**Lista de Exercícios - Composição e Polimorfismo**

## Herança

**Q1)** Para um jogo de RPG crie uma sala que pode ser de dois tipos. Uma sala santuário, ou uma câmara de gás. Ao entrar na sala, cada tipo deve exibir uma mensagem distinta. A sala santuário apresenta a mensagem *“Descanse sua fadiga nobre aventureira(o). Aqui estás protegida(o)”*. A câmara de gás, por outro lado, exibe a mensagem *“Sua jornada acaba aqui, junto com seu último suspiro”*.

**Q2)** Crie um programa para calcular o preço de dois tipos de produtos: Alimento e Limpeza. Todos os produtos possuem um nome, uma quantidade (em inteiro), e um preço base. Cada tipo apresenta um cálculo distinto. Os produtos de Limpeza fornecem um desconto de 10% para cada produto acima de 4. Portanto, se a cliente compra 7 detergentes, a R$ 3,50 cada, no lugar de pagar (7 \* 3.5 = 24.5) ela pagará 24.5 - [0.1\*(7 - 4) \* 24.5] = 24.5 - 7.35 = 17.35.

Os produtos do tipo Alimento, por sua vez, podem conter, ou não, gluten. Se o alimento contiver gluten, o cliente consegue 50% de desconto no preço total do produto. Portanto, 5 pacotes de tapioca (que não contém gluten), a 2.40 cada, resulta num total de 0.5\* 2.4 \* 5 = 6.

**Q3)** Para começar a expandir o jogo criado na Lista 1, está na hora de começar a definir os diferentes papéis que o seu personagem pode possuir. Todo papel é construindo recebendo um valor de poder (*power*) e defesa (*armor*). Baseado nesses atributos, todos os papéis podem realizar três ações distintas: ataca(), defende() e cura(), que, respectivamente, retornam uma quantidade negativa de dano baseado no poder, retornam a quantidade de defesa, e retornam um valor positivo correspondente ao poder do papel. Posteriormente, o getAtaque e getCura serão usados junto com o atributo vida (*health points*, ou, HP) do Heroi.

São definidos três papéis distintos: Protetor, Curandeiro e Assassino. O protetor recebe um aumento de 50% no seu valor de defesa; o curandeiro recebe um bônus de 40% no seu poder ao chamar o método cura, porém seu ataque reduz em 50%; e por fim, o assassino recebe um bônus de 45% no poder de seu método atacar, e uma penalidade de 50% na sua armadura resultante do defende.

Crie as classes e o polimorfismo para os diferentes papéis. Por enquanto, não precisa criar a classe Heroi.

**Q4)** Crie classes para representar diferentes contas bancárias. Uma conta bancária deve possuir um saldo (que inicia em zero), o nome e a renda mensal de um titular e a sua renda . Além disso deve possuir os métodos depositar e sacar que, respectivamente, adiciona e remove dinheiro da conta. A conta fornece uma projeção de lucros baseado em uma taxa de juros que aumenta o saldo em uma quantidade fixa percentual que varia de acordo com as contas. Para isso, deve ser informada a quantidade de meses da projeção, e um valor referente ao investimento mensal. Por exemplo, uma projeção da conta normal (10% de juros) considerando 3 meses para uma aplicação inicial de R$ 100,00 fornece:

Mês 1 = 100 + (100\*0.1) = 110;

Mês 2 = 110 + (110\*0.1) = 121;

Mês 3 = 121 + (121\*0.1) = 133.1;

Outra funcionalidade é o empréstimo que retorna um valor baseado nas informações de cada conta.

Existem três tipos distintos de conta:

* **Conta Normal:** Os juros da projeção são de 10%, e o financiamento é equivalente à renda total do titular.
* **Conta Estudante:** Os juros são baixos (2% ao mês) para a projeção de lucro. O financiamento será a metade da renda do titular da conta.
* **Conta Poupança:** Não há financiamento (ou seja, o financiamento é zero), porém os juros da projeção sobem para 30% ao mês.
* **Conta Elite:** A taxa de juros é de 15%, e o financiamento é a soma da renda do titular e 10% de uma projeção de 10% da renda do cliente durante 5 meses.

