Filas
(IED-001)

Prof. Dr. Silvio do Lago Pereira

Departamento de Tecnologia da Informação

Faculdade de Tecnologia de São Paulo

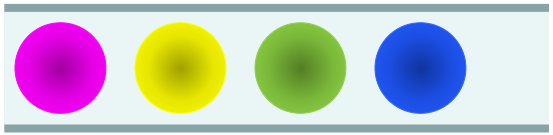
Filas

Fila

é uma lista de itens em que as inserções são feitas num extremo, denominado **final**, e as remoções são feitas no extremo oposto, denominado **início**.

Funcionamento:

- Quando um item é inserido numa fila, ele é colocado no seu final.
- Apenas o item no início da fila pode ser removido.
- Essa política de acesso é denominada **FIFO** (*First-In/First-Out*).



Exemplos de aplicação:

- Fila de impressão de arquivos num sistema operacional.
- Fila de escalonamento de processos num sistema operacional.

Fila é útil em qualquer situação em que precisamos **manter** a ordem de uma sequência!

Filas

Operações em filas

- **`fila(m)`**

  Cria e devolve uma fila vazia **`F`**, com tamanho **`m`**.

- **`vaziaf(F)`**

  Devolve verdade se, e só se, a fila **`F`** estiver vazia.

- **`cheiaf(F)`**

  Devolve verdade se, e só se, a fila **`F`** estiver cheia.

- **`enfileira(x,F)`**

  Insere o item **`x`** no final da fila **`F`**.

- **`desenfileira(F)`**

  Remove e devolve o item que estiver no início de **`F`**.

- **`destroif(&F)`**

  Destrói a fila **`F`**.

Exemplo 1. Operação, efeito e resultado

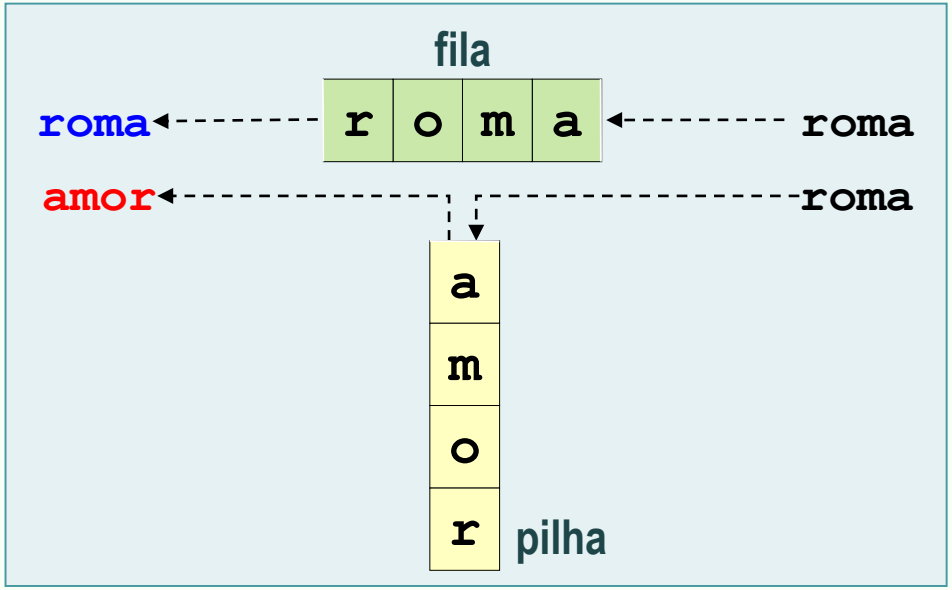
`F = fila(3)`	`[]`	–
`vaziaf(F)`	`[]`	1
`cheiaf(F)`	`[]`	0
`enfileira(1,F)`	`[1]`	–
`enfileira(2,F)`	`[1,2]`	–
`enfileira(3,F)`	`[1,2,3]`	–
`vaziaf(F)`	`[1,2,3]`	0
`cheiaf(F)`	`[1,2,3]`	1
`desenfileira(F)`	`[2,3]`	1
`desenfileira(F)`	`[3]`	2
`enfileira(4,F)`	`[3,4]`	–
`enfileira(desenfileira(F),F)`	`[4,3]`	–
`vaziaf(F)`	`[4,3]`	0
`cheiaf(F)`	`[4,3]`	0
`destroif(&F)`	inexistente	–

Vamos começar supondo que esse tipo de dados está disponível no arquivo **`fila.h`**!

