Estrutura de Dados

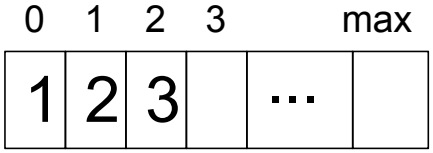
Estrutura Fila

(Implementação Estática/Sequencial)

Prof. Luiz Gustavo Almeida Martins

Introdução

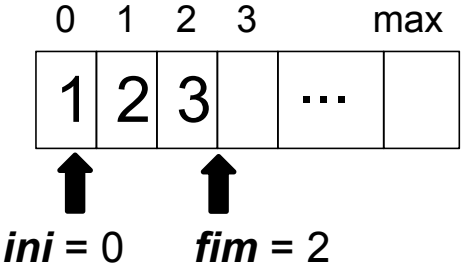
- Aloca posição para todos os elementos quando a estrutura é criada
  - Utiliza um **vetor com *max* posições**



Introdução

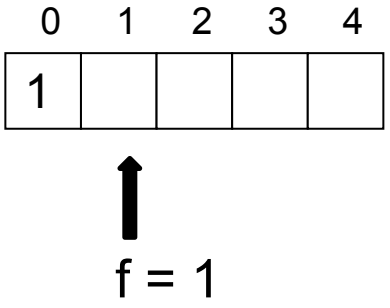
- Aloca posição para todos os elementos quando a estrutura é criada
  - Utiliza um **vetor com *max* posições**

- Eficiência depende do acesso rápido às extremidades da fila
  - Inserção precisa conhecer o **final da fila**
  - Remoção precisa conhecer o **início da fila**



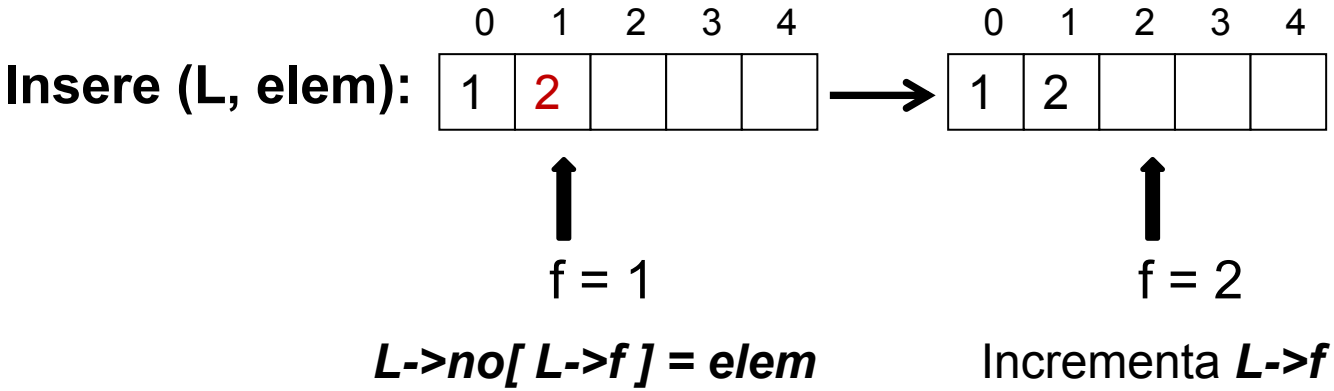
Formas de Implementação

- **Ideia 1: utilizar uma lista linear simples**
  - **Início fixo** na posição ZERO (dispensa campo)



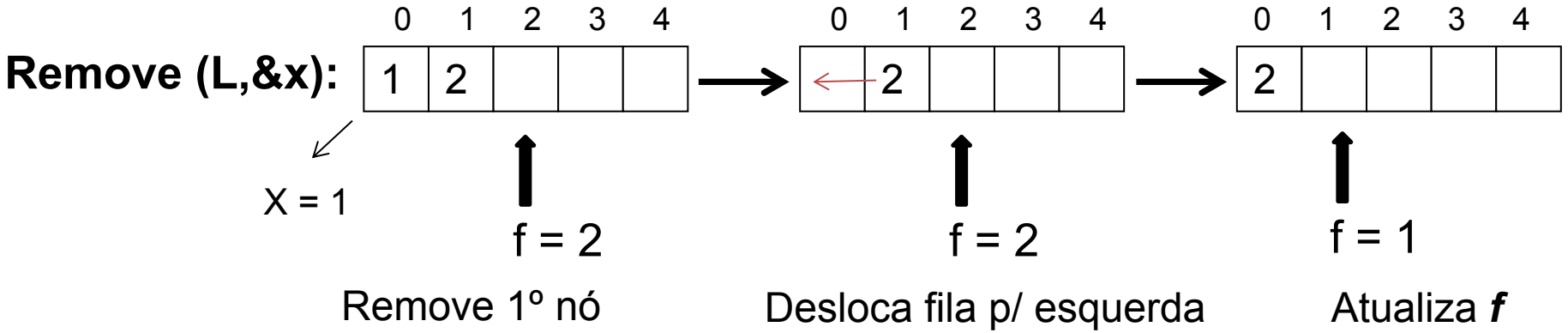
Formas de Implementação

- **Ideia 1: utilizar uma lista linear simples**
  - **Início fixo** na posição ZERO (dispensa campo)
  - **Inserir no final** é uma **operação simples**



