UTFPR
UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
Câmpus Campo Mourão

Ciência da Computação

Algoritmos e Estrutura de Dados 1

Fila

Uma Lista Linear com restrições

Prof. Rafael Liberato
liberato@utfpr.edu.br

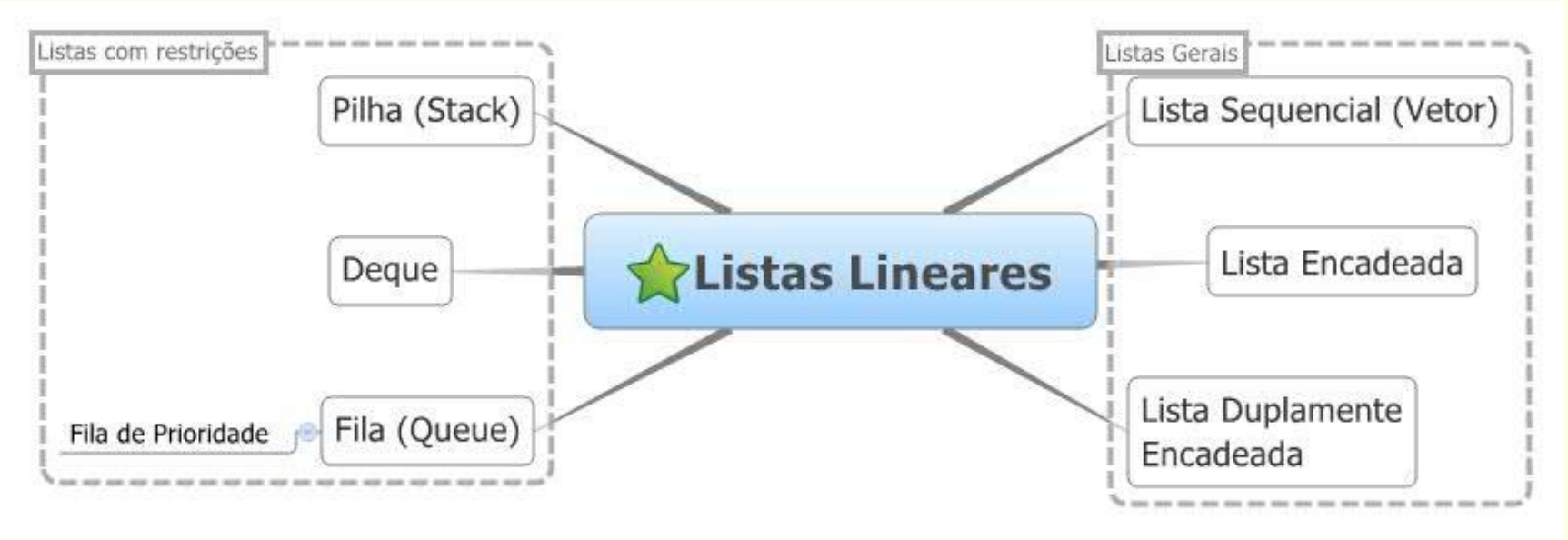
Objetivos

- Definir o TAD Fila
  - Entender a estrutura de dados utilizada na fila, tanto estática quanto dinâmica.
  - Entender qual é a responsabilidade de cada operação, independente da estrutura utilizada.

Roteiro

- **Conceito**
- **Definindo uma Fila**
- **TAD Fila (Dados/Operações)**
  - Definindo a estrutura que armazenará os **dados**
  - Transformando **operações** em Funções
- **Descrição dos dados**
- **Descrição das operações da Fila**

Relembrando...



