Relatório

Introdução

O problema apresentado se caracterizava pelo desenvolvimento de um programa para o gerenciamento de vôos em aeroportos brasileiros. Funções para inserção de aeroportos, inserção e remoção de vôos e busca por vôos, impressão de vôos a partir de um aeroporto específico e impressão de todos os vôos deveriam ser implementadas nesse problema, sendo a mais complexa aquela que faz a busca por vôos.

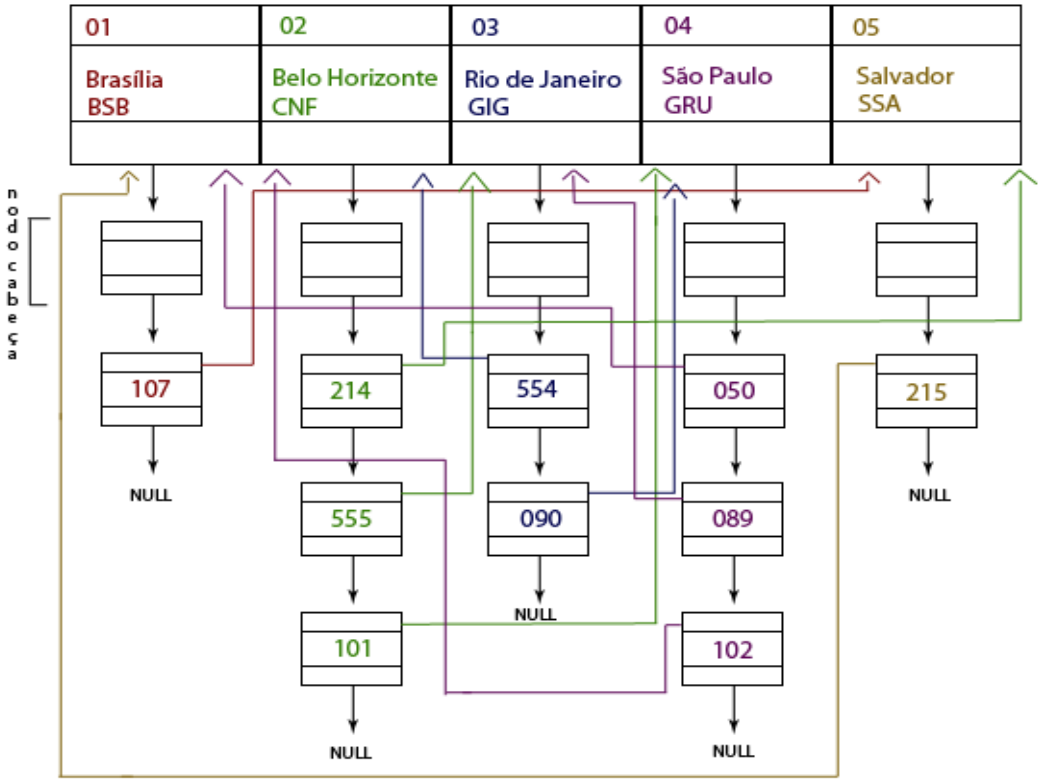
Implementação

Estruturas

Para este programa, foram utilizadas estruturas de dados do tipo lista implementadas de duas formas diferentes. A primeira lista foi implementada como um vetor de estruturas, onde ficaram armazenados os aeroportos. Esta lista é iniciada com o tamanho de dez elementos, e cada elemento possui uma ID, o nome da cidade, o código do aeroporto e uma lista encadeada com todos os vôos que partem daquele nodo em específico.

Logo que o vetor de aeroportos é criado, ele é também iniciado com as IDs dos aeroportos em ordem crescente, os campos cidade e código são preenchidos com a string "<vazio>", para indicar que ainda não há nenhum aeroporto cadastrado naquela posição. É criado também o nodo cabeça da lista encadeada de adjacências. Nesta lista de adjacências, cada nodo é um vôo e possui em sua estrutura o número do vôo, um ponteiro para o próximo vôo (ou para nulo, se for o último item da lista) e um segundo ponteiro para a lista de adjacencias aeroporto de destino. O nodo cabeça está sempre vazio, e é nele que novos vôos são inseridos sendo criado, em seguida, um novo nodo cabeça, que será colocado no lugar do anterior.

A figura 1 mostra um exemplo de como as estruturas de dados foram montadas. Cada posição do vetor é ocupado por um aeroporto, com ID, nome da cidade, código e lista de adjacências. Cada vôo aponta para seu aeroporto de destino.



Funções

- Cadastra Aeroporto

  Essa é a função que insere os aeroportos no vetor. Primeiramente ela faz uma busca para encontrar alguma posição vazia, se não houver posição vazia, retorna uma mensagem de erro, se houver, insere o novo aeroporto e imprime a ID com que o aeroporto foi cadastrado.

- Cadastra Vôo

  Primeiramente, busca o aeroporto de origem e o de destino. O número do vôo é inserido no nodo cabeça do aeroporto de origem e é colocado um ponteiro para o aeroporto de destino. Um novo nodo cabeça é alocado e passa apontar para o nodo que contém o vôo que foi cadastrado.

- Remove Vôo

  Busca o aeroporto de origem, então percorre a lista de adjacências procurando pelo vôo a ser removido, usando como referência no número do vôo, sempre armazenando o nodo anterior. Quando encontra o nodo a ser removido, faz o nodo anterior apontar para o que

o nodo a ser removido apontava, então libera a memória alocada para o nodo.

- Imprime Vôos

  Percorre o vetor buscando o aeroporto passado como parâmetro, então percorre a lista de adjacências, imprimindo toda a lista encadeada de adjacências.

- Imprime Tudo

  Percorre todo o vetor imprimindo a lista de adjacência de cada um deles.

- Procura Vôo

  Função mais complexa do programa, ela deve buscar todos os caminhos possíveis de um aeroporto ao outro, com um número máximo de vôos, que é passado como parâmetro. Para isso, é usado um algoritmo de *backtracking* que, por sua vez, utiliza recursão. A figura 1 ilustra o funcionamento desse algoritmo.

Conclusão

A complexidade do programa reside, sobretudo, na função ProcuraVoo, que possui várias chamadas recursivas, sendo também o procedimento mais trabalhoso de se desenvolver, graças à sua implementação, que usa técnicas de força bruta – mais especificamente "*backtracking*". As funções restante, basicamente, trabalham com listas, tanto encadeadas quanto implementadas com vetores.