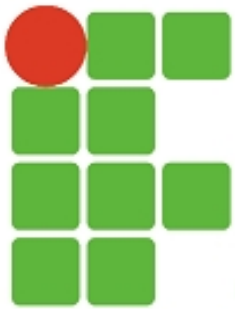


INSTITUTO FEDERAL DE
EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
PARAÍBA
Campus Campina Grande

6

Curso Engenharia da Computação
Programação e Estruturas de Dados

Filas – Fundamentos e Implementações

Prof. César Rocha
cesarocha@ifpb.edu.br

Objetivos

- *Explorar os conceitos fundamentais acerca do uso de **filas** utilizando a linguagem C*
  - *Organização e implementação, características, vantagens, desvantagens, regras de utilização, operações básicas e algoritmos de implementação*
- *Neste módulo, serão abordadas ainda as seguintes implementações de filas: **seqüencial, encadeada e circular***
- *Este módulo será utilizado como referência na entrega dos futuros projetos*
  - *Implementação das estruturas e algoritmos, criação das **bibliotecas** e práticas de laboratório*

Motivação

- *Uma das estruturas de dados mais utilizadas em programação é a **fila***
  - *Um exemplo de utilização em computação é a implementação de uma fila de impressão*
- *Assim como na pilha, os acessos aos elementos em uma fila também seguem uma regra :*
  - *Lembre-se da pilha, onde: **L**ast **I**n **F**irst **O**ut*
  - *Porém, uma fila é: **F**irst **I**n **F**irst **O**ut*
- ***A idéia fundamental***
  - *Só podemos inserir um novo nó no final da fila*
  - *Só podemos retirar a partir do início da fila*

Estratégia

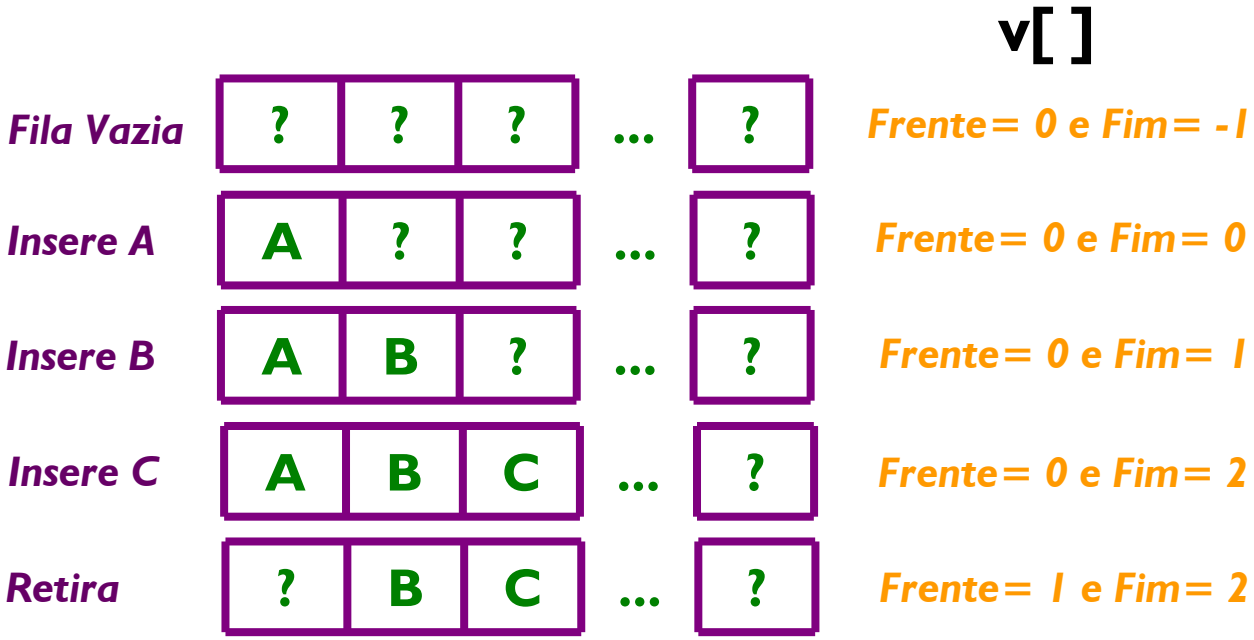
- Pode-se fazer uma analogia com uma fila de clientes em um banco qualquer:
  - o primeiro a entrar na fila, será o primeiro a sair e ser atendido (retiramos ele do início ou frente da fila)
  - o último cliente deve entrar no final da fila e, portanto, será o último a sair (ou ser atendido) na fila
- Portanto:
  - Não podemos retirar qualquer cliente da fila!
    - Apenas clientes que estão na frente da fila
  - Não podemos inserir um cliente na fila de tal forma que ele não seja o último no conjunto de clientes

