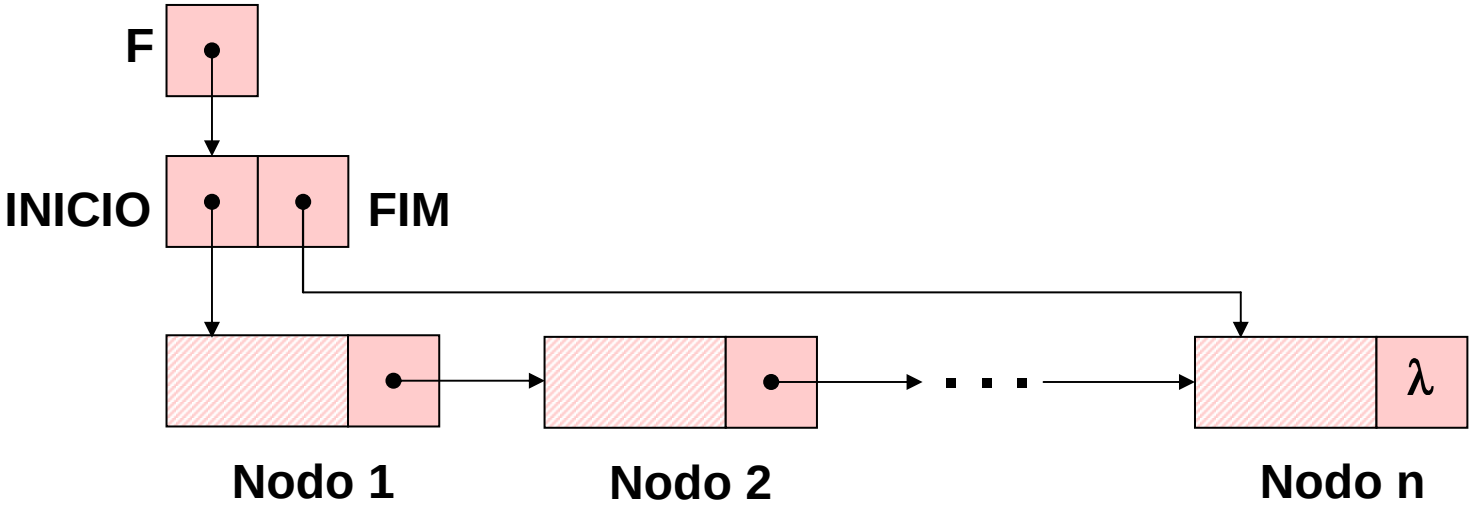
Alocação Encadeada

Com já discutimos, a alocação seqüência apresenta algumas desvantagens. Em virtude disso, podemos nós utilizar de uma lista encadeada para armazenarmos uma fila. Como operaremos em ambas as extremidades da lista, devemos facilitar o acesso ao último nodo. Uma estratégia, muito utilizada, é a utilização de uma representação baseada em um descritor contendo duas referências, ao primeiro e ao último nodo.

Alocação Encadeada



Desta forma, definirmos e implementaremos, agora, o TAD FILA_ENC (de valores inteiros).

```
typedef struct nodo
{
   int inf;
   struct nodo * next;
}NODO;
typedef struct
{
   NODO *INICIO;
   NODO *FIM;
}DESCRITOR;
typedef DESCRITOR * FILA_ENC;
void cria_fila (FILA_ENC *);
int eh_vazia (FILA_ENC);
void ins (FILA_ENC, int);
int cons (FILA_ENC);
void ret (FILA_ENC);
int cons_ret (FILA_ENC);
```

