Ficha de Trabalho – 3

Objectivos: Esta ficha lida com a associação simples entre objectos. As associações podem tomar essencialmente duas formas: a agregação e a composição.

Exercício 1:

Assuma a existência da classe Prego (passe o código para o seu computador). Os objectos desta classe representam pequenos objectos metálicos afiados que se afixam a paredes em coordenadas x,y. Os pregos podem mudar de sítio (removem-se e voltam-se a afixar).

```
class Prego
{
        int x,y;
        public:
        Prego(int a, int b)
        {
                x = a; y = b;
                cout << "construindo prego em " << x << "," << y << "\n";
        }
        ~Prego() {      cout << "destruindo prego em " << x << "," << y << "\n";    }
        void mudaDeSitio(int a, int b) {      x = a; y = b;    }
}
```

Pretende-se agora uma classe Aviso com as seguintes características:

- Tem um texto.
- Está preso a um prego.
- O prego não pertence ao aviso.
- O prego já existia antes e pode estar associado a mais do que um aviso.
- Quando é criado, o aviso tem sempre um texto e é sempre posto num determinado prego que já se encontra afixado na parede. Não é preciso nem possível mudar o aviso de um prego para outro.
- Sempre que o objecto é criado ou destruído deve ser apresentada uma mensagem no ecrã que inclua o texto do aviso (para distinguir os objectos, se existirem vários).
- Deve ser possível saber as coordenadas x e y do prego em que o aviso está pendurado.
- Vários avisos podem estar pendurados no mesmo prego. Trata-se do mesmo objecto prego e não de cópias de pregos que por acaso estão no mesmo sítio na parede.
- Se se mudar um prego de sítio, o aviso que está pendurado muda implicitamente para o novo sítio. Se está a pensar acrescentar um mecanismo de "mudaDeSitio" à classe Aviso, pense novamente: quem muda de sítio é o prego e não o aviso. Este apenas vai atrás. Além disso o aviso não sabe directamente em que sítio está. Apenas sabe que está num determinado prego, e este é que sabe onde está. Mais, a mudança de sítio de um prego pode afectar mais do que um aviso (afecta todos os que estão lá pendurados).

Faça agora um programa que vá usar pregos e avisos.
- Deve colocar vários avisos na parede em vários pregos.
- Deve alterar a posição de um prego e todos os avisos nesse prego são alterados.
- Determinar qual o prego que tem mais avisos.
- Remover um dado aviso.

Exercício 2:

Pretende-se representar a informação existente num banco. O banco armazena contas. Uma conta é um artefacto cuja existência só faz sentido no contexto do banco. Cada conta diz respeito a uma pessoa, a qual já existe, quer esteja associada a uma conta, quer não. Existem então três conceitos: Banco, Conta e Pessoa.

Usando a classe Pessoa, escreva a classe Conta tendo em atenção o seguinte:
- Os atributos de Conta são: o saldo (inteiro) e uma pessoa. A pessoa não pertence à conta. Pelo contrário: o objecto Pretende-se conseguir chegar de Conta ao objecto Pessoa directamente e não através de pesquisas.
- Deve ser possível criar uma conta apenas mediante a indicação da pessoa titular. O saldo inicial é zero.
- Deve ser possível aumentar ou diminuir o saldo. O valor mínimo do saldo é zero.
- Deve ser possível obter a pessoa titular da Conta, mas essa informação é fornecida como constante para prevenir modificações indesejadas.

(Numa segunda evolução do exercício, pense em contas à Ordem e contas a Prazo!)

Defina uma classe Banco.
Um banco é caracterizado por um nome e um conjunto de contas. Recorde que as contas são artefactos que apenas fazem sentido no contexto do banco que as têm. Ao criar um banco deve ser indicado o seu nome. Inicialmente não tem nenhum cliente, ou seja, nenhuma conta. Não existe limite quanto ao número de contas. O banco deve permitir:
- Acrescentar uma conta.
- Encontrar/obter uma conta dado o BI do seu titular.
- Eliminar uma conta dado o BI do seu titular.
- Depositar (Levantar) uma quantia na conta de um cliente dado o seu BI e o valor a depositar (levantar).
- Obter a soma dos saldos de todos os clientes do banco;
- Obter uma string com todos os nomes dos clientes do banco, separados por vírgulas.

Acrescente uma função main para testar as classes que fez neste exercício.
- Crie na função main várias pessoas e dois bancos. Faça com que uma pessoa seja cliente dos dois bancos. Mude o nome dessa pessoa. Confirme que essa alteração é automaticamente visível na listagem dos clientes de ambos os bancos. Se assim não for, o seu código não está!

- Acrescente à função main código para ler um conjunto de objectos Pessoa a partir de um ficheiro de texto. Esta alínea serve apenas para começar a treinar o uso de classes biblioteca para ler ficheiros de texto e aproveita-se o contexto deste exercício. O formato do ficheiro é o seguinte: cada linha diz respeito a uma pessoa. A primeira palavra em cada linha é o nome da pessoa (para simplificar, é só o primeiro nome), a segunda palavra é o BI, e a terceira é o número de contribuinte. Crie um ficheiro assim com o notepad ou algo igualmente simples e use-o. Devem ser lidas e criadas tantas pessoas até um máximo de 10 (menos se o ficheiro não tiver tantas). Use essas pessoas como clientes no banco tal e qual como na alínea anterior.

Exercício 3:

Construa em C++ o conceito de Turma de Alunos.

Uma turma é um conjunto de pessoas que se encontram a estudar a mesma disciplina com o mesmo professor. Usando a classe Pessoa que aparece nos dois últimos exercícios, Escreva a classe Turma, a qual deve cumprir os seguintes requisitos:

- A turma tem um objecto Pessoa que é o professor dessa turma e um conjunto de objectos Pessoa que são os alunos. Tanto o professor como os alunos têm vida própria para além da turma e a suas existências não podem ser controladas pelo objecto Turma.
- A criação da turma obriga a indicação do seu professor. Os alunos são acrescentados mais tarde.
- Deve ser possível acrescentar alunos até ao máximo de 20.
- Deve ser possível remover um aluno dado o seu BI.
- Deve ser possível listar os nomes dos alunos na turma.

Acrescente uma função main e teste a funcionalidade da classe.

Declare várias pessoas e pelo menos duas turmas. Inscreva um aluno nas duas turmas e depois experimente mudar o seu nome para confirmar que essa alteração é automaticamente visível na listagem de alunos das duas turmas, provando assim que essas turmas não "têm" cópias das pessoas mas sim os originais.