Propriedades

- *Propriedades de uma fila:*
  - *Existem **n** elementos enfileirados;*
  - ***El** é o elemento no início (ou frente) da fila;*
  - ***En** é o último elemento da fila;*
- *A inserção de um primeiro elemento **El** em uma fila vazia, torna-o o primeiro a sair da estrutura*
- *Não se pode consultar qualquer elemento*
- *A inserção é sempre feita depois do elemento **En***
- *A retirada é sempre feita no elemento **El***

Tipos de filas

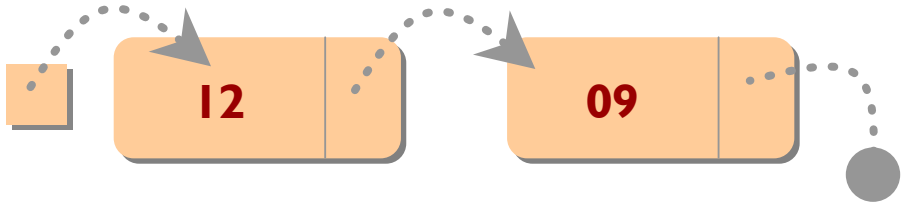
- *Neste estágio, estaremos trabalhando com três tipos de filas:* ***seqüenciais***, ***encadeadas*** e ***circulares***
  1. ***Seqüencial:*** *neste TAD, os elementos da fila são armazenados em endereços seqüenciais. Materializada na forma de um* ***vetor*** *(arranjo ou matriz).*



- *É preciso controlar o início e o final da fila. Note os campos frente e fim.*
- *Como poderíamos adaptar uma lista seqüencial à uma fila?*

Tipos de filas

- *(Continuação.)*
  - ❷ ***Encadeada:*** *neste TAD, os elementos da fila são elementos encadeados por* ***ponteiros***



  - ❸ ***Circular:*** *caso particular de filas. Iremos levantar e comentar a necessidade da implementação circular mais à frente*

- ***Importante:*** *são apenas diferentes maneiras de se implementar o TAD, porém todas as três formas seguem a mesma filosofia* ***F****irst* ***I****n* ***F****irst* ***O****ut*

Pense um pouco...