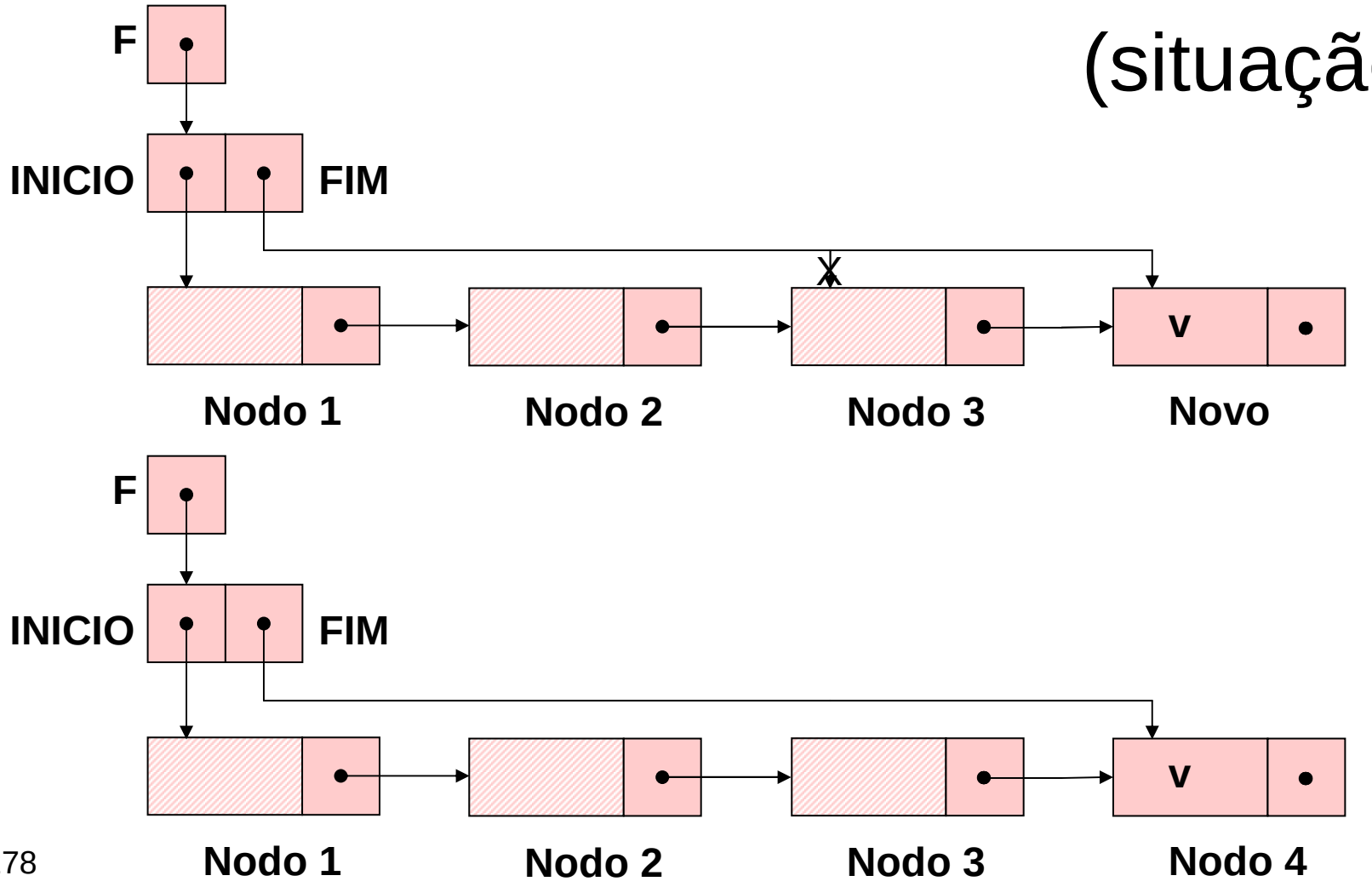
```
void cria_fila (FILA_ENC *pf)
{
   *pf=(DESCRITOR *)malloc(sizeof(DESCRITOR));
   if (!*pf)
   {
      printf ("\nERRO! Memoria insuficiente!\n");
      exit (1);
   }
   (*pf)->INICIO=(*pf)->FIM=NULL;
}
```

```
int eh_vazia (FILA_ENC f)
{
   return (f->INICIO == NULL);
}
```

Alocação Encadeada

Esquema do processo de inserção de um elemento na fila encadeada.

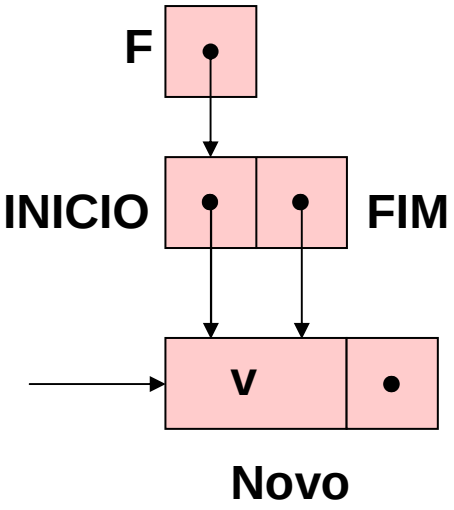
(situação 1)

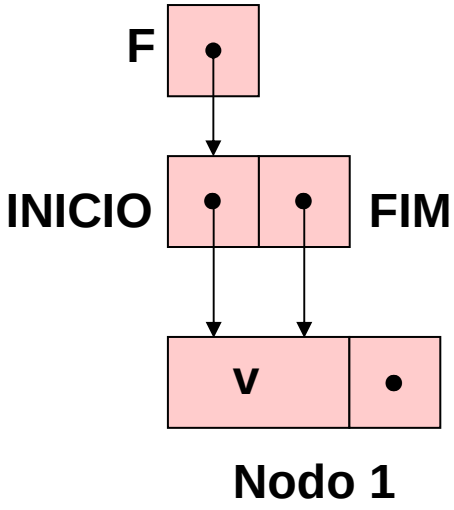


Alocação Encadeada

Esquema do processo de inserção de um elemento na fila encadeada.

