Filas

Fabrício J. Barth

BandTec - Faculdade de Tecnologia Bandeirantes

Março de 2011

Tópicos Principais

- Introdução
- Interface do tipo fila
- Implementação de fila com vetor
- Implementação de fila com lista encadeada

Introdução - Fila

- A estrutura de fila é uma analogia natural com o conceito de fila que usamos no nosso dia-a-dia: quem primeiro entra numa fila é o primeiro a ser atendido (a sair da fila).
- Sua ideia fundamental é que só podemos inserir um novo elemento no final da fila e só podemos retirar o elemento do início.

Interface do tipo Fila

public interface Fila

- **cria()** : cria uma estrutura de fila (em linguagens O.O. isto pode ser feito no construtor da classe);
- **inseri(v)**: inseri um elemento no fim da lista;
- **retira()**: retira um elemento no início da lista;
- **vazia()**: informa se a fila está ou não vazia;
- **libera()**: destrói a fila, liberando toda a memória usada pela estrutura.

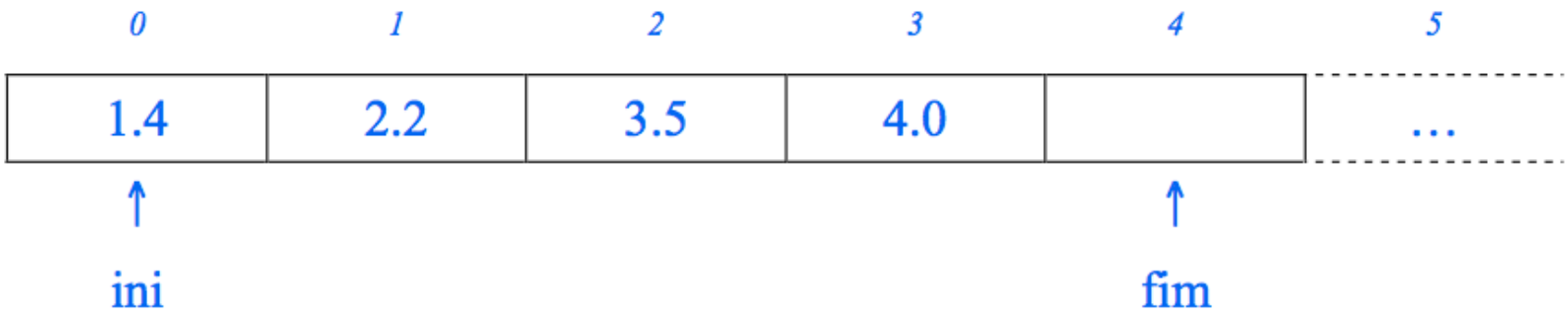
Interface do tipo Fila

```
1  public interface Fila {
2      public Fila cria();
3      public void inseri(float v);
4      public float retira();
5      public boolean vazia();
6      public void libera();
7  }
```

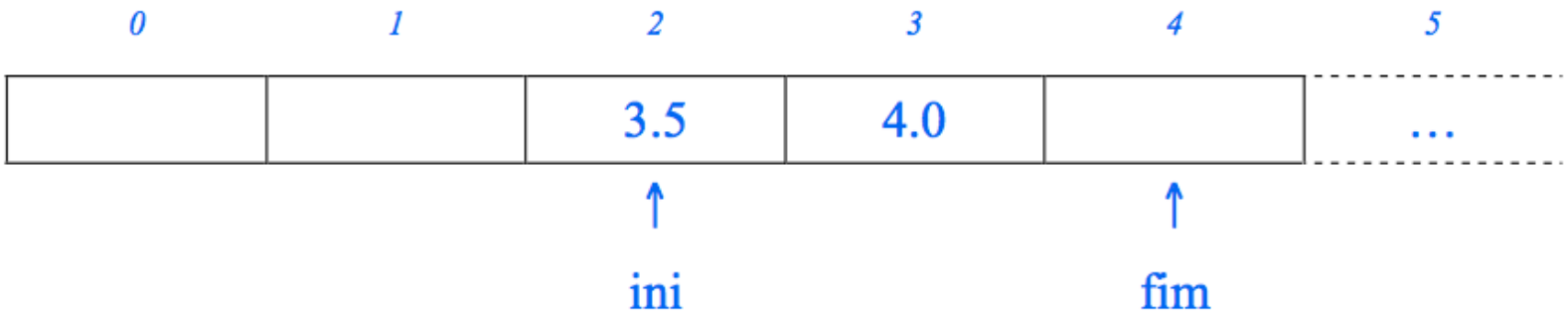
Implementação de fila com vetor

- Utiliza uma estrutura finita de tamanho N.
- O processo de inserção e remoção em extremidades opostos fará a fila **andar** no vetor.

