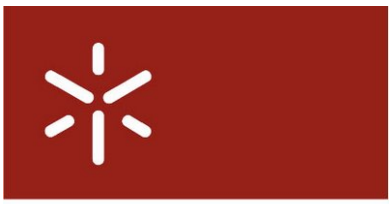


Universidade do Minho

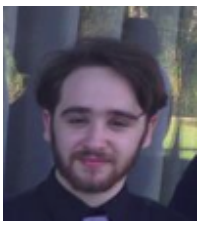
Laboratórios de Informática 3

MIEI - 2º ano - 2º semestre
Universidade do Minho

Trabalho Prático - Java

João Vieira
A76516

Rui Vieira
A74658



António Gomes
AE3654

12 de Junho de 2018

Conteúdo

1. *Introdução*

No âmbito da unidade curricular de Laboratórios de Informática III, foi nos proposto o desenvolvimento de um sistema capaz de lidar com um enorme volume de dados relacionados com uma das maiores plataformas de troca de conhecimento, *Stack Overflow*. Este projeto considera-se um grande desafio para nós pelo facto de passarmos a programar em grande escala, uma vez, que o projeto exige lidar com grandes volumes de dados e com uma complexidade algoritmica e estrutural mais elevada. Nesse sentido, o desenvolvimento deste programa será realizado a partir dos princípios da modularidade (divisão do código fonte em unidades separadas coerentes), de encapsulamento (garantia de proteção e acessos controlados aos dados), da conceção de código reutilizável e escolha otimizada das estruturas de dados.

Este relatório trata a explicação da implementação do trabalho proposto no paradigma da linguagem Java.

2. *Abordagem do problema e Descrição das estruturas criadas*

2.1 Problema

O projeto tem como principal objetivo responder a onze interrogações sobre a plataforma de troca de conhecimento, *Stack Overflow*. De forma a responder às interrogações deve ser desenvolvido um programa em Java bem estruturado e o mais otimizado possível utilizando conhecimentos adquiridos diversas unidades curriculares frequentadas até à presente data. A informação é nos dada em vários ficheiros XML, que guardam diferentes dados, depois de analisar com bastante cuidado o enunciado e ficheiros notamos que só necessitávamos de fazer o *parsing* dos ficheiros Posts.xml, Users.xml e Tags.xml. Esse *Parser* irá inserir a informação em estruturas de dados e respetivas sub estruturas.

2.2 Tipos Concretos de dados

2.2.1 TCD

O tipo concreto de dados, TCD,é constituído pelas principais estruturas dados definidas para realização do projeto. Para estruturas principais optamos pela utilização de quatro HashMap referentes a Users, Respostas, Perguntas e Tags.

Para os Users, optamos pela utilização de uma HashMap usando o ID como 'key', pois permite mapeamento direto, e uma sub-estrutura User como 'value'.

Para Respostas, optamos pela utilização de uma HashMap usando o ID como 'key', pois permite mapeamento direto, e uma sub-estrutura Resposta como 'value'.

Para Perguntas, optamos pela utilização de uma HashMap usando ID como 'key', pois permite mapeamento direto, e uma sub- estrutura Pergunta como 'value'.

Para Tags, optamos pela utilização de uma HashMap usando a designação da tag como 'key', e como 'value' usamos apenas long indicando quantas vezes essa mesma tag foi usada.

3. *Modularidade e encapsulamento*

3.1 Encapsulamento

De forma a manter o encapsulamento, definimos funções para leitura, escrita, manipulação, comparação e respetivos auxiliares de cada estrutura de suporte nas respetivas livrarias garantindo assim o encapsulamento e preservação do conteúdo das mesmas.

3.2 Abstração de dados

O tipo abstrato de dados (TAD) é uma especificação de um conjunto de dados e operações que podem ser executadas sobre esses dados sem referência a detalhes da implementação. Esta abstração permitiu uma maior limpeza e legibilidade do código desenvolvido.

