INSTITUTO FEDERAL DE
EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
SUL-RIO-GRANDENSE

Estrutura de Dados

Pilhas, Filas e Filas duplas

Prof. Silvana Teodoro

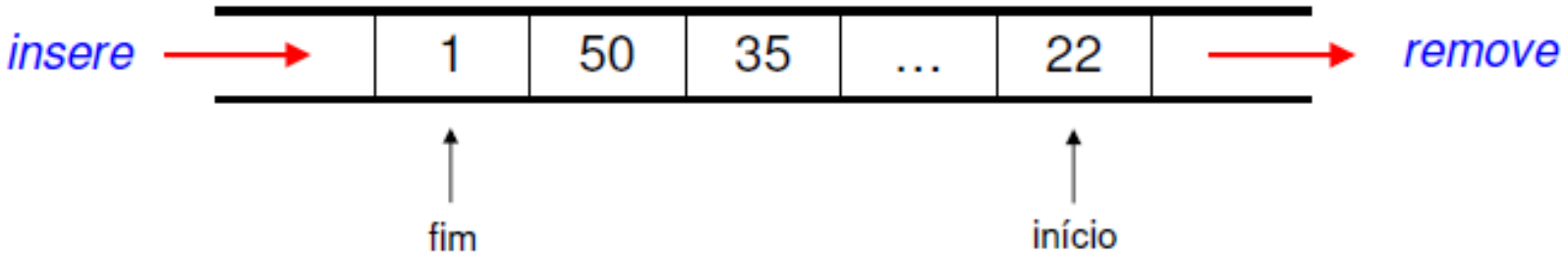
silvanateodoro@charqueadas.ifsul.edu.br

Sumário

- Revisão
- Filas duplas
- Funções
- Atividade
- Referências
- Leitura recomendada

Revisão: Filas

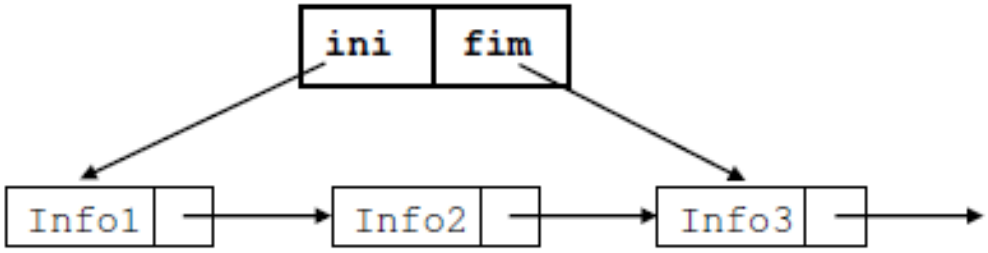
- Um novo elemento é inserido no final da fila e um elemento é retirado do início da fila.



- Diferença:
  - **Fila = "o primeiro que entra é o primeiro que sai" (FIFO)**
  - **Pilha = "o último que entra é o primeiro que sai" (LIFO)**

Revisão: Implementação da fila com listas

- Características:
  - Elementos da fila armazenados na lista
  - Usa dois ponteiros
    - ini aponta para o primeiro elemento da fila
    - fim aponta para o último elemento da fila



Revisão: Implementação da fila com listas

- Características:
  - Fila representada por um ponteiro para o primeiro nó da lista

```
/* nó da lista para armazenar valores reais */
struct lista {
   float info;
   struct lista* prox;
};
typedef struct lista Lista;

/* estrutura da fila */
struct fila {
   Lista* ini;
   Lista* fim;
};
```

Revisão: Implementação da fila com listas

- Função **fila_cria**
  - Cria aloca a estrutura da fila
  - Inicializa a lista como sendo vazia

```
Fila* fila_cria (void)
{
   Fila* f = (Fila*) malloc(sizeof(Fila));
   f->ini = f->fim = NULL;
   return f;
}
```

Revisão: Implementação da fila com listas

- Função **fila_insere**
  - Insere novo elemento n no final da lista

```
void fila_insere (Fila* f, float v)
{
  Lista* n = (Lista*) malloc(sizeof(Lista));
  n->info = v;              /* armazena informação              */
  n->prox = NULL;           /* novo nó passa a ser o último     */
  if (f->fim != NULL)       /* verifica se lista não estava vazia */
    f->fim->prox = n;
  else                      /* fila estava vazia                */
    f->ini = n;
  f->fim = n;               /* fila aponta para novo elemento   */
}
```

Revisão: Implementação da fila com listas

- Função **fila_retira**
  - **Retira** o elemento do **início** da lista

```
float fila_retira (Fila* f)
{  Lista* t;
   float v;
   if (fila_vazia(f)) {  printf("Fila vazia.\n");
                         exit(1); }          /* aborta programa          */
   t = f->ini;
   v = t->info;
   f->ini = t->prox;
   if (f->ini == NULL)                       /* verifica se fila ficou vazia */
      f->fim = NULL;
   free(t);
   return v;
}
```

