Escola Superior de Gestão e Tecnologia [IPSantarém]

Licenciatura em Informática

Ano Letivo 2019/2020

Linguagens de Programação

Percurso de entregas

Discentes:

Miguel Nunes – 170100142

Raul Boal – 170100102

Docentes:

Prof. António Roberto

Prof. Ricardo Rodrigues

Santarém, 15 de junho de 2020

Resumo

Para o nosso Projeto de Linguagens de Programação tínhamos duas possibilidades, ou fazíamos um projeto sobre grafos, onde nós não estávamos muito familiarizados, ou fazíamos um projeto sobre árvores binárias ou n-árias.

Visto que o nosso tema é as “entregas ao domicilio”, fizemos um programa Java que nos desse as entregas que uma certa Rua tem.

Criamos então uma árvore binária com 2 classes, RuaTree, que é a nossa classe da árvore binária e a NoRua que é a classe auxiliar da nossa árvore com algumas variáveis como “private String Rua”, “private int RuaTam” (que nos dá o Tamanho da Rua), entre outras variáveis, mais 1 interface Rua com os métodos da nossa árvore binária.

Índice

Introdução

“Quantas encomendas pode uma rua receber?”

O nosso Projeto procura responder a este problema, mostrando de uma maneira muito simples, inserindo as Ruas que vão ser alvo das encomendas ou removendo Ruas caso estas já não tenham encomendas e também inserindo ou removendo as Encomendas.

Fizemos uma árvore, com 14 métodos, para simplificar ao máximo a nossa ideia original, que era o Percurso de entregas ao Domicílio. Para isso fizemos uma pesquisa intensiva sobre árvores em java e decidimos fazer a nossa árvore baseada numa árvore AVL, ou seja, uma árvore binária com autobalanceamento, usada em árvores de pesquisa com backup em memória, visto que esta não foi retratada em aula e que a conseguimos aproveitar muito bem para os nossos objetivos, dentro do projeto.

Materiais e Métodos

Todo o nosso projeto foi feito na integra no Eclipse IDE e depois entregue pela plataforma GitHub, através da utilização do Git na linha de comandos.

Utilizamos, como já foi referido anteriormente 14 métodos neste projeto.

Dos quais temos:

➔ inserir();

Método que insere as Ruas.

➔ remover();

Método que remove as Ruas.

➔ removerNo();

Método que remove os Nós.

➔ altura();

Método que retorna a altura da árvore (quantos níveis tem).

➔ rotacaoDireita();

Método que balanceia a árvore para a direita quando um nó esta "desbalanceado" em linha reta para a esquerda.

➔ rotacaoEsquerda();

Método que balanceia a árvore para a esquerda quando um nó esta "desbalanceado" em linha reta para a direita.

➔ rotacaoDuplaEsquerdaDireita();

Método que balanceia a árvore quando um nó estiver "desbalanceado" e o seu filho estiver inclinado no sentido inverso ao pai da Esquerda para a Direita.

➔ rotacaoDuplaDireitaEsquerda ();

Método que balanceia a árvore quando um nó estiver "desbalanceado" e o seu filho estiver inclinado no sentido inverso ao pai da Direita para a Esquerda.

➔ inserEncomendas();

Método que insere as Encomendas.

➔ removEncomendas();

Método que remove as Encomendas.

➔ procurar();

Método que procura a Rua onde se insere as Encomendas.

➔ Balanceado();

Método que faz o Balanceamento da árvore com ajuda dos métodos de Rotação.

➔ isEmpty();

Método que retorna nulo se não houver árvore.

➔ encontraCurta();

Método que encontra a String mais pequena das Ruas.

Considerações Finais

Inicialmente o nosso Trabalho tinha como objetivo a construção de 3 árvores, duas árvores binárias e uma árvore n-ária (com “n” filhos), que comunicassem entre si de modo a que retornassem uma descrição pormenorizada do percurso necessário para o estafeta entregar, quer sejam cartas ou pacotes à pessoa e porta indicada.

A primeira árvore binária era uma árvore de Ruas onde, dentro dos nós, continha uma certa rua que o estafeta teria de percorrer e tinha uma subárvore para os edifícios (segunda árvore binária) que seria sempre nulo a não ser que o estafeta estivesse no final da rua.

A segunda árvore binária dentro dos nós, continha os edifícios da rua em questão e outra subárvore, a árvore dos pacotes de encomenda (a árvore n-aria), onde dentro dos nós desta árvore teríamos duas ArrayLists, uma para guardar os nomes das pessoas que recebiam os pacotes e outra para guardar os nomes das pessoas que recebiam as cartas, que acabamos por modificar e deixar apenas um ArrayList<String> Entregas que englobavas ambas as cartas e os pacotes.

Esta implementação deu problemas pois havia duas árvores binárias a comunicarem com uma árvore não binária. Tínhamos de ter em conta o facto que estas árvores necessitam de regras muito diferentes o que pode dar e deu, no nosso caso alguns problemas.

Tendo em conta isto tudo, foi nos sugerido pelo Professor Ricardo simplificar o nosso projeto e foi o que fizemos. Alteramos na integra a estrutura do projeto. Invés de termos 3 árvores, passamos para 1 árvore binária.

Conclusão

Dentro dos possíveis, não totalmente satisfeitos por termos desistido da nossa ideia original, acabámos por conseguir chegar aos objetivos que definimos para este projeto de árvore binária. Conseguimos pegar num conceito que não foi dado em aula, que foi a árvore AVL, estudamo-lo e aperfeiçoamo-lo de maneira a que fosse a melhor opção ou pelo menos, uma boa opção para demonstrar as nossas capacidades e fazer o nosso projeto, para além do facto de demonstrar que nós conseguimos adaptar, com ajuda dos docentes, a ideia original para esta ideia “convertida”.

Concluímos por tanto que este Projeto, apesar das várias adversidades desde o início até ao final do mesmo, foi um Projeto que nos ajudou bastante a progredir dentro das árvores, quer estas sejam binárias, n-árias ou AVL e também nos mostrou que nem sempre a nossa primeira ideia é a melhor. Há todo um processo por trás, que nos faz “limar arestas”.