Filas

Exemplo 2. Cadeia palíndroma

```
#include <stdio.h>
#include <ctype.h>
#include "pilha.h"
#include "fila.h"
int main(void) {
   char s[256];
   Fila F = fila(256);
   Pilha P = pilha(256);
   printf("\nFrase? ");
   gets(s);
   for(int i=0; s[i]; i++)
      if( isalpha(s[i]) ) {
         enfileira(s[i],F);
         empilha(s[i],P);
      }
   while( !vaziaf(F) && desenfileira(F)==desempilha(P) );
   if( vaziaf(F) ) puts("A frase e palindroma");
   else puts("A frase nao e palindroma");
   destroif(&F);
   destroip(&P);
   return 0;
}
```

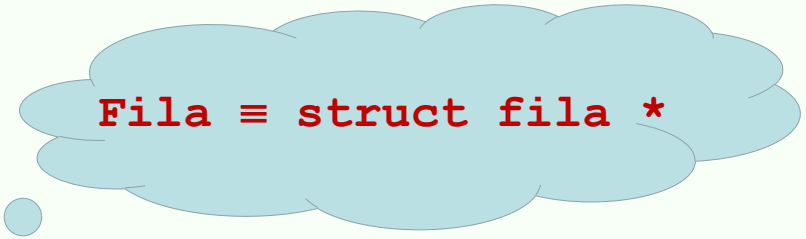


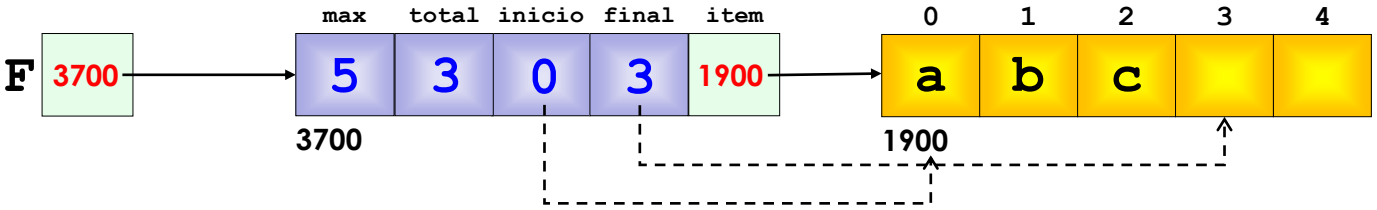
Filas

Exemplo 3. O tipo `Fila`

```
typedef char Itemf;

typedef struct fila {
    int     max;
    int     total;
    int     inicio;
    int     final;
    Itemf *item;
} *Fila;
```



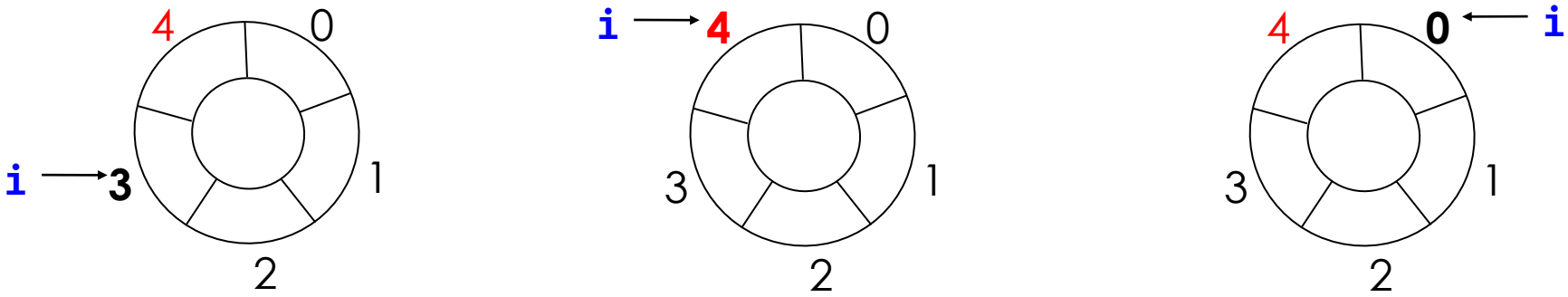


Note que o tipo **`Itemf`** pode ser redefinido, em função da aplicação que usa o tipo **`Fila`**!