4. *Estruturas e sub-estruturas*

4.1 Conjugação de estruturas e sub-estruturas

Tal como referido anteriormente, no TCD, implementamos quatro HashMap que utilizam sub-estruturas definidas por nós que explicaremos de seguida com mais detalhe.

4.2 Sub-estruturas

4.2.1 Users

Para representação dos dados referentes a um utilizador criamos uma estrutura users que se insere dentro da HashMap de users (usando o seu id como key) tendo como conteúdo :

- **Id**- long usado como identificador do utilizador.
- **Bio**- string de descrição do utilizador.
- **Rep**- int representado a reputação do utilizador.
- **Name**- string nome do utilizador.
- **Views**- int representando o numero de views do utilizador.
- **VoteDif**- int representando a diferença de votos (positivos-negativos) do utilizador.
- **NrPosts**- int representando o numero de posts efetuado pelo utilizador. É de notar que este numero é calculado no parser para todos os utilizadores.

4.2.2 Resposta

Para representação de dados referentes a uma resposta criamos uma estrutura Resposta, que se insere dentro da HashMap de respostas (usando o seu id como key) . A Respostas tem como conteúdo:

- **Id**- long usado como identificador da resposta.
- **ParentId**- long usado como identificador da pergunta à qual pertence esta resposta.
- **CreationDate**- LocalDate representando a data de criação da resposta.
- **Score**- int representando o score da resposta.
- **OwnerUserId**- long usado como identificador do utilizador a que a resposta pertence.
- **CommentCount**- int representando o numero de comentários à resposta.
- **Rate**- double representando o rate da resposta. É de notar que este rate é calculado no parser para todas as respostas.

4.2.3 Pergunta

Para representação de dados referentes a uma pergunta criamos uma estrutura Pergunta, que se insere dentro da HashMap de perguntas (usando o seu id como key) tendo como conteúdo:

- **Id**- long usado como identificador da pergunta.
- **CreationDate**- LocalTime representando a data de criação da pergunta.
- **Score**- int representando o score da resposta.
- **Title**- String representando Titulo da pergunta.
- **Tags**- set de tags utilizadas na Pergunta
- **CommentCount**- int representando o número de comentários à pergunta.
- **respostas**- set de Respostas da respetiva pergunta.

5. *Interrogações*

Após a devida analise/reflexão do problema e armazenamento dos dados, estão reunidas as condições necessárias para responder às interrogações referidas no enunciado.

5.1 Interrogação 1 - info from post

Esta interrogação procura a partir do id de um post, retornando o titulo do post e nome do seu autor.

5.1.1 Estratégia

Devido ao uso de HashMap a resposta a esta interrogação é quase trivial, bastando procurar direta na HashMap de perguntas pelo id dado, caso não exista fazemos o mesma na HashMap de respostas encontrando assim a sub estrutura do post da qual usamos o OwnerUserId para mais uma vez procura na HashMap de users e saber o seu nome. Caso o post seja uma respostas obriga ainda a uma procura da pergunta a partir do ParentID indo buscar o titulo da respetiva pergunta.

5.2 Interrogação 2 - top most active

Esta interrogação encontra os top N users com o maior número de posts de sempre.

5.2.1 Estratégia

De forma a responder a esta interrogação será sempre necessário percorrer toda a HashMap de users filtrando os que têm o valor mais alto de número de posts, valor este que é calculado no parser.

5.3 Interrogação 3 - total posts

Esta interrogação retorna o número total de posts entre um dado intervalo de tempo.

5.3.1 Estratégia

De forma a responder a esta interrogação optamos pelo uso de um contador que é incrementado sempre que encontra um post (pergunta ou resposta) dentro do dado intervalo. Bastando apenas uma travessia da HashMap de Perguntas e Respostas respetivamente.