Fila- Conceito

Conceito

- **Fila ou Queue é uma lista linear com restrições.**
  - A inserção é realizada no final
  - A remoção é realizada no início

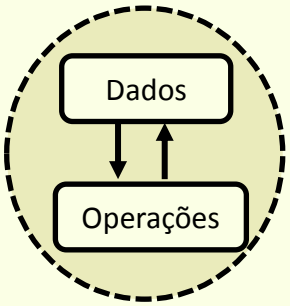
- **Política FIFO (First-In-First-Out)**
  - O primeiro que entra é o primeiro que sai

Aplicação

- **Aplicações**
  - Filas de espera para diversos domínios
  - Compartilhamento e distribuição de recursos
  - Escalonadores
- **Aplicações Indiretas**
  - Estruturas de dados auxiliares para algoritmos
  - Componentes para outras estruturas de dados

Definindo uma FILA

- **Agora que conhecemos um pouco mais sobre a FILA, vamos especificar o que necessitamos que elas façam**
- **Para isso, precisamos:**
  1. definir o que vamos armazenar na Fila
  2. definir quais operações serão realizadas na Filla



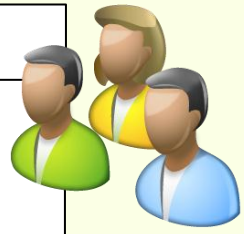
- **Quando levantamos essas informações, nós definimos o TAD**
  - **Neste momento, não vamos nos preocupar em COMO implementar, e sim em O QUE precisamos**

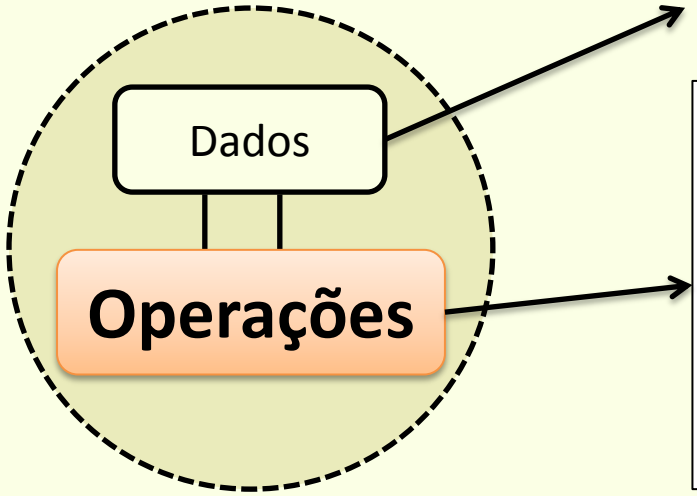
Definindo uma FILA

- **Vamos especificar o Tipo Abstrato de Dados chamado Fila para definir o que precisamos**

1. Quais dados serão manipulados?
2. Quais operações serão disponibilizadas para manipular os dados? O que precisamos?

Aluno
- ra: int - nome: String - email: String - notas: float[ ]