Filas

Exemplo 4. Simulação de vetor circular

```
#define avanca(i)  (i = (i+1) % F->max)
```



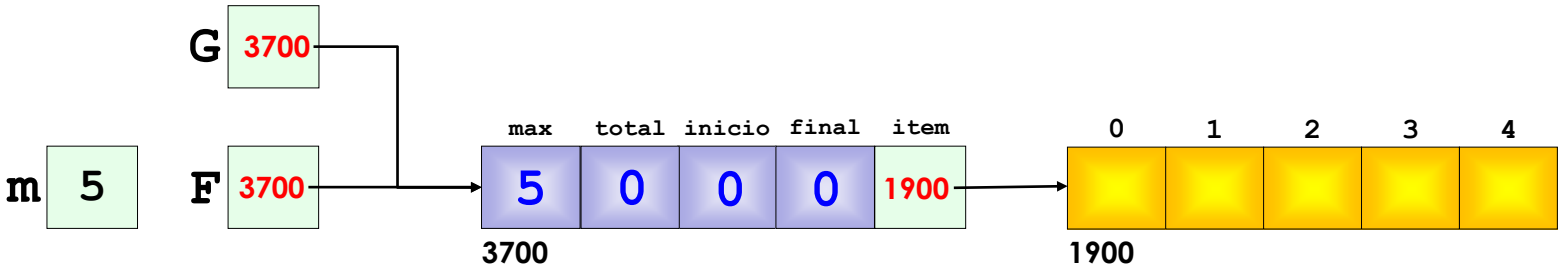
A simulação de vetor circular permite um melhor aproveitamento do espaço de memória!

Filas

Exemplo 5. Criação de fila vazia

```
Fila fila(int m) {
    Fila F = malloc(sizeof(struct fila));
    F->max    = m;
    F->total  = 0;
    F->inicio = 0;
    F->final  = 0;
    F->item   = malloc(m*sizeof(Itemp));
    return F;
  }
```

```
Fila G = fila(5);
```

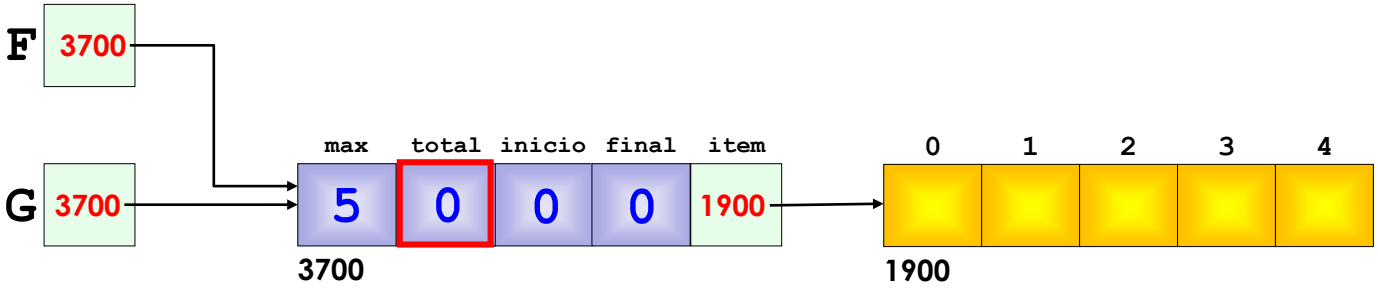


Note que o nome **`Fila`** identifica o tipo, enquanto o nome **`fila`** identifica a função!

Exemplo 6. Teste de fila vazia

```
int vaziaf(Fila F) {
    return (F->total == 0);
}
```

```
vaziaf(G)
```



