Anhanguera

Aqui o seu esforço ganha força.

Filas com alocação dinâmica

Prof. Esp. Rodrigo Hentz

Filas

- Estudamos filas anteriormente partindo-se do princípio que a fila era implementada utilizando-se um vetor.
- Depois estudamos a alocação dinâmica de memória e como era possível trabalhar com uma lista encadeada com alocação dinâmica de memória.
- Hoje retornaremos ao assunto de filas, mas utilizando alocação dinâmica de memória para sua implementação.

Filas

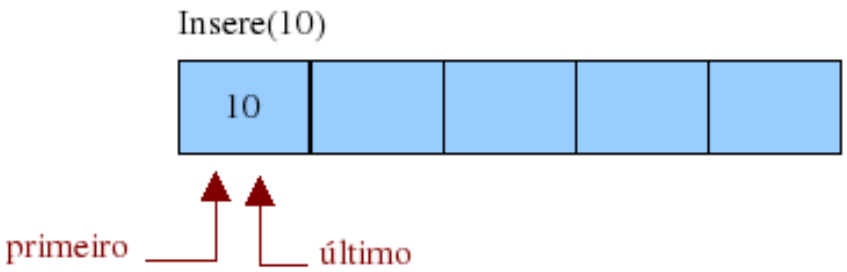
- Uma fila é simplesmente uma lista linear de informações, que é acessada na ordem primeiro a entrar, primeiro a sair, sendo chamada, algumas vezes, de **FIFO** (first in, first out).
- Isso é, o primeiro item