U.PORTO
FEUP FACULDADE DE ENGENHARIA
UNIVERSIDADE DO PORTO

Algoritmos e Estruturas de Dados
EIC0013

Ano Letivo 2014/2015

T2 - GESTÃO DE UMA APP STORE

FEUP_AEDA1415_2MIEIC03_D

Carlos Soares - up201305514@fe.up.pt

Diogo Marques - up201305642@fe.up.pt

Fábio Carneiro - ei11029@fe.up.pt

30 de Dezembro de 2014

Índice

Introdução ao tema

Uma *App Store* é constituída por várias *apps* desenvolvidas por *developers* externos e que podem ser compradas pelos seus clientes, sendo uma parte do valor da venda retida pela *App Store*, e o restante devolvido aos developers.

De modo a apresentar o top 10 de apps aos clientes, decide-se manter a informação das apps numa árvore binária de pesquisa, sendo a sua ordenação efetuada por classificação da app, preço e categoria. Deve ser permitido adicionar apps, remover apps, e alterar a informação das apps. Devem ainda ser permitidas listagens várias tirando partido da ordenação da árvore..

Considere que as apps que o developer decidiu deixar de vender devem ser guardadas numa tabela de

dispersão. Deve ser permitido, na tabela de dispersão: a inserção de apps que o developer decidiu deixar

de vender; a remoção de apps, caso o developer decida publicá-las novamente para venda; e a alteração

das características das apps que estão na tabela. Deve ainda ser possível listar as apps presentes na tabela

de dispersão.

Considere que a appstore se tornou bastante popular e que um crescente número de developers a procura

para vender as suas apps. A publicação de apps é agora sujeita a um processo de validação antes da sua

publicação na appstore. Para isso mantenha, mantenha as apps submetidas para validação numa fila de

prioridade. A ordenação das apps na fila deve obedecer aos seguintes critérios: data de submissão, preço

da app e nome da app. Deve ser possível inserir, remover e alterar as apps guardadas na fila de prioridade.

É possível gerir toda a informação dos developers, clientes, vendas, *apps* disponíveis, *apps* submetidas e *apps* removidas, e guardar toda a informação presente na mesma num ficheiro e ler a informação guardada.

Solucão Implementada

Devido ao aumento da popularidade da App Store, e um crescente número de *developers* à procura para vender as suas classes, os *developers* deixaram de poder publicar apps diretamente na *App Store*. Estas são agora colocadas numa fila de espera para serem posteriormente validadas pelo proprietário da *App Store*. Todas as apps que o *developer* submeter na *App Store* ficam sujeitas a este processo de validação, que é necessário para as *apps* ficarem disponíveis para compra. Não há nada que o *developer* possa fazer para garantir que a sua app seja seleccionada, existe um critério de ordenação que define a prioridade de cada *app* na fila de espera, e o proprietário da *App Store* pode e deve servir-se desta ordenação na selecção das apps que devem ser publicadas na sua loja.

Assim sendo, como o objetivo desta segunda parte era complementar a *App Store* implementada anteriormente, esta conta agora com uma árvore binária de pesquisa (BST), que mantém referências para as apps existentes no vetor *apps*, ordenadas, e de uma fila de prioridade que mantém as apps que ainda não foram sujeitas ao processo de validação referido em cima.

Developers

A classe *Developer* possui uma nova estrutura de dados (tabela de dispersão) onde são guardadas todas as aplicações que o *developer* deixou de vender e aquelas que foram removidas pelo proprietário da *App Store.* A tabela de *apps* removidas serve principalmente para o *developer* manter um registo das suas vendas para futura referência e do *feedback* dado pelos clientes.

Após fazer login no sistema, o *developer* pode publicar uma *app* na *App Store*, alterar informações relativas às *apps* que publicou, deixar de vender as suas *apps*, e cancelar o processo de validação das *apps* que submeteu. Entende-se por cancelar o processo de validação retirar a *app* da fila de prioridade e colocar na tabela das *apps* removidas.

O *developer* pode voltar a publicar as apps que deixou de vender a qualquer momento. No entanto, o *developer* deve ter em atenção que as *apps* passam a ser consideradas como *apps* novas assim que as volta a submeter (os registos de vendas, as classificações e os comentários são apagados) e ficam igualmente sujeitas ao processo de validação associado a *apps* recentes. Não existe um limite para o número de vezes que o *developer* pode retirar as apps da loja, e voltar a submetê-las.

É possível llistar as apps removidas por nome, e por preço, e por número de unidades vendidas, bem como alterar as informações relativas às mesmas. Ainda nas listagens, o *developer* pode pedir a listagem das *apps* que se encontram à espera de serem aprovadas (por nome, preço, data , e listar as suas *apps* (as que estão disponíveis na *App Store*) por ordem alfabética e por número de unidades vendidas.

Quando um *developer* é removido da loja pelo proprietário, toda a informação associada ao mesmo é apagada (*apps* existentes no vetor apps, na árvore de pesquisa e na fila de prioridade, bem como na tabela onde se encontravam guardadas as *apps* que deixou de vender).