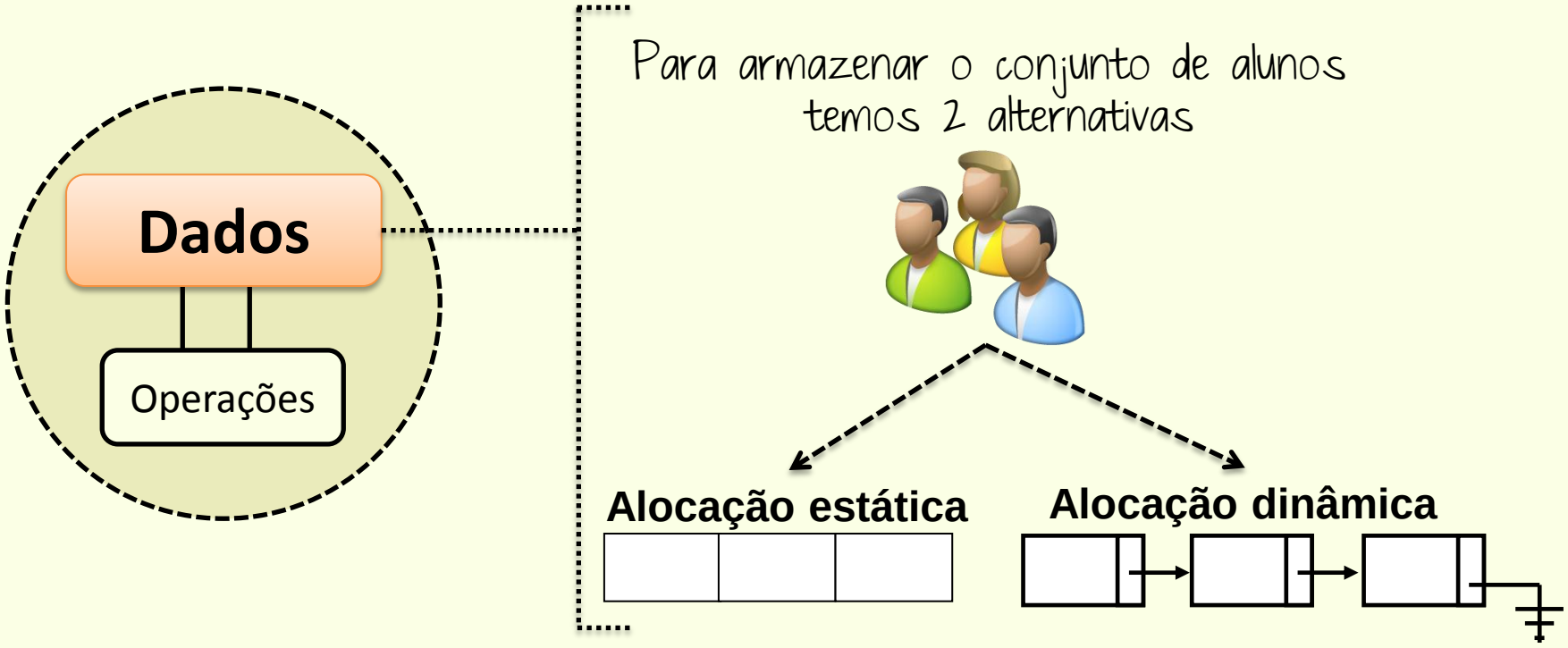


Conjunto de Alunos

- Criar a fila
- Inserir um aluno na fila;
- Remover um aluno da fila;
- Recuperar informações do aluno que está no inicio da pilha;
- Verificar se um aluno está na fila;
- Verificar quantos alunos existem na fila;
- Verificar se a fila está vazia;
- Ver todo o conteúdo da fila.

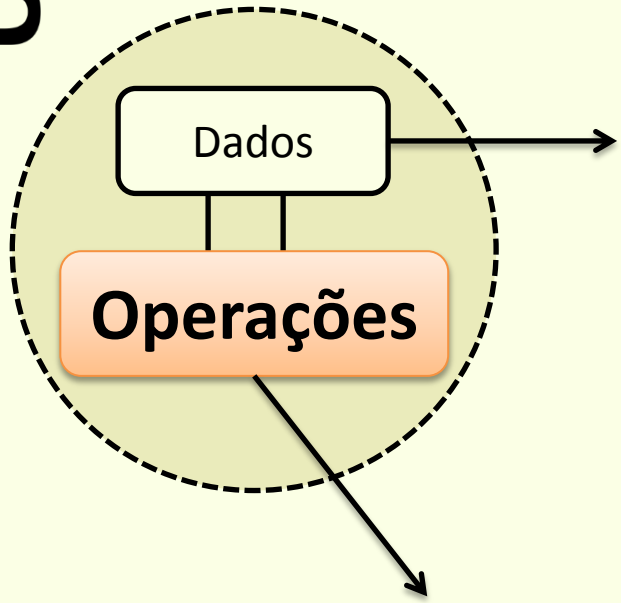
TAD FILA

- **E quanto a estrutura que armazenará os dados?**
  - Ela será definida quando o TAD for materializado (implementado) por uma Estrutura de Dados



TAD Fila

TAD FILA



Queue será o nome da `struct` que organizará os dados da Fila, independentemente da estratégia de alocação de memória utilizada.

```
Queue *createQueue ();                    // Criar a fila
void initializeQueue(Queue *q);           // Inicializa a fila
int enqueue(Queue * q, ItemType e);       // Inserir um elemento na fila;
int dequeue(Queue* q, ItemType* e);       // Remover um elemento da fila
int peek(Queue* q, ItemType* e);          // Recuperar informações do primeiro da  fila
int contains(Queue* q, ItemType *e);      // Verificar se um elemento  está na fila
int sizeQueue(Queue* q);                  // Verificar quantos elementos  existem na fila
int isEmptyQueue(Queue* q);               // Verificar se a fila está vazia
void printQueue(Queue* q);                // Ver todo o conteúdo da fila.
```

