Métodos Computacionais

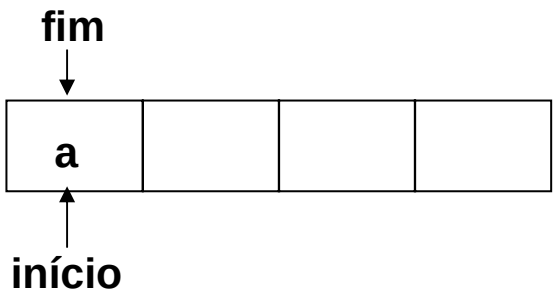
Fila

Definição de Fila

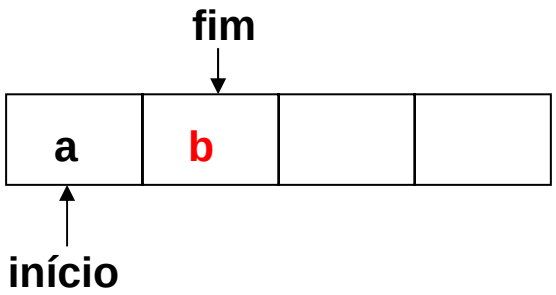
- Fila é uma estrutura de dados dinâmica onde:
  - Inserção de elementos se dá no final e a remoção no início
  - O primeiro elemento que entra é o primeiro que sai (FIFO)
  - Ex de uso : fila de impressão

Funcionamento da Fila

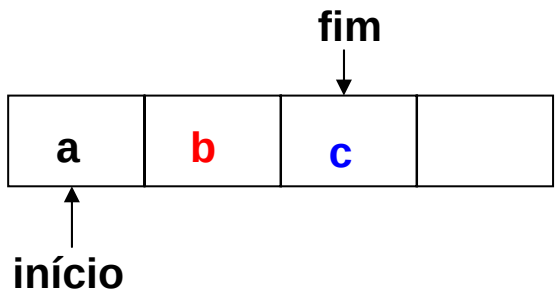
1- insere(a)



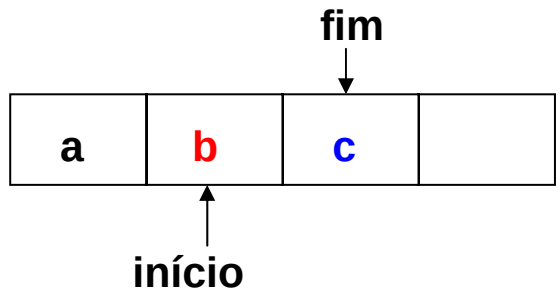
2- insere(b)



3- insere(c)



4- retira( )



Interface do tipo Fila

- Criar uma fila vazia
- Inserir um elemento no fim
- Retirar o elemento do início
- Verificar se a fila está vazia
- Liberar a fila

- Em C

```
/* fila.h */
typedef struct fila Fila;
Fila* fila_cria();
void fila_insere(Fila* f,float v);
float fila_retira(Fila* f);
int fila_vazia(Fila* f);
void fila_libera(Fila* f);
```

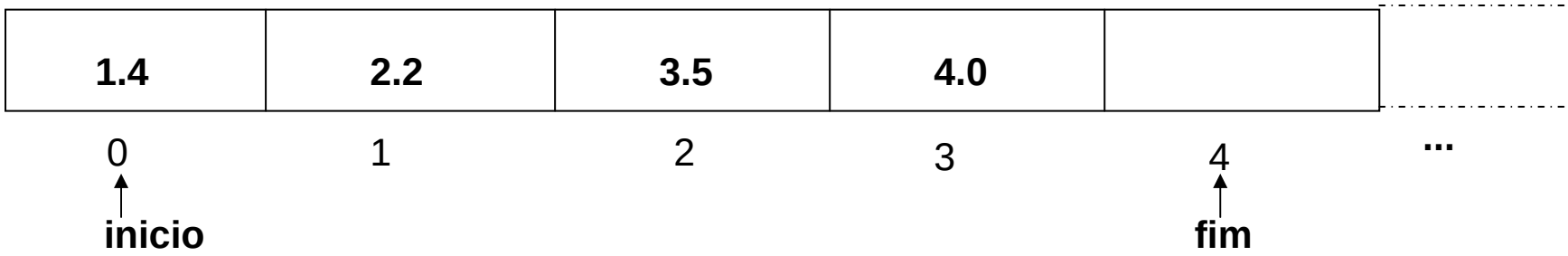
Implementando Fila com Vetor

- Estrutura que representa fila deve ser composta pelo número de elementos da fila (**n**), um vetor que armazena os elementos (**vet**) e o índice da posição do vetor que armazena o primeiro elemento da fila (**ini**)

