RELATÓRIO A RESPEITO DO TRABALHO 1 – LISTA ORDENADA

INE5609-03238A (20181) – Estruturas de Dados

João Grasel Cariolato – Thiago Brezinski

Pode-se afirmar que este foi dividido em duas partes, sendo elas “Definições” e “Refatoração”.

Pode-se dizer que a parte de Definições foi baseada em alguns princípios precipitados para o projeto, o que acabou refletindo em retrabalho. Logo de início, definimos alguns pontos necessários e que fundamentam a maioria dos métodos desenvolvidos, como a inicialização dos parâmetros dos seus construtores, onde, por exemplo, a classe Caixa recebe como parâmetros o “elemento” e o “ID” e utiliza "null" para os parâmetros “próxima” e “anterior”, e a classe Lista, quando instanciada, define os parâmetros “atual”, “primeira” e “ultima” como "null" e a “quantidade” com 0. Neste primeiro momento, o entendimento era de que os IDs eram controlados por uma variável global da Lista chamada “proximoID”, inicializada como 1 e que era utilizada como parâmetro para criação da Caixa. Definimos também que quando a caixa atual fosse deletada, o cursor apontaria para a caixa anterior (como pode ser visto no esquema da Figura 2). E, por ultimo, conversamos também sobre o que fazer quando o usuário pedir para que o cursor avançasse ou retrocedesse uma quantidade maior de Caixas que a lista realmente possuía a frente ou atrás do cursor (dependendo da ação escolhida). Acabamos por definir que mesmo que o cursor chegasse na ultima ou primeira posição e tivesse mais unidades para avançar ou retroceder, ele continuaria avançando ou retrocedendo até que não houvesse mais unidades para percorrer e, assim, ao chegar na ultima posição ele aponta diretamente para a primeira e vice-versa (para mais fácil entendimento visualizar a Figura 3).

Assim, começou-se o projeto criando uma classe Caixa e uma classe Lista duplamente encadeada (como as classes vistas em sala de aula e que podem ser visualizadas nos esquemas apresentados na Figura 1), porém como já comentado, por falta de um entendimento necessário a respeito das exigências mínimas do projeto, desenvolvemos uma caixa que aceitava unicamente elementos inteiros e somente uma lista diretamente ordenada.

Após uma conversa com o professor e um entendimento maior sobre as expectativas, adentrou-se na segunda parte do projeto, onde ocorreu a refatoração do código, onde a partir de uma lista obtivemos duas, sendo uma lista duplamente encadeada e uma lista ordenada, que herda a primeira e possui os métodos de inserirOrdenado, adicionarFake, removerFake e imprimirListaOrdenada. Assim, desacoplamos o código e facilitamos o reaproveitamento do mesmo em algum caso futuro. Implementamos também

uma interface IBuscavel, aplicada no elemento da Caixa, o que permite que qualquer tipo de elemento seja adicionado a uma caixa e cumprindo assim a exigência de que a lista seja genérica. Também retiramos a variável “ID” da classe Caixa, uma vez que o elemento já possui um, justamente por implementar uma interface que obriga a implementação de um método “getID”.

Desenvolvemos duas classes extras (uma classe Pessoa e uma classe Veículo, as quais recebem como parâmetro no construtor um ID) para fins de testes, os quais consistiram no seguinte procedimento: instanciar uma lista, instanciar alguns objetos Pessoas e alguns objetos Veículos e adicioná-los à lista instanciada de maneira ordenada.

Por fim, concluímos que além do desenvolvimento prático do que foi visto em teoria na sala de aula, o trabalho foi muito importante para visualizarmos que apesar de nos precipitar as vezes quanto a requisitos do trabalho, é imprescindível quando existirem dúvidas que exista uma validação e um esclarecimento das mesmas, o que provavelmente acarretará em um retrabalho, porém garantirá que os requisitos mínimos foram cumpridos, assim como as expectativas.

ANEXOS

Figura 1 - Estrutura de uma lista duplamente encadeada.

