Aula 4: Filas

Professor(a): Virgínia Fernandes Mota
http://www.dcc.ufmg.br/∼virginiaferm

ALGORITMOS E ESTRUTURAS DE DADOS - SETOR DE INFORMÁTICA

Filas

- Uma estrutura de dados bastante usada em computação é a **fila**.
- Enquanto na pilha "o último que entra é o primeiro que sai"na fila tempos "o primeiro que entra é o primeiro que sai".
- FIFO - *first in first out*.
- Exemplo de uso: Fila de impressão.

Filas

- Consideraremos duas estratégias para implementação: com **vetor** ou com **lista encadeada**.
- Uma estrutura de fila pode ser composta pelas seguintes operações:
  - criar uma fila vazia;
  - inserir um elemento no fim;
  - retirar o elemento do início;
  - verificar se a fila está vazia;
  - liberar a fila.

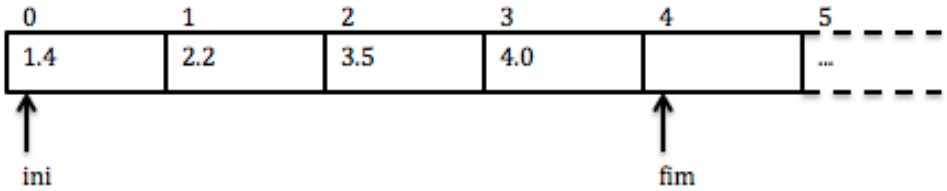
- O arquivo *fila.h* pode conter o seguinte código.

```
1 typedef struct fila Fila;
2
3 Fila *fila_cria();
4 void fila_insere(Fila *f, float v);
5 float fila_retira(Fila *f);
6 int fila_vazia(Fila *f);
7 void fila_libera(Fila *f);
```

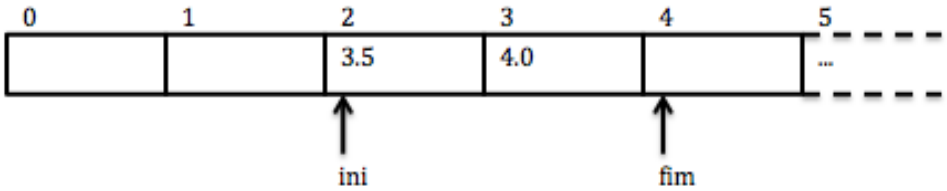
- O tipo fila criado pode ser implementado usando **vetor** ou **lista encadeada**

Implementação de fila com vetor

- Assim como na pilha, devemos fixar o número máximo N de elementos na fila.



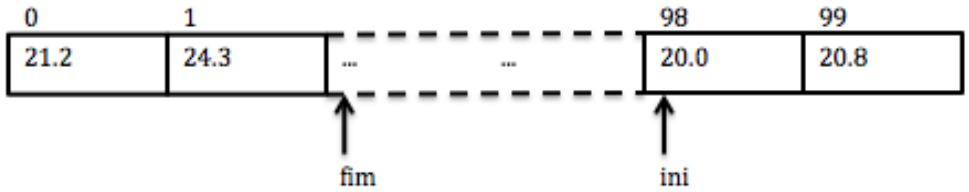
Fila após inserção de quatro novos elementos



Fila após retirar dois elementos

Implementação de fila com vetor

- É fácil observar que, em um dado instante, a parte ocupada do vetor pode chegar à última posição.
- Para reaproveitar as primeiras posições livres do vetor sem implementar uma re-arrumação complicada dos elementos, podemos incrementar as posições do vetor de forma "circular".



Fila com incremento circular: Se o último elemento da fila ocupa a última posição do vetor, inserimos os novos elementos a partir do início do vetor.

