UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

Câmpus Campo Mourão

Ciência da Computação

Algoritmos e Estrutura de Dados 1

Fila com alocação estática

Prof. Rafael Liberato

liberato@utfpr.edu.br

Objetivos

- Entender o funcionamento de uma Fila Estática
- Ser capaz de implementar as operações definidas no TAD Lista manipulando uma estrutura estática de armazenamento.

Roteiro

- TAD Fila
- Fila Estática
- Simulação
- Implementação

TAD Fila

TAD Fila

```
#define ItemType int

typedef struct{

}Queue;

Queue *createQueue ();
void initializeQueue(Queue *q);
int enqueue(Queue * q, ItemType e);
int dequeue(Queue* q, ItemType* e);
int peek(Queue* q, ItemType* e);
int contains(Queue* q, ItemType *e);
int sizeQueue(Queue* q);
int isEmptyQueue(Queue* q);
void printQueue(Queue* q);
```

Vamos identificar os atributos que representarão a Fila estática

Estrutura utilizada para armazenar os dados

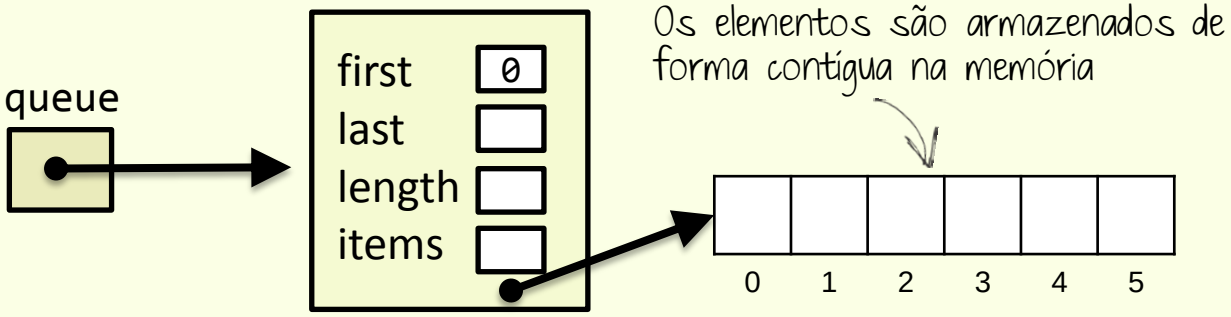
Fila Estática

Fila Estática

- A **Fila Estática** utiliza uma estrutura de alocação estática de memória para o armazenamento dos dados
- A linguagem nos fornece essa estrutura por meio dos arranjos unidimensionais (vetores)
  - ➔ Os elementos da Fila são armazenados em um vetor
- Para manter a eficiência da remoção na **Fila Estática** vamos utilizar uma variável para representar o primeiro elemento.
  - ➔ Dessa forma, para remover um elemento basta incrementar essa variável.

Fila Estática

- **Atributos**
  - O atributo `items` armazena o endereço do array
  - O atributo `length` armazena a quantidade de espaços do array
  - O atributo `first` armazena a posição do primeiro elemento da Fila
  - O atributo `last` armazena a posição da primeira posição vazia da Fila. (last-1 é a posição do último elemento)




```
typedef struct{
    int first;
    int last;
    int length;
    ItemType *items;
}Queue;
```

Simulação

Simulação

- Utilize a simulação para entender o comportamento das funções e auxiliá-lo na implementação.

Simulação

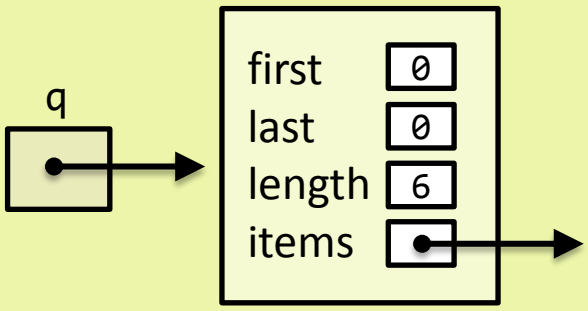
```
Queue *q = createQueue();
enqueue(q,10);
enqueue(q,20);
enqueue(q,30);
ItemType removed;
dequeue(q, &removed);
dequeue(q, &removed);
dequeue(q, &removed);
```

