Teste Listas Circulares[1] - Estrutura de Dados

15 / 05 / 2024

---

1 - Framework e Playlist

Elon Lages Lima, grande fã de Twenty One Pilots, Taylor Swift e Barões da Pisadinha, já projetou seu próprio player de músicas em C++. O código, contudo, não era open source e foi apenas descoberto por emapianos durante uma feira de doações de materiais acadêmicos do IMPA. Além disso, o código é carente de um seletor de músicas, já que Elon, ao perder a paciência com C++, decidiu voltar a estudar topologia[2] .

1.a - Criando a Roda

Para completar esse código[3] , implementaremos um seletor de músicas na forma de uma **lista circular duplamente encadeada** que contém IDs numéricos referentes a certas músicas. Ou seja, devemos implementar o seguinte:

- A estrutura dos nós e da lista;
- Uma função para criar essa lista a partir de uma array de IDs;
- Uma função que retorne o ID atual e avance a lista;
- Uma função que retorne o ID atual e retroceda a lista.

Note que o avanço e retrocesso devem ser operações sobre os nós e não sobre os valores deles.

1.b - Quebrando *(graciosamente)* a Roda

É natural que numa playlist em loop, algumas músicas vão começar a ficar repetitivas. Além disso, quando o player parar de ser usado, precisamos limpar a lista. Implemente, então, o seguinte:

- Uma função que retira um nó por ID;
- Uma função que deleta a lista e libera sua memória.

1.c - Juntando Rodas

Suponha agora que você possui duas playlists muito boas — Elon Lages certamente tinha — e queira juntá-las em uma só. Uma forma de fazer isso é intercalando uma música de cada lista, criando uma nova lista circular. Caso uma das listas seja maior que a outra, o que restar da lista maior deve simplesmente ser adicionado ao final da seção intercalada.

- Implemente a função que realize essa junção de listas.

---

[1] Duplamente encadeadas

[2] Uma escolha sábia e que certamente favoreceu o avanço da humanidade.

[3] Que é completamente fictício e, portanto, não está disponível para consulta.

2 - Are we in the loop?

Durante as implementações de listas duplas, inevitavelmente um emapiano cometeu um erro na implementação do código que levou a um dos ponteiros da nossa lista a apontar para um nó anterior (ou posterior, qualquer que esteja mais errado!). O resultado disso é um loop.

Crie uma função que checa se uma lista encadeada possui um loop. Isso é, se existe um nó a partir do qual avançar ou retroceder na lista levará eventualmente a ele mesmo. Note que apenas uma dessas opções precisa ser verdade, e.g. avançar leva ao mesmo nó, mas retroceder leva ao `HEAD`.

A função deve receber um nó inicial da lista encadeada (um `HEAD`) e retornar um valor booleano (`True` ou `False`).

3 - Helping an Ouroboros

Ouroboros (do grego Οὐροβόρος: 'que consome a cauda') é um símbolo antigo frequentemente representado por uma serpente — ou por um dragão — que morde a própria cauda. O Ouroboros costuma ser representado pelo círculo, o que parece indicar, além do eterno retorno, o ciclo da vida.

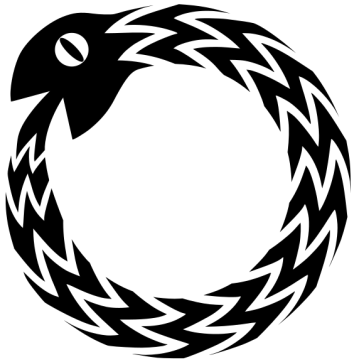

Figura 1: Representação artística de um ouroboros.

Neste exercício, um ouroboros é representado por uma lista circular de n números inteiros não-negativos. Devido a sua natureza autofágica[4], a cabeça (`HEAD`) do ouroboros começa a comer sua cauda (`TAIL`). Esse processo se dá seguinte forma:

- Se o valor da `HEAD` é a, e o valor da `TAIL` é b, então o valor da `HEAD` é substituído por $2^a \cdot (2b + 1)$, e o nó da cauda é deletado.

Esse processo se repete até só sobrar um nó no ouroboros, com um valor final restando.

Crie uma função que, ao receber um valor **final** e o tamanho (n) do ouroboros que o originou, retorna uma lista circular que representa os valores presentes no ouroboros originalmente.

[4] Autofágico: relativo àquilo que consome da própria carne.