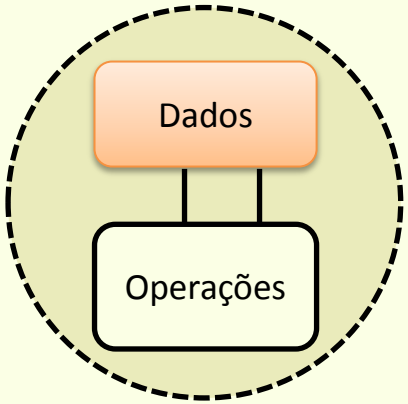
Descrição dos dados

Estrutura de dados FILA

- Baseado nessas informações, vamos implementar a Estrutura de dados **Fila** nas versões estática e dinâmica

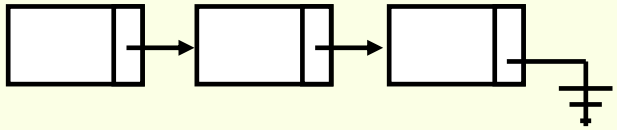
Fila Estática

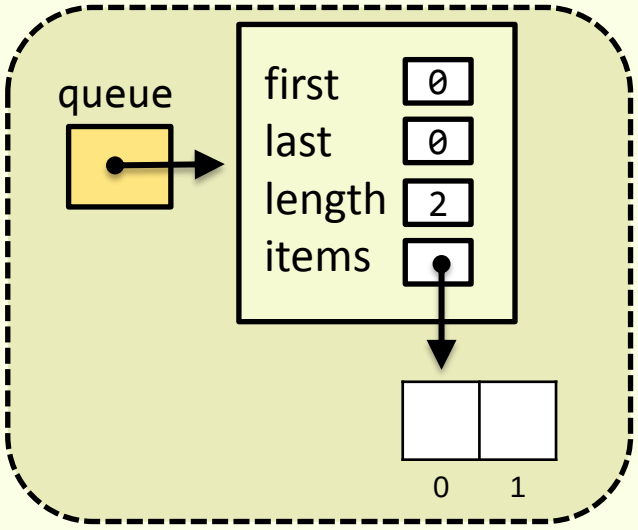
Utilizando alocação estática

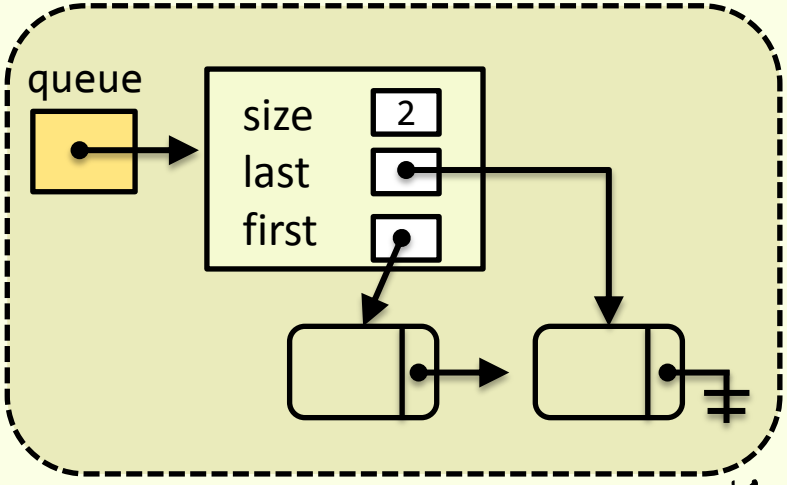


Fila Dinâmica

Utilizando alocação dinâmica

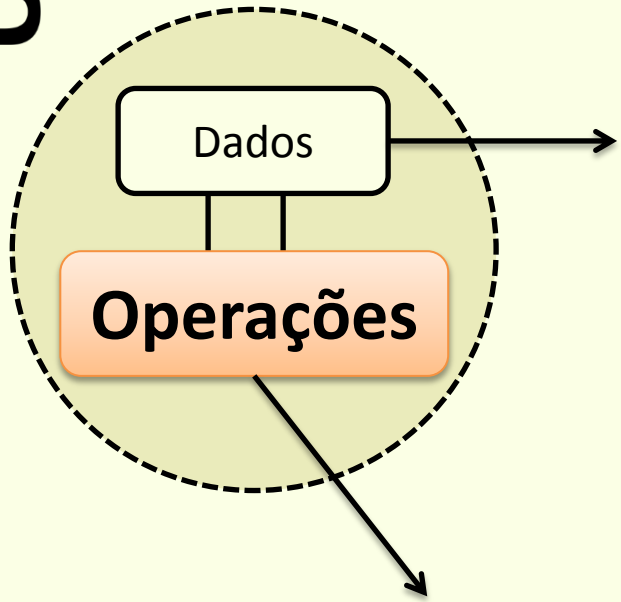






Descrição das operações

TAD FILA



Queue será o nome da `struct` que organizará os dados da Fila, independentemente da estratégia de alocação de memória utilizada.

```
Queue *createQueue ();                    // Criar a fila
void initializeQueue(Queue *q);           // Inicializa a fila
int enqueue(Queue * q, ItemType e);       // Inserir um elemento na fila;
int dequeue(Queue* q, ItemType* e);       // Remover um elemento da fila
int peek(Queue* q, ItemType* e);          // Recuperar informações do primeiro da  fila