(situação 2)





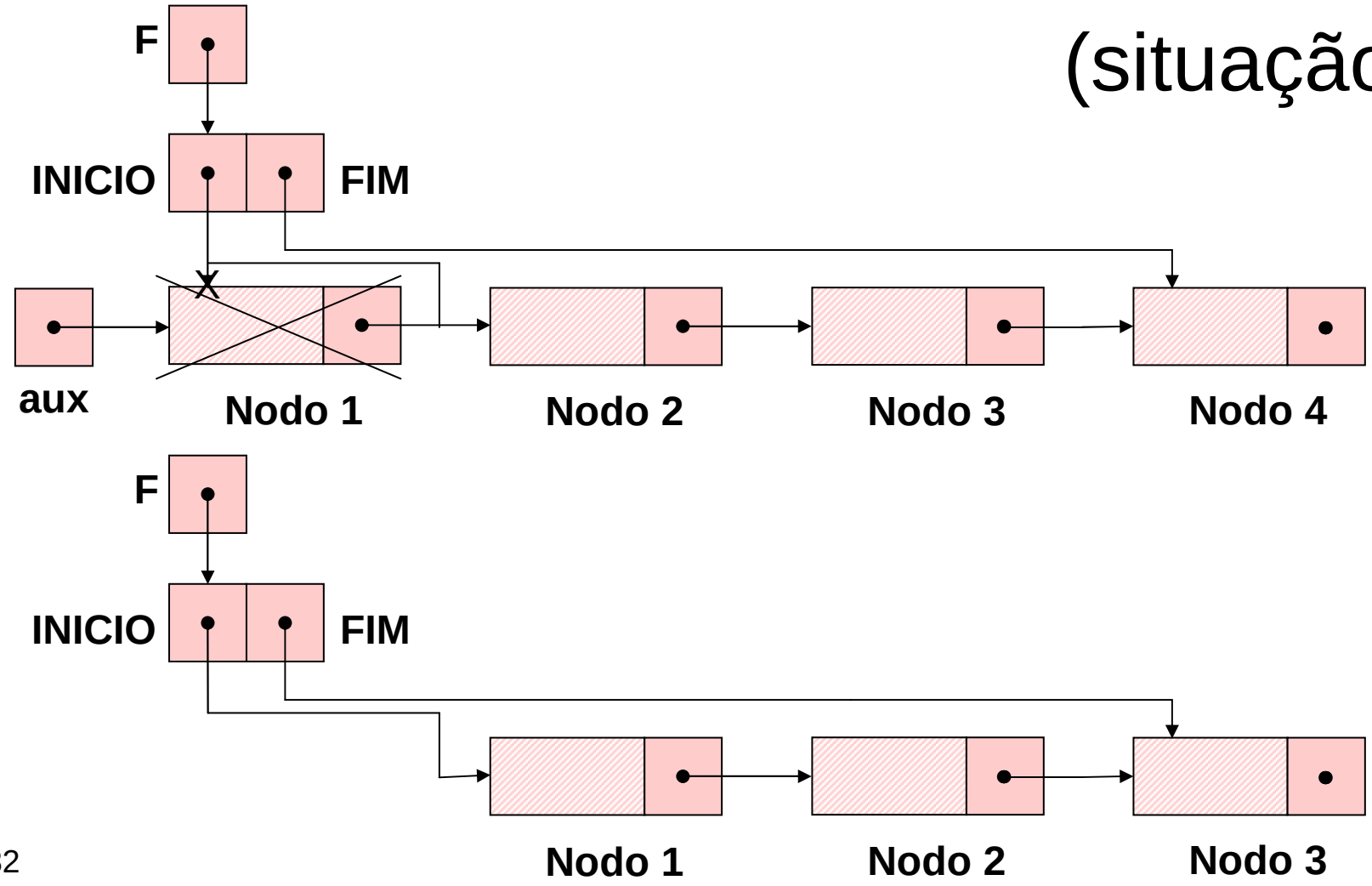
```
void ins (FILA_ENC f, int v)
{
   NODO *novo;
   novo = (NODO *) malloc (sizeof(NODO));
    if (!novo)
    {
       printf ("\nERRO! Memoria insuficiente!\n");
       exit (1);
    }
   novo->inf = v;
   novo->next = NULL;
   if (eh_vazia(f))
      f->INICIO=novo;
   else
      f->FIM->next=novo;
   f->FIM=novo;   }
```

```
int cons (FILA_ENC f)
{
   if (eh_vazia(f))
   {
      printf ("\nERRO! Consulta em fila vazia!\n");
      exit (2);
   }
   else
      return (f->INICIO->inf);
}
```

Alocação Encadeada

Esquema do processo de retirada de um elemento na fila encadeada.

