Ficha de Trabalho – 5

Objectivos: Exercícios com classes cujos objectos que interagem de forma bidireccional

Exercício 1:

Escreva um programa que simule a existência de peixes num aquário. Deve modelizar cada um destes conceitos (Peixe e Aquário) em classes distintas e aplicar devidamente os conceitos do encapsulamento. Significa isto que não vai ter o aquário a manipular ou alterar o estado interno dos peixes aquário como se estes fossem marionetas transparentes (por exemplo, fazer o aquário incrementar o peso do peixe quando este é alimentado – errado: o peixe come; se transforma essa comida em peso, ou se tem um problema de metabolismo e deita tudo fora, o peixe é que sabe, não o aquário).

Acerca dos peixes sabe-se o seguinte:

Um peixe tem:

- Nome da espécie (texto), Cor (texto), Peso (gramas), Numero de série (inteiro sempre crescente)

Faz / permite:

- Ser alimentado com uma dada quantidade em gramas de alimento que é somada ao seu peso. Se exceder 50g e estiver num aquário, o peixe considera que precisa de um "quintal" maior e foge do aquário, ou seja, remove-se/pede para ser removido dele. Nota importante: sair/não sair do aquário é uma decisão da inteira responsabilidade do peixe. Não é o aquário que toma essa decisão e iniciativa. Considere que pode haver várias espécies de peixe e de alguma forma a decisão variava em função da espécie. A função do aquário é simplesmente "dizer" ao peixe "toma esta comida, come e faz o que entenderes com ela, eu vou passar ao próximo peixe". A alternativa "toma isto, come. Estás com mais de 50 gramas? Então sai" vai contra os princípios do encapsulamento e é considerada um erro (e dá mais trabalho a fazer). O peixe decide sair do aquário. Eventualmente pede ao aquário para o tirar (não se tira a ele próprio – o peixe não faz ideia como é que o aquário guarda os peixes internamente).

  Obter a descrição textual em string.

Obedece às seguintes regras:

- A construção dos objectos exige sempre o nome da espécie. A cor é opcional, sendo "cinzento" se nada for especificado. O peso inicial é de 10 g. O número de série é um valor automaticamente atribuído aumentando de um em um a cada novo peixe criado explicitamente (a construção por cópia não aumenta este valor). O primeiro peixe tem o valor 500.
- Quando são criados os peixes não estão necessariamente em aquário nenhum. Podem ser postos num aquário à posteriori. Quando um peixe é posto no aquário, este assume a sua posse e controlo total, passando a comandar o destino do peixe (poder de vida e morte sobre o peixe).

Acerca dos aquários sabe-se que:

Um aquário tem:

- Uma quantidade de peixes. Eventualmente muitos.

Faz / permite:

- Inserir um peixe novo.

- Verificar se um peixe se encontra no aquário dado o número de série dele.
- Alimentar os peixes todos com uma determinada quantidade de gramas (a mesma quantidade para todos, um por um).

Um aquário obedece às seguintes regras:

- Os aquários são criados inicialmente sem peixe nenhum.

Observações

Este exercício introduz o aspecto extremamente importante de interacção complexa e bidireccional entre duas classes e recomenda-se a quem não vá às aulas que o resolva todas as alíneas na mesma e peça depois uma opinião qualitativa ao docente sobre o código elaborado. A interacção bidireccional é a seguinte:

1. O aquário indica aos peixes que têm mais uma dose de alimento (aquário → peixe).
2. O peixe indica ao aquário que deseja ser removido (peixe → aquário).

Esta situação pressupõe o conhecimento mútuo entre ambos os objectos.

Implemente as classes descritas anteriormente...

Coloque cada classe num par de ficheiros .h/.cpp independente ficando o projecto com 5 ficheiros ao todo: um .h e um .cpp por cada classe e um .cpp para a função main. Esta organização irá fazer surgir uma situação de inclusão circular A precisa de B mas B precisa de A (o aquário tem peixes mas o peixe tem que saber em que aquário se encontra). Esta situação é inevitável dada a interacção existente entre peixes e aquário e o seu aparecimento neste exercício é propositado para ver como se resolve. Trata-se de uma situação muito comum que é necessário saber resolver.

Nota: a inclusão circular A.h inclui B.h e B.h inclui A.h não funciona. Das duas, uma:

- Ou ambos os ficheiros vão ser incluídos repetidamente até ao infinito: A.h inclui B.h que inclui A.h (2ª vez) que inclui B.h (2ª vez) que inclui A.h (3ª vez) etc. → Não funciona, não é aceite.
- Ou os ficheiros estão protegidos contra inclusão repetida (directiva pragma ou a construção #ifndef - #define - #endif): neste caso, A.h inclui B.h que já não consegue incluir A.h e portanto B.h não compila porque lhe falta A.h. → Não funciona.
- A solução passa por um dos ficheiros A ou B desistir de incluir o outro e mencione apenas a existência da classe que precisa. No caso de apenas precisar de um ponteiro, basta mencionar que a classe (objectos) apontada existe (se precisar de um objecto mesmo, então terá que incluir mesmo o .h todo).

Escreva uma função main que permita a seguinte funcionalidade:

- Adicionar peixes a um aquário mediante dados especificados pelo utilizador;
- Listar os peixes que estão no aquário (ver a descrição textual de cada um);
- Alimentar todos os peixes dada uma quantidade de comida especificada pelo utilizador.
- Verificar se um determinado peixe se encontra no aquário dado o seu número de série.
- Remover um peixe dado o seu número de série.
- etc….