Implementação de fila com vetor

- Podemos definir uma função auxiliar responsável por incrementar o valor de um índice em uma unidade.
- Com o uso do operado módulo, em geral optamos por dispensar a função auxiliar.

```
1 static int incr (int i){
2       if (i == N - 1)
3             return 0;
4       else
5             return i+1;
6 }
```

```
1 static int incr (int i){
2       return (i+1) % N
3 }
```

Implementação de fila com vetor

- Podemos declarar o tipo fila como sendo uma estrutura com três componentes:
  - um vetor *vet* de tamanho N;
  - um inteiro n que representa o número de elementos armazenados na fila; e,
  - um índice *ini* para o início da fila.
- *ini* marca a posição do próximo elemento a ser retirado da fila e *fim* marca a posição (vazia) em que será inserido o próximo elemento.
- Podemos calcular o índice *fim* incrementando *ini* de n unidades:
  $fim = (ini + n)\%N$

Implementação de fila com vetor

- A estrutura de fila é então:

```
1 #define N 100
2
3 struct fila{
4    int n;
5    int ini;
6    float vet[N];
7 }
```

Implementação de fila com vetor - Função Cria

```
1 Fila *fila_cria(){
2   Fila *f = (Fila *) malloc (sizeof(Fila));
3   f->n = 0;
4   f->ini = 0;
5   return f;
6 }
```

Implementação de fila com vetor - Função Insere

```c
void fila_insere(Fila *f, float v){
  int fim;
  if (f->n == N){
    printf("Capacidade da fila estourou \n");
    exit(1);
  }
  fim = (f->ini + f->n) % N;
  f->vet[fim] = v;
  f->n++;
}
```

Implementação de fila com vetor - Função Retira

```
1  float fila_retira(Fila *f){
2    float v;
3    if (fila_vazia(f)){
4      printf("Fila vazia \n");
5      exit(1);
6    }
7    v = f->vet[f->ini];
8    f->ini = (f->ini + 1) % N;
9    f->n--;
10   return v;
11 }
```

Implementação de fila com vetor - Função Vazia

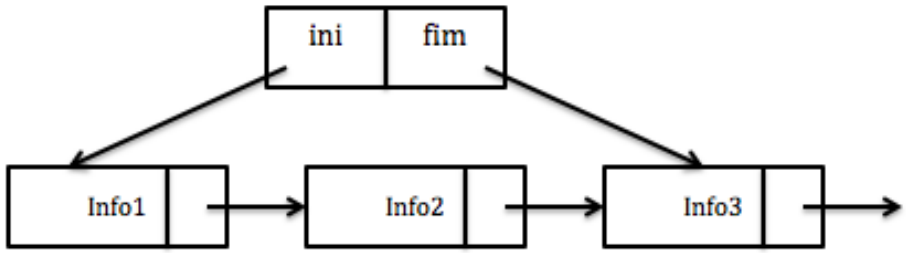
```
1 int fila_vazia(Fila *f){
2   return (f->n == 0);
3 }
```

Implementação de fila com vetor - Função Libera

```
1 void fila_libera(Fila *f){
2   free(f);
3 }
```

Implementação de fila com lista

- Vamos agora ver como implementar uma fila usando lista encadeada!



Implementação de fila com lista

- A estrutura de fila usando lista encadeada é então:

```
1 struct lista{
2   float info;
3   struct lista *prox;
4 };
5
6 struct fila{
7   Lista *ini;
8   Lista *fim;
9 }
```

Implementação de fila com lista - Função Cria

```
1 Fila *fila_cria(){
2   Fila *f = (Fila *) malloc (sizeof(Fila));
3   f->ini = f->fim = NULL;
4   return f;
5 }
```

Implementação de fila com lista - Função Insere

```
1  void fila_insere(Fila *f, float v){
2    Lista *n = (Lista *) malloc (sizeof(Lista));
3    n->info = v;
4    n->prox = NULL; //novo nó passa a ser o último
5    if (f->fim != NULL) //verifica se a fila não está vazia
6      f->fim->prox = n;
7    else
8      f->ini = n;
9    f->fim = n;
10 }
```

Implementação de fila com lista - Função Retira

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

Implementação de fila com lista - Função Vazia

```
1 int fila_vazia(Fila *f){
2   return (f->ini == NULL);
3 }
```