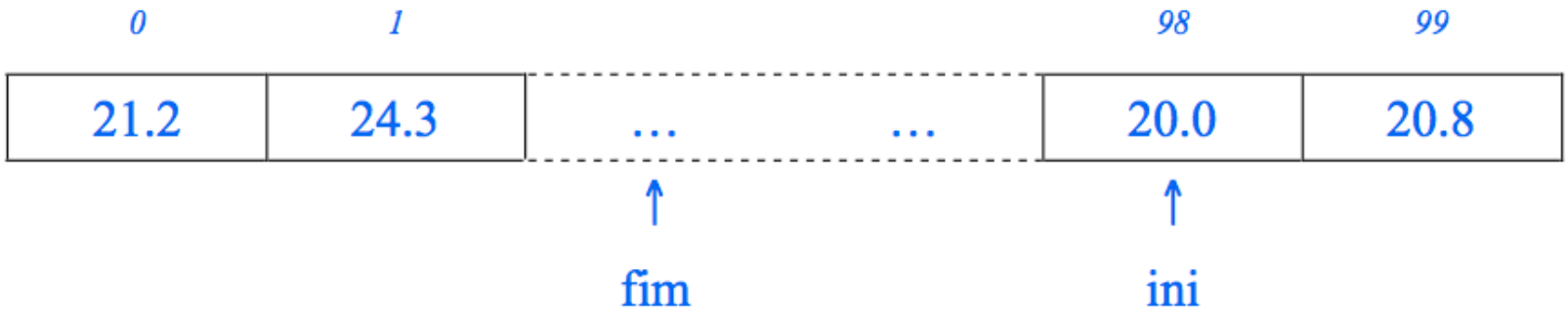
Fila após inserção de quatro novos elementos



Fila após retirar dois elementos



Fila com incremento circular



Incremento circular

Para esta implementação, os índices do vetor são incrementados de maneira que seus valores progridam "circularmente".

```
1  private int incr(int i){
2      if(i == N)
3          return 0;
4      else
5          return i+1;
6  }
```

Essa mesma função pode ser implementada de uma forma mais compacta, por meio de um operador módulo (que retorna o resto de uma divisão):

```
1  private int incr(int i){
2      return (i+1) % N;
3  }
```

Com o uso do operador módulo, em geral optamos por dispensar a função auxiliar e escrever diretamente o incremento circular:

$$i = (i + 1)\%N;$$

Implementação de fila com vetor

```
1  public class FilaVetor implements Fila{
2      /*determina a qtd max de itens que a fila pode ter*/
3      private static final int N = 100;
4      /*armazena os elementos*/
5      private float[] elementos;
6      /*determina a possicao de inicio da fila*/
7      private int inicio;
8      /*registra a qtd de itens armazenados na fila*/
9      private int n;
```