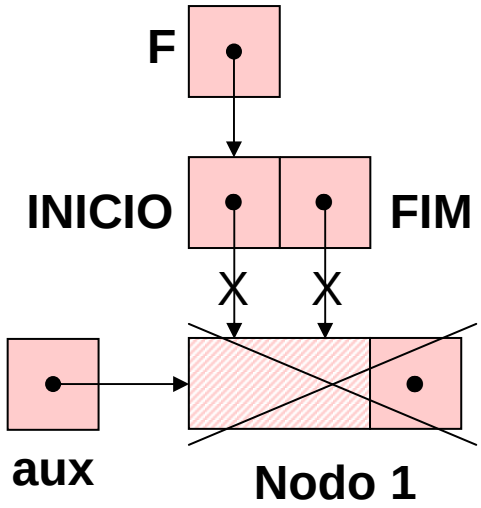
(situação 1)

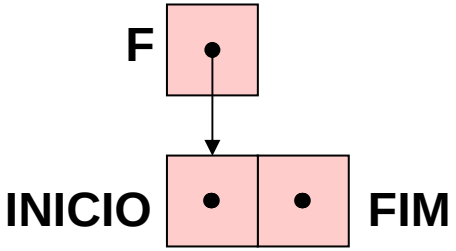


Alocação Encadeada

Esquema do processo de retirada de um elemento na fila encadeada.

(situação 2)





```
void ret (FILA_ENC f)
{
   if (eh_vazia(f))
   {
      printf ("\nERRO! Retirada em fila vazia!\n");
      exit (3);
   }
   else
   {
      NODO *aux=f->INICIO;
      f->INICIO=f->INICIO->next;
      if (!f->INICIO)
         f->FIM=NULL;
      free (aux);    }  }
```

```
int cons_ret (FILA_ENC f)
{
   if (eh_vazia(f))
   {  printf ("\nERRO! Consulta e retirada em fila vazia!\n");
      exit (4);
   }
   else
  {
      int v=f->INICIO->inf;
      NODO *aux=f->INICIO;
      f->INICIO=f->INICIO->next;
      if (!f->INICIO)
         f->FIM=NULL;
      free (aux);
      return (v);    }   }
```

Alocação Encadeada

Uma aplicação interessante para filas é a ordenação por distribuição, descrita a seguir. Seja uma lista l composta de n chaves, cada qual representada por um inteiro numa base b > 1. O problema consiste em ordenar essa lista. O algoritmo utiliza b filas, denotadas por f_i, $0 \leq i \leq b-1$. Seja d o comprimento máximo da representação das chaves na base b. O algoritmo efetua d iterações, em cada uma das quais a tabela é percorrida. A primeira iteração destaca, em cada

Alocação Encadeada

nó, o dígito menos significativo da representação b-ária de cada chave. Se este for igual a k, a chave correspondente será inserida na fila f_k. Ao terminar o percurso da tabela, esta se encontra distribuída pelas filas, que devem então ser concatenadas em seqüência, isto é, f_0, depois f_1, f_2, etc. Para essa tabela, já disposta numa ordem diferente da original, o processo deve ser repetido levando-se em consideração o segundo dígito da

Alocação Encadeada

representação, e assim sucessivamente até que tenham sido feitas tantas distribuições quantos são os dígitos na chave de ordenação. Observaremos agora um exemplo dessa ordenação, onde b=10 e d=2.

lista: 19 13 05 27 01 26 31 16 02 09 11 21 60 07

Interação 1: 1ª distribuição (unidades simples)

$fila_0$: 60

$fila_1$: 01, 31, 11, 21

$fila_2$: 02

$fila_3$: 13

$fila_4$:

$fila_5$: 05

$fila_6$: 26, 16

$fila_7$: 27, 07

$fila_8$:

$fila_9$: 19, 09

lista: 60 01 31 11 21 02 13 05 26 16 27 07 19 09

Interação 2: 2ª distribuição (dezenas simples)

$fila_0$: 01, 02, 05, 07, 09

$fila_1$: 11, 13, 16, 19

$fila_2$: 21, 26, 27

$fila_3$: 31

$fila_4$:

$fila_5$:

$fila_6$: 60

$fila_7$:

$fila_8$:

$fila_9$:

tabela: 01 02 05 07 09 11 13 16 19 21 26 27 31 60

Alocação Encadeada - Exercício

Utilizando-se dos TAD´s LISTA_ENC e FILA_ENC, implemente a operação ord_por_dist a qual recebe como entrada uma referência para uma lista encadeada de inteiros e a ordena através do processo de ordenação por distribuição.

OBS: b=10 e d=4.