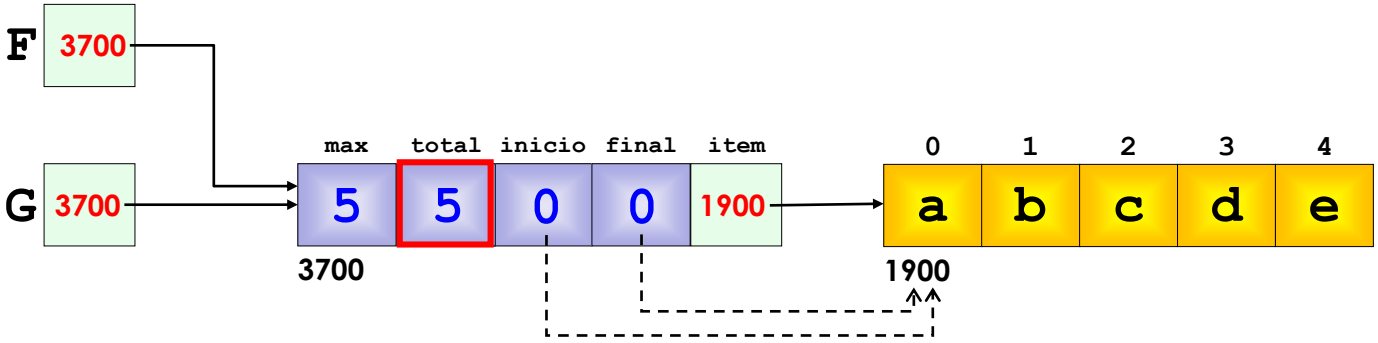
A fila está vazia quando o seu campo **total** tem valor 0!

Filas

Exemplo 7. Teste de fila cheia

```
int cheiaf(Fila F) {
    return (F->total == F->max);
}
```

```
cheiaf(G)
```

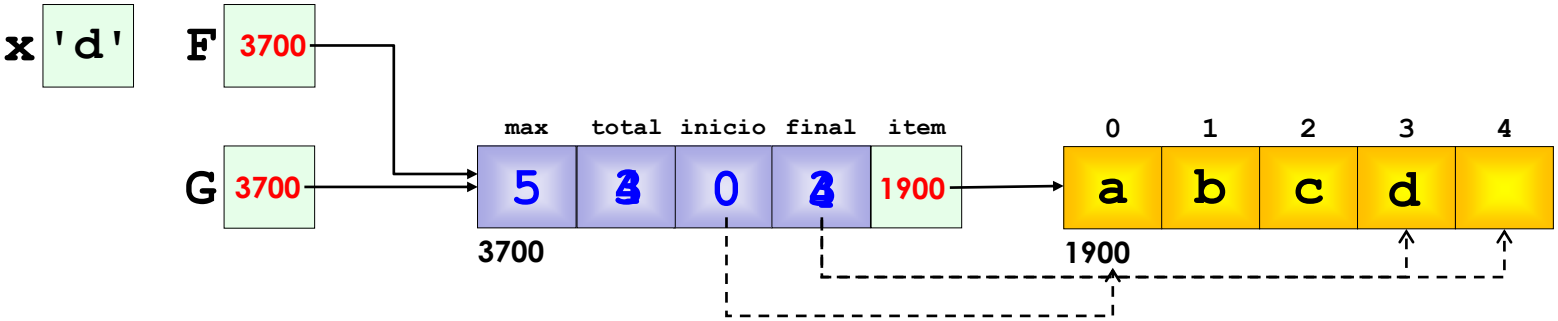


A fila está cheia quando o seu campo **total** tem valor **max**!

Exemplo 8. Inserção de item em fila

```
void enfileira(Itemf x, Fila F) {
    if( cheiaf(F) ) {
        puts("fila cheia!");
        abort();
    }
    F->item[F->final] = x;
    avanca(F->final);
    F->total++;
}
```

```
enfileira('d',G)
```



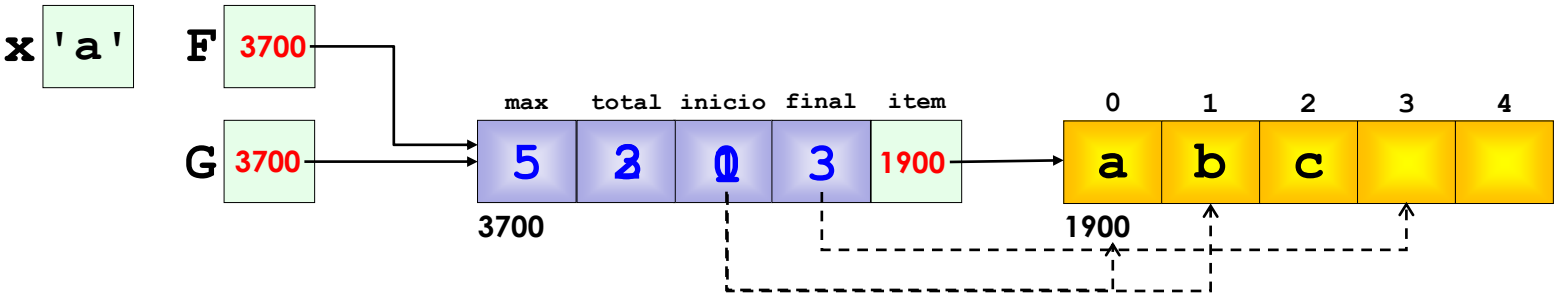
A tentativa de **inserção** em fila cheia causa um erro!