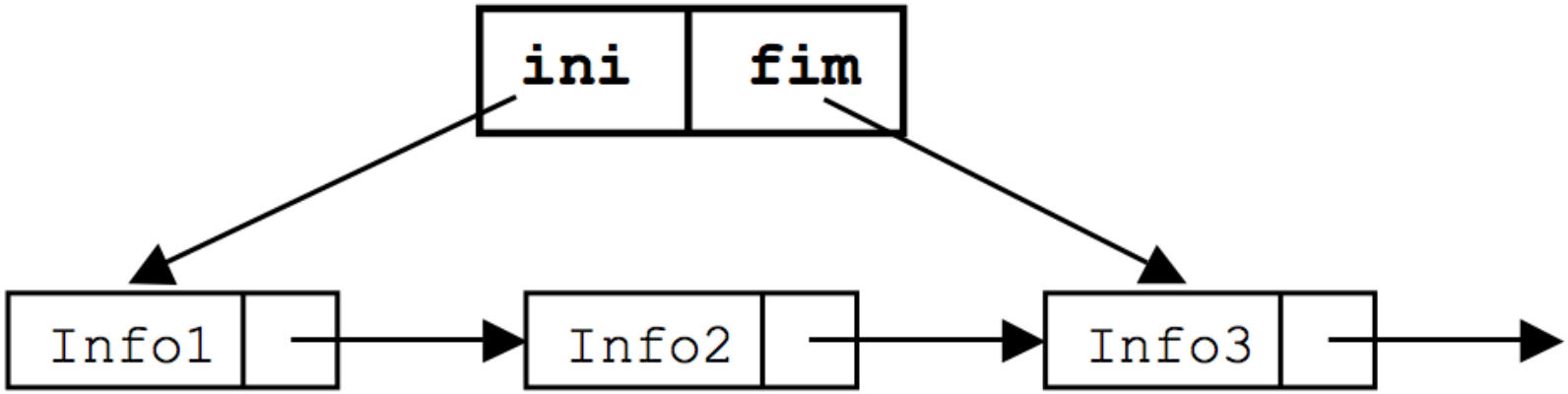
```
1  public Fila cria() {
2      this.elementos = new float[N];
3      this.inicio = 0;
4      this.n = 0;
5      return this;
6  }
7
8  public boolean vazia() { return (this.n == 0);}
9
10 public void libera() {
11     this.elementos = new float[N];
12     this.inicio = 0;
13     this.n = 0;
14 }
```

```
1  public void inseri(float v) {
2      int fim;
3      if(this.n == N){
4          System.out.println("Capacidade da fila estorou."
5              + "Nao eh posivel inserir mais elementos.");
6          return;
7      }
8      fim = (this.inicio + this.n) % N;
9      this.elementos[fim] = v;
10     this.n++;
11 }
```

```
1  public float retira() {
2      float v;
3      if(vazia()){
4          System.out.println("Fila vazia."
5             + "Nao eh possivel remover nenhum elemento.");
6          return 0;
7      }
8      /*retira elemento do inicio da fila*/
9      v = this.elementos[inicio];
10     this.inicio = (this.inicio + 1) % N;
11     this.n--;
12     return v;
13 }
```

Implementação de fila com lista

- Deve utilizar dois objetos que apontam um para o início da lista e o outro para o fim.



Implementação de fila com lista

```
1  public class FilaLista implements Fila{
2
3      private No inicio;
4      private No fim;
```

```
1   public Fila cria() {
2       this.inicio = null;
3       this.fim = null;
4       return this;
5   }
6
7   public boolean vazia() { return (this.inicio == null);}
8
9   public void libera() {
10      this.inicio = null;
11      this.fim = null;
12  }
```

```
1  public void inseri(float v) {
2      No novo = new No();
3      novo.setInfo(v);
4      novo.setProx(null);
5      /*verifica se lista nao esta vazia*/
6      if(this.fim != null)
7          this.fim.setProx(novo);
8      else
9          /*fila estava vazia*/
10         this.inicio = novo;
11     /*fila aponta para novo elemento*/
12     this.fim = novo;
13 }
```

```
1  public float retira() {
2      No temp;
3      float v;
4      if(vazia()){
5          System.out.println("A fila esta vazia");
6          return 0;
7      }
8      temp = this.inicio;
9      v = temp.getInfo();
10     this.inicio = temp.getProx();
11     /*verifica se a fila ficou vazia*/
12     if(this.inicio == null)
13         this.fim = null;
14     return v;
15 }
```

Exemplo de uso

```
1  public TestaFilaLista(){
2      Fila f = new FilaLista();
3      f.cria();
4
5      for(int i=0; i<100; i++)
6          f.inseri(i);
7
8      for(int i=0; i<110; i++)
9          System.out.println("Retirou: "+f.retira());
10 }
```

Material de **consulta** e **referência**

- Capítulo 12 do livro: "Introdução a Estruturas de Dados" do Waldemar Celes, Renato Cerqueira e José Lucas Rangel.