Programação Orientada a Objetos
Lista de exercícios sobre Classes Abstratas e Interfaces

1. Responda às seguintes questões:

a. O que é uma classe abstrata?
( ) Uma classe abstrata é uma sem classes filhas.
( ) Uma classe abstrata é qualquer classe pai com mais de uma classe filha.
( ) classe abstrata é classe que não pode ser instanciada, mas pode ser uma classe base.
( ) Uma classe abstrata é outro nome para "classe base".

b. O que é um método abstrato?
( ) Um método abstrato é qualquer método em uma classe abstrata.
( ) Um método abstrato é um método que não pode ser herdado.
( ) Um método abstrato é um método sem um corpo declarado e com a palavra reservada *abstract.*
( ) Um método abstrato é um método na classe filho que anula um método pai.

c. Uma classe abstrata pode definir métodos abstratos e métodos não abstratos?
( ) Não, deve ter tudo um ou outro.
( ) Não, deve ter todos os métodos abstratos.
( ) Sim - mas as classes filhas não herdam métodos abstratos.
( ) Sim - as classes filhas herdam ambas.

d. Uma classe abstrata pode ter filhos não-abstratos?
( ) Não, um pai abstrato deve ter apenas filhos abstratos.
( ) Não, um pai abstrato não pode ter filhos.
( ) Sim, todas as crianças de um pai abstrato devem ser não-abstratas.
( ) Sim, um pai abstrato pode ter filhos abstratos e não-abstratos

e. Qual dos seguintes é FALSO sobre classes abstratas em Java.
( ) Se derivarmos uma classe abstrata e não implementarmos todos os métodos abstratos, a classe derivada também deve ser marcada como abstrata usando a palavra-chave 'abstract'.
( ) As classes abstratas podem ter construtores.
( ) Uma classe pode ser abstrata sem nenhum método abstrato.
( ) Uma classe pode herdar de várias classes abstratas.

f. O que a seguir é verdadeiro sobre interfaces em java.
( ) Uma interface é semelhante a uma classe, mas só pode conter membros públicos, estáticos e atributos finais (ou seja, constantes)
( ) .... métodos públicos, abstratos (ou seja, apenas cabeçalhos de métodos, sem corpos)
( ) Pode-se criar uma instância de interface.
( ) Uma classe pode implementar várias interfaces.
( ) Muitas classes podem implementar a mesma interface.

g. Escolha a opção incorreta:
( ) Uma classe pode implementar mais de uma interface
( ) Uma classe pode estender mais de uma classe
( ) Uma interface pode estender mais de uma interface
( ) Uma interface não pode estender uma classe

h. Uma subclasse é colocada em um pacote diferente da super-classe. Você deseja permitir que a subclasse acesse um método definido na super-classe. O(s) especificador(es) de acesso correto(s) para este método é/são:
( ) apenas público
( ) protegido apenas
( ) público e protegido
( ) private, público e protegido

i. Os únicos campos que podem aparecer em uma interface devem ser declarados estático ou final.
( ) Verdade
( ) Falso

j. Os métodos em uma interface são implicitamente *protected.*
( ) Verdade
( ) Falso

k. Uma interface pode herdar de uma outra interface.
( ) Verdade
( ) Falso

l. Uma classe abstrata pode herdar de uma interface.
( ) Verdade
( ) Falso

m. Uma classe pode implementar uma interface.

( ) Verdade
( ) Falso
n. *public interface Hockey extends Sports, Football*. Isso é correto?
( ) Verdade
( ) Falso

2. Escreva uma classe abstrata Forma com os seguintes partes:
- Atributo: numLados
- Construtor: inicialize numLados
- Métodos concretos: get e set
- Métodos abstratos: getArea (), getPerimetro()
2.1. Escreva uma subclasse concreta Retangulo
- Atributos: largura, altura
2.2. Escreva uma subclasse concreta Triangulo
- atributos: base, altura
2.3. Em outra classe, escreva um método *main()* para definir um retângulo e um triângulo.

3. Escreva a interface *Redimensionavel*
- Tem um método redimensionar (double x) que redimensiona uma forma pelo fator x
3.1 Faça *Retangulo* implementar *Redimensionável*
3.2 Escreva um método *main()* para:
- Definir um retângulo (largura = 2, altura = 3)
- Imprima a área e o perímetro do retângulo
- Redimensionar o retângulo por fator de 2
- Reimprimir a área e o perímetro do retângulo
3.3 Polimorfismo: modifique o método main() acima para:
    - Criar um retângulo e um triangulo
    - Adicioná-los a um ArrayList de *Forma* (pesquise sobre ArrayList)
    - Iterar através das Formas no ArrayList:
        a) Se Forma for Redimensionável, redimensioná-la por um fator de 0,5
        b) Imprimir perímetro e área

4) preencha as lacunas, conforme se trate de classes (A)bstratas ou (I)nterfaces:

( ) Atributos estáticos e de instância
( ) Métodos concretos e/ou abstratos
( ) Atributos estáticos finais (constantes)
( ) Todos os métodos são abstratos
( ) Herança única (via **extends**)
( ) Possuem construtor
( ) Herança múltipla (via **implements**)
( ) Não possuem construtor

5) Método *toString().* Pesquise sobre esse método da classe *Object* e sobrescreva-o para as classes concretas Retângulo e Triângulo. A implementação do mesmo deve imprimir na tela a área e o perímetro do objeto em questão.

6) A seguinte interface é válida?

```
public interface Marcador {
}
```