Revisão: Implementação da fila com listas

- Função **fila_libera**
  - Libera a fila depois de liberar todos os elementos da lista

```
void fila_libera (Fila* f)
{
  Lista* q = f->ini;
  while (q!=NULL) {
    Lista* t = q->prox;
    free(q);
    q = t;
  }
  free(f);
}
```

Fila dupla

- Características:
  - Fila na qual é possível:
    - **Inserir** novos elementos no **início** e no **fim**
    - **Retirar** elementos de **ambos os extremos**
  - **Simula**, dentro de uma mesma estrutura, **duas filas**, com os elementos em ordem inversa uma da outra.

Funções para filas duplas

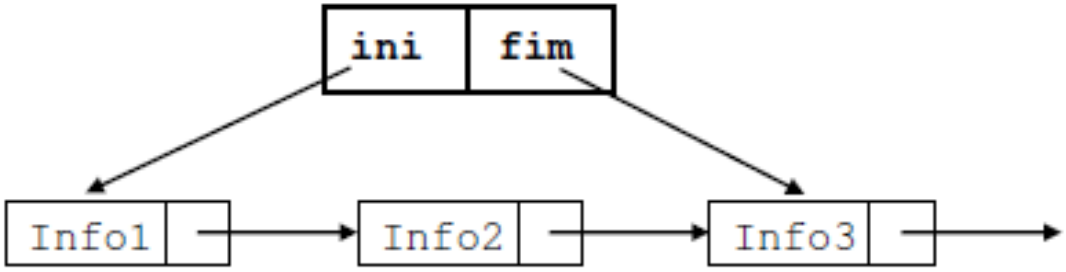
- Função **fila2_cria**
  - **Aloca** dinamicamente a estrutura da fila
  - **Inicializa** seus campos e retorna seu ponteiro
- Função **fila2_insere_fim** e função **fila2_retira_ini**
  - **Insere no fim** e **retira do início**, respectivamente, um valor na fila
- Função **fila2_insere_ini** e função **fila2_retira_fim**
  - **Insere no início** e **retira do fim**, respectivamente, um valor na fila
- Função **fila2_vazia**
  - Informa se a fila **está ou não vazia**
- Função **fila2_libera**
  - **Destrói** a fila, liberando toda a memória usada pela estrutura

Interfaces para filas duplas

```
typedef struct fila2 Fila2;

Fila2* fila2_cria (void);

void fila2_insere_ini (Fila2* f, float v);

void fila2_insere_fim (Fila2* f, float v);

float fila2_retira_ini (Fila2* f);

float fila2_retira_fim (Fila2* f);

int fila2_vazia (Fila2* f);

void fila2_libera (Fila2* f);
```

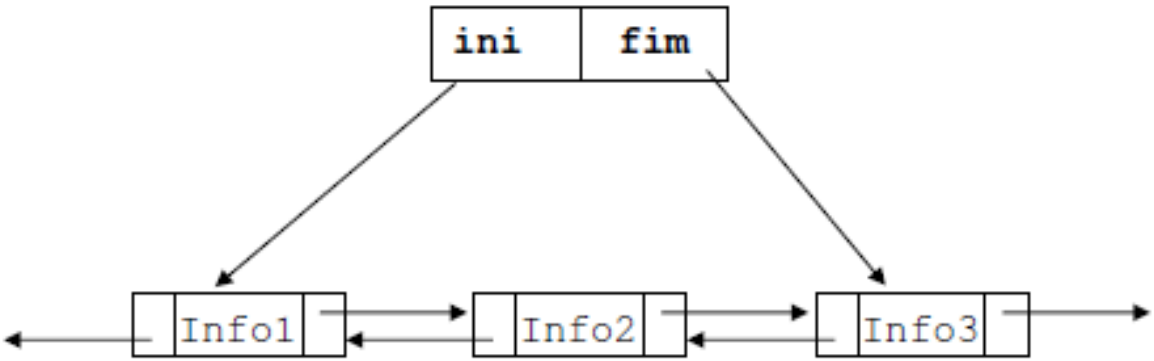
Implementação de fila dupla com lista duplamente encadeada

- Função para **retirar do fim** (na lista **SIMPLESMENTE** encadeada):
  - Não pode ser implementada de forma eficiente
  - Dado o ponteiro para o último elemento da lista, **não é possível acessar de forma eficiente o anterior**, que passaria a ser o último elemento



Implementação de fila dupla com lista duplamente encadeada

- Função para **retirar do fim** (na lista **DUPLAMENTE** encadeada):
  - dado o ponteiro de um **nó**, é possível **acessar ambos os elementos adjacentes**
  - **resolve** o problema de acessar o elemento **anterior ao último**