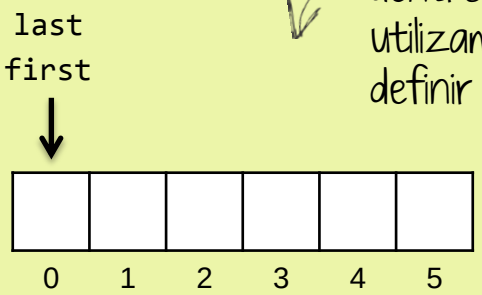
```
enqueue(q,40);
enqueue(q,50);
enqueue(q,60);
enqueue(q,70);
```

Simulação

```
Queue *q = createQueue();
enqueue(q,10);
enqueue(q,20);
enqueue(q,30);
ItemType removed;
dequeue(q, &removed);
dequeue(q, &removed);
dequeue(q, &removed);
```

```
enqueue(q,40);
enqueue(q,50);
enqueue(q,60);
enqueue(q,70);
```





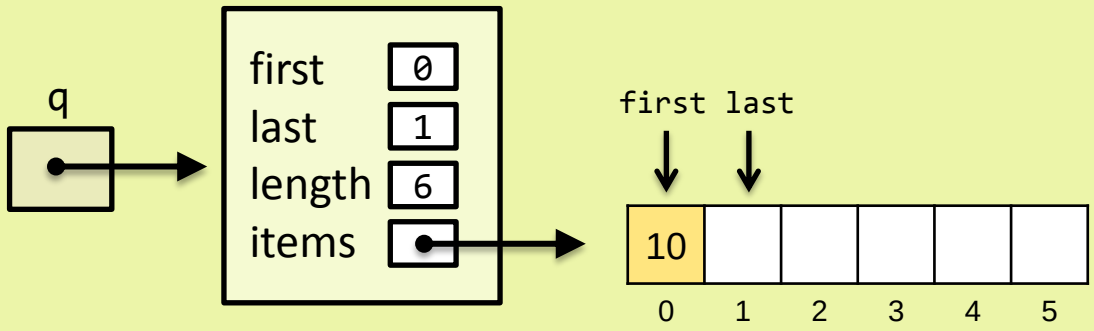
O tamanho do vetor é definido dentro da função. Normalmente utilizamos uma constante para definir o tamanho do vetor

FILA VAZIA

Simulação

```
Queue *q = createQueue();
enqueue(q,10);
enqueue(q,20);
enqueue(q,30);
ItemType removed;
dequeue(q, &removed);
dequeue(q, &removed);
dequeue(q, &removed);
```

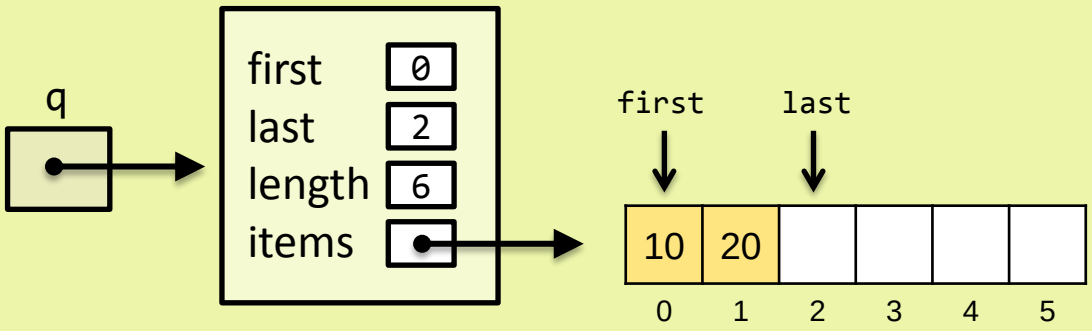
```
enqueue(q,40);
enqueue(q,50);
enqueue(q,60);
enqueue(q,70);
```



Simulação

```
Queue *q = createQueue();
enqueue(q,10);
enqueue(q,20);
enqueue(q,30);
ItemType removed;
dequeue(q, &removed);
dequeue(q, &removed);
dequeue(q, &removed);
```

