Algoritmos e Estruturas de Dados

Gestão de uma Biblioteca (Parte 2)

Turma 4 - Grupo B

José Peixoto	200603103	ei12134@fe.up.pt
Paulo Faria	201204965	ei12135@fe.up.pt
Pedro Moura	201306843	up201306843@fe.up.pt

29 de Dezembro de 2014

Conteúdo

Resumo

Foi proposta a extensão das funcionalidades de uma aplicação em `C++` para gestão de uma biblioteca que evidenciasse o conhecimento e um uso correcto de novas estruturas de dados: árvores binárias de pesquisa, filas de prioridade e tabelas de dispersão. Os livros são agora dispostos de forma ordenada pelo ano de edição numa árvore binária de pesquisa, os leitores inactivos são armazenados numa tabela de dispersão e os pedidos de livros já emprestados são registados numa fila de prioridade.

1 Descrição da solução implementada

Para implementar a ordenação pedida dos livros, introduziu-se um novo membro-dado da classe livro `editionYear` que representa o seu ano de edição e funciona como primeiro parâmetro de decisão no operador menor dessa classe. A árvore binária de pesquisa ordena os livros pelo ano de edição, pelo título, pelos nomes dos autores e pelo `bookID`, de modo a permitir distinguir entre cópias do mesmo exemplar.

Para permitir a classificação de leitores como inactivos de forma manual, recorreu-se a uma variável booleana `inactive`, membro-dado da classe `Reader` e para verificar de forma automática os leitores inactivos, usou-se outro novo membro-dado, data da última actividade do leitor `lastActivity`. Recorreu-se ao número de identificação de cada leitor para o indexar na tabela de dispersão, uma vez que cada leitor tem um `card` com um número único.

Na construção de uma fila de prioridade que armazenasse os pedidos de empréstimo, definiu-se uma nova classe `Request` que contivesse apontadores para um leitor, para o livro pedido e respectiva data. Ordenou-se a fila de prioridade, implementando o operador menor da classe `Request`, priorizando os pedidos pela data e depois pelas idades dos leitores: crianças ($idade \leq 12$), estudantes ($12 < idade \leq 23$) e, por fim, adultos ($idade > 23$).

2 Lista de casos de utilização da aplicação

A aplicação permite a criação, leitura, edição e remoção de livros, leitores, funcionários, empréstimos e pedidos de empréstimo.

Alterações como a remoção ou adição de livros são espelhadas na árvore binária. É possível listar a árvore binária de forma completa ou parcial, pesquisando pelo ano de edição, pelos autores ou pelo título.

É permitida a atribuição manual do estado de inactividade pela data da última actividade e/ou pela configuração do estado de inactividade guardado em `inactive`. A atribuição automática dos estados de inactividade `Update inactives` sobrepõe-se às configurações manuais, usando como critério único a última actividade do leitor.

É criado e adicionado um novo pedido à fila de espera, de forma automática, aquando da tentativa de empréstimo de um livro que esteja indisponível. Após a devolução de qualquer livro é feita uma busca na fila pelo primeiro leitor em espera que possa requisitar o livro (desde que não tenha excedido o máximo de empréstimos em simultâneo).

3 Relato das principais dificuldades encontradas no desenvolvimento do trabalho

Encontraram-se algumas dificuldades na nomenclatura e no adicionar de novos métodos e membros-dado às classes anteriormente definidas, pelo aumento da complexidade na leitura e escrita de ficheiros e na gestão da estabilidade das funcionalidades já existentes.

Foi difícil a implementação do operador menor da classe `Request` que permitisse uma correcta ordenação na fila de prioridades.

4 Indicação do esforço dedicado por cada elemento do grupo

4.1 Notas em grupo

Resolvemos sequencialmente os itens do enunciado. Fomos trabalhando ora por turnos ora conferenciando via *Skype* ou *Google Hangouts*.

A UML