Filas Dinâmicas em C

INF0286 | INF0447 – Algoritmos e Estruturas de Dados I

Prof. Me. Raphael Guedes

raphaelguedes@ufg.br

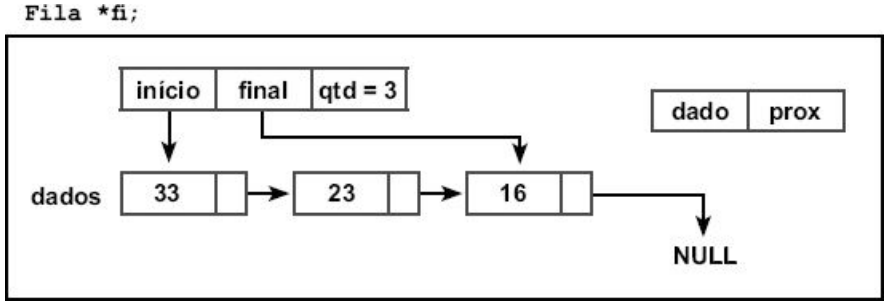
2024

INF
INSTITUTO DE INFORMÁTICA

UFG
UNIVERSIDADE FEDERAL DE GOIÁS

Filas Dinâmicas

- Uma fila dinâmica encadeada é uma fila definida utilizando alocação dinâmica e acesso encadeado dos elementos.
- Cada elemento da fila é alocado dinamicamente:
  - sempre que novos dados são inseridos na fila;
  - e tem sua memória liberada sempre que são removidos.
- O **elemento** é um ponteiro para uma estrutura contendo dois campos de informação:
  - um campo **dado**, para armazenar a informação inserida na fila;
  - um campo **prox**, o qual é um ponteiro que aponta para o próximo elemento na fila.



Filas Dinâmicas

- A implementação de uma fila dinâmica encadeada é **praticamente** igual à implementação de uma lista dinâmica encadeada.
- A diferença é que uma fila possui uma regra (política) para inserção e outra para remoção (*first-in, first-out*).

Filas Dinâmicas

- A fila apresentada neste slide, além da estrutura que define seus elementos, utiliza um nó descritor para guardar o início, o final e a quantidade de elementos (dados) inseridos na fila.

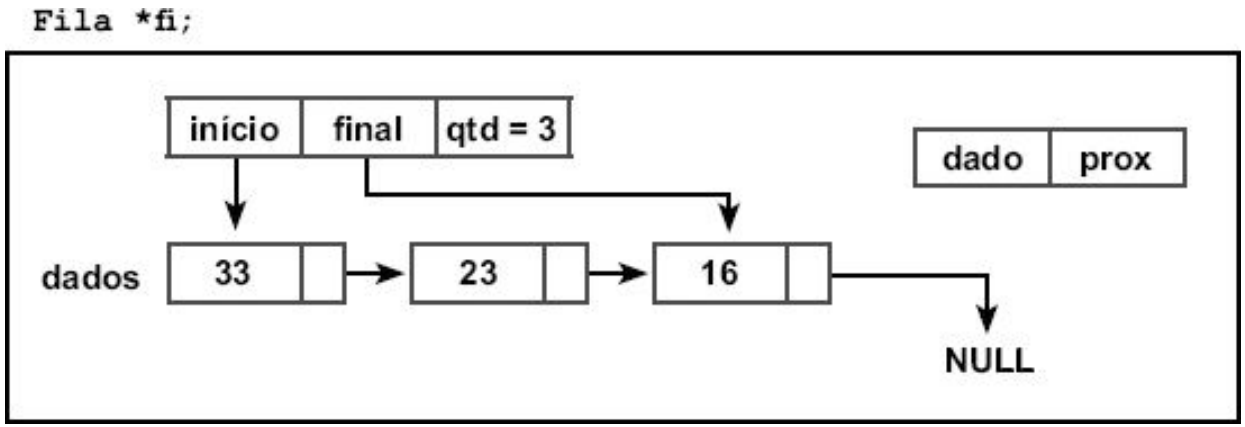
- Um **nó descritor** é um elemento especial da fila.
  - No nó descritor podemos armazenar qualquer informação que julgarmos necessária sobre a fila.

Filas Dinâmicas

- A principal vantagem de se utilizar uma abordagem dinâmica e encadeada na definição da fila é a melhor utilização dos recursos de memória.
- Não é preciso definir previamente o tamanho da fila.

