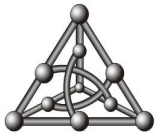
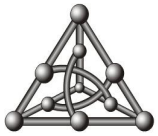


UNIVERSIDADE FEDERAL DE MATO GROSSO DO SUL
Faculdade de Computação

Orientações iniciais:

- A prova é individual e sem consulta;
- A prova vale **100 pontos**, que equivalem à nota **10**;
- Desligue todos os equipamentos eletrônicos (Smartphones, Tablets, Smartwatches, etc...)
- Consulta à Internet é proibida.
- Distribua bem o tempo gasto em cada questão. A quantidade de pontos de cada questão corresponde ao tempo que você deveria gastar em cada uma delas;
- A prova tem duração de mínima de **40 minutos** e duração máxima de **1 hora e 40 minutos**;
- Caso você tenha alguma dúvida referente ao enunciado das questões durante a prova, pergunte em voz alta. O professor não irá atendê-lo individualmente.
- Perguntas do tipo: Qual o nome daquela classe que faz tal coisa? Meu programa não está compilando, o que estou errando?. Não serão respondidas.
- Crie uma pasta com o seu nome na sua Área de Trabalho e armazene todas as implementações da prova dentro dessa pasta. Você pode utilizar um editor de texto comum ou o Eclipse. Crie dentro dessa pasta um arquivo chamado *README.txt*, que contém um descritivo de como você organizou os exercícios (1 projeto no eclipse, vários projetos no eclipse, utilizou editor de texto comum, etc.).
- Se o código não compilar não será corrigido.
- Não utilize interface gráfica.
- Comentários e Indentação serão considerados na correção.

BOA PROVA!



UNIVERSIDADE FEDERAL DE MATO GROSSO DO SUL
Faculdade de Computação

Prova 1

Linguagem de Programação Orientada a Objetos — 2014

Turma P03

1. **[50 pontos]** Escreva uma classe Java chamada Vector, que representa um Vetor de Inteiros NÃO-NEGATIVOS. Classe Vector deve armazenar informações sobre a capacidade de números inteiros que ela é capaz de armazenar. Além disso, a classe vector deve possuir uma informação que indique quantos elementos atualmente ela possui. E por fim, a classe Vector deve possui um vetor de números inteiros. Todos esses elementos devem estar devidamente encapsulados e com os corretos modificadores de visibilidade.

   É proíbido o uso de qualquer classe pronta do JAVA, por exemplo qualquer classe do pacote java.util.*

   A classe Vector deve possuir os seguintes métodos:

   (a) `print`
   Imprime todos os elementos do vetor de inteiros encapsulado.

   (b) `addAtHead`
   Adiciona um elemento no início do vetor.

   (c) `addAtTail`
   Adiciona um elemento no fim do vetor.

   (d) `add`
   Adiciona um elemento no início do vetor.

   (e) `removeAt`
   Remove um elemento do vetor em uma posição que deve ser informada na chamada do método. Caso a remoção seja possível, retorne TRUE, caso contrário retorne FALSE.

   (f) `removeTail`
   Remove um elemento do fim do vetor. Caso a remoção seja possível, retorne TRUE, caso contrário retorne FALSE.

   (g) `removeHead`
   Remove um elemento do início do vetor. Caso a remoção seja possível, retorne TRUE, caso contrário retorne FALSE.

   (h) `indexOf`
   O método recebe um número inteiro a ser procurado no vetor de inteiros e retorna a posição onde este elemento está. Se não encontrar, retorne -1.

   (i) `remove`
   O método recebe um número inteiro a ser removido no vetor de inteiros. Se a remoção foi feita com sucesso retorne TRUE, caso contrário retorne FALSE;

(j) `equals`
O método recebe um Objeto do tipo Vector e informa se o objeto passado como parâmetro é igual ao objeto que invocou ao método equals(). Se forem iguais retorne TRUE, caso contrário retorne FALSE.

Você deve criar uma classe main **VectorTest.java** para testar todos os itens anteriores.

2. **[50 pontos]** Pokemon é uma criatura que possui poderes especiais e diferentes, e são baseadas em animais. Foram criadas por Satoshi Tajiri e desenhadas por Ken Sugimori, estas criaturas logo se tornaram os preferidos entre o público e a franquia já conta com bilhões de euros em lucros. Em tempos antigos e desconhecidos, várias criaturas diferentes surgiram no planeta. Estas criaturas desenvolveram-se e logo nasceram várias sub-espécies, com vários tipos, como Normal, Eléctrico, Lutador, etc.

   - Todo Pokemon tem um nome, nível de vida (inteiro entre 0 e 100) e força (inteiro entre 0 e 100). Todo Pokemon tem um comportamento chamado `attack`. O método `attack` recebe como parâmetro um outro Pokemon que se deseja atacar, e nenhum outro parâmetro.
   - Os Pokemons subdividem-se em Pokemons da Água, Terra, Eletricidade, Fogo e Normal. Existem outros 11 grupos, mas por simplicidade vamos adotar apenas estes cinco.

   Abaixo veja uma tabela que indica o que acontece se um Pokemon da classe que está na linha atacar outro que está na coluna.

	Água	Terra	Eletricidade	Normais	Fogo
Água	-15/-10	0/-25	-25/0	0/-10	0/-25
Terra	-25/0	-15/-10	0/-25	0/-10	0/-25
Eletricidade	0/-25	-25/0	-15/10	0/-10	0/-10
Normais	-10/0	-10/0	-10/0	-15/10	-5/-10
Fogo	-25/0	-25/0	-10/0	-10/-5	-15/-10

Tabela 1: Tabela de dano ao atacar oponentes.

Os valores separados por uma '/' significam respectivamente: quanto de dano (diminuição da vida) o Pokemon da linha vai receber e se atacar um Pokemon da coluna, e quanto o Pokemon da coluna receberá de dano caso for atacado por um Pokemon da linha. Exemplo: Se um Pokemon da água atacar um Pokemon da Terra ele não recebe nenhum dano, mas causará -25 pontos de dano no oponente que é da Terra.

- Você deve modelar este problema Orientado a Objetos, implementando as classes necessárias.
- Implemente uma classe **PokemonTest.java** para realizar os testes necessários.
- Esta solução deve ser desenvolvida, obrigatoriamente, com Herança.

- Se o nível de vida do Pokemon estiver abaixo de 25%, ele desiste do ataque e não causa dano ao oponente.
- Cada Pokemon deve possuir um método chamado `iChooseYou()`, onde cada Pokemon deve imprimir apenas o seu nome na tela.