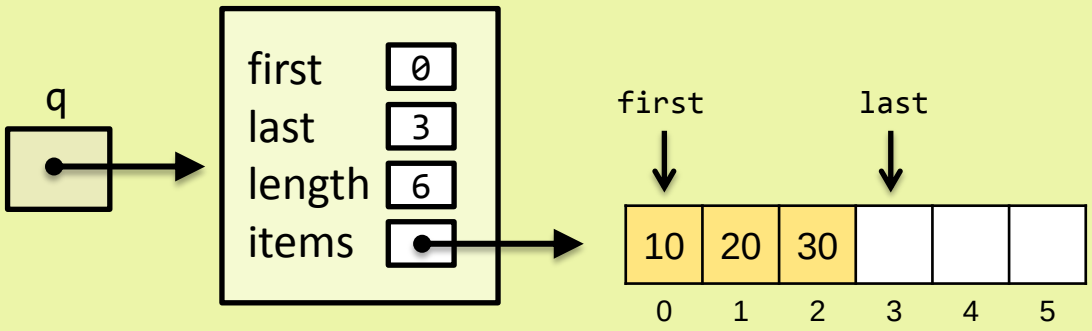
```
enqueue(q,40);
enqueue(q,50);
enqueue(q,60);
enqueue(q,70);
```



Simulação

```
Queue *q = createQueue();
enqueue(q,10);
enqueue(q,20);
enqueue(q,30);
ItemType removed;
dequeue(q, &removed);
dequeue(q, &removed);
dequeue(q, &removed);
```

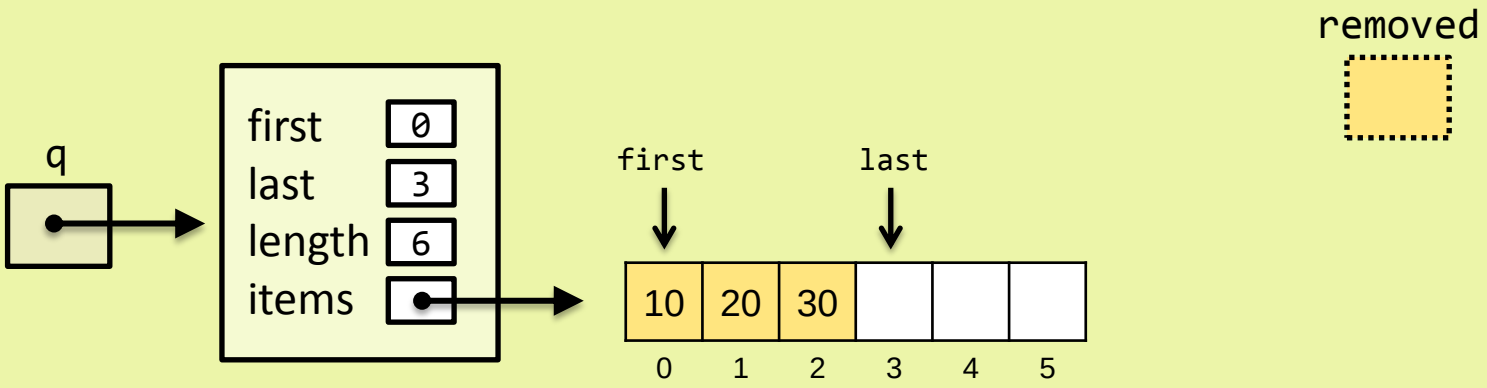
```
enqueue(q,40);
enqueue(q,50);
enqueue(q,60);
enqueue(q,70);
```



Simulação

```
Queue *q = createQueue();
enqueue(q,10);
enqueue(q,20);
enqueue(q,30);
ItemType removed;
dequeue(q, &removed);
dequeue(q, &removed);
dequeue(q, &removed);
```