int contains(Queue* q, ItemType *e);      // Verificar se um elemento  está na fila
int sizeQueue(Queue* q);                  // Verificar quantos elementos  existem na fila
int isEmptyQueue(Queue* q);               // Verificar se a fila está vazia
void printQueue(Queue* q);                // Ver todo o conteúdo da fila.
```

Descrição das Operações

```
Queue*createQueue();
```

Aloca dinamicamente uma Fila, inicializa-a e retorna seu endereço.

Parâmetros:

Nenhum

Retorno:

O endereço de memória da Fila criada e inicializada.

Descrição das Operações

```
void initializeQueue(Queue *q);
```

Inicializa uma especificada Fila.

Parâmetros:

- **Queue *q**: endereco da Fila a ser inicializada

Retorno:

nenhum

Descrição das Operações

```
int enqueue(Queue * q, ItemType e);
```

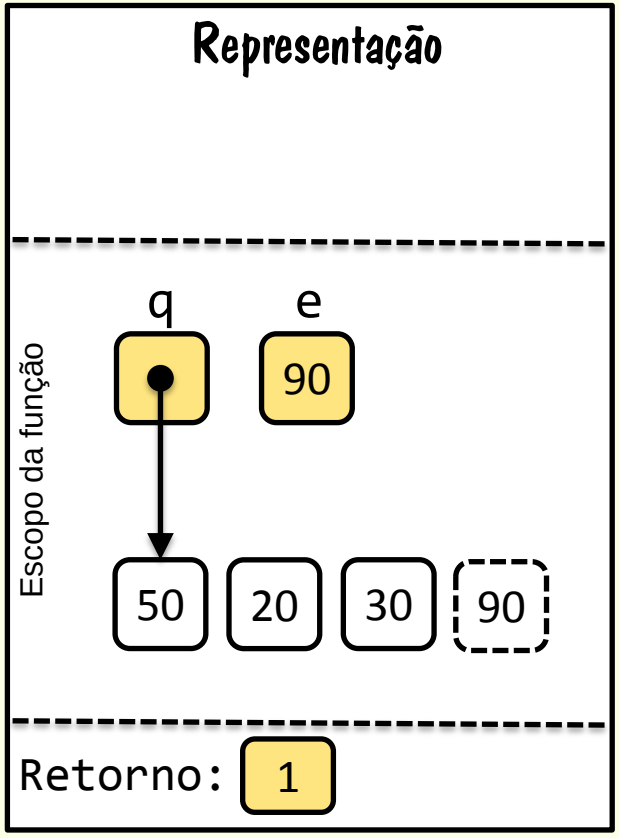
Insere o especificado elemento na Fila

Parâmetros:

- **Queue *q:** endereço da Fila a ser manipulada.
- **ItemType e:** elemento a ser inserido.