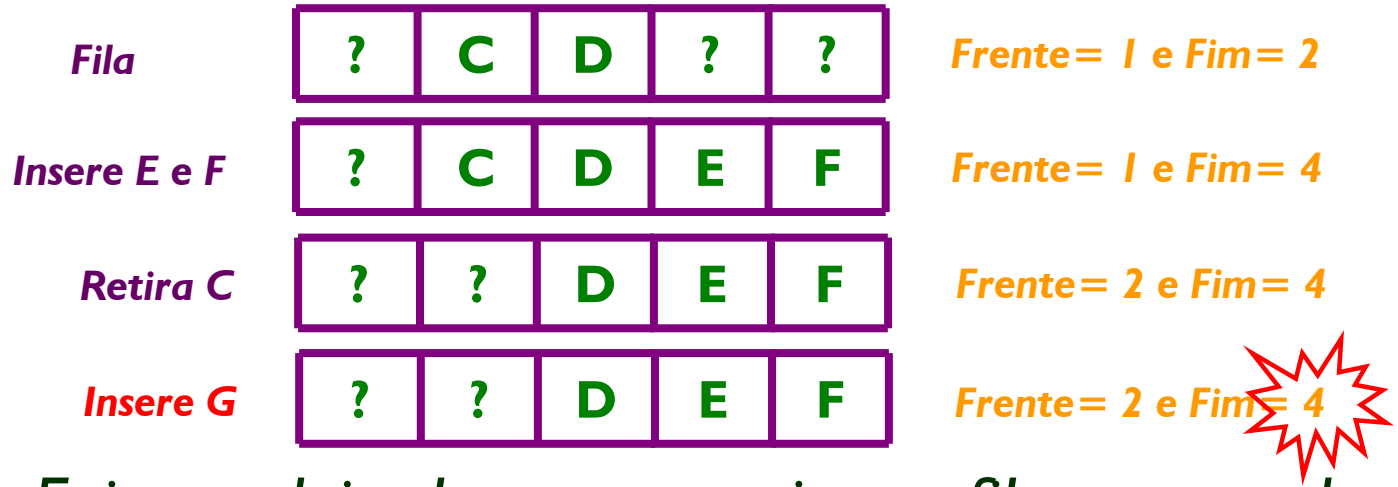
- *O que você acha que seria necessário para implementar uma biblioteca de um novo TAD que representasse uma fila seqüencial?*
  1. *um vetor de elementos (tamanho pré-definido)*
  2. *duas variáveis que controlem a frente e o fim da fila*

```
/* estruturação */
typedef struct fila {
   int elementos[MAX];
   int frente;
   int fim;
}TFila;
```

```
/* filaseq.h */
void criarFila(TFila *fila);
int filaVazia(TFila *fila);
int filaCheia(TFila *fila);
int enfileirar(TFila *fila, int dado);
int desenfileirar(TFila *fila, int *d);
void imprimir(TFila *fila);
```

Mas... há problemas!

- *Você deve ter percebido que a parte ocupada do vetor pode chegar à ultima posição*
  - *Ou seja, a fila seqüencial vai se deslocando da esquerda (frente) para à direita (fim) em cada retirada*



  - *Existem dois elementos vazios na fila, mas nenhum outro elemento pode mais ser inserido!*
    - **fila.fim** *chegou à condição de limite (MAX -1)*

Mas... há problemas!

- **E agora?**
  - *Do jeito que está, pode-se chegar a uma situação (absurda) em que a fila estará vazia, mas nenhum elemento novo poderá mais ser inserido!*
  - *Conclusão: a representação seqüencial, descrita anteriormente, parece não ser muito boa!*
- *Poderíamos mudar a função desenfileirar(…)*
  - *E fazer o deslocamento à esquerda de todos os elementos a cada retirada na frente da fila, mas...*
    - *...e se a fila tivesse 100 elementos?*
  - *Havíamos criticado o deslocamento da lista seqüencial!*

Solução

- *O que se quer é reaproveitar as primeiras posições livres do vetor sem implementar um re-arranjo (trabalhoso) de todos os elementos a cada retirada*
  - *Para isso, pode-se incrementar as posições do vetor de maneira "circular"*
  - *Ou seja, se o último elemento da fila ocupar a última posição do vetor, inserimos os novos elementos a partir do início do vetor!*
- ***Graficamente:***