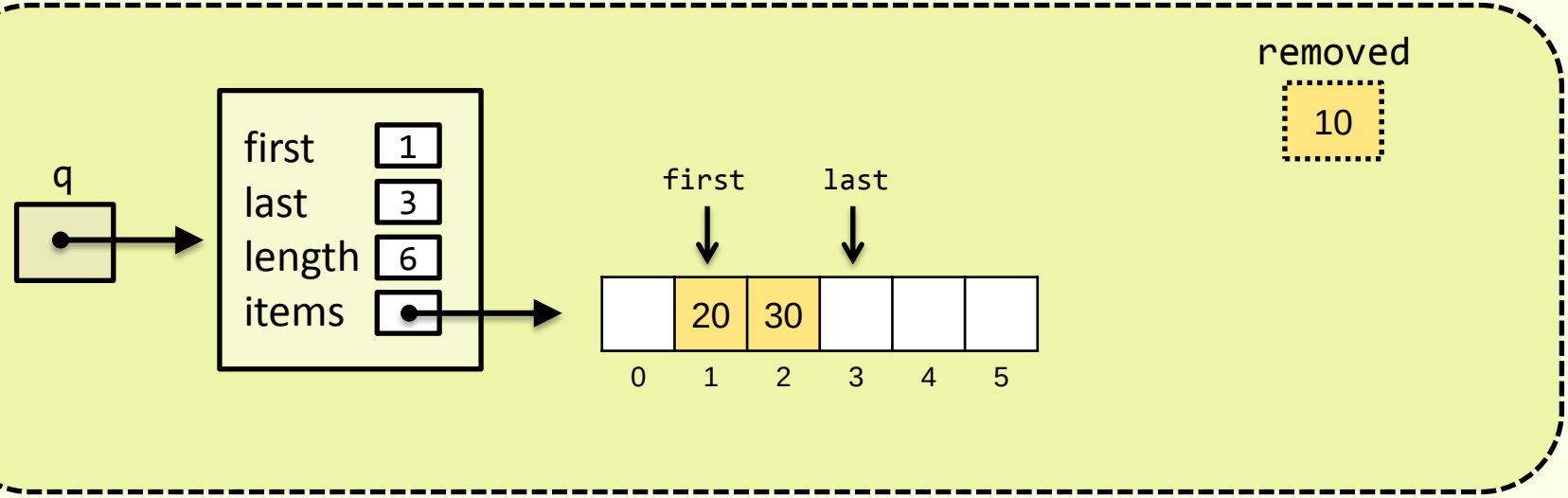
```
enqueue(q,40);
enqueue(q,50);
enqueue(q,60);
enqueue(q,70);
```



Simulação

```
Queue *q = createQueue();
enqueue(q,10);
enqueue(q,20);
enqueue(q,30);
ItemType removed;
dequeue(q, &removed);
dequeue(q, &removed);
dequeue(q, &removed);
```

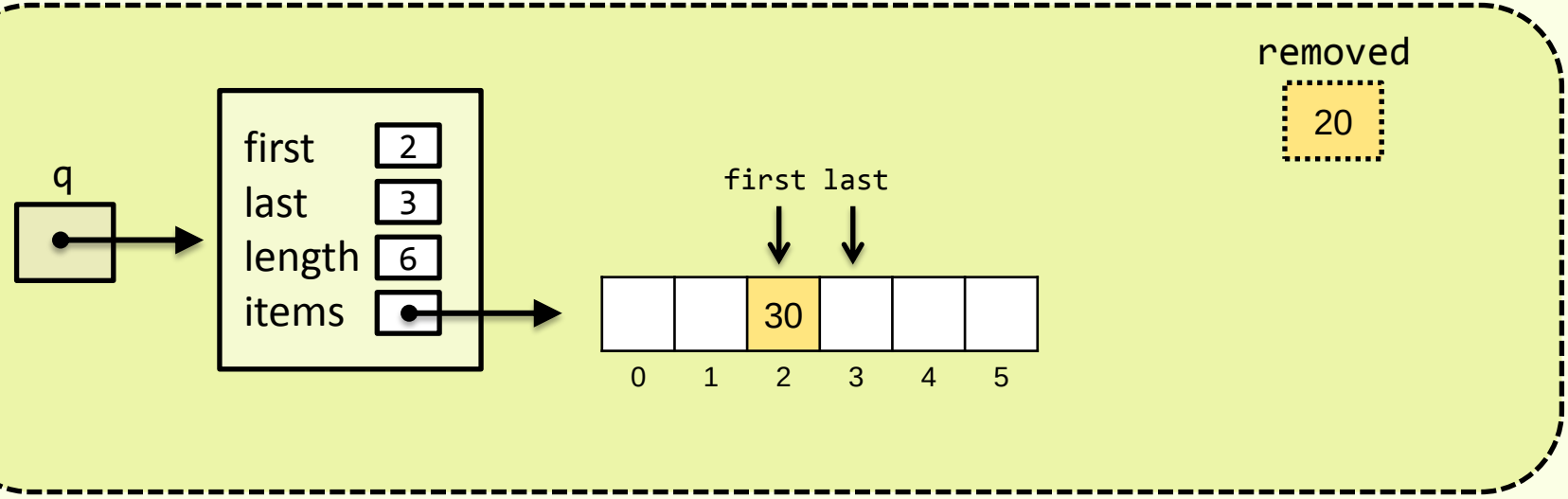
```
enqueue(q,40);
enqueue(q,50);
enqueue(q,60);
enqueue(q,70);
```