## Composição

**Q5)** Para melhorar o seu código dos produtos, crie a classe Carrinho que é inicia vazio, e cujas funcionalidades são: adicionar, remover e consultar os produtos de um carrinho (use o nome como chave de busca). A outra funcionalidade é calcular o preço total de todos os produtos do carrinho. Também crie dois métodos utilizando downcast para retornar a coleção específica de produtos do tipo Alimento e Limpeza presentes no carrinho. Explique, como comentário do seu código, como a composição é usada no seu design, especificando que é o objeto composto, o wrapper e como foi feita a delegação (forwarding).

**Q6)** Crie a classe Banco que pode criar, armazenar e fechar contas. Além disso o banco pode fornecer o financiamento e projeção de lucro de qualquer conta, por meio do nome do titular da conta. Além disso, o banco pode também calcular o total de dinheiro necessário para fornecer o financiamento a todos os seus clientes (cada cliente possui apenas uma conta no banco).

**Q7)** Está na hora de unir o Heroi ao seu papel. Codifique uma composição de forma que cada heroi possua apenas um papel. O heroi deve possuir os atributos: nome, vida, destreza (valor de 1 a 100) e energia. O Heroi pode atacar um outro Heroi, receber um ataque com um dano específico, e curar um outro Heroi. O tipo do papel do Heroi é definido na sua construção. Todo Heroi começa com vida e energia 100. Cada ataque e cura gastam 20 de energia, e o Heroi possui um método recuperaEnergia que recarrega 50 de energia do Heroi, de forma que a Energia não pode ficar acima de 100. A vida, por outro lado, pode continuar aumentando acima de 100 à medida em que o Heroi é curado. Ao atingir 0 de vida, o Heroi não pode atacar, curar, ou defender.

Faça os métodos ataca(Heroi outroHeroi), recebeDano(int dano), cura(Heroi outroHeroi) que se comunicam, respectivamente, com os métodos: ataca, defende, cura. Note que cada método tem uma chance de ocorrer que corresponde à destreza do Heroi (1 a 100%). Caso encontre dificuldades, resolva o exercício 8 da lista 1 para entender melhor a comunicação entre os objetos do tipo Heroi.

**Q8)** Os Herois agora podem ser de três raças distintas: Elfo, Humano e Anão. Cada raça apresenta benefícios distintos. Os anões são seres robustos, e portanto ganham +30% de vida extra ao serem criados (HP). Os humanos são muito ativos e conseguem fazer mais coisa com menos energia, o que resulta em uma redução de 50% nos seus gastos de energia. Por fim, os elfos são Herois muito ágeis, e portanto podem fazer o teste de destreza duas vezes (caso a primeira não seja bem sucedida).

Inclua as diferentes raças no seu design para manter o funcionamento do Personagem e de seus Papeis.

## Composição Polimorfica

**Q9)** Agora o seu personagem pode mudar de papel de forma dinâmica. Crie os métodos: transformaEmProtetor, transformaEmCurandeiro e transformaEmAssassino que modifica o papel do Heroi. Use o debug do Eclipse para entender como a mudança de instância afeta a chamada dos métodos.

## Interface

**Q10)** Use o Comparable para ordenar suas Contas bancárias, no Banco, de acordo com a renda dos titulares da conta.

**Q11)** Use o Comparable e o Comparator no seu Carrinho para ordenar os produtos de diferentes formas. Por default eles são ordenados por preço base. Porém forneça o método que ordena seus produtos de acordo com as suas respectivas quantidades.

**Q12)** Use o Comparable e o Comparator no seu Heroi para permitir a comparação de Herois por diferentes atributos como vida e destreza. Use forwarding para retornar o poder de um Heroi baseado em seu papel para que você possa fornecer a ordenção baseado em poder e defesa.