Implementação de fila dupla com lista duplamente encadeada

```
/* nó da lista para armazenar valores reais */
struct lista2 {
   float info;
   struct lista2* ant;
   struct lista2* prox;
};
typedef struct lista2 Lista2;

/* estrutura da fila */
struct fila2 {
   Lista2* ini;
   Lista2* fim;
};
```

Implementação de fila dupla com lista duplamente encadeada

- Função auxiliar: **insere no início**
  - Insere novo elemento n no inicio da lista duplamente encadeada

```
/* função auxiliar: insere no início */
static Lista2* ins2_ini (Lista2* ini, float v)
{
   Lista* p = (Lista2*) malloc(sizeof(Lista2));
   p->info = v;
   p->prox = ini;
   p->ant = NULL;
   if (ini != NULL)            /* verifica se lista não estava vazia */
      ini->ant = p;
   return p;
}
```

Implementação de fila dupla com lista duplamente encadeada

- Função auxiliar: **insere no fim**
  - Insere novo elemento n no fim da lista duplamente encadeada

```
/* função auxiliar: insere no fim */
static Lista2* ins2_fim (Lista2* fim, float v)
{
  Lista2* p = (Lista2*) malloc(sizeof(Lista2));
  p->info = v;
  p->prox = NULL;
  p->ant = fim;
  if (fim != NULL)          /* verifica se lista não estava vazia */
    fim->prox = p;
  return p;
}
```

Implementação de fila dupla com lista duplamente encadeada

- Função auxiliar: **retira do início**
  - Retira elemento do inicio da lista duplamente encadeada

```
/* função auxiliar: retira do início */
static Lista2* ret2_ini (Lista2* ini)
{
   Lista2* p = ini->prox;
   if (p != NULL)    /* verifica se lista não ficou vazia */
      p->ant = NULL;
   free(ini);
   return p;
}
```

Implementação de fila dupla com lista duplamente encadeada

- Função auxiliar: **retira do fim**
  - Retira elemento do fim da lista duplamente encadeada

```
/* função auxiliar: retira do fim */
static Lista2* ret2_fim (Lista2* fim)
{
   Lista2* p = fim->ant;
   if (p != NULL)                /* verifica se lista não ficou vazia */
      p->prox = NULL;
   free(fim);
   return p;
}
```

Implementação de fila dupla com lista duplamente encadeada

- Funções **fila2_insere_ini** e **fila2_insere_fim**

```
void fila2_insere_ini (Fila2* f, float v) {
   f->ini = ins2_ini(f->ini,v);
   if (f->fim==NULL)          /* fila antes vazia? */
      f->fim = f->ini;
}


void fila2_insere_fim (Fila2* f, float v) {
   f->fim = ins2_fim(f->fim,v);
   if (f->ini==NULL)          /* fila antes vazia? */
      f->ini = f->fim;
}
```

Implementação de fila dupla com lista duplamente encadeada

- Função **fila2_retira_ini**

```
float fila2_retira_ini (Fila2* f) {
   float v;
   if (fila2_vazia(f)) {
      printf("Fila vazia.\n");
      exit(1);                    /* aborta programa */
   }
   v = f->ini->info;
   f->ini = ret2_ini(f->ini);
   if (f->ini == NULL)            /* fila ficou vazia? */
      f->fim = NULL;
   return v;
}
```

Implementação de fila dupla com lista duplamente encadeada

- Função **fila2_retira_fim**

```
float fila2_retira_fim (Fila2* f) {
   float v;
   if (vazia(f)) {
      printf("Fila vazia.\n");
      exit(1);                    /* aborta programa */
   }
   v = f->fim->info;
   f->fim = ret2_fim(f->fim);
   if (f->fim == NULL)            /* fila ficou vazia? */
      f->ini = NULL;
   return v;
}
```

Implementação de fila dupla com lista duplamente encadeada

- Implemente as funções de manipulação de filas duplas apresentadas em aula de forma que elas sejam aplicadas no gerenciamento de ordens de serviço de uma empresa (utilize um número inteiro para representar o código de cada ordem de serviço).
- Implemente uma função que, ao receber o código de uma ordem de serviço, retorna o número total de ordens que estão à sua frente ou -1 caso ela não tenha sido encontrada.
- Implemente uma função que, ao receber o código de uma ordem de serviço, imprima os códigos (em ordem de atendimento previsto) de todas as demais ordens que estão enfileradas. Caso a ordem informada não seja encontrada, informe o usuário sobre esta situação.

Referências

- CELES, Waldemar; CERQUEIRA, Renato; RANGEL, José Lucas. **Introdução a Estrutura de Dados.** Editora Campus, 2004.
- Material didático do Departamento de Informática da PUC-Rio (2014).

Leitura complementar

- Capítulo 12 – Filas. In: CELES, Waldemar; CERQUEIRA, Renato; RANGEL, José Lucas. **Introdução a Estrutura de Dados.** Editora Campus, 2004.