Simulação

```
Queue *q = createQueue();
enqueue(q,10);
enqueue(q,20);
enqueue(q,30);
ItemType removed;
dequeue(q, &removed);
dequeue(q, &removed);
dequeue(q, &removed);
```

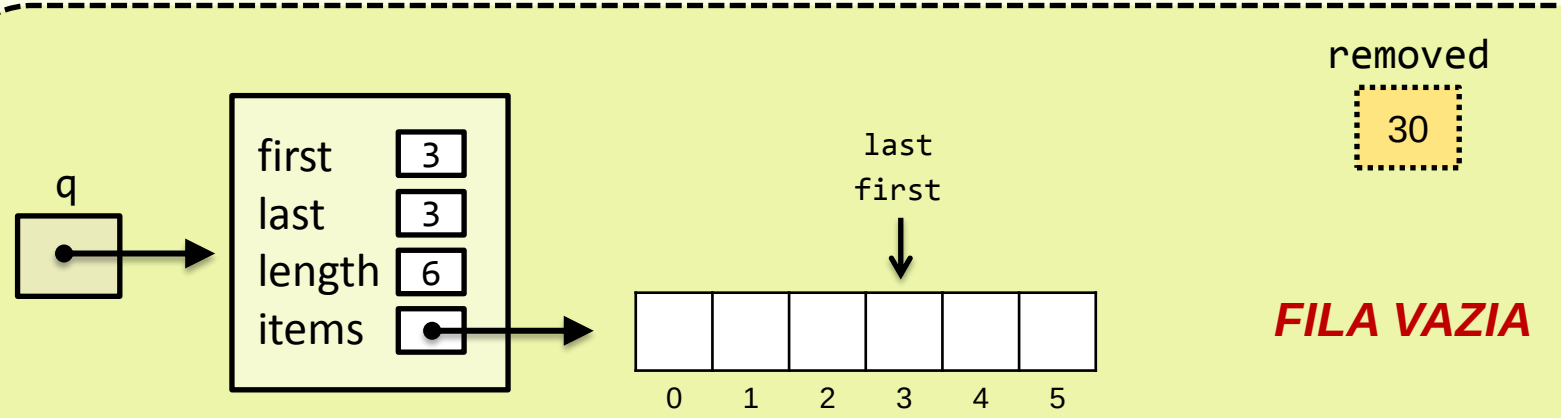
```
enqueue(q,40);
enqueue(q,50);
enqueue(q,60);
enqueue(q,70);
```



Simulação

```
Queue *q = createQueue();
enqueue(q,10);
enqueue(q,20);
enqueue(q,30);
ItemType removed;
dequeue(q, &removed);
dequeue(q, &removed);
dequeue(q, &removed);
```

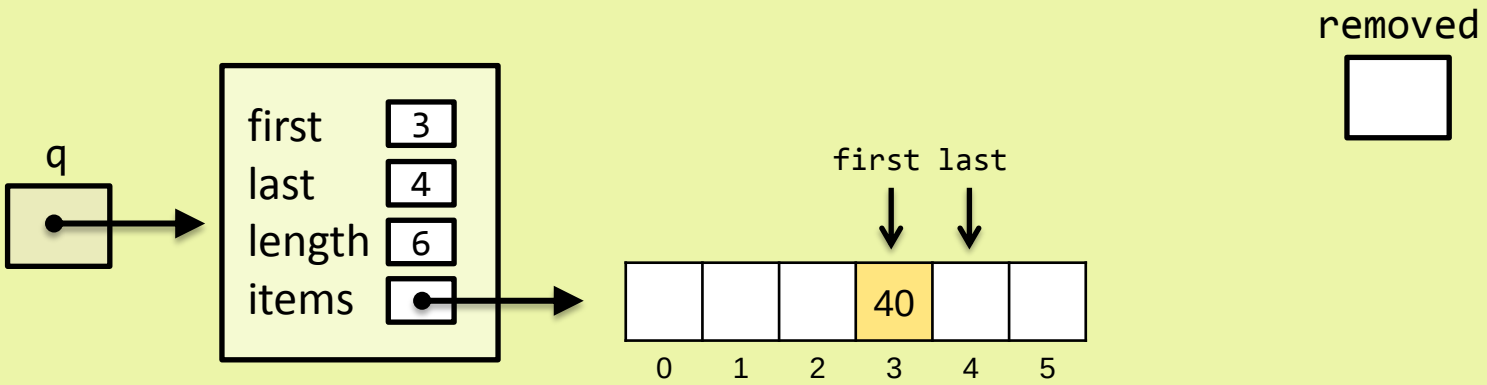
```
enqueue(q,40);
enqueue(q,50);
enqueue(q,60);
enqueue(q,70);
```