- A principal desvantagem é a necessidade de percorrer toda a fila para destruí-la.

Filas Dinâmicas: estrutura com nó descritor



Filas Dinâmicas: estrutura com nó descritor

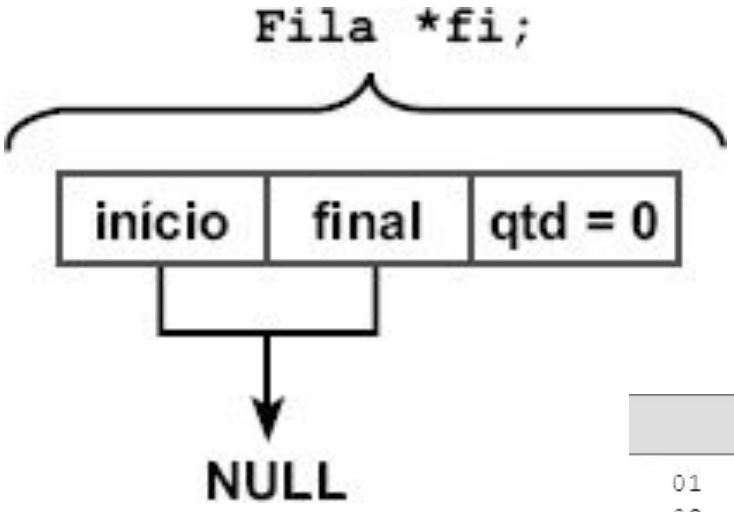
Arquivo FilaDin.h

```
01  struct aluno{
02      int matricula;
03      char nome[30];
04      float n1,n2,n3;
05  };
06  typedef struct fila Fila;
07
08  Fila* cria_Fila();
09  void libera_Fila(Fila* fi);
10  int consulta_Fila(Fila* fi, struct aluno *al);
11  int insere_Fila(Fila* fi, struct aluno al);
12  int remove_Fila(Fila* fi);
13  int tamanho_Fila(Fila* fi);
14  int Fila_vazia(Fila* fi);
15  int Fila_cheia(Fila* fi);
```

Arquivo FilaDin.c

```
01  #include <stdio.h>
02  #include <stdlib.h>
03  #include "FilaDin.h" //inclui os protótipos
04  //definição do tipo Fila
05  struct elemento{
06      struct aluno dados;
07      struct elemento *prox;
08  };
09  typedef struct elemento Elem;
10  //definição do nó descritor da fila
11  struct fila{
12      struct elemento *inicio;
13      struct elemento *final;
14      int qtd;
15  };
```

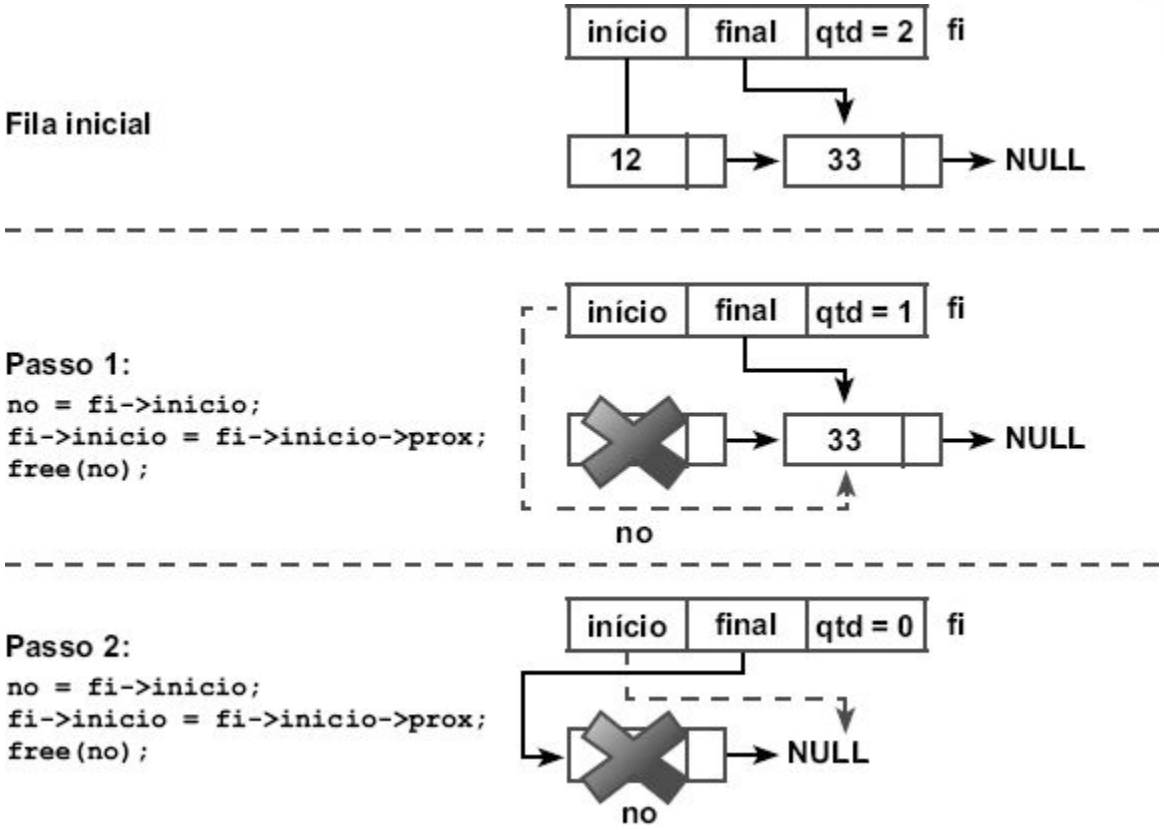
Filas Dinâmicas: criação



Criando uma fila

```
01    Fila* cria_Fila(){
02        Fila* fi = (Fila*) malloc(sizeof(struct fila));
03        if(fi != NULL){
04            fi->final = NULL;
05            fi->inicio = NULL;
06            fi->qtd = 0;
07        }
08        return fi;
09    }
```

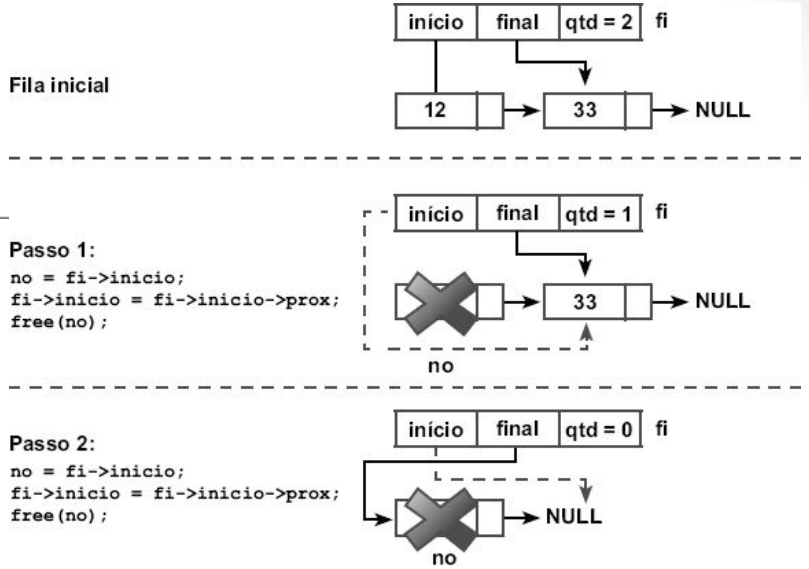
Filas Dinâmicas: destruição



Filas Dinâmicas: destruição

Destruindo uma fila

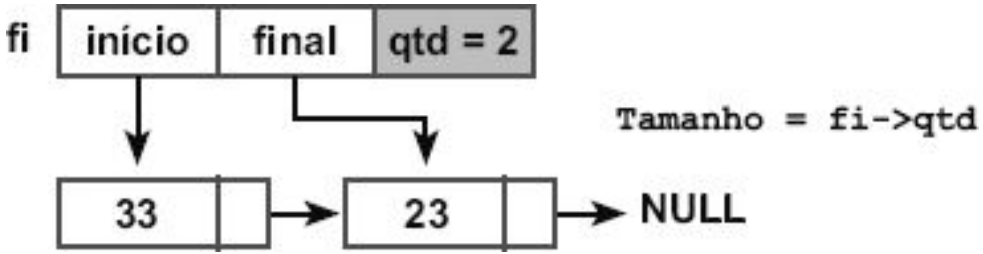
```
01    void libera_Fila(Fila* fi){
02        if(fi != NULL){
03            Elem* no;
04            while(fi->inicio != NULL){
05                no = fi->inicio;
06                fi->inicio = fi->inicio->prox;
07                free(no);
08            }
09            free(fi);
10        }
11    }
```



Filas Dinâmicas: tamanho

Tamanho da fila

```
01    int tamanho_Fila(Fila* fi){
02        if(fi == NULL)
03            return 0;
04        return fi->qtd;
05    }
```



Filas Dinâmicas: fila cheia

Não teríamos uma abordagem melhor?

Relembrar da fila estática

E se tentarmos alocar um nó, resolve?

Retornando se a fila está cheia

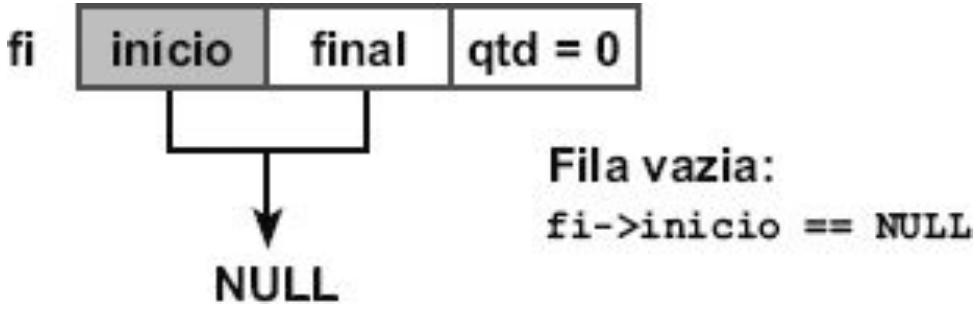
```
01    int Fila_cheia(Fila* fi){
02        return 0;
03    }
```

Filas Dinâmicas: fila vazia

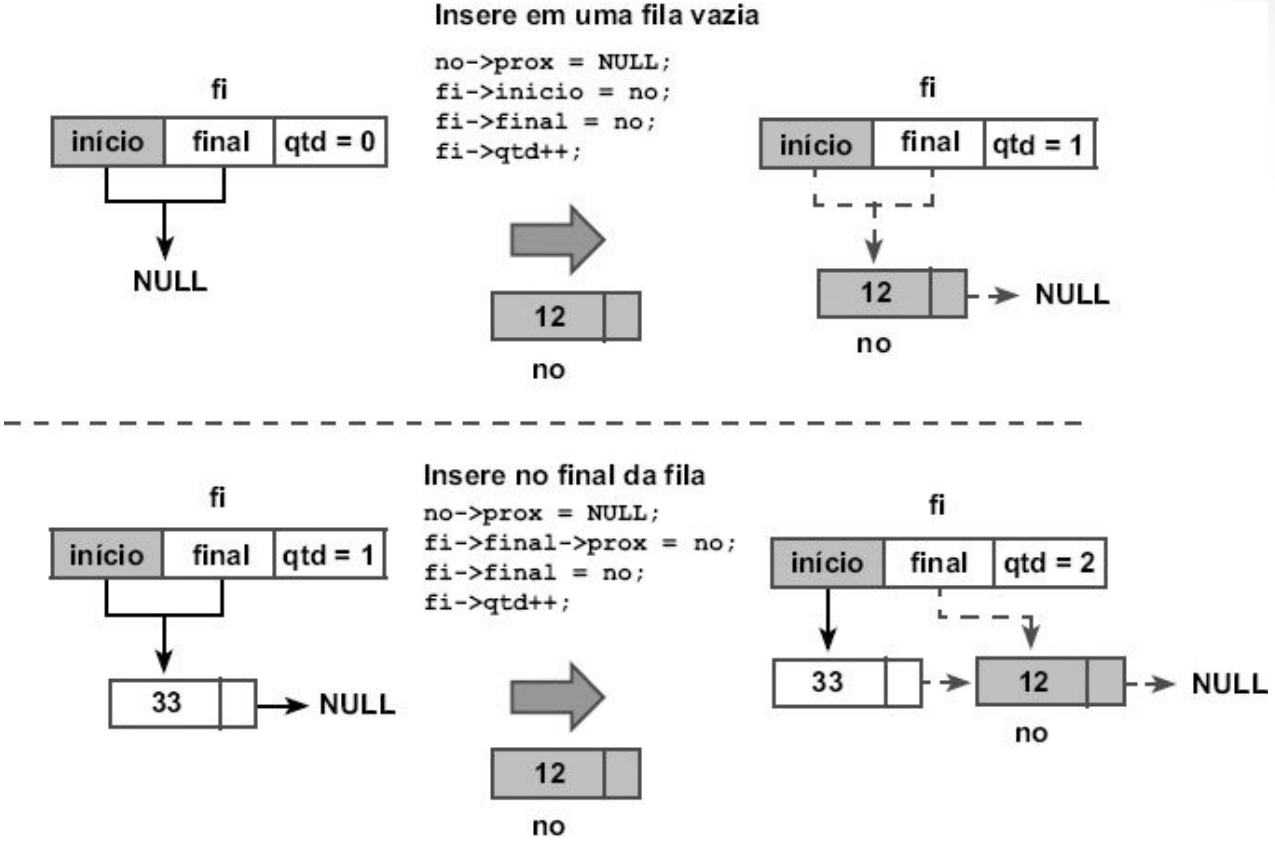
Retornando se a fila está vazia

```
01    int Fila_vazia(Fila* fi){
02        if(fi == NULL)
03            return -1;
04        if(fi->inicio == NULL)
05            return 1;
06        return 0;
07    }
```

E se o final também apontar para NULL?



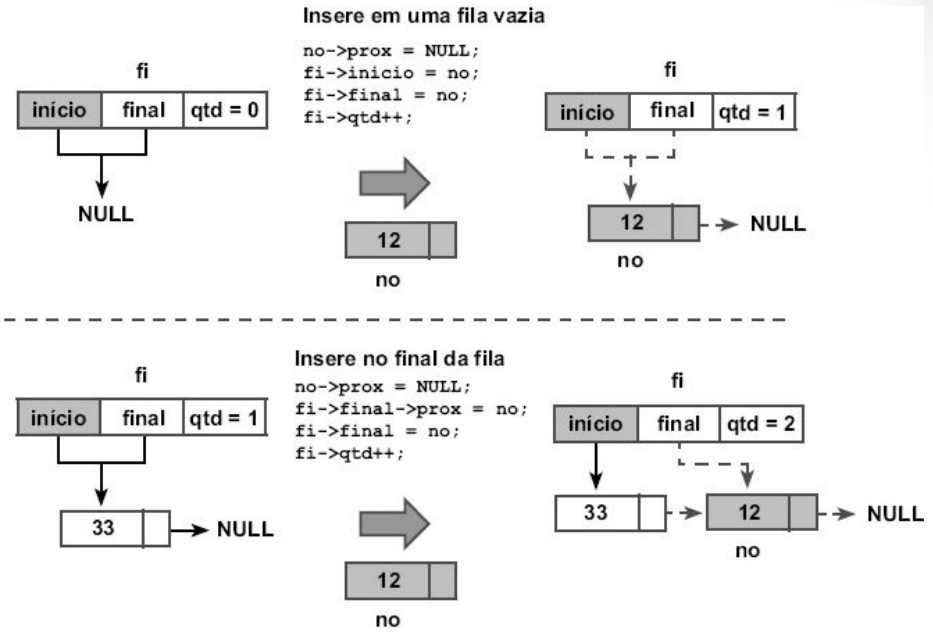
Filas Dinâmicas: inserção (sempre no fim)



Filas Dinâmicas: inserção (sempre no fim)

Inserindo um elemento na fila

```
01  int insere_Fila(Fila* fi, struct aluno al){
02      if(fi == NULL)
03          return 0;
04      Elem *no = (Elem*) malloc(sizeof(Elem));
05      if(no == NULL)
06          return 0;
07      no->dados = al;
08      no->prox = NULL;
09      if(fi->final == NULL)//fila vazia
10          fi->inicio = no;
11      else
12          fi->final->prox = no;
13      fi->final = no;
14      fi->qtd++;
15      return 1;
16  }
```



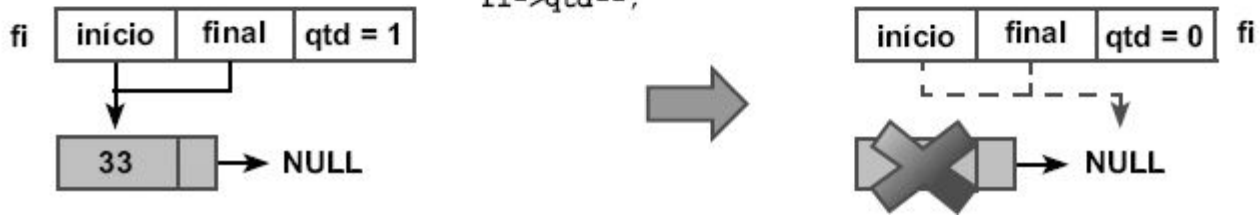
Filas Dinâmicas: remoção (sempre no início)

Remove do final da fila:

```
fi->inicio = fi->inicio->prox;
free(no);
fi->qtd--;
```

Remove e a fila fica vazia:

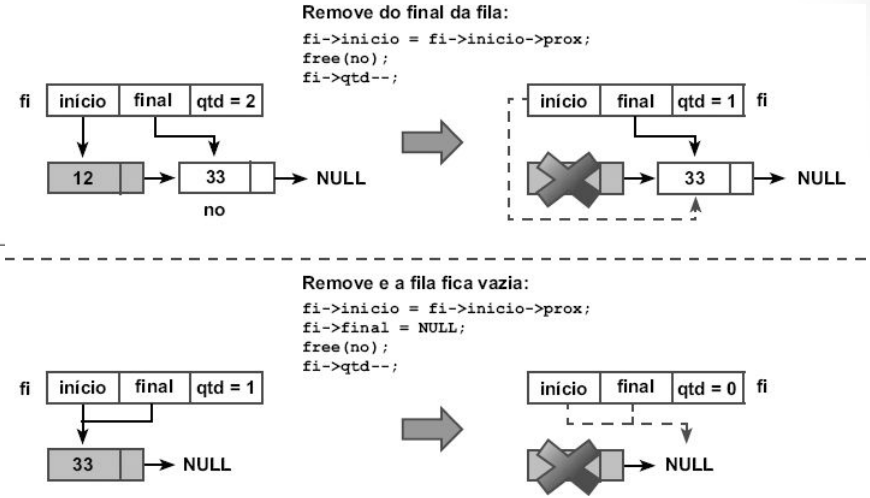
```
fi->inicio = fi->inicio->prox;
fi->final = NULL;
free(no);
fi->qtd--;
```



Filas Dinâmicas: remoção (sempre no início)

Removendo um elemento da fila

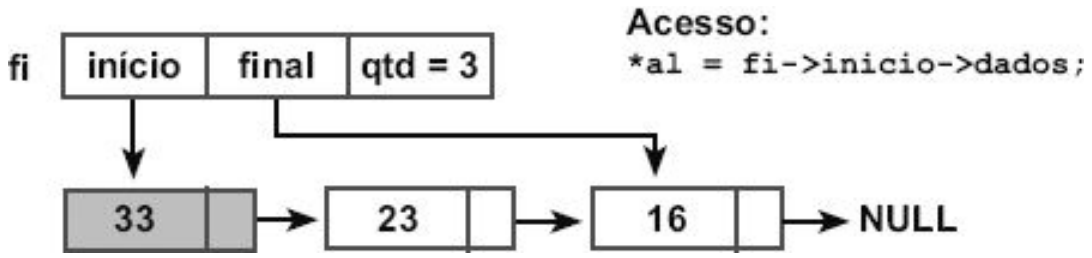
```
01    int remove_Fila(Fila* fi){
02        if(fi == NULL)
03            return 0;
04        if(fi->inicio == NULL)//fila vazia
05            return 0;
06        Elem *no = fi->inicio;
07        fi->inicio = fi->inicio->prox;
08        free(no);
09        if(fi->inicio == NULL)//fila ficou vazia
10            fi->final = NULL;
11        fi->qtd--;
12        return 1;
13    }
```



Filas Dinâmicas: consulta (sempre no início)

Consultando a fila

```
01    int consulta_Fila(Fila* fi, struct aluno *al){
02        if(fi == NULL)
03            return 0;
04        if(fi->inicio == NULL)//fila vazia
05            return 0;
06        *al = fi->inicio->dados;
07        return 1;
08    }
```



Referências

- BACKES, André Ricardo. **Algoritmos e Estruturas de Dados em C**. Rio de Janeiro: LTC, 2023.

Obrigado!

raphaelguedes@ufg.br
raphaelguedes@inf.ufg.br

INF
INSTITUTO DE INFORMÁTICA

UFG
UNIVERSIDADE FEDERAL DE GOIÁS