Programação Orientada a Objetos

Prof. Paulo Henrique Pisani

Prof. Saul de Castro Leite

http://professor.ufabc.edu.br/~paulo.pisani/

UFABC

julho/2018

Exercício 1 – Figuras Planas

Ricardo Drudi (PRAE)

Você ficou encarregado de criar um sistema para calcular a área e o perímetro de **figuras planas convexas**. Seu trabalho é:

- Criar uma hierarquia de classes que inclua **círculo**, **triângulo**, **quadrado** e **retângulo**.
- As figuras triângulo, quadrado e retângulo devem ser representadas pelo comprimento de seus lados. O círculo será representado pelo seu raio.
- Crie métodos para retornar as medidas de área e perímetro (note que para o triângulo, é possível calcular a área conhecendo apenas seus lados com a *fórmula de Heron*).
- Sobrecarregue o método toString() para mostrar informação detalhada sobre cada figura (nome, área e perímetro).

UFABC

Atenção: todas as figuras devem estar no **pacote geometria**.

Exercício 1 – Figuras Planas

Crie um programa que tenha um vetor de figuras planas. Nesse vetor, instancie uma figura de cada tipo e imprima na tela as informações de cada figura.

O programa ficará no pacote “criafiguras”.

Este é o mesmo exercício da última aula prática. Mas agora vimos classes abstratas e podemos aprimorar o código.

Faça o download do código da última aula e aplique o conceito de classes/métodos abstratos.

UFABC

Exercício 2

Um sistema usado na Empresa de Entregas ABC depende da implementação de uma classe para guardar objetos da classe Encomenda. Contudo, um funcionário perdeu o código fonte da classe! Por sorte, os projetistas do sistema guardara a interface que essa classe implementava:

```
public interface GuardaEncomenda{
    void adiciona(Encomenda e); //adiciona uma nova encomenda (não aceita
                                //  encomenda duplicada)
    int getSize(); //retorna o número de Encomendas (não nulas)
    void ordena(); //ordena as Encomendas (ponteiros para null vêm por último)
}
```

O seu objetivo é fazer uma nova implementação (seguindo esta interface). O nome da sua classe deve ser **MeuGuardaEncomenda** (para simplificar o exercício, você pode usar um vetor de Encomendas de tamanho máximo 1000 para guardar os objetos).

A classe Encomenda possui o método **compareTo()**, que pode ser usado para **comparação** de instancias da classe Encomenda:

```
public int compareTo(Encomenda e)
```

Retornos do método:

- -1: objeto passado como argumento > instância atual
- 0: objeto passado como argumento = instância atual
- +1: objeto passado como argumento < instância atual

Exercício 2

Além de implementar a interface, o objeto MeuGuardaEncomenda também sobrescreve o método **toString():**

```
@Override
public String toString(){
     String ret = "";
     for(int i = 0; i < tamanho; ++i){
          ret = ret + vetorEncomendas[i] + ", ";
     }
     return ret;
 }
```

Neste método, tamanho é quantidade de encomendas não nulas no vetorEncomendas.

Para testar sua implementação, baixe os arquivos (bytecode) *SistemaABC.class*, *Encomenda.class e GuardaEncomenda.class* e teste sua implementação (disponíveis no site da disciplina). O resultado deve ser o seguinte:

Enc.#5, Enc.#36, Enc.#40, Enc.#41, Enc.#56, Enc.#62, Enc.#78, Enc.#79, Enc.#83, NUM:99230

UFABC

Exercício 3 – Plano errado

- O professor ABC resolveu escrever o código de um sistema para um plano de aulas (link “PlanoErrado.zip”, no site da disciplina). Entretanto, o código está com vários erros... Você poderia ajudar o professor a corrigir seu programa?

Faça apenas mudanças pontuais, não é preciso reestruturar o programa!

Importante: você pode alterar modificadores de acesso, mas altere o menos possível e tente manter o encapsulamento! E não adicione atributos!

UFABC

Exercício 3 – Plano errado

- A saída do programa deveria ser essa:

```
Plano da disciplina: POO
- [Teorica] Introducao
- [Prova] Avaliacao 1
- [Pratica] Classes
- [Teorica] Construtor
- [Teorica] Polimorfismo
- [Projeto] Avaliacao 2
- [Prova] Avaliacao 3
Formula atual = 0.5 x Avaliacao 1 + 0.5 x Avaliacao 2 + 0.8 x Avaliacao 3

Plano da disciplina: Fisica
- [Teorica] Introducao
- [Pratica] Classes
- [Teorica] Construtor
- [Teorica] Polimorfismo
- [Projeto] Avaliacao 2
Formula depois de cancelar provas = 0.5 x Avaliacao 2
```

UFABC