Simulação

```
Queue *q = createQueue();
enqueue(q,10);
enqueue(q,20);
enqueue(q,30);
ItemType removed;
dequeue(q, &removed);
dequeue(q, &removed);
dequeue(q, &removed);
```

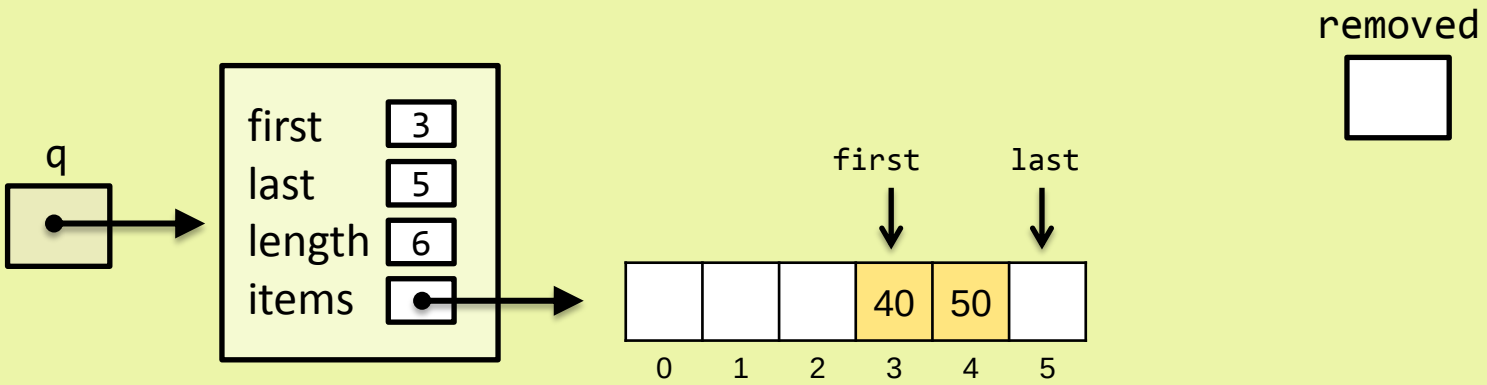
```
enqueue(q,40);
enqueue(q,50);
enqueue(q,60);
enqueue(q,70);
```



Simulação

```
Queue *q = createQueue();
enqueue(q,10);
enqueue(q,20);
enqueue(q,30);
ItemType removed;
dequeue(q, &removed);
dequeue(q, &removed);
dequeue(q, &removed);
```

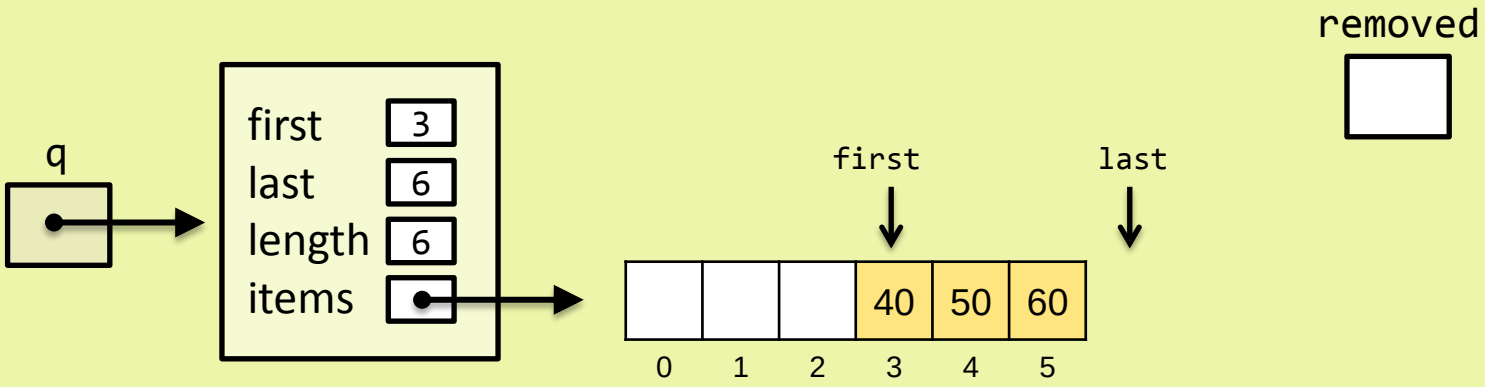
```
enqueue(q,40);
enqueue(q,50);
enqueue(q,60);
enqueue(q,70);
```