Filas

Exemplo 9. Remoção de item em fila

```
Itemf desenfileira(Fila F) {
    if( vaziaf(F) ) {
        puts("fila vazia!");
        abort();
    }
    Itemf x = F->item[F->inicio];
    avanca(F->inicio);
    F->total--;
    return x;
}
```

```
desenfileira(G)
```



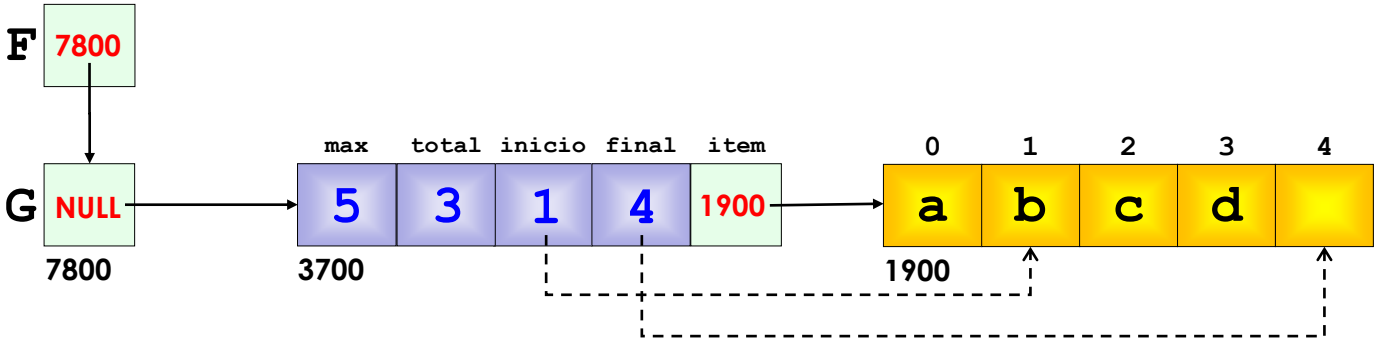
A tentativa de **remoção** em fila vazia causa um erro!

Filas

Exemplo 10. Destruição de fila

```
void destroif(Fila *F) {
    free((*F)->item);
    free(*F);
    *F = NULL;
}
```

```
destroif(&G)
```



Após a destruição da fila, ela não pode mais ser acessada!

Filas

Exercício 1. Cadeias palíndromas

O programa do Exemplo 2 não reconhece **"Amor a Roma"** como uma cadeia palíndroma. Use a função **toupper()**, declarada em **ctype.h**, para resolver esse problema (essa função converte uma letra minúscula em maiúscula).

Exercício 2. Saída de programa

Qual a saída exibida pelo programa a seguir?

```
#include <stdio.h>
#include "fila.h"
int main(void) {
   Fila F = fila(5);
   for(int i=0; i<=3; i++)
      enfileira('A'+i,F);
   while( !vaziaf(F) )
      printf("%c\n",desenfileira(F));
   destroif(&F);
   return 0;
}
```

Filas

Exercício 3. Compartilhamento de CPU

Esse programa simula o compartilhamento de uma CPU entre vários processos. Qual a saída?

```
#include <stdio.h>
#include "fila.h"
#define TIMESLICE 3 // tempo máximo de uso ininterrupto de CPU
int main(void) {
   Fila F = fila(5);
   enfileira(17,F); // 1° processo requer 7s para concluir sua execução
   enfileira(25,F);
   enfileira(39,F);
   enfileira(46,F);
   while( !vaziaf(F) ) {
      int x = desenfileira(F);
      int p = x/10;
      int t = x%10;
      if( t>3 ) enfileira(p*10+(t-TIMESLICE),F);
      else printf("Processo %d concluido\n",p);
   }
   destroif(&F);
   return 0;
}
```

Fim