Clientes

Os clientes podem agora consultar o top 10 das *apps* disponíveis na App Store, pedir listagens de *apps* por data de lançamento (da mais antiga para a mais recente e vice-versa) e por número de unidades vendidas.

App

Agora cada vez que uma *app* é comprada é incrementado o seu membro-dado *sale*.

Foi adicionado um campo para guardar a data de lançamento da app. A data de lançamento reflete o momento em que a *app* foi aprovada pelo proprietário da *App* Store, ou a última vez que foi atualizada desde que passou a estar disponível na loja. Logo, sempre que um *developer* ou o proprietário alterar as informações de uma *app*, a data de lançamento é também atualizada para refletir a data da última atualização da *app*.

Existe também um vector com as classificações atribuídas pelos clientes. As classificações, ao contrário dos comentários, são anónimas e servem apenas de estatística para o proprietário da *App Store* e como feedback para o *developer*. Uma app que nunca tenha sido classificada possui agora uma classificação por defeito de 3 em 5, para facilitar a ordenação de *apps* na árvore binária de pesquisa. A pontuação mínima que um cliente pode atribuir a uma *app* é 1. Qualquer utilizador pode classificar uma app mais do que uma vez, a nossa implementação preza a simplicidade e não colocou esta restrição.

Quando uma *app* é removida da loja, o programa apaga todas as ocorrências dessa *app* dentro do vector "*apps*", da árvore binária e da fila de prioridade, isto é, tanto as *apps* publicadas deixam de estar disponívéis para compra, e as *apps* que ainda não foram aprovadas pelo proprietário são removidas e transferidas para a tabela de *apps* removidas do respetivo *developer*. No entanto o registo de vendas dessas *app* mantém-se, ou seja, as

apps continuam na posse dos clientes que a compraram anteriormente, e são “devolvidas” aos seus respetivos *developers*

Tratamento de exceções

Para o tratamento de exceções relacionadas com a criação, manutenção e remoção de *apps* foram criadas mais três classes: *AppExiste, DeveloperExiste e ClienteExiste, que* abrangem situações contrárias às classes do tipo *Inexistente* já implementadas na primeira parte do trabalho. Estas exceções são lançadas, por exemplo, quando o proprietário tenta adicionar uma *app* quando uma com o mesmo nome já existe, ou quando o cliente tenta comprar uma app que já possui ou que já adicionou ao seu carrinho de compras. Foi também criada a classe *AppExisteNoCarrinho*, lançada sempre que o cliente tenta adicionar ao carrinho uma app já existente no mesmo, e a PermissoesInsuficientes, lançada quando o *Developer* tenta alterar ou apagar uma app de outro *Developer* ou sem *Developer* associado.

"O proprietário"

Tanto o *developer* como o proprietário da *App Store* podem publicar *apps* na loja. Quando o proprietário da *App Store* cria uma *app*, além do nome é também pedido para associar a *app* a um developer (obrigatório), e esta é imediatamente adicionada às apps disponíveis na *App Store*, sem qualquer aprovação. Quando um *developer* adiciona uma *app*, esta fica automaticamente vinculada ao *developer*, mas no entanto estará sujeita a um processo de validação pelo proprietário.