Simulação

```
Queue *q = createQueue();
enqueue(q,10);
enqueue(q,20);
enqueue(q,30);
ItemType removed;
dequeue(q, &removed);
dequeue(q, &removed);
dequeue(q, &removed);
```

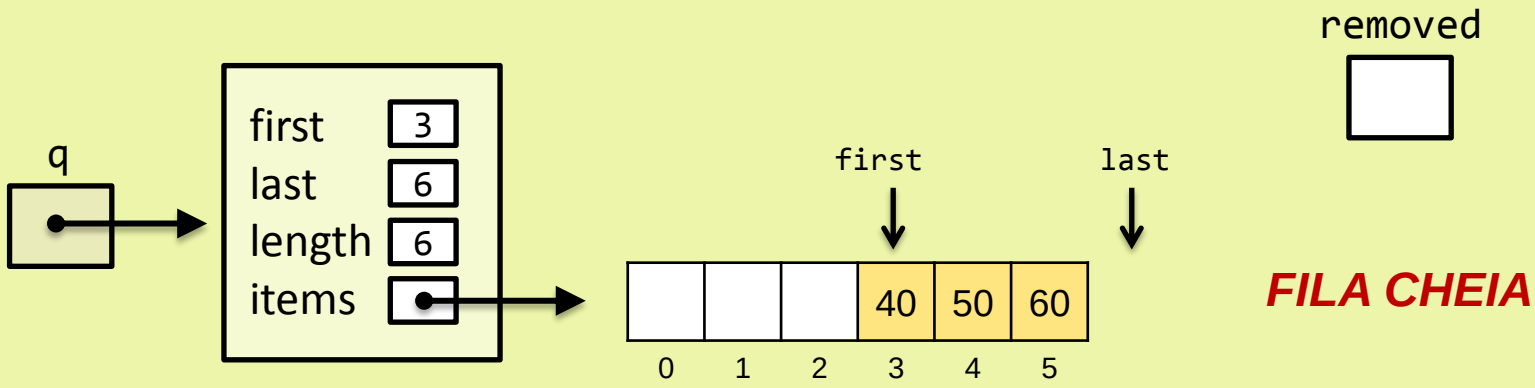
```
enqueue(q,40);
enqueue(q,50);
enqueue(q,60);
enqueue(q,70);
```



Simulação

```
Queue *q = createQueue();
enqueue(q,10);
enqueue(q,20);
enqueue(q,30);
ItemType removed;
dequeue(q, &removed);
dequeue(q, &removed);
dequeue(q, &removed);
```

```
enqueue(q,40);
enqueue(q,50);
enqueue(q,60);
enqueue(q,70);
```

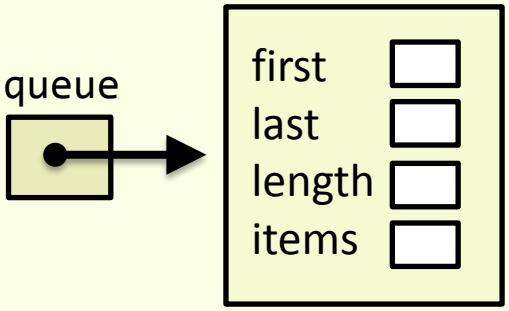


Implementação

Implementação

- A partir dessa simulação é possível extrair o comportamento das funções sobre os atributos da **Fila Estática**

```
Queue *createQueue ();
void initializeQueue(Queue *q);
int enqueue(Queue * q, ItemType e);
int dequeue(Queue* q, ItemType* e);
int peek(Queue* q, ItemType* e);
int contains(Queue* q, ItemType *e);
int sizeQueue(Queue* q);
int isEmptyQueue(Queue* q);
void printQueue(Queue* q);
```




```
typedef struct{
    int first;
    int last;
    int length;
    ItemType *items;
}Queue;
```

Implementação

LET'S DO IT

Referências