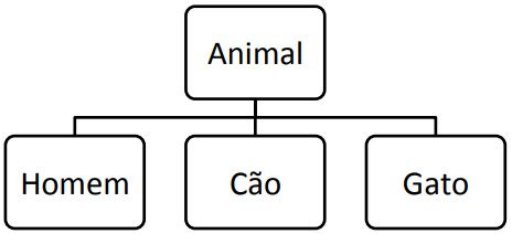
Análise e Desenvolvimento de Software
Programação Orientada a Objetos
Prof: Vinícius Fritzen Machado

INSTITUTO FEDERAL
Rio Grande do Sul
Campus Osório

Lista de Exercícios 04

1. Desenvolva o seguinte programa:
    a. Faça uma classe Animal com um método abstrato “fala”
    b. Faça as classes Homem, Cao e Gato, herdando de animal, redefinindo o método “fala” para retornar “OI”, “Au au”, e “Miau”, respectivamente.
    c. Crie um vetor de 10 Animais e instancie os três tipos de subclasses nesse vetor.
    d. Faça um loop por todos os animais do vetor, pedindo para eles falarem.



2. Faça um programa que calcule a área de uma figura geométrica. Aceite quatro tipos de figura geométrica: quadrado, retângulo, triângulo, e círculo. use herança e polimorfismo.

3. Identifique as classes e implemente um programa para a seguinte especificação: “O supermercado vende diferentes tipos de produtos. Cada produto tem um preço e uma quantidade em estoque. Um pedido de um cliente é composto de itens, onde cada item especifica o produto que o cliente deseja e a respectiva quantidade. Esse pedido pode ser pago em dinheiro, cheque ou cartão.”

4. Faça um programa para representar a árvore genealógica de uma família. Para tal, crie uma classe Pessoa que permita indicar, lám de nome e idade, o pai e mãe. Tenha em mente que pai e mãe também são do tipo Pessoa. Crie um método estático na aplicação que recebe por parâmetro uma Pessoa e mostra todas as gerações passadas.

5. Crie uma classe que implementa um array de escrita única de valores de um tipo numérico qualquer. Um array de escrita única é um array cujos elementos só possam ser modificados uma única vez. Para a implementação, devemos ter, para cada elemento do array, um valor booleano associado que diz se o elemento pode ser modificado ou não. Quando instâncias dessa classe forem criadas, todos os elementos do array poderão ser modificados, mas assim que um elemento for

modificado pela primeira vez o seu valor booleano associado será modificado de forma que da próxima vez que o elemento for modificado nada ocorrerá. Crie também uma aplicação que demonstre o uso desta classe.

6. Usando o exercício anterior como base, crie uma classe que implemente um array cujos elementos só podem ser modificados um certo número limitado de vezes. Quando instâncias dessa classe forem criadas, elas devem receber um valor que diz quantas vezes um dado elemento do array encapsulado pode ser modificado. Dica: O valor deve ser o mesmo para todos os elementos do array, mas cada elemento individual do array poderá ser modificado o número de vezes que for especificado

7. Desenvolver uma aplicação que simule uma Agenda.
    a. Implementar a classe Contato (nome, email, telefone)
    b. Implementar a classe Agenda
    c. Implementar o método verContatos
    d. Criar uma aplicação Main para testar a Agenda

    Parte II:
    a. Fazer uma classe abstrata Ordenador com a lógica de ordenação, porém chamando um método abstrato compara(Pessoa, Pessoa)
    b. Criar duas classes extras, OrdenadorNome e OrdenadorEmail, que herdam de Ordenador e implementam o método abstrato compara(Pessoa, Pessoa)
    c. Instanciar o ordenador apropriado para listar os contatos por nome ou por email de acordo com a vontade do usuário

    Parte III:
    a. Integre a solução da Agenda com um sistema de arquivos que carrega os contatos quando o programa é executado e salva ao terminar.

8. Implemente a hierarquia de classes ContaBancaria (superclasse), ContaCorrente (com senha, número, saldo e quantidade de transações realizadas) e ContaPoupanca (com senha, número, saldo e taxa de rendimento).

    • quando uma ContaBancaria for criada, informe a senha da conta por parâmetro.
    • na classe ContaBancaria, crie os seguintes métodos abstratos:
        saca(double valor)
        deposita(double valor)
        tiraExtrato()

    • nesta mesma classe, crie o método alteraSenha, que recebe uma senha por parâmetro e deve confirmar a senha anterior (via teclado), e somente se a senha anterior estiver correta a senha recebida por parâmetro deve ser atribuída.

    • implemente os métodos abstratos nas classes ContaCorrente e ContaPoupanca.

    • crie os métodos de acesso para os atributos de ContaCorrente e ContaPoupanca.

    PARTE II - Crie uma classe de teste para testar a hierarquia da descrição acima.

• pergunte (via teclado) quantas contas o usuário deseja criar e crie-as (com a utilização de arrays para armazenar as contas.

• a cada conta criada, pergunte ao usuário se trata-se de uma ContaCorrente ou de uma ContaPoupanca, e crie a conta de acordo com o informado pelo usuário.

• após as contas terem sido criadas, informe a taxa de rendimento de cada ContaPoupanca armazenada.
• realize saques, depósitos e extratos nestas contas.

• imprima a quantidade de transações realizadas nas contas correntes e as taxas de rendimento das contas poupança.

9. Dado o trecho de código abaixo, determinar:
    a. Todas as chamadas a função f
    b. Saída do programa
    c. Uma breve explicação sobre a funionamento do método f.

```
1  public class ExercicioRecursivo {
2
3      public static int i = 0;
4
5      public static int f(int v) {
6
7          if (v == 0 || v == 1) {
8              return 1;
9          }
10         i++;
11         return f(v-2) + f(v-1);
12     }
13
14
15     public static void main(String[] args) {
16         System.out.println(f(6));
17         System.out.println(i);
18     }
19
20 }
```

10. Para pesquisar: qual a diferença entre abstração, encapsulamento e modularidade?