Aula Prática 2

Objetivos

Análise e desenho de aplicações segundo o paradigma *Object Oriented*.
Utilizar mecanismos de Encapsulamento *(Information Hiding)* e Visibilidade.
Revisão dos conceitos *this* e *static*.

Nota: Não se esqueça que lápis e papel são elementos fundamentais nas aulas práticas.

Problema 2.1

Pretende-se fazer um sistema de informação para gestão do videoclube de uma associação de estudantes. Esse videoclube contém um catálogo de vídeos e um conjunto de utilizadores.

Um vídeo é caracterizado por um ID sequencial, um título, uma categoria (Ação, Comédia, Infantil, Drama, etc.) e uma idade (ALL, M6, M12, M16, M18).

Os clientes deste videoclube terão um número de sócio sequencial, data de inscrição de sócio e poderão ser de dois tipos:

- Estudantes: caracterizados por um nome, número de CC, data de nascimento, número mecanográfico e curso;
- Funcionários: caracterizados por um nome, número de CC, data de nascimento, número de funcionário e número fiscal.

a) Comece por identificar todas as entidades envolvidas neste sistema de informação, assim como as tarefas associadas a cada uma delas.

b) Defina relações entre essas entidades e efetue o desenho do sistema. Para tal, utilize um diagrama que represente os vários objetos (classes) e relações entre eles. Sugestão: UML.

c) Faça um programa que, de uma forma interativa (recorrendo a um menu), permita gerir o catálogo de vídeos, utilizadores e empréstimos. O programa deve permitir a introdução e remoção de utilizadores, pesquisar e listar vídeos disponibilizados para determinado utilizador. Relativamente aos vídeos, o programa deve permitir a introdução e remoção de filmes do sistema, verificar a disponibilidade de determinado item do catálogo, efetuar empréstimo (*checkout*) e devolução (*checkin*) de vídeos, assim como identificar todos os utilizadores que requisitaram determinado vídeo.
   Sugestão: Recomenda-se que as operações de empréstimo e devolução sejam efetuadas com base no ID do vídeo e no número de sócio do cliente.

d) Altere a aplicação para permitir a introdução de um sistema de *rating* associado a cada vídeo do catálogo. Assim, na devolução (*checkin*) de vídeos, os utilizadores devem atribuir um nível de qualidade (1 a 10) ao filme. Cada vídeo passará a ter um *rating* total e um *rating* médio.

e) Altere o menu principal para passar a disponibilizar informação de *rating* associado a cada item do catálogo de vídeos. Introduza uma funcionalidade para listar os vídeos por *rating*.

f) Altere a aplicação para permitir a listagem do histórico de vídeos emprestados a determinado cliente.

g) Altere a aplicação para passar a incluir um mecanismo de quotas de empréstimo. Assim, cada utilizador só poderá requisitar simultaneamente um máximo de N vídeos. O valor de N será definido aquando da inicialização do videoclube.

Problema 2.2

Escreva um programa para resolver o jogo da *Sopas de Letras*.

a) A entrada do programa é um único ficheiro de texto contendo o puzzle e as palavras a encontrar. Por exemplo:

```
QLXRXBXCLH
YHQTOPJQUJ
VFMHSNZZZL
DWPPEUEUQK
MKCATSILTC
LFTRTRVLRX
RNHGKMEQUM
VODQBYCEKQ
CWVFKCTKVU
WUXIMOZESS
list, set
graph, stack, queue, tree
```

O programa deve assumir que:

- O puzzle é sempre quadrado;
- O tamanho máximo do puzzle é 80x80;
- As letras do puzzle estão em maiúscula;
- A lista de palavras pode estar só em minúsculas, ou misturadas;
- As palavras podem começar ou acabar por espaços em branco;
- As palavras são compostas por caracteres alfabéticos;
- As palavras têm de ter pelo menos 2 caracteres;
- A lista de palavras não contém linhas em branco;
- Cada linha pode ter várias palavras separadas por vírgula, espaço ou ponto e vírgula;
- Todas as palavras da lista têm de estar no puzzle;
- A lista de palavras não pode conter palavras duplicadas ou frases redundantes (por exemplo, não pode conter BAG e RUTABAGA ao mesmo tempo);
- Cada palavra só aparece uma vez no puzzle.

b) A saída do programa é a lista de palavras com a sua localização (linha, coluna) e a orientação no puzzle. Exemplo de execução do caso anterior:

```
$java WSSolver sdl_01.txt
LIST            5,8             left
SET             3,5             down
GRAPH           7,4             up
STACK           5,6             left
QUEUE           4,9             left
TREE            5,5             downright
```