Retorno:

- **true**: inserção realizada
- **false**: inserção **não** realizada



Descrição das Operações

```
int dequeue(Queue* q, ItemType* e);
```

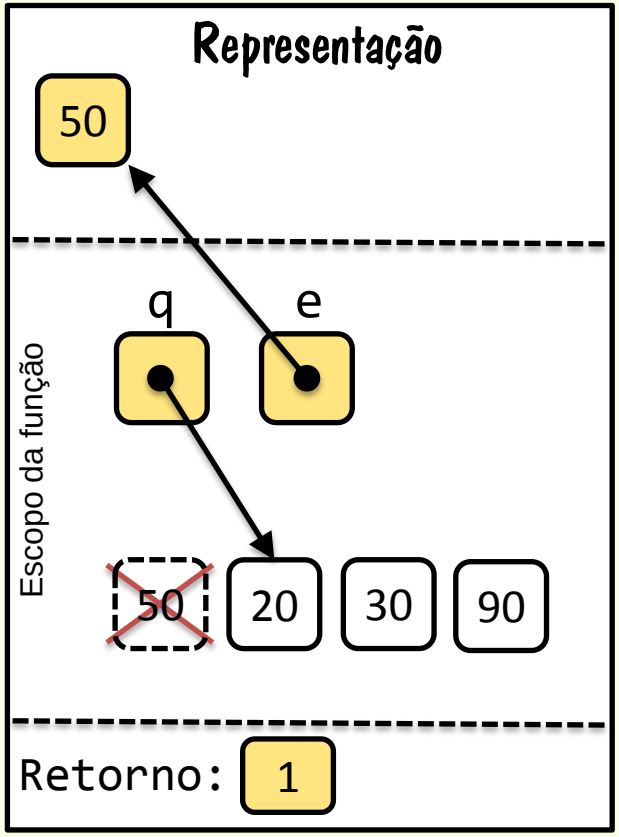
Remove o primeiro elemento da Fila. A função copia o elemento removido para o endereço **e**

Parâmetros:

- **Queue *q:** endereço da Fila a ser manipulada
- **ItemType *e:** endereço utilizado para o armazenamento do elemento removido

Retorno:

- **true**: remoção realizada
- **false**: remoção **não** realizada



Descrição das Operações

```
int peek(Queue* q, ItemType* e);
```

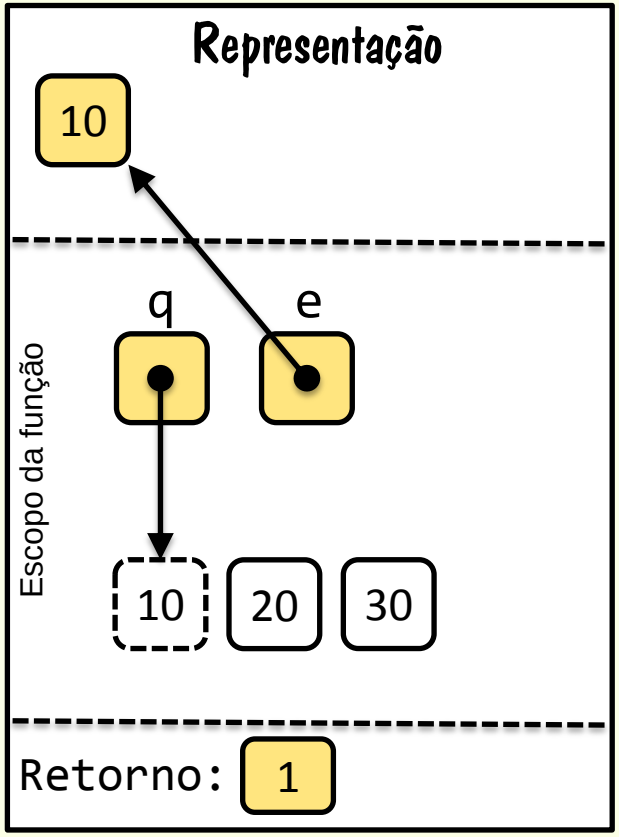
Recupera um elemento da lista de uma especificada posição.

Parâmetros:

- **Queue *q:** endereço da Fila a ser manipulada.
- **ItemType *e:** endereço utilizado para armazenar o primeiro elemento da Fila.