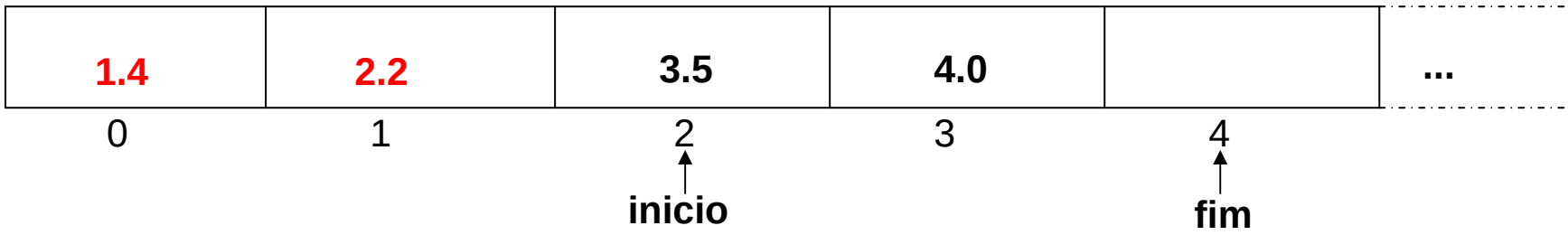
```
#define N 100
struct fila {
   int n;
   int ini;
   float vet[N];
};
```

Implementando Fila com Vetor

Fila após a inserção de quatro novos elementos



Fila após a remoção de dois elementos



- A parte ocupada do vetor pode chegar à última posição e ainda haver espaço antes do início da fila.

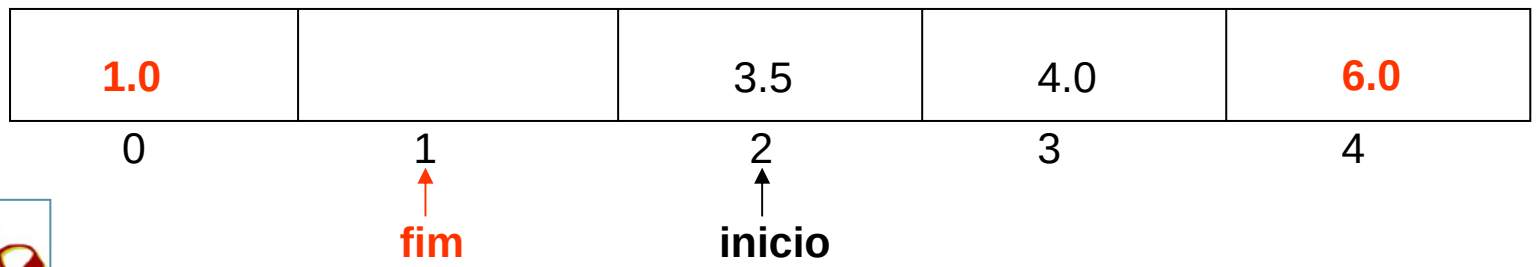
Implementando Fila com Vetor

- **Solução:** usar uma estratégia circular (se o último elemento da fila ocupa a última posição do vetor, insere-se novos elementos a partir do início do vetor, caso haja espaço)
  - Utilização do operador módulo (%)

O índice para a posição livre após o último elemento da fila pode ser calculado como:

`fim = (ini+n)%N` N → tamanho do vetor

Fila após a inserção de dois novos elementos



Implementando Fila com Vetor

- Função de Criação

```
typedef struct fila Fila;
Fila* fila_cria() {
  Fila* f = (Fila*) malloc(sizeof(Fila));
  f->n = 0;
  f->ini = 0;
  return f;
}
```

Inicializa número de elementos e índice de início com 0

Implementando Fila com Vetor

- Função de Inserção

Verifica se tem espaço disponível

```
void fila_insere (Fila* f, float v) {
   int fim;
   if (f->n == N) {
      printf("Capacidade da fila estourou.\n");
      exit(1);
   }
   /* insere elemento na próxima posição livre */

   fim = (f->ini + f->n)%N;
   f->vet[fim] = v;
   f->n++;
}
```

fim é o índice para a próxima posição livre após o último elemento da fila

Implementando Fila com Vetor

- Função de Remoção

```
float fila_retira (Fila* f) {
   float v;
   if (fila_vazia(f)) {
      printf("Fila vazia.\n");
      exit(1);
   }
   /*Retira elemento do inicio */
   v = f->vet[f->ini];
   f->ini = (f->ini+1)%N;
   f->n--;
   return v;
}
```