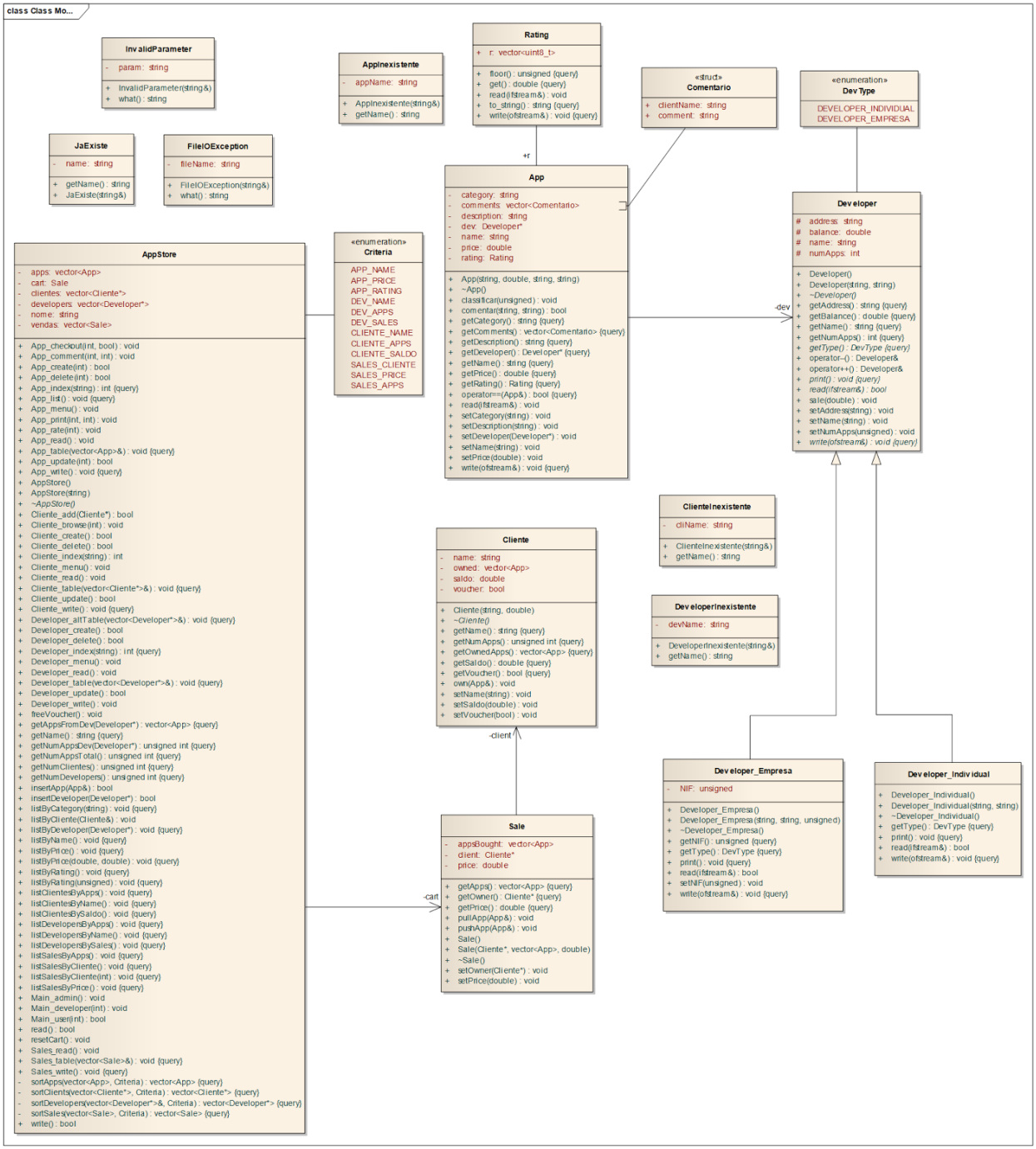
Estrutura dos ficheiros

A informação das aplicações é guardada no ficheiro “Apps.bin”, e tem a seguinte estrutura: nome, developer, categoria, descrição, preço, classificação, comentários. As classificações são constituídas pelo número de elementos do vetor e pelo vetor das classificações em si, enquanto que cada entrada do vetor de comentários apresenta o nome do cliente e o comentário.

Os ficheiros “Clientes.bin” e “Sales.bin” mantêm a estrutura definida na primeira parte deste trabalho.

O ficheiro encontra-se organizado da seguinte forma: nome do cliente, valor total da compra, número de *apps* que comprou, e uma lista com os nomes das *apps* compradas.

Diagrama UML



Casos de utilização

Um cliente pode agora:

- Listar apps por número de únidades vendidas
- Listar apps por data de lançamento (da mais antiga para a mais recente e vice-versa)
- Consultar o top 10 das apps disponíveis na App Store

Um developer pode agora:

- Submeter uma nova app na App Store, estando sujeita a um processo de validação
- Alterar informações relativas a uma app removida
- Remover uma app sua existente na App Store, sendo esta transferida para a sua tabela de *apps* removidas
- Listas as *apps* submetidas por número de vendas
- Listar as *apps* pendentes por ordem alfabética e data de lançamento
- Listar as *apps* pendentes por preço e prioridade na fila de espera
- Listar as *apps* removidas por ordem alfabética e por preço
- Listar as *apps* removidas por número de unidades vendidas

O proprietário da App Store pode agora:

- Seleccionar as *apps* a serem publicadas na sua loja
- Listar developers por ordem alfabética
- Listar developers por ordem decrescente do número de apps publicadas na App Store
- Listar developers por ordem decrescente do lucro em vendas de apps
- Listar vendas por cliente, por volume (número de apps/transação) e montante
- Visualizar informação sobre o número de apps existentes na loja
- Visualizar informação sobre o número de clientes e developers inscritos

Dificuldades

Em relação à implementação do código, algumas das principais dificuldades foram:

- o uso de referências/apontadores para *apps* na árvore binária (BST) - para ser possível ordenar os seus elementos de acordo com os critérios definidos no enunciado e manter esta estrutura de dados, foi necessário alterar a implementação da BST para desreferenciar os apontadores e comparar dois objetos da árvore

- encontrar a melhor forma de implementar as alterações aproveitando ao máximo o código anterior - para tornar a nossa implementação em código mais eficiente, mas mantendo compatibilidade com as novas estruturas, os vetores de *apps* foram convertidos para vectores de apontadores

Distribuição das tarefas pelos elementos do grupo

As tarefas relacionadas com o nosso trabalho começaram por ser divididas, inicialmente, por 3 elementos do grupo quer com reuniões quer com encontros online.

No entanto, as tarefas foram distribuídas da melhor maneira (de acordo com o grupo) e cada um deu o seu melhor para alcançar as metas pré-definidas. De uma maneira geral, todos participaram ativamente sendo notável uma maior dedicação por parte de um dos elementos do grupo, Diogo Marques.

Com tudo isto, o trabalho em grupo acabou por ser uma experiencia positiva onde houve, principalmente, cooperação e entendimento entre os elementos do grupo.