Retorno:

- **true:** o elemento foi armazenado com sucesso.
- **false:** o elemento **não** foi recuperado.



Descrição das Operações

```
int containsQueue(Queue* q, ItemType *e);
```

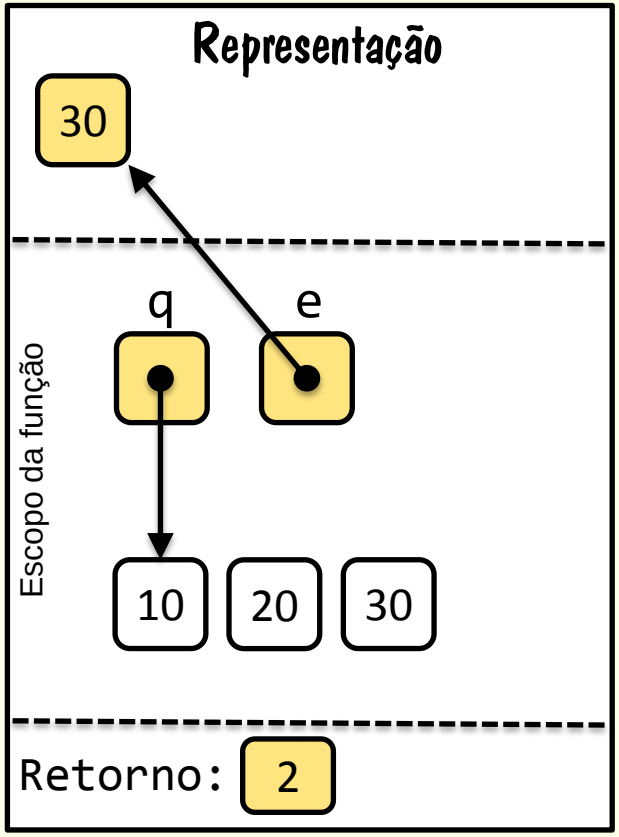
Verifica se o especificado elemento está contido na Fila.

Parâmetros:

- **Queue *q:** endereço da Fila a ser manipulada.
- **ItemType *e:** endereço que contém o elemento desejado

Retorno:

- **true:** se o elemento está contido na Fila.
- **false:** se o elemento **não** está contido.



Descrição das Operações

```
int sizeQueue(Queue* q);
```

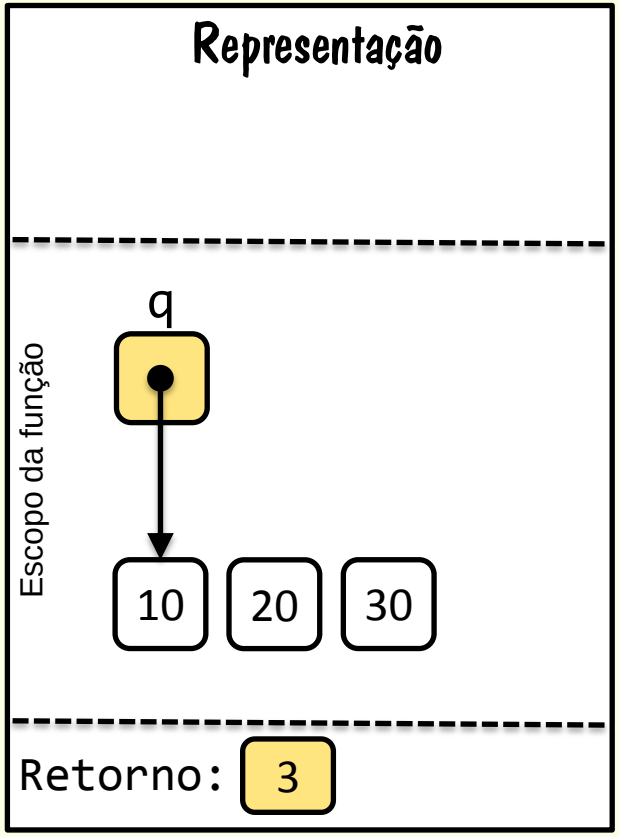
Retorna a quantidade de elementos da Fila

Parâmetros:

- **Queue *q:** endereço da Fila a ser manipulada.