Verifica se a fila está vazia

Equivalente a:

```
if (f->ini == N-1)
     f->ini = 0;
else
      f->ini++;
```

Incremento circular do índice de início da Fila

Implementando Fila com Vetor

- Função que testa se fila está vazia

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

- Função que libera memória alocada para a fila

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

Implementando Fila com Vetor

- Outras Funções Utilitárias

```
/*Função que informa o primeiro elemento da fila */
float fila_primeiro (Fila* f) {
  return (f->vet[f->ini]);
}
```

```
/* Função que imprime os elementos da fila
void fila_imprime (Fila* f) {
   int i;
   for (i=0; i<f->n; i++)
       printf("%f\n", f->vet[(f->ini+i)%N]);
}
```

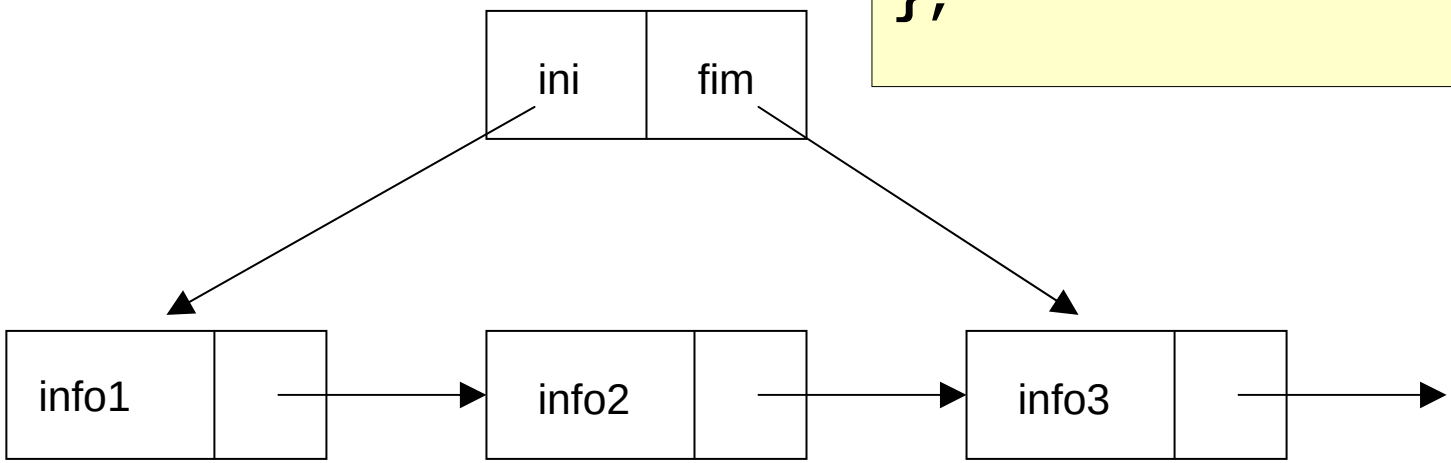
Incremento circular do índice de início da Fila

Implementando Fila com Lista

- Estrutura que representa fila deve ser composta por 2 ponteiros para a lista que armazena os elementos.
  - Um aponta para o primeiro elemento e o outro para o último elemento

```
typedef struct lista{
   float info;
   struct lista *prox;
} Lista;

struct fila {
   Lista *ini;
   Lista *fim
};
```



Implementando Fila com Lista

- Função de Criação

```
typedef struct fila Fila;

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

ini e fim são inicializados com NULL

Implementando Fila com Lista

- Função de Inserção

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

   f->fim = novo;
}
```

Se a fila estava vazia novo elemento passa a ser o começo da fila

Novo elemento sempre é o último da fila

Implementando Fila com Lista

- Função de Remoção

```
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)
       f->fim = NULL;
   free(t);
   return v;
}
```

Se a fila ficou vazia, deve-se atualizar ponteiro para o fim

Implementando Fila com Lista

- Função que testa se a fila está vazia

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

- Função que libera memória alocada para a fila

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

Deve-se liberar todos os elementos da lista primeiro

Implementando Fila com Lista

- Outras Funções Utilitárias

```
/* Função que informa o primeiro elemento da fila
   */
float fila_primeiro (Fila* f) {
  return (f->ini->info);
}
```

```
/* Função que imprime os elementos da fila
void fila_imprime (Fila* f) {
   Lista* q;
   for (q=f->ini; q!=NULL; q =q->prox)
         printf("%f\n", q->info);
}
```