EMA915115 – Linguagens Comerciais de Programação - 2022
Prof. Lucas Guerreiro

Lista de Exercícios 01

Data limite de entrega: 15/05/2022

Atenção: Os códigos deverão ser enviados via GitHub. Criar um repositório em seu GitHub com o nome LCP2022 e criar uma pasta Lista01, os arquivos .java deverão estar dentro desta pasta. Enviar um email com o título LCP2022 e com os seguintes dados: Nome Completo, RA e o link do seu GitHub para lucas.guerreiro@unesp.br.

1) Escreva um programa que receba um número inteiro *i* entre 1 e 20 como entrada pelo usuário (utilizando *Scanner*) e que imprima asteriscos de 1 a *i* no console em cada linha. Por exemplo, se a entrada for 3, a saída deve ser:

```
*
**
***
```

Caso *i* não seja um número entre 1 e 20 deve-se exibir uma mensagem informando ao usuário que o número inserido é inválido. O nome da classe deve ser *Exercicio01.java*.

2) Escreva um programa que irá receber do usuário três números e irá calcular e exibir em métodos separados: a soma deles, o produto dos três números, a média deles, os números ordenados (usando *if/else*). O nome da classe deve ser *Exercicio02.java*.

3) Escreva um programa de cálculo de contribuição que envolva contribuições em cima do salário do funcionário que serão pagam tanto pelo próprio funcionário como pela empresa. Esta contribuição será feita somente em cima de um salário de até R$5000, ou seja, se o funcionário recebe até R$5000, a contribuição será em cima do salário todo, mas caso ele receba R$6000, por exemplo, a incidência de imposto recairá sobre R$5000, mas os R$1000 excedentes não serão impactados. A cobrança leva em conta a idade do funcionário e segue a Tabela:

Idade do Funcionário	Contribuição do Funcionário	Contribuição da Empresa
Até 30 anos	20%	18%
Entre 30 e 40 anos	18%	15%
Entre 40 e 50 anos	12%	15%
Entre 50 e 60 anos	7%	10%
Acima de 60 anos	3%	4%

Com isso, o programa irá utilizar como atributos o *nome*, *idade* e *salário* de um funcionário e fará o cálculo das contribuições da empresa e do funcionário conforme as regras definidas

anteriormente, bem como qual será o salário que o funcionário irá receber descontando o valor da sua própria contribuição. No método *main* da classe *Exercicio03*, implementar ao menos 1 objeto para cada faixa etária (definindo no corpo do código e não recebendo via Scanner) e exibir as informações através da sobreposição do método *toString*.

4) Implementar o conceito de herança considerando uma superclasse *Animal* e subclasses *Cachorro*, *Gato* e *Passaro*. As características comuns a todos os animais devem estar na superclasse *Animal*, enquanto métodos específicos devem ser implementados nas subclasses conforme regras abaixo:

- Cachorro: Possui nome e idade, emite som "Au au", pode correr.
- Gato: Possui nome e idade, emite som "Miau", pode correr e pular.
- Passaro: Possui nome e idade, emite som "Piu", pode voar.

Além disso, criar uma classe para testar os animais com o nome *ChamaAnimal* que deverá instanciar objetos referentes a cada um e que use seus respectivos métodos com polimorfismo, ou seja, para cada animal os métodos devem ser chamados com o mesmo nome.

5) Utilizando os conceitos de classe abstrata, crie classes para uma biblioteca. A classe abstrata base deverá se chamar *Livro* e terá um atributo *titulo*, sendo que o método *getter* deste atributo será implementado pela classe abstrata. A classe abstrata terá também a assinatura do método *setter* do *titulo* e será implementada pelas classes herdadas. Implementar duas classes que herdarão de Livro: *LivroGratuito* e *LivroPago*. A classe *LivroGratuito* irá definir o atributo *titulo* ao implementar o método *setter* com um parâmetro *String*. Na classe *LivroPago* devemos definir também o método *setter* do atributo *titulo* e também métodos *getter/setter* para o valor do livro. Criar, por fim, uma classe *Biblioteca* que será utilizada para criar instâncias de *LivroGratuito* e *LivroPago* e definir seus atributos bem como exibir em tela. Não há necessidade de uso de Scanner neste exercício, tudo pode ser definido no método principal da classe *Biblioteca*.

6) Criar uma interface que se chama *Geometria* e que terá os métodos *defineArea()* e *definePerimetro()*. Além disso, criar 4 classes que implementem a classe *Geometria*, sendo as classes denominadas: *Quadrado*, *Retangulo*, *Triangulo* e *Circulo*. Cada uma destas classes deverá ter um construtor que receberá as medidas dos objetos e demais métodos necessários. Deve-se solicitar o menor número possível de parâmetros para os construtores, por exemplo: para o quadrado vamos solicitar apenas um lado. Por fim, criar uma classe que se chama *TestaGeometria* e que deverá criar objetos das classes implementadas, criar ao menos 2 objetos de cada classe e exibir os valores de área e perímetro de cada um deles.

7) Criar um programa para calcular a soma de números no formato: $x + y^2 + z^3 + w^4$, através do método *Math.pow()*. O programa deve aceitar entre 1 e 4 parâmetros e calcular de acordo com o número de entradas, ou seja, se receber 2 parâmetros, por exemplo, deverá então calcular apenas $x + y^2$. O nome da classe deverá ser *Exercicio07.java*.

8) Criar um programa para identificar se um número atende a seguinte premissa: seja *i* a quantidade de dígitos em um número inteiro *x*, se a soma de cada dígito elevado a potência *i* for

igual a *x*, a premissa é verdadeira e deve-se exibir que o número é válido, caso contrário, informar que o número é inválido. Por exemplo: o número *153* atende a premissa, pois: $1^3 + 5^3 + 3^3 = 153$. O nome da classe deverá ser Exercicio08.java.

Observação: Poucos números inteiros são válidos. Exemplos de números que o resultado será verdadeiro: *371, 9474, 54748*.

Dica: Notem que podemos calcular divisões e resto em Java da seguinte forma:

```
int a = 12/7; // a = 1
float b = (float) 12/7; // b = 1.7142857
int c = 12%7; // c = 5
```