INSTITUTO FEDERAL
Catarinense
Campus Blumenau

CCA0916 - Estruturas de dados I - BCC

Bacharelado em Ciência da Computação

Prof. Dr. Paulo César Rodacki Gomes

Lista de exercícios - 05 - Filas (com arrays)

O objetivo desta atividade prática em laboratório é realizar a implementação de Filas utilizando arrays, também chamadas de Filas Circulares devido ao fato da fila correr de forma circular dentro do array. Por motivo de simplicidade, vamos implementar filas para armazenar valores do tipo primitivo `int`, de acordo com o diagrama de classes da figura 1 (interface `Fila` e classe `FilaVetor`).

Considerações sobre a linguagem de programação utilizada

- **Java**: Se você implementar o exercício em Java, crie a interface `Fila`.
- **C++**: Por outro lado, se você implementar em C++, `Fila` deve ser uma classe puramente abstrata.
- **C padrão ANSI**: se você implementar em C, procure adaptar o enunciado do exercício para as particularidades da linguagem. Por exemplo, `FilaVetor` passa a ser um dado estruturado (com `struct` e `typedef`. Os métodos passam a ser funções.

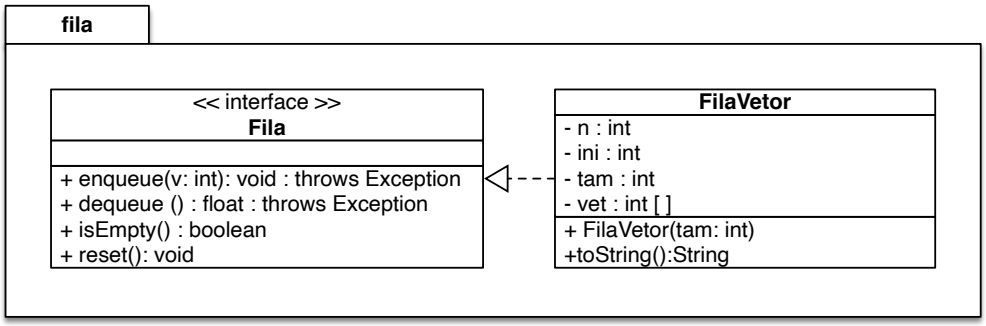

Figura 1: Diagrama de classes do pacote fila

Atributos:

A classe `FilaVetor` deve ter os seguintes atributos privados:

- `int tam`: especifica a capacidade máxima da fila, este valor é passado ao método construtor da classe, e é usado para criação do array (ou vetor) de elementos da fila;
- `int n`: quantidade atual de elementos da fila, também utilizado para saber a posição atual do final da fila;
- `ini`: índice da posição de ínício da fila. O início da fila é a posição do próximo elemento a ser retirado da fila;
- `int vet[]`: array de números inteiros (estrutura usada para armazenar os elementos da fila).

Observação: note que a classe `FilaVetor` não possui um atributo para marcar o final da fila. Este valor deve ser calculado dinamicamente sempre que precisar ser utilizado.

Métodos a serem implementados:

- `enqueue(int v)`: insere o valor v no final da fila. Se a fila já estiver cheia, este método deve lançar uma exceção;
- `dequeue()`: retira e retorna o elemento do início da fila. Se a fila estiver vazia, o método deve lançar uma exceção;
- `toString()`: retorna uma string com os elementos da fila na ordem de início ao fim. Note que isto é diferente de simplesmente listar os elementos do vetor. **Observação**: em C++ a funcionalidade equivalente pode ser implementada por meio de sobrecarga do operador `<<`;
- `isEmpty()`: retorna um valor booleano indicando se a fila está vazia (`true`) ou não vazia (`false`);
- `reset()`: limpa a fila (deixa a estrutura com zero elementos e reinicializa os atributos `ini` e `n`).
- `FilaVetor concatena(FilaVetor f2)`: tem por finalidade *criar uma nova fila* a partir da concatenação de duas filas previamente existentes. Observe que a fila resultante deve ter tamanho total igual à soma dos tamanhos das duas filas originais, e seus elementos devem estar corretamente inseridos conforme ilustrado na figura 2.

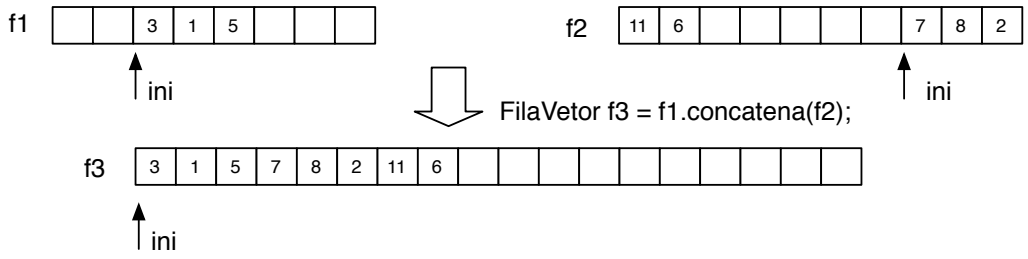

Figura 2: Demonstração da operação concatena

- `FilaVetor merge(FilaVetor f2)` tem por finalidade *criar uma nova fila* a partir da concatenção intercalada de duas filas previamente existentes. Observe que a fila resultante deve ter tamanho total igual à soma dos tamanhos das duas filas originais, e seus elementos devem estar corretamente inseridos conforme ilustrado na figura 3.

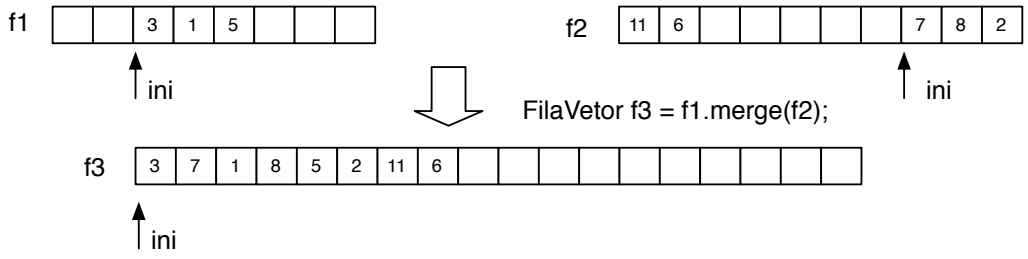

Figura 3: Demonstração da operação merge

Observação: Implemente uma classe Java ou programa principal em C++ chamado `filaMain`, que deve conter o método `main` para testar e demonstrar o funcionamento da fila implementada.