Prática 7 Herança e Polimorfismo

Exercício 7.1

Reimplemente o exercício 5.3 com uma aproximação orientada por objetos. Nesse sentido, introduza a classe abstrata Forma com os serviços perímetro e área na sua interface pública. Faça com que as classes das figuras (círculo, triângulo e retângulo) sejam descendentes desta classe, e adapte o programa para esta nova solução.
Inclua um novo atributo, cor (String), comum a todas as figuras, considerando agora que são diferentes duas figuras que tenham cores diferentes, mesmo que tenham em comum as restantes características.

Exercício 7.2

Retome o exercício 5.1, mudando o nome da classe *Date* para *DateYMD*, e acrescentando uma nova classe *Date*, abstrata (i.e. não instanciável), e que não tenha nenhuma representação interna. Deve implementar nesta classe o máximo de serviços público que conseguir. Os métodos estáticos devem manter-se nesta classe abstrata.
A classe *DateYMD* deve ser implementada como sendo descendente da classe *Date*, e contendo a representação interna de dia, mês e ano (como no exercício 5.1).
O programa com o menu do exercício 5.1 deve ser copiado e adaptado (qb) para testar as alterações feitas, mantendo o tipo de dados *Date* como abstração a ser utilizada (alterando, obviamente, a instanciação do objeto para *DateYMD*).

b) Construa uma nova classe *DateND* em que a data é representada internamente como sendo a distância (em número de dias) a 1 de Janeiro de 2000. Esta classe deve descender de *Date* e implementar todos os serviços públicos necessários.

Exercício 7.3

Pretende-se desenvolver um programa que possibilite a gestão de alguns produtos numa agência de viagens. As entidades principais neste sistema de informação são alojamentos (apartamentos e quartos em hotel) e carros que se podem alugar. Devem ser suportadas as caraterísticas seguintes:

- A agência de viagens, para além de conter um conjunto de alojamentos e um conjunto de viaturas de aluguer, tem um nome (String) e um endereço (String);
- Um alojamento tem um código (String), nome (String), local (String), preço por noite (double), disponibilidade (booleano) e avaliação (double, entre 1.0 e 5.0). Deve permitir as operações de check-in e check-out.
- Um apartamento é um alojamento mas inclui também informação sobre o número de quartos.
- Um quarto de hotel é um alojamento mas tem mais um campo que indica o tipo (single, double, twin, triple).
- Um carro tem classe (char, de 'A' a 'F') e indicação do tipo de motorização (gasolina/ diesel/híbrido/elétrico). Deve permitir as operações de levantar e entregar.

Represente adequadamente todas estas entidades. Crie construtores, métodos set/get que lhe pareçam adequados, bem como métodos que sejam fundamentais para cada classe.
Teste as classes desenvolvidas usando um programa em que simule a interface com o(a) funcionário(a) da agência (por exemplo, um menu), envolvendo necessariamente a criação dos diversos objetos referidos acima. Simule algumas operações de reserva e entregas e imprima no final a informação atual sobre a agência.

Exercício 7.4 (Opcional)

Considere as seguintes entidades que fazem parte de um jogo de futebol robótico:

- Um Objeto móvel é caracterizado por coordenadas (x e y) e pela distância percorrida durante o jogo.
  Um objeto móvel deve permitir a operação move(int newX, int newY). Sempre que esta operação é chamada, deve ser calculada automaticamente a distância percorrida.
- Robô, é um objeto móvel caraterizado ainda por um id (String), pelo tipo de jogador (uma String que pode ser GuardaRedes, Avancado, Defesa, Medio) e número de golos marcados. Um robô deve permitir ainda a operação de marcar um golo.
- Bola, objeto móvel caraterizado ainda por uma cor (String);
- Equipa, caracterizada por um nome (String), nome do responsável (String), total de golos marcados, total de golos sofridos e um conjunto de robôs.
- Jogo, caracterizado pela informação relativa a duas equipas, por uma bola, duração do jogo e tempo decorrido. Explore como pode ir calculando automaticamente o tempo decorrido.

Represente adequadamente todas estas entidades. Crie construtores, métodos set/get que lhe pareçam adequados, bem como métodos que sejam fundamentais para cada classe.
Teste cada uma das classes desenvolvidas construindo um programa para esse efeito, envolvendo a criação dos diversos objetos. Evolua este programa para que permita simular um jogo entre duas equipas, cada uma com 3 robôs (e.g. simule algumas movimentações, marcação de golos, etc.).