ESCOLA POLITÉCNICA DA UNIVERSIDADE DE SÃO PAULO
Departamento de Engenharia de Computação e Sistemas Digitais

PCS 3111 - Laboratório de Programação Orientada a Objetos para Engenharia Elétrica

Aula 08 – Classes Abstratas, Herança Múltipla e métodos e atributos estáticos

Atenção

1. Código inicial a ser usado na resolução dos exercícios encontra-se **disponível no e-Disciplinas**.
2. Os nomes, os atributos, os métodos, e as respectivas assinaturas das classes dadas **devem seguir o especificado** em cada exercício para fins de correção automática.

AVISO: para evitar problemas de compilação no Judge, envie os exercícios à medida que implementá-los!

Exercício 1

a) Modifique a classe **Recurso**, de modo a torná-la abstrata. Adicione o método

```
double getCusto(int dias)
```

que deve ser um método abstrato.

b) Implemente a classe **Pessoa**, derivada de **Recurso** e concreta, com os seguintes métodos públicos específicos à classe:

```
Pessoa(string nome, double valorPorHora, int horasDiarias);
virtual ~Pessoa();

virtual double getValorPorHora();
virtual int getHorasDiarias();
```

Uma **Pessoa** recebe em seu construtor os parâmetros **valorPorHora** e **horasDiarias**, que devem ser retornados pelos *getters* respectivos.

O método `double getCusto(int dias)` deve calcular o custo do uso do recurso durante uma quantidade de dias, de forma que:

$$custo = dias * horasDiarias * valorPorHora$$

Exercício 2

a) Implemente a classe **FuncionarioUSP** (cuja definição é entregue – adicione-a ao projeto), que possui um número USP.

```
FuncionarioUSP(int nusp);
virtual ~FuncionarioUSP();

int getNusp();
```

b) Além de ser derivada de **Recurso**, faça a classe **Pessoa** também ser derivada de **FuncionarioUSP**. O construtor da classe Pessoa será modificado para receber o número usp como parâmetro:

```
Pessoa(string nome, double valorPorHora, int horasDiarias, int nusp);
```

Exercício 3

Considere agora que a classe **Pessoa** possua uma sobrecarga de seu construtor, que não receba o parâmetro **valorPorHora** (ou seja, a classe terá dois construtores).

Adicione e implemente os métodos, deixando os já existentes:

```
Pessoa(string nome, int horasDiarias, int nusp);

virtual bool recebeValorPadrao();
static void setValorPorHoraPadrao(double valor);
static double getValorPorHoraPadrao();
```

- Ao se chamar esse construtor, a **Pessoa** deve receber um valor por hora padrão. Utilize o método **getValorPorHoraPadrao** para obtê-lo.
- Inicialmente, o valor padrão deve ser igual a **8**;
- Caso a **Pessoa** utilize o valor por hora padrão, o método **recebeValorPadrao** deve retornar **true**.
- O método **setValorPorHoraPadrao** permite alterar o valor por hora de todas as **Pessoas** que recebem o valor padrão. Ao ser chamado, o novo valor deve ser aplicado estendendo-se para todos os objetos dessa classe (mesmo se o objeto foi criado antes da alteração do valor). Por exemplo, se for criado um objeto para a **Pessoa** João que recebe o valor padrão, incialmente será apresentado 8 ao imprimir o valor por hora que ele recebe. Se alterarmos o valor por hora padrão para 10, ao imprimir o valor por hora recebido por João deverá aparecer 10.
    - **Dica**: crie um novo atributo bool recebePadrao.

Testes do Judge

Exercício 1

- Recurso: é classe abstrata;
- Pessoa: é classe derivada de Recurso;
- Pessoa: getters;
- Pessoa: getCusto;

Exercício 2

- FuncionarioUSP: getters

- Pessoa: é classe derivada de Recurso e de FuncionarioUSP;
- Pessoa: getNusp;

Exercício 3

- Pessoa: setValorPorHoraPadrao e getValorPorHoraPadrao são métodos estáticos;
- Pessoa: valorPorHoraPadrao com valor inicial;
- Pessoa: valorPorHoraPadrao é atributo estático;
- Pessoa: setValorPorHoraPadrao antes de instanciar uma Pessoa;
- Pessoa: getters com valorPorHoraPadrao;
- Pessoa: setValorPorHoraPadrão modifica todas as Pessoas que recebem o valor padrão;