Universidade Federal de Itajubá
Campus Itabira

Programação Orientada a Objetos

Lista de exercícios 2

1. Considere uma classe que armazena dois atributos privados: um vetor de 10 células inteiras e a quantidade de células deste vetor que estão ocupadas. Agora considere que esta classe contém um método *calculaMedia*, que retorna um valor inteiro contendo a média dos valores armazenados pelo vetor. Caso o método *calculaMedia* seja invocado e a quantidade de valores armazenadas seja zero, qual o valor a ser retornado? Como resolver este problema? Implemente em `Java` uma classe com a solução e um método *main* para testá-la.

2. Implemente a classe **Agenda**, que armazena uma lista de contatos, cada um contendo as seguintes informações: nome, email, telefone residencial, telefone celular e endereço. Implemente na classe Agenda um método *add*, que adicione um novo contato e o método *save*, que receba como parâmetro um nome de um arquivo e armazene as informações de todos os contatos no arquivo. Obs.: as informações do endereço devem ser armazenadas por uma classe Endereco.

3. Implemente a classe **Par** que armazena como atributos privados dois valores correspondentes a um par ordenado no $\mathbb{R}^2$. Implemente a classe **ListaPar** que armazene uma lista de pares ordenados (pode usar qualquer estrutura de Java para isto). A classe **ListaPar** deve ter um método *ordena* que realiza a ordenação dos pares em ordem crescente. Obs.: Sejam $p_1 = (x_1, y_1)$ e $p_2 = (x_2, y_2)$ dois pares ordenados quaisquer, diz-se que $p_1 < p_2$ se $x_1 < x_2$ ou $x_1 = x_2$ e $y_1 < y_2$.

4. Considerando que a seguinte interface esteja disponível

```
public interface Max { public boolean maior(Max m); }
```

implemente uma classe genérica denominada **Heap** composta por um vetor de 10 elementos e um valor inteiro, que determina quantas posições deste vetor estao ocupadas. Os elementos do vetor devem **obrigatoriamente** ter implementado a interface Max. Implemente um construtor para esta classe e dois métodos: *adicionar* que recebe um elemento, o coloca no vetor (caso exista espaço disponível) e retorna verdadeiro ou falso dizendo se a operação foi realizada com sucesso ou não; e *remover* que retorna o maior elemento do vetor (ou **NULL** caso nao existam elementos no vetor).
Obs.: A cada operação, os elementos no vetor do **Heap** devem satisfazer a seguinte propriedade: o maior elemento sempre se encontra na primeira célula do vetor.

5. Defina **classe abstrata** e **interface**, elucidando suas diferenças e apresente um examplo (código) onde cada uma pode ser usada.

6. Considerando o tratamento de exceções em Java, defina a semântica das palavras-chave: **throws**, **throw**, **try ... catch**, **finally**. Codifique um exemplo qualquer onde todas elas são usadas.

7. Implemente uma classe `Usuario` em Java para realizar o cadastro de um usuário com as informações: **nome**, **idade** e **cpf** (que podem ser representadas da maneira mais conveniente). O construtor da classe deverá validar os valores dos atributos **idade** e **cpf**. Caso algum destes atributos contenha valores incorretos, uma exceção deverá ser lançada e tal fato informado ao usuário (explicitando o erro ocorrido). Para isto, seu programa deverá implementar uma classe que represente estas exceções e usá-la na validação. Implemente também um método *public static void main* para testar a classe.

- **Obs1.:** Considere válida uma idade entre 1 e 120 anos
- **Obs2.:** Embora exista uma regra real para definir se um número de CPF pode ou não ser válido, assuma nesta questão que para um CPF ser válido os 2 últimos dígitos sejam iguais à soma dos 9 primeiros (assuma que o usuário sempre digitará 11 digitos para o cpf).

8. Implemente uma classe `ListaUsuarios` que armazene como atributos uma lista (de tamanho indeterminado) com objetos do tipo `Usuario` da questão (1), além da quantidade de usuários jovens (idade até 17 anos), adultos (idade entre 18 e 60 anos) e idosos (idade acima de 60 anos). Implemente o construtor da classe `ListaUsuarios` e um método `addUsuario`, que receba como parâmetro um objeto do tipo `Usuario` e o insira na classe.

9. Implemente uma função em java que receba um vetor de objetos de qualquer classe (desde a classe tenha implementado a interface Comparable$< T >$) e ordene os elementos deste vetor. Mostre como seria a implementação da interface Comparable$< T >$ na classe Usuário, para ordená-los pela idade.

10. Responda V (verdadeiro) ou F (falso) para cada uma das afirmações a seguir, justificando quando sua escolha for F.

1. ___ Toda classe abstrata deve ter um método abstrato
2. ___ Toda classe que contenha um (ou mais) método(s) abstrato(s) deve ser abstrata
3. ___ Classes abstratas não podem ser instanciadas nem apontar para objetos
4. ___ Uma classe que estende uma classe abstrata se torna automaticamente abstrata
5. ___ Uma classe que estende uma classe abstrata deve obrigatoriamente implementar seus métodos abstratos
6. ___ Em uma interface não é possível declarar membros de dados, apenas métodos e constantes
7. ___ Uma classe que implementa uma interface com métodos se torna automaticamente abstrata