Retorno:

- **Número de elementos da Fila**



Descrição das Operações

```
int isEmptyQueue(Queue* q);
```

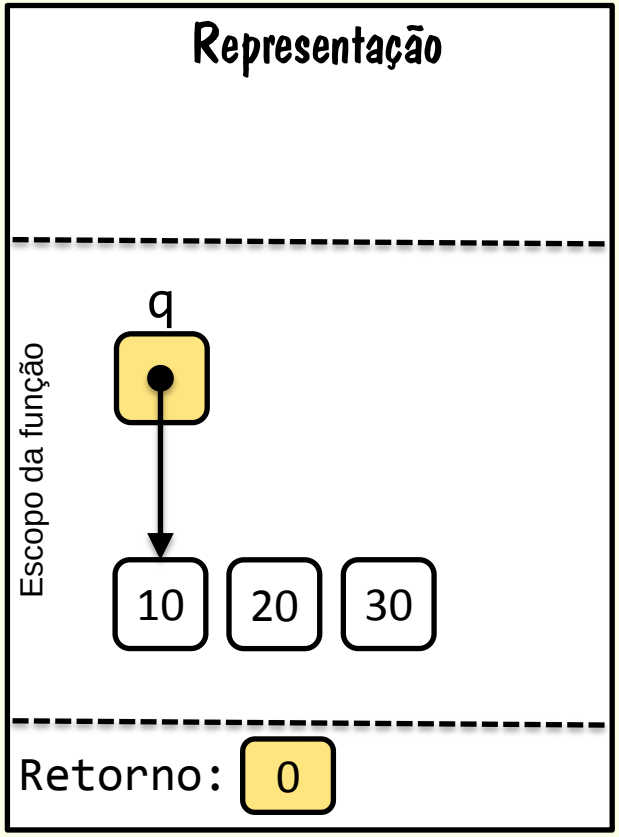
Verifica se a lista está vazia.

Parâmetros:

- **Queue *q:** endereço da Fila a ser manipulada.

Retorno:

- **true:** a Fila está vazia
- **false:** a Fila **não** está vazia.



Descrição das Operações

```
void printQueue(Queue* q);
```

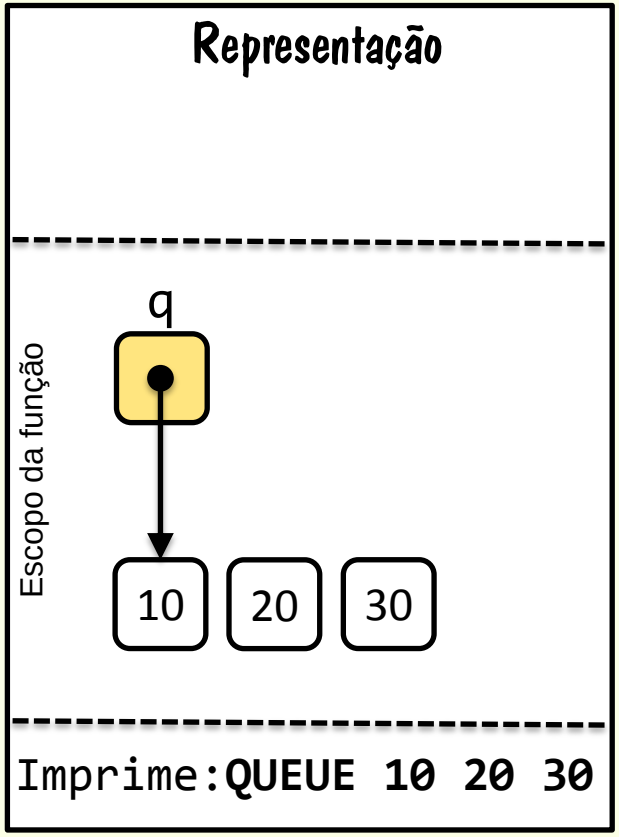
Imprime todos os elementos da Fila.

Parâmetros:

- **Queue *q:** endereço da Fila a ser manipulada.

Retorno:

nenhum



Referências

- http://www.ime.usp.br/~pf/algoritmos/aulas/fila.html