Parte prática. Duração: 2h00m

Um Clube de Vídeo aluga filmes em DVD e pretende implementar um sistema para gestão dos seus clientes e alugueres de filmes. O clube pode possuir vários exemplares de um mesmo filme; no caso de todos os exemplares já terem sido alugados, é mantida uma fila de pedidos. À devolução do primeiro exemplar de um pedido realizado, o primeiro cliente que fez a solicitação do filme passa a alugá-lo.

Assim, a classe **ClubeVideo** guarda informações sobre os seus clientes (objetos da classe **Cliente**), os filmes (objetos da classe **Filme**) disponíveis no clube para aluguer, e os pedidos em esperada (objetos da classe **PedidoCliente**), que ficam guardados numa fila, por ordem de pedido realizado, aquando da tentativa de aluguer de um filme que não tem exemplares disponíveis.

O acervo de filmes do clube de vídeo é formado pelos filmes disponíveis na loja, para pronto aluguer, e pelos filmes em posse dos clientes que os alugaram. Os clientes guardam os filmes alugados numa pilha: sempre que um novo filme é alugado, é colocado no topo da pilha, e quando quer devolver qualquer filme, precisa retirar os filmes de cima. Quando um cliente devolve um filme ao clube de vídeo, este passa a estar disponível para novo aluguer, caso não haja clientes com pedidos na fila de espera. Se houver um pedido em espera, o exemplar do filme devolvido passa imediatamente ao primeiro cliente que estava à sua espera, que o colocará no topo da sua pilha de filmes.

Quando um cliente, ainda não pertencente ao clube, tenta alugar um filme, este cliente passa automaticamente a pertencer ao clube, sendo incluído no vetor de clientes do clube. No caso de algum cliente tentar alugar um filme que não existe, ou seja, que não está disponível na loja e nem tem exemplares alugados a outros clientes, é lançada uma exceção do tipo **FilmeInexistente**.

As classes **ClubeVideo**, **Cliente**, **Filme,** e **PedidoCliente** estão parcialmente definidas a seguir.

```
class ClubeVideo {
  vector<Filme> filmes;
  vector<Cliente> clientes;
  queue<PedidoCliente> pedidos;
public:
  ClubeVideo(){}
  //...
  bool existeCliente(string umNome) const;
  bool existeFilme(string umTitulo) const;
  string imprimirFilmes() const;
  list<string> titulosDisponiveis() const;
  bool tituloDisponivel(string umTitulo) const;
  string imprimirListaDeEspera() const;
  void alugar(string umNome, string umTitulo);
  void devolver(string umNome, string umTitulo);
};
```

```
class PedidoCliente {
  string nome;
  string titulo;
public:
  PedidoCliente(string nomeC, string tituloF);
  string getNomeCliente() const;
  string getTituloFilme() const;
  bool operator==(const PedidoCliente& pc1);
  friend ostream& operator<<(ostream& os,
    const PedidoCliente& pc);
};
```

```
class Cliente {
  string nome;
  stack<Filme> filmesEmprestados;
public:
  Cliente(string umNome);
  string getNome() const;
  void addFilme(Filme umFilme);
  stack<Filme> getFilmesEmprestados() const;
  Filme devolver(string umTitulo);
  bool operator==(const Cliente& c1);
  friend ostream& operator<<(ostream& os,
    const Cliente& c1);
};
```

```
class Filme {
        string titulo;
        int emprestimos;
public:
        Filme(string umTitulo);
        string getTitulo() const;
        int getEmprestimos() const;
        void addEmprestimos();
        bool operator==(const Filme& f1) const;
        friend ostream& operator<<(ostream& os, const Filme& dvd);
};
```

Parte prática. Duração: 2h00m

**a)** [**2.5 valores**] Implemente na classe **ClubeVideo** os membros-função seguintes:

*bool existeCliente(string umNome) const*

*bool existeFilme(string umTitulo) const*

O método *existeCliente* retorna verdadeiro ou falso, caso o cliente de nome *umNome* exista ou não no vetor de clientes do clube de vídeo, respetivamente. O método *existeFilme* faz o mesmo, para um filme chamado *umTitulo*, e procura-o quer entre os filmes da loja quer entre os exemplares alugados aos clientes.

**b)** [**2.5 valores**] Implemente na classe **ClubeVideo** o membro-função:

*string imprimirFilmes() const*

Este método retorna uma *string* com a informação (disponibilizada por "`operator <<`" e separada por "\n") de todos os exemplares de filmes que o clube de vídeo tem, estejam eles disponíveis na loja ou alugados pelos seus clientes.

**c)** [**3 valores**] Implemente na classe **ClubeVideo** o membro-função:

*list<string> titulosDisponiveis() const*

Esta função copia os títulos de todos os seus filmes disponíveis na loja, para aluguer imediato, sem considerar exemplares repetidos. A lista deve estar ordenada alfabeticamente e, no caso de um filme ter múltiplos exemplares, o seu título será contabilizado uma única vez.

**d)** [**2.5 valores**] Implemente na classe **ClubeVideo** o membro-função:

*bool tituloDisponivel(string umTitulo) const*

Este método retorna verdadeiro caso haja pelo menos um exemplar do filme de *umTitulo*, disponível para aluguer imediato na loja, sem considerar os exemplares já alugados.

**e)** [**3 valores**] Implemente na classe **ClubeVideo** o membro-função:

*string imprimirListaDeEspera() const*

Este método retorna uma *string* com a informação dos títulos de todos os filmes em lista de espera, ordenados alfabeticamente.

**f)** [**3.5 valores**] Implemente na classe **ClubeVideo** o membro-função:

*void alugar(string umNome, string umTitulo)*

A função *alugar* tenta realizar o aluguer de um filme de título *umTitulo* por um cliente de nome *umNome*. Para um novo cliente, que não está ainda no vector de clientes do clube de vídeo, este passará a pertencer automaticamente. No caso de haver um exemplar do filme disponível na loja do clube de vídeo, este exemplar é alugado ao cliente (que passa a empilhá-lo na sua pilha de filmes alugados) e é retirado do vetor de filmes da loja. O exemplar alugado também tem o seu membro dado *empréstimos* acrescido de uma unidade, correspondendo a mais um aluguer daquele exemplar. No caso de não haver exemplares disponíveis, ou seja, todos os exemplares do filme já estão alugados a outros clientes, um *PedidoCliente* é registado e colocado em fila de espera. Assim que um exemplar for devolvido, no caso de não haver outros clientes em espera adiante na fila, o cliente poderá alugá-lo.

Implemente também a exceção abaixo:

*FilmeInexistente*

No caso do filme de *umTitulo* não existir, isto é, não tendo exemplares disponíveis na loja ou alugados a outros clientes, uma exceção do tipo *FilmeInexistente* deve ser lançada adequadamente. Consulte a invocação dos seus métodos no ficheiro de teste para a sua correta implementação. Novos clientes não devem ser adicionados ao vector de clientes caso ocorra esta exceção.

**g)** [**3 valores**] Implemente na classe **ClubeVideo** o membro-função:

*void devolver(string umNome, string umTitulo)*

Esta função devolve ao clube de vídeo um exemplar do filme de título *umTitulo* pelo cliente de nome *umNome*. Ao realizar a devolução de um filme, o cliente deverá retirá-lo da sua pilha de filmes emprestados. No caso de haver pedidos em fila de espera, o primeiro cliente da fila a tê-lo solicitado alugará o filme. Caso não haja pedidos em fila de espera, o exemplar do filme passará a estar disponível na loja do clube para futuros alugueres.