Implementação de fila com lista - Função Libera

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

Implementação de fila - Impressão

```c
//imprime : versão com vetor
void fila_imprime_vet(Fila *f){
  int i;
  for (i = 0; i < f->n; i++)
    printf("%f \n", f->vet[(f->ini + i)%N]);
}
```

```c
//imprime : versão com lista
void fila_imprime_lista(Fila *f){
  Lista *q;
  for (q = f->ini; q != NULL; q = q-> prox)
    printf("%f\n", q->info);
}
```

Implementação de fila

```c
#include <stdio.h>
#include "fila.h"

int main(){
   Fila *f = fila_cria();
   fila_insere(f, 20.0);
   fila_insere(f, 42.0);
   fila_insere(f, 13.0);
   printf("Primeiro elemento: %f", fila_retira(f));
   fila_imprime(f);
   fila_libera(f);
   return 0;
}
```

Fila dupla

- A estrutura de dados que chamamos de *fila dupla* consiste em uma fila na qual é possível inserir novos elementos nas duas extremidades (início e fim).
- Da mesma forma, na *fila dupla* é possível retirar elementos dos dois extremos.
- Uma estrutura de fila dupla pode ser composta pelas seguintes operações:
  - criar uma estrutura de fila dupla;
  - inserir um novo elemento no início;
  - inserir um novo elemento no fim;
  - retirar o elemento do início;
  - retirar o elemento do fim;
  - verificar se a fila está vazia;
  - liberar a fila.

Fila dupla

- O arquivo *fila2.h* pode conter o seguinte código.

```
1 typedef struct fila2 Fila2;
2
3 Fila2 *fila2_cria();
4 void fila2_insere_ini(Fila2 *f, float v);
5 void fila2_insere_fim(Fila2 *f, float v);
6 float fila2_retira_ini(Fila2 *f);
7 float fila2_retira_fim(Fila2 *f);
8 int fila2_vazia(Fila2 *f);
9 void fila2_libera(Fila2 *f);
```

Fila dupla com vetor

- Vamos analisar as duas novas funções: *insere_ini* e *retira_fim*.

```
1  void fila2_insere_ini(Fila *f, float v){
2    int prec;
3    if (f->n == N){
4      printf("Capacidade da fila estourou. \n");
5      exit(1);
6    }
7    //insere elemento na posição precedente ao início
8    prec = (f->ini - 1 + N) % N;
9    f->vet[prec] = v;
10   f->ini = prec;
11   f->n++;
12 }
```

Fila dupla com vetor

```
1  float fila2_retira_fim(Fila *f){
2    int ult;
3    float v;
4    if (fila2_vazia(f)){
5      printf("Fila vazia. \n");
6      exit(1);
7    }
8    ult = (f->ini + f->n - 1) % N;
9    v = f->vet[ult];
10   f->n--;
11   return v;
12 }
```

Exercícios

1. Implementar as funções/procedimentos apresentados em sala para a manipulação de filas. Crie *filas.c* e *filas.h* para a manipulação das filas (com vetor e com lista) e *main.c* para testes. Crie um *makefile* para compilar o código.
2. Implementar as funções/procedimentos apresentados em sala para a manipulação de filas duplas. Crie *filas2.c* e *filas2.h* para a manipulação das filas (com vetor e com lista) e *main.c* para testes. Crie um *makefile* para compilar o código.

Exercícios

3. Vamos montar um estacionamento! Compramos um terreno que possui uma entrada e uma saída. Quando chega um novo carro, este é estacionado no terreno, um atrás do outro. Quando um carro precisa sair, os carros do terreno são retirados pela saída, dão uma volta na quadra e são colocados no final da fila pela entrada do estacionamento. Faça um programa que inclua carros no estacionamento informando o número da placa e retire carros usando o identificador (placa). Depois de ter informado a placa, exiba o estado do estacionamento.

Na próxima aula...

Prova