5.4 Interrogação 4 - question with tag

Esta interrogação retorna os ids das perguntas que usaram uma dada tag num dado intervalo de tempo.

5.4.1 Estratégia

De forma a responder a esta interrogação criamos uma lista na qual é inserido o id de uma pergunta se a mesma estiver no dado intervalo e se usar uma dada tag. Necessitando apenas de uma travessia da HashMap de Perguntas.

5.5 Interrogação 5 - get user info

Esta interrogação retorna a informação relativa a um certo user e a lista dos seus últimos 10 posts, podendo estes ser tanto perguntas como respostas.

5.5.1 Estratégia

A informação relativa a qualquer user de um certo id pode ser obtida através de mapeamento direto utilizando o seu id na hash de users, uma das vantagens da utilização de HashMap. De forma a obter a lista dos últimos 10 posts do utilizador optamos pela criação de três listas uma onde eram guardadas todas perguntas desse mesmo utilizador assim como uma onde guardamos todas as respostas do mesmo e ainda uma terceira onde colocamos os ids dos 10 últimos posts filtrando as duas listas anteriores.

5.6 Interrogação 6 - most voted answers

Ids das respostas com mais score obtido, dentro de um determinado intervalo de tempo.

5.6.1 Estratégia

De forma a responder a esta interrogação fazemos apenas uma travessia à HashMap de respostas guardando todas as que se encontrem entre o dado intervalo de tempo ordenando posteriormente por score.

5.7 Interrogação 7 - most answered questions

Ids das perguntas com maior número de respostas efetuadas dentro de um dado intervalo de tempo.

5.7.1 Estratégia

De forma a responder a esta interrogação basta apenas fazer uma travessia à Hash de Perguntas guardando as que se encontram dentro do dado intervalo ordenando posteriormente pelo numero de respostas que é dado pelo tamanho do set respostas presente na sub-estrutura Pergunta.

5.8 Interrogação 8 - contains word

Dado uma palavra são procuradas N perguntas que contêm a palavra procurada no seu título.

5.8.1 Estratégia

À semelhança das interrogações anteriores, nesta também é feita uma travessia da Hash de perguntas, guardando as mesmas caso se verifique a existência de uma dada palavra no título e posteriormente ordenando por cronologia inversa.

5.9 Interrogação 9 - both participated

Esta interrogação devolve uma lista de tamanho N com ids de perguntas onde dois dados utilizadores participam.

5.9.1 Estratégia

De forma a responder a esta interrogação, guardamos uma lista para ambos utilizadores de todas as perguntas em que cada um participa verificando de seguida se o outro é autor de alguma resposta nessa lista, guardando o id da pergunta numa terceira lista, caso se verifique. Finalmente, num 3º caso, percorremos a HashMap de perguntas e verificamos se ambos os utilizadores são autores das respetivas respostas para cada pergunta, em caso afirmativo guardamos numa lista o id da respetiva pergunta.

5.10 Interrogação 10 - better answer

Esta interrogação devolve a melhor resposta, baseado num rate, a partir do id de uma dada pergunta.

5.10.1 Estratégia

De forma a responder a esta interrogação basta uma procura direta na Hash de perguntas retornando de seguida a resposta com melhor rate calculado, a partir do set de respostas da estrutura Pergunta.

6. *Conclusão*

A realização do projeto em Java a nosso ver foi mais simples, talvez por já se ter realizado o mesmo em C que obriga a outro tipo de cuidado no que toca a memória e a falta de estruturas com boas APIs. É de notar que as nossas estruturas sofreram ligeiras alterações, de forma a tirar melhor partido do paradigma utilizado, optando por retirar a árvore de respostas referentes a uma pergunta (sub-estrutura), substituindo a mesma por um set com as respetivas respostas. Em suma, sentimos-nos satisfeitos com o trabalho realizado tendo verificado que paradigmas diferentes apresentam dificuldades diferentes na realização do mesmo projeto.