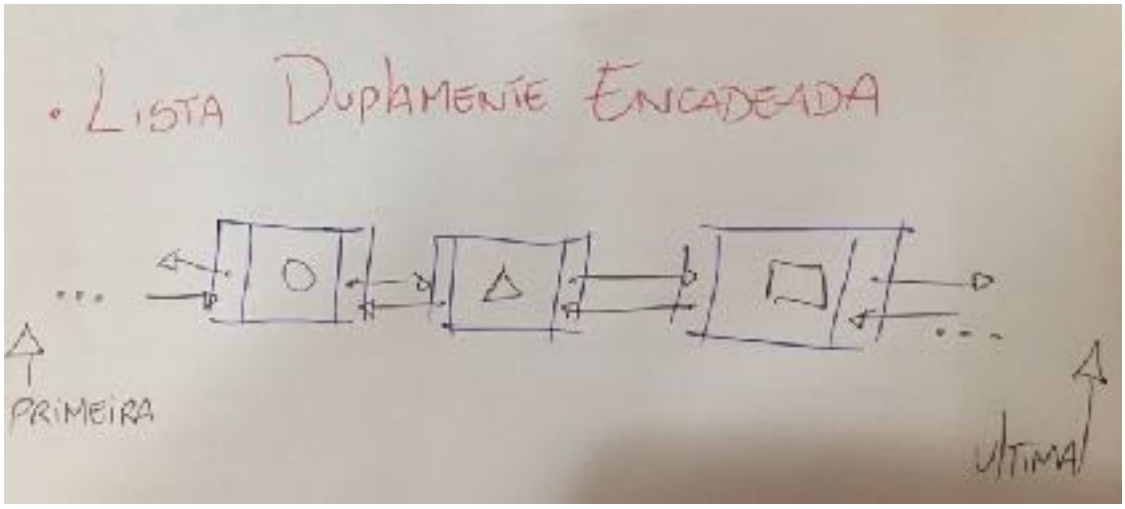

Figura 2 – Procedimento correspondente ao ato de excluir a caixa apontada pelo cursor.

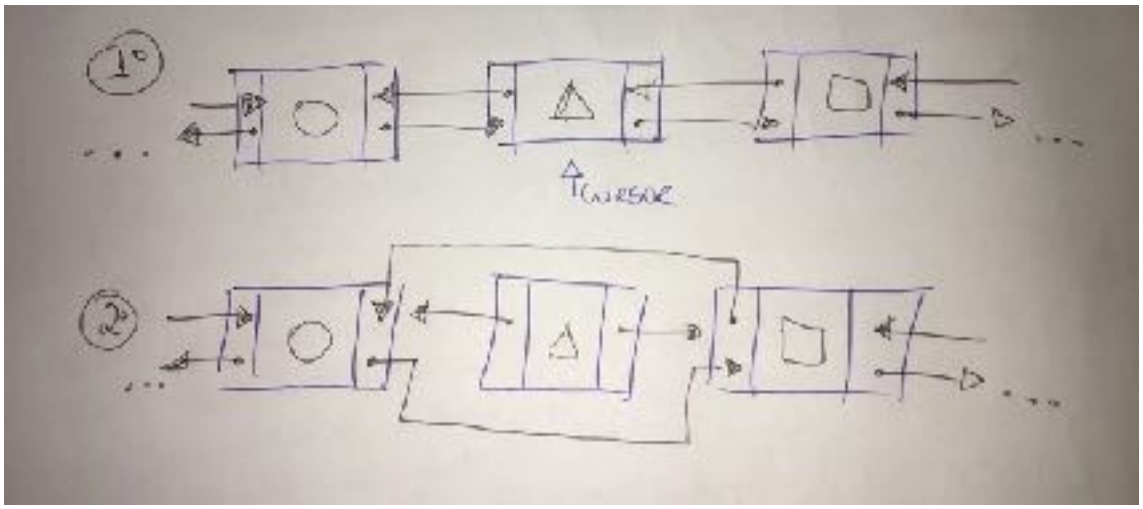

Figura 3 – Estrutura referente ao avanço ou retrocesso do cursor.