Filas encadeadas

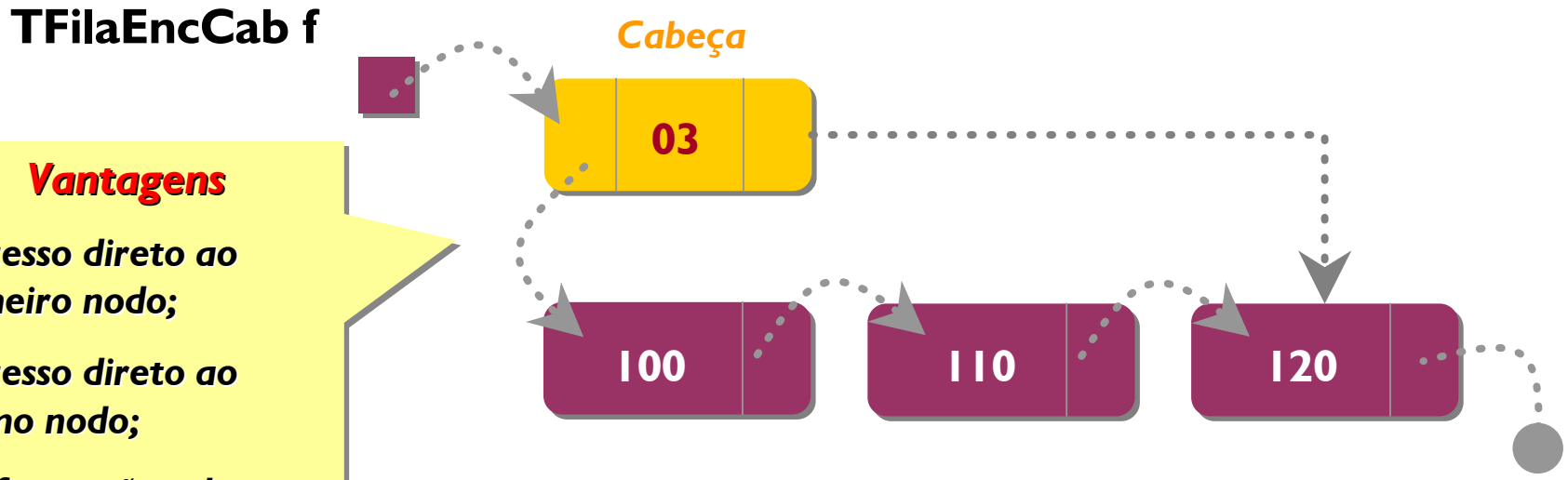
- *O que fazer quando o número máximo de elementos na fila não é conhecido?*
  - *Devemos implementar a fila usando uma estrutura de dados dinâmica (com **alocação dinâmica**)*
  - *Podemos empregar os conceitos vistos nas listas simplesmente (ou duplamente) encadeadas*
- *Entretanto, as regras de inserção/remoção, agora, irão mudar!*
  - *Algumas funções da lista encadeada (inserir ou remover mediante uma posição qualquer, entre outras) deverão ser eliminadas neste novo TAD*

Filas encadeadas - modus operandi

- O primeiro elemento (início) da lista encadeada poderá representar a frente atual da fila
  - Assim, cada novo elemento será inserido no final da lista. Conseqüentemente e, sempre que solicitado, retiramos o elemento a partir do início da lista
- Desta forma, vamos precisar de apenas duas funções auxiliares na lista:
  1. uma para inserir no fim (enfileirar)
  2. outra para remover do início (desenfileirar)
- Há apenas uma desvantagem:
  - Para cada elemento a ser inserido na fila, teremos que percorrer todos os nodos até encontrar o último.

Fila encadeada com nó descritor

- *Vamos utilizar os conhecimentos auferidos em técnicas de encadeamento!*
- *Neste caso em particular, iremos alocar um nó "cabeça" para que possamos ter acesso direto ao final da fila (promover performance)*



Vantagens

- ***Acesso direto ao primeiro nodo;***
- ***Acesso direto ao último nodo;***
- ***Informação sobre o tamanho da lista.***

Algoritmos em C

- *O que deverá ser feito pelo aluno:*
  - *Escolha e instalação do ambiente a ser trabalhado no laboratório*
  - *Modelagem deste TAD (dados e operações)*
  - *Implementação dos algoritmos de operações básicas vistos em sala de aula na linguagem C*
  - *Utilização das regras de modelagem vistas no módulo anterior (criação de bibliotecas) e modularização*
  - *Implantação de código legível e bem documentado*
  - *Nomes de variáveis condizentes com o problema*
  - *Prática de laboratório*

Para um bom aproveitamento:

- *O aluno deve identificar a relação entre TAD (biblioteca e modularização) com a implementação da fila no código!*
- *Resolva todas as questões da* ***prática de laboratório de filas circular e encadeada***
- *Procure o professor ou monitor da disciplina e questione conceitos, listas, etc.*
- *Não deixe para codificar tudo e acumular assunto para a primeira avaliação.*
  - *Este é apenas um dos assuntos abordados na prova!*