Prof. Kalil Araujo Bispo
Programação Orientada a Objetos

Lista de Exercício 6 – Pilares da Programação Orientada a Objetos (Parte 1)

1. Crie uma classe Pessoa com atributos para nome e data de nascimento. Crie também um construtor padrão e um construtor parametrizado, com todos os atributos. Adicione métodos de acesso apropriados. Em seguida, crie uma subclasse Alunos, com os atributos para 3 notas. Crie também seus construtores e métodos de acesso parametrizados. Depois crie uma classe com um método main() para mostrar a média das notas de um aluno, seu nome e sua data de nascimento.

2. O trecho de código a seguir é valido em Java
   Integer i = new Integer(10);
   Object [] data = {“Lista 5”, new Object(), i}

   Faça um programa em Java que informe ao usuário que tipo de objeto ele está acessando no array.

3. Crie a classe Imovel, que possui um endereço e um preço. Em seguida, crie uma classe ImovelNovo, que herda Imovel e possui um adicional de 10% no preço. Depois, crie métodos de acesso e impressão deste valor adicional. Crie uma classe ImovelVelho, que herda Imovel e possui um desconto de 30% no preço. Crie métodos de acesso e impressão para este desconto. Por fim, faça um programa que crie um imóvel. Peça para o usuário digitar 1 para novo e 2 para velho. Conforme a definição do usuário, imprima o valor final do imóvel.

4. Criar uma classe Conta, que possua um saldo e métodos para verificar o saldo, depositar e sacar. Adicionar um método na classe Conta, que atualiza essa conta de acordo com uma taxa percentual fornecida. Criar duas subclasses da classe Conta: ContaCorrente e ContaPoupanca. Ambas terão o método atualizar reescrito: A ContaCorrente deve atualizar-se com o dobro da taxa e a ContaPoupanca deve atualizar-se com o triplo da taxa. Além disso, a ContaCorrente deve reescrever o método depositar, a fim de retirar uma taxa bancária de dez centavos de cada depósito. Faça um programa em Java que crie uma conta-corrente e uma poupança, atualize seus dados, faça depósitos e saques, e por fim, exiba os valores dos saldos.

5. Crie a classe Pessoa com os atributos nome e dataNascimento. A classe Pessoa deve conter:

a) Um construtor que recebe como parâmetros duas strings e inicializa os campos nome e dataNascimento
b) O método toString, que não recebe parâmetros e retorna um objeto da classe String na seguinte forma:
   Nome: <nome da pessoa>
   Data de Nascimento: <data de nascimento da pessoa>

Crie a classe PessoaIMC que herde da classe Pessoa e contenha os atributos peso e altura, ambos do tipo double. O construtor desta classe deve receber como parâmetros duas strings e dois valores do tipo double e inicializar os campos nome, dataNascimento, peso e altura. A classe PessoaIMC deve conter os seguintes métodos:

c) calculaIMC() que recebe como parâmetros a altura e o peso e retorna um valor do tipo double correspondente ao IMC (Índice de Massa Corporal = peso / altura ao quadrado) calculado.
d) resultIMC() que não recebe parâmetros e retorna uma String com o IMC da pessoa
e) O método toString() desta classe deve retornar uma string da seguinte forma:
   Nome: <nome da pessoa>
   Data de Nascimento: <sua data de nascimento>
   Peso: <seu peso>
   Altura: <sua altura>

Crie as classes Homem e Mulher, herdeiras de PessoaIMC. Cada uma deve reimplementar o método resultIMC(), que realiza o calculo do IMC e exibe uma mensagem do resultado acordo com o valor obtido.

Para Homem:	Para Mulher:
IMC < 20.7: Abaixo do peso ideal	IMC < 19: Abaixo do peso ideal
20.7 < IMC < 26.4: Peso ideal	19 < IMC < 25.8: Peso ideal
IMC > 26.4: Acima do peso ideal	IMC > 25.8: Acima do peso ideal

Crie uma classe para o programa principal, com o método main(), que crie instâncias das classes Homem e Mulher e armazene essas instâncias em um array do tipo PessoaIMC, com 5 posições. O programa deve preencher o array com objetos apropriados. Após o armazenamento de todos os objetos, o programa deve ler cada posição do array, imprimindo os dados do objeto ali contido, calculando e exibindo seu IMC.

6. Criar uma estrutura hierárquica que contenha as seguintes classes: Veiculo, Bicicleta e Automóvel. Os métodos da classe Veiculo possuem a seguinte assinatura:
   a) listarVerificacoes()
   b) ajustar()
   c) limpar()

Todos os métodos exibem uma mensagem nesse formato: *Classe.método()*. Estes métodos são reimplementados nas subclasses Automóvel e Bicicleta. Acrescentar na classe Automóvel o método mudarOleo().

Para desenvolver a classe Principal, que é apresentada a seguir, é necessário criar também a classe Oficina, que terá dois métodos:

d) proximo(), que retorna aleatoriamente um objeto do tipo bicicleta ou automóvel
e) manutencao(Veiculo v), que recebe como parâmetro um objeto do tipo Veiculo e chama todos os métodos definidos na classe Veiculo. Se o veiculo for do tipo Automóvel, deve também chamar o método mudarOleo()

```
public class Principal{
    public static void main(String args[]) {
        Oficina o = new Oficina();
        Veiculo v;

        for(int i = 0; i < 5; ++i){
            v = o.proximo();
            o.manutencao(v);
        }
    }
}
```

```
//Código que gera números aleatórios

Random r = new Random();
int valor = r.nextInt();
```