Formas de Implementação

- **Ideia 1: utilizar uma lista linear simples**
  - **Início fixo** na posição ZERO (dispensa campo)
  - **Inserir no final** é uma **operação simples**
  - **Problema:** remoção envolve **deslocamento de todos os elementos** restantes (**ineficiente**)



Formas de Implementação

- **Ideia 2: utilizar uma lista linear circular**
  - Usa campo ***ini*** para indicar 1º nó da fila
  - Adota um **incremento circular (** $x = x \oplus 1$**):**

$$x = \begin{cases} x + 1, & \text{se } x+1 < max \\ 0, & \text{se } x+1 = max \end{cases}$$

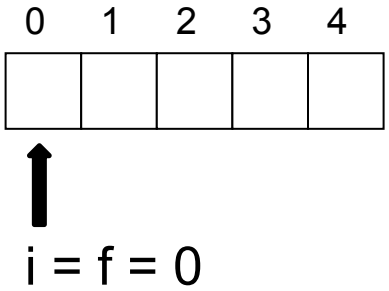
OU

$$x = (x+1) \ \% \ max$$

Fila Circular

Exemplo:

Fila vazia:

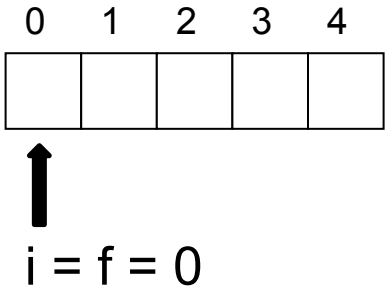


Max = 5

Fila Circular

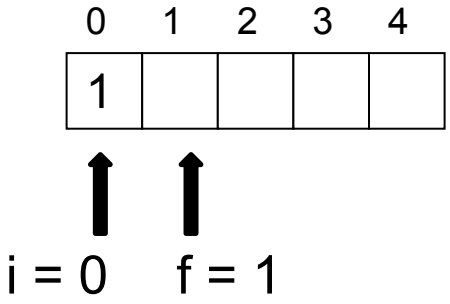
Exemplo:

Fila vazia:



Max = 5

Insere 1:



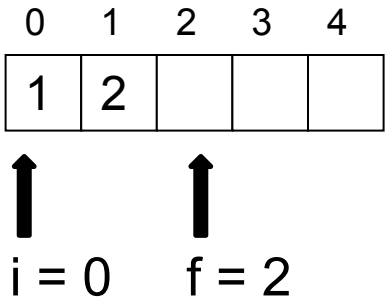
Vet[f] = E
f = f ⊕ 1

Incremento Circular

Fila Circular

Exemplo:

Insere 2:

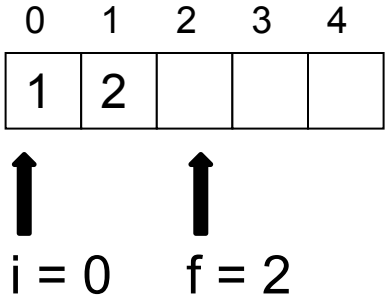


Vet[f] = E
f = f ⊕ 1

Fila Circular

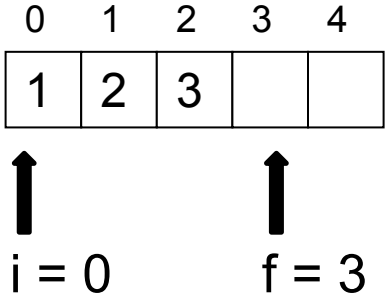
Exemplo:

Insere 2:



Vet[f] = E
f = f ⊕ 1

Insere 3:

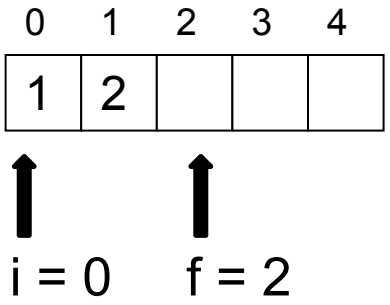


Vet[f] = E
f = f ⊕ 1

Fila Circular

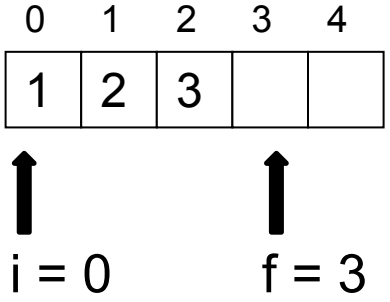
Exemplo:

Insere 2:



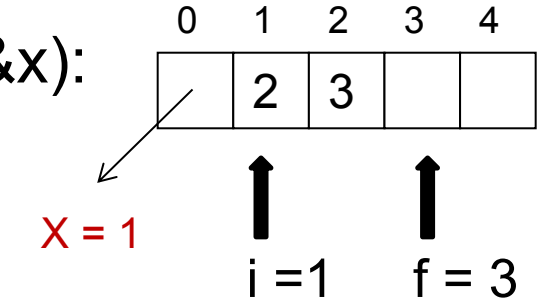
Vet[f] = E
f = f ⊕ 1

Insere 3:



Vet[f] = E
f = f ⊕ 1

Remove (&x):

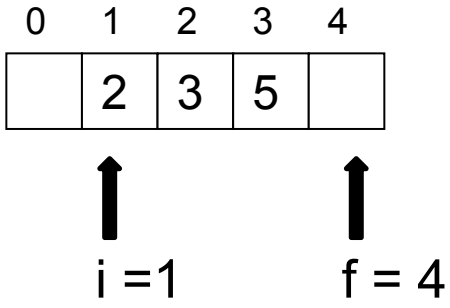


E = Vet[i];
i = i ⊕ 1

Fila Circular

Exemplo:

Insere 5:

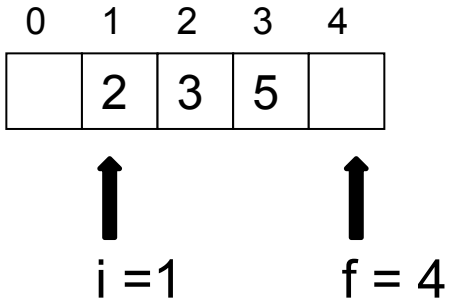


Vet[f] = E
f = f ⊕ 1

Fila Circular

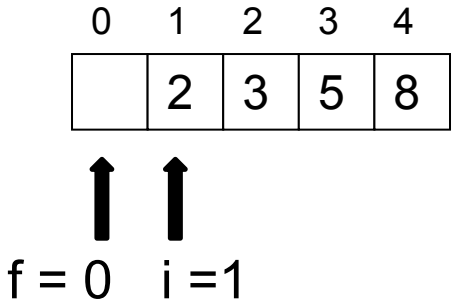
Exemplo:

Insere 5:



Vet[f] = E
f = f ⊕ 1

Insere 8:

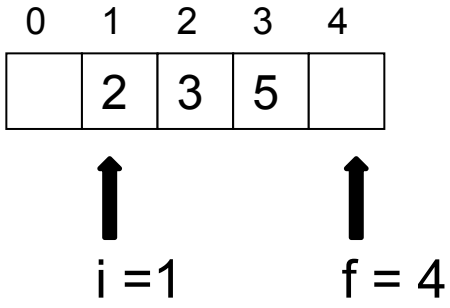


Vet[f] = E
f = f ⊕ 1

Fila Circular

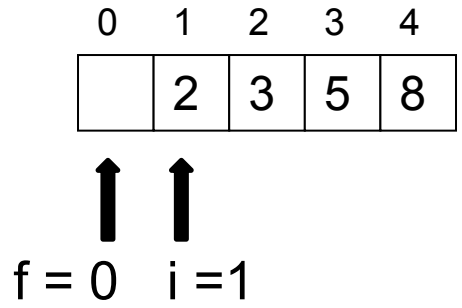
Exemplo:

Insere 5:



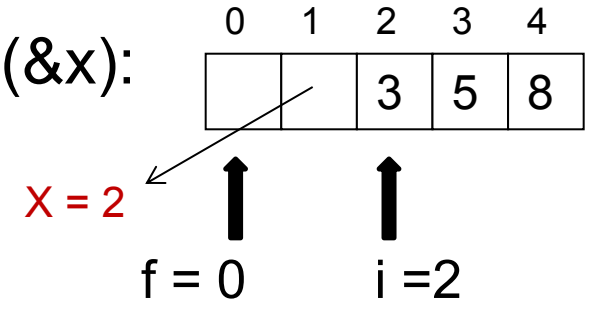
Vet[f] = E
f = f ⊕ 1

Insere 8:



Vet[f] = E
f = f ⊕ 1

Remove (&x):

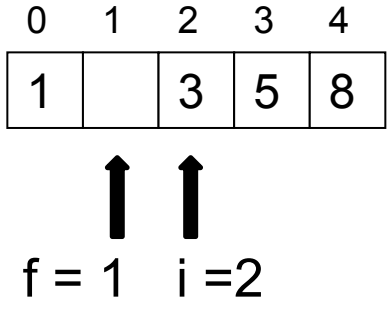


E = Vet[i];
i = i ⊕ 1

Fila Circular

Exemplo:

Insere 1:

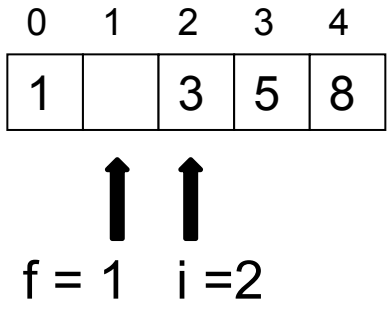


Vet[f] = E
f = f ⊕ 1

Fila Circular

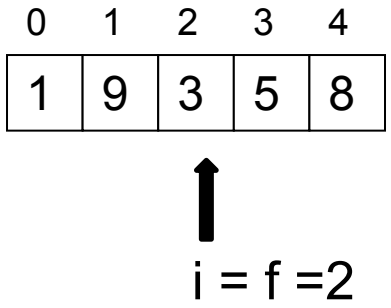
Exemplo:

Insere 1:



Vet[f] = E
f = f ⊕ 1

Insere 9:

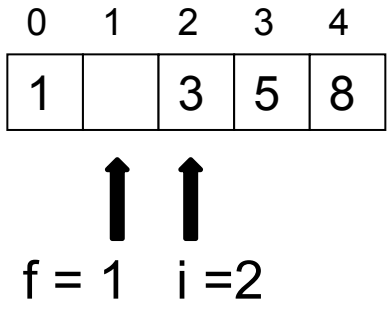


Vet[f] = E
f = f ⊕ 1

Fila Circular

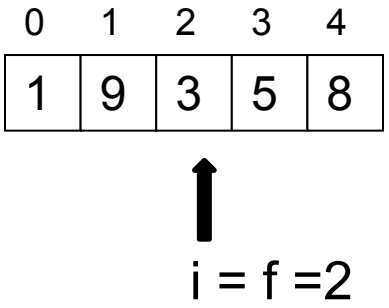
Exemplo:

Insere 1:



Vet[f] = E
f = f ⊕ 1

Insere 9:



Vet[f] = E
f = f ⊕ 1

- **Problema:** Diferenciar fila vazia e fila cheia
  - Ambos casos são ***ini*** = ***fim***

Fila Circular

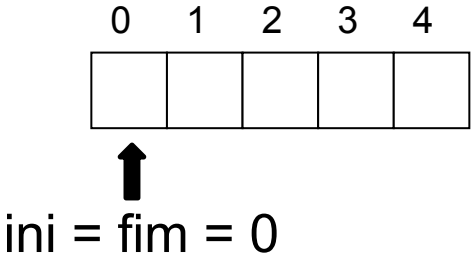
- **Existem 2 soluções:**
  - **Abordagem 1:** desperdício de 1 posição
  - **Abordagem 2:** uso de um contador

Fila com Desperdício de Posição

- São armazenados ***max-1*** elementos
  - Último nó da fila fica sempre vazio

Fila com Desperdício de Posição

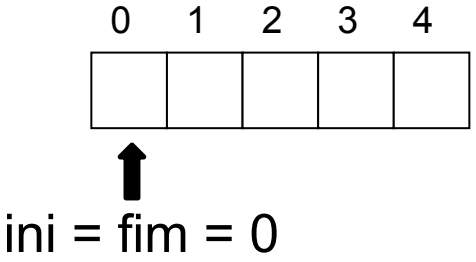
- São armazenados ***max-1*** elementos
  - Último nó da fila fica sempre vazio
  - **Fila vazia:**



ini = fim

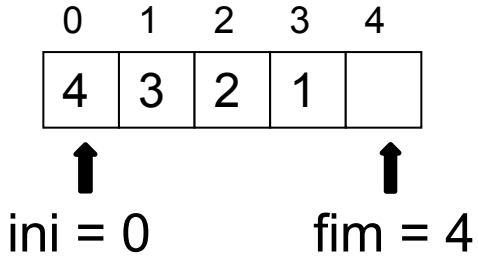
Fila com Desperdício de Posição

- São armazenados ***max-1*** elementos
  - Último nó da fila fica sempre vazio
  - **Fila vazia:**



ini = fim

  - **Fila cheia:**

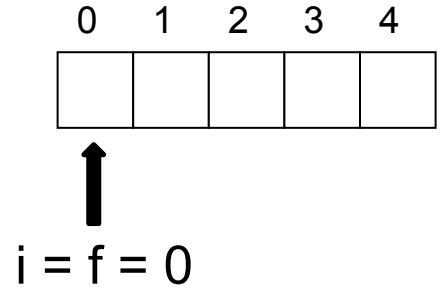


ini = fim ⊕ 1

Fila com Desperdício de Posição

- **Exemplo:**

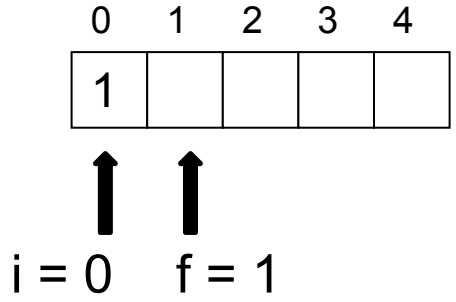
Início:



i = 0

f = 0

Insere 1:



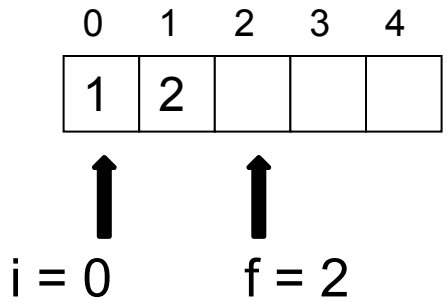
V[f] = E

f = f ⊕ 1

Fila com Desperdício de Posição

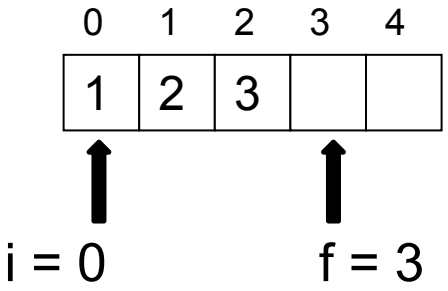
- **Exemplo:**

Insere 2:



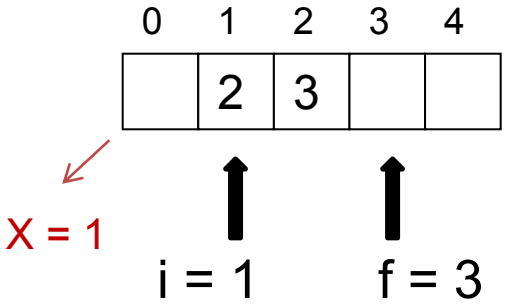
V[f] = E
f = f ⊕ 1

Insere 3:



V[f] = E
f = f ⊕ 1

Remove (&x):

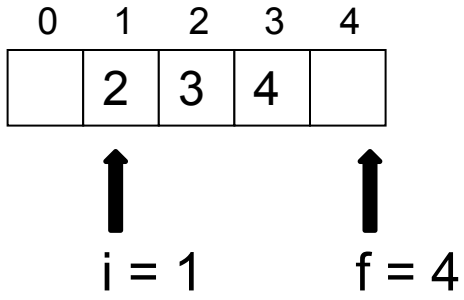


*E = V[i]
i = i ⊕ 1

Fila com Desperdício de Posição

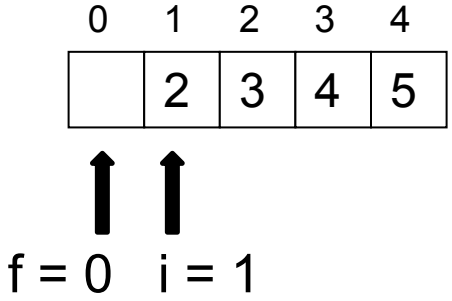
- **Exemplo:**

Insere 4:



V[f] = E
f = f ⊕ 1

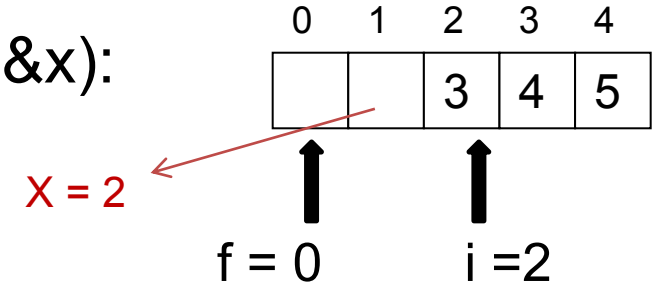
Insere 5:



V[f] = E
f = f ⊕ 1

FILA ESTÁ CHEIA

Remove (&x):

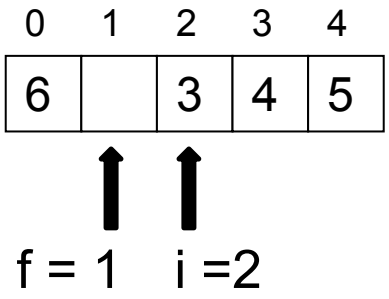


*E = V[i]
i = i ⊕ 1

Fila com Desperdício de Posição

- **Exemplo:**

Insere 6:



V[f] = E
f = f ⊕ 1

FILA ESTÁ CHEIA

Insere 7:

Operação falha
(FILA CHEIA)

Fila com Desperdício de Posição

- **Estrutura de representação:**
  - Vetor com ***max*** posições

Fila com Desperdício de Posição

- **Estrutura de representação:**
  - Vetor com ***max*** posições
  - Campo para indicar o início da fila (***ini***)

Fila com Desperdício de Posição

- **Estrutura de representação:**
  - Vetor com ***max*** posições
  - Campo para indicar o início da fila (***ini***)
  - Campo para indicar o fim da fila (***fim***)

Fila com Desperdício de Posição

- **Estrutura de representação:**
  - Vetor com ***max*** posições
  - Campo para indicar o início da fila (***ini***)
  - Campo para indicar o fim da fila (***fim***)

Exemplo: fila de inteiros

fila.c

```
# define max 20
struct fila {
        int vetor [max];
        int ini, fim;
};
```

fila.h

```
typedef struct fila * Fila;
```

Fila com Desperdício de Posição

- **Operação cria_fila():**
  - Aloca todas as posições da estrutura fila (**vetor**)

Fila com Desperdício de Posição

- **Operação cria_fila():**
  - Aloca todas as posições da estrutura fila (**vetor**)
  - *Coloca a fila no estado de vazia (**ini** = **fim**)*
    - Inicializados como **ZERO** ou ***max*-1**

Fila com Desperdício de Posição

- **Operação cria_fila():**
  - Aloca todas as posições da estrutura fila (**vetor**)
  - *Coloca a fila no estado de vazia (**ini = fim**)*
    - Inicializados como **ZERO** ou ***max*-1**

```
Fila cria_fila() {
    Fila f;
    f = (Fila) malloc(sizeof(struct fila));
    if (f != NULL) {
        f->ini = 0;
        f->fim = 0;
    }
    return f;
}
```

Fila com Desperdício de Posição

- **Operação fila_vazia():**
  - Verifica se a fila está no estado de vazia *(**ini=fim**)*
    - *Independe do valor atual dos campos (não precisa ter o mesmo valor usado na inicialização)*

```
int fila_vazia(Fila f) {
    if (f->ini == f->fim)
        return 1;
    else
        return 0;
}
```

Fila com Desperdício de Posição

- **Operação fila_cheia():**
  - Verifica se a fila está no estado de cheia

    (ini = fim ⊕ 1)

```
int fila_cheia(Fila f) {
    if (f->ini == (f->fim+1)%max)
        return 1;
    else
        return 0;
}
```

Fila com Desperdício de Posição

- **Operação insere_fim():**
  - Insere o elemento no final da fila
    - Posição indicada pelo campo ***fim***

Fila com Desperdício de Posição

- **Operação insere_fim():**
  - Insere o elemento no final da fila
    - Posição indicada pelo campo ***fim***
  - Incrementa o campo ***fim*** (**incremento circular**)

Fila com Desperdício de Posição

- **Operação insere_fim():**
  - Insere o elemento no final da fila
    - Posição indicada pelo campo ***fim***
  - Incrementa o campo ***fim*** (**incremento circular**)

```
int insere_fim(Fila f, int elem) {
    if (fila_cheia(f) == 1)
        return 0;
    // Insere elemento no final
    f->no[f->fim] = elem;
    f->fim = (f->fim+1)%max; // Incremento circular
    return 1;
}
```

Fila com Desperdício de Posição

- **Operação remove_ini():**
  - Retorna o valor do elemento no início da fila
    - Posição indicada pelo campo ***ini***

Fila com Desperdício de Posição

- **Operação remove_ini():**
  - Retorna o valor do elemento no início da fila
    - Posição indicada pelo campo ***ini***
  - Incrementa o campo ***ini*** (**incremento circular**)

Fila com Desperdício de Posição

- **Operação remove_ini():**
  - Retorna o valor do elemento no início da fila
    - Posição indicada pelo campo ***ini***
  - Incrementa o campo ***ini*** (**incremento circular**)

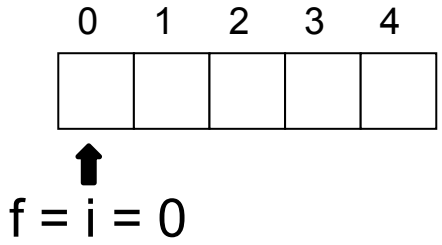
```
int remove_ini(Fila f, int *elem) {
    if (fila_vazia(f) == 1)
        return 0;
    // Remove o elemento do inicio
    *elem = f->no[f->ini];
    f->ini = (f->ini+1)%max; // Incremento circular
    return 1;
}
```

Fila com Uso do Contador

- Utiliza um campo para contar elementos
  - ***Fim*** obtido a partir do ***início*** e do ***contador*** (***fim = ini ⊕ cont = (ini + cont) % max***)

Fila com Uso do Contador

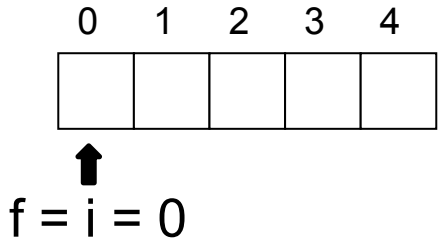
- Utiliza um campo para contar elementos
  - ***Fim*** obtido a partir do ***início*** e do ***contador*** (***fim = ini ⊕ cont = (ini + cont) % max***)
  - **Fila vazia:**



cont = 0

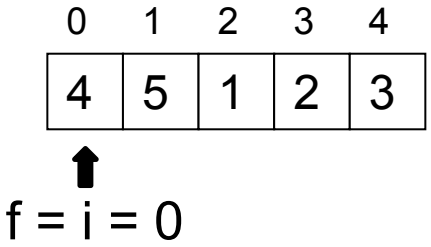
Fila com Uso do Contador

- Utiliza um campo para contar elementos
  - ***Fim*** obtido a partir do ***início*** e do ***contador*** (***fim = ini ⊕ cont = (ini + cont) % max***)
  - **Fila vazia:**



cont = 0

  - **Fila cheia:**

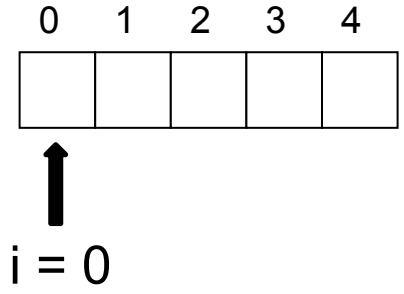


cont = max

Fila com Uso do Contador

- **Exemplo:**

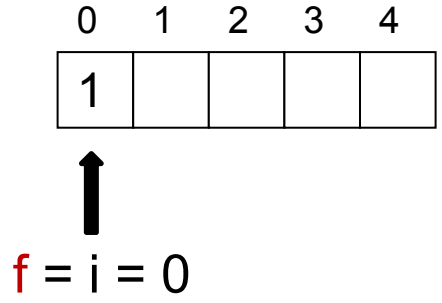
Início:



i = 0

cont = 0

Insere 1:



f = i ⊕ cont

V[f] = E

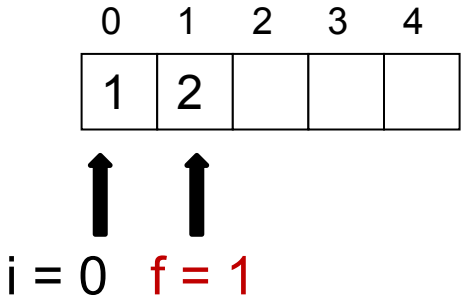
cont++

(cont = 1)

Fila com Uso do Contador

- **Exemplo:**

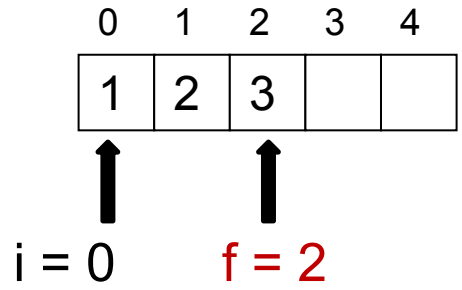
Insere 2:



f = i ⊕ cont
V[f] = E
cont++

(cont = 2)

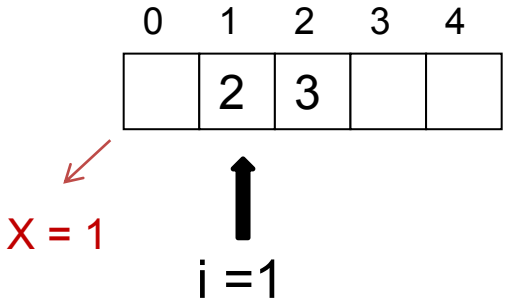
Insere 3:



f = i ⊕ cont
V[f] = E
cont++

(cont = 3)

Remove (&x):



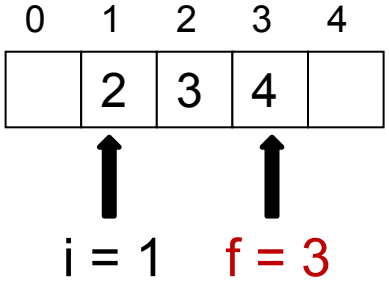
*E = V[i]
i = i ⊕ 1
cont--

(cont = 2)

Fila com Uso do Contador

- **Exemplo:**

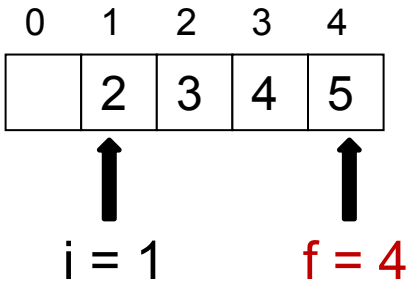
Insere 4:



f = i ⊕ cont
V[f] = E
cont++

(cont = 3)

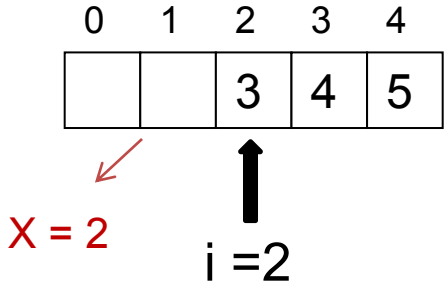
Insere 5:



f = i ⊕ cont
V[f] = E
cont++

(cont = 4)

Remove (&x):



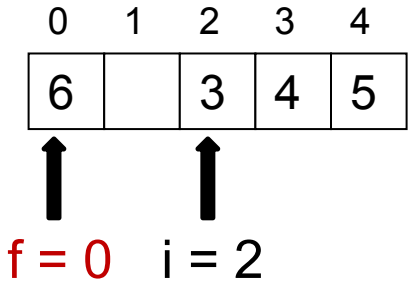
*E = V[i]
i = i ⊕ 1
cont--

(cont = 3)

Fila com Uso do Contador

- **Exemplo:**

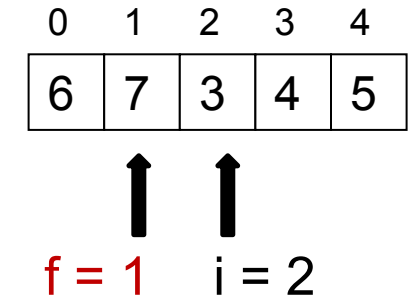
Insere 6:



f = i ⊕ cont
V[f] = E
cont++

(cont = 4)

Insere 7:



f = i ⊕ cont
V[f] = E
cont++

(cont = 5)

Insere 8: Operação falha (**FILA ESTÁ CHEIA**)

Fila com Uso do Contador

- **Estrutura de representação:**
  - Vetor com ***max*** posições

Fila com Uso do Contador

- **Estrutura de representação:**
  - Vetor com ***max*** posições
  - Campo para indicar o início da fila (***ini***)

Fila com Uso do Contador

- **Estrutura de representação:**
  - Vetor com ***max*** posições
  - Campo para indicar o início da fila (***ini***)
  - Campo para contar os elementos (***cont***)
    - ***Fim*** obtido a partir dos campos ***ini*** e ***cont***

Fila com Uso do Contador

- **Estrutura de representação:**
  - Vetor com ***max*** posições
  - Campo para indicar o início da fila (***ini***)
  - Campo para contar os elementos (***cont***)
    - ***Fim*** obtido a partir dos campos ***ini*** e ***cont***

Exemplo: fila de inteiros

fila.c

```
# define max 20
struct fila {
        int vetor [max];
        int ini, cont;
};
```

fila.h

```
typedef struct fila * Fila;
```

Fila com Uso do Contador

- **Operação cria_fila():**
  - Aloca todas as posições da estrutura fila (**vetor**)

Fila com Uso do Contador

- **Operação cria_fila():**
  - Aloca todas as posições da estrutura fila (**vetor**)
  - *Coloca a fila no estado de vazia*
    - ***Ini*** e ***cont*** inicializados com **ZERO**

Fila com Uso do Contador

- **Operação cria_fila():**
  - Aloca todas as posições da estrutura fila (**vetor**)
  - *Coloca a fila no estado de vazia*
    - ***Ini*** e ***cont*** inicializados com **ZERO**

```
Fila cria_fila() {
    Fila f;
    f = (Fila) malloc(sizeof(struct fila));
    if (f != NULL) {
        f->ini = 0;
        f->cont = 0;
    }
    return f;
}
```

Fila com Uso do Contador

- **Operação fila_vazia():**
  - Verifica se a fila está no estado de vazia *(**cont=0**)*
    - *Independe dos valores dos campos **ini** e **fim***

```
int fila_vazia(Fila f) {
    if (f->cont == 0)
        return 1;
    else
        return 0;
}
```

Fila com Uso do Contador

- **Operação fila_cheia():**
  - Verifica se a fila está cheia *(**cont = max**)*
    - ***max*** corresponde a quantidade máxima de elementos permitidos na fila

```
int fila_cheia(Fila f) {
    if (f->cont == max)
        return 1;
    else
        return 0;
}
```

Fila com Uso do Contador

- **Operação insere_fim():**
  - Insere o elemento no final da fila
    - **Posição calculada** a partir dos campos ***ini*** e ***cont***

Fila com Uso do Contador

- **Operação insere_fim():**
  - Insere o elemento no final da fila
    - **Posição calculada** a partir dos campos ***ini*** e ***cont***
  - Incrementa (**NÃO circular**) o campo ***cont***

Fila com Uso do Contador

- **Operação insere_fim():**
  - Insere o elemento no final da fila
    - **Posição calculada** a partir dos campos ***ini*** e ***cont***
  - Incrementa (**NÃO circular**) o campo ***cont***

```
int insere_fim(Fila f, int elem) {
    if (fila_cheia(f) == 1)
        return 0;
    // Insere elemento no final
    f->no[(f->ini+f->cont)%max] = elem;
    f->cont++; // Incremento normal
    return 1;
}
```

Fila com Uso do Contador

- **Operação remove_ini():**
  - Retorna o valor do elemento no início da fila
    - Posição indicada pelo campo ***ini***

Fila com Uso do Contador

- **Operação remove_ini():**
  - Retorna o valor do elemento no início da fila
    - Posição indicada pelo campo ***ini***
  - Incrementa o campo ***ini*** (**incremento circular**)

Fila com Uso do Contador

- **Operação remove_ini():**
  - Retorna o valor do elemento no início da fila
    - Posição indicada pelo campo ***ini***
  - Incrementa o campo ***ini*** (**incremento circular**)
  - Decrementa o campo ***cont***

Fila com Uso do Contador

- **Operação remove_ini():**
  - Retorna o valor do elemento no início da fila
    - Posição indicada pelo campo ***ini***
  - Incrementa o campo ***ini*** (**incremento circular**)
  - Decrementa o campo ***cont***

```
int remove_ini(Fila f, int *elem) {
    if (fila_vazia(f) == 1)   return 0;
    // Remove o elemento do inicio
    *elem = f->no[f->ini];
    f->ini = (f->ini+1)%max; // Incremento circular
    f->cont--; // Decremento não circular
    return 1;  }
```

Exercícios

1. *Implementar, utilizando a implementação **estática/sequencial com desperdício de posição**, o TAD fila de números inteiros. Essa implementação deve contemplar as operações básicas: criar_fila, fila_vazia, fila_cheia, insere_fim e remove_ini. Além disso, desenvolva um programa aplicativo que permita ao usuário criar uma fila, inserir e remover elementos, e imprimir a fila.*

   Teste este programa com a seguinte seqüencia de operações:

   - *Cria fila*
   - *Imprime fila*
   - *Insere os elementos {4,8,-1,19,2,7,8,5,9,22,45};*
   - *Imprime fila*
   - *Remove elemento*
   - *Imprime fila*

2. *Altere o código anterior de modo a implementar a **fila estática/sequencial com o uso do campo contador**.*

Referências

- *Backes, André, Linguagem C Descomplicada, portal de vídeo-aulas, https://programacaodescomplicada.wordpress.com/, acessado em 09/03/2016.*

- *Celes, W., Cerqueira, R. e Rangel, J. L. Introdução a estruturas de dados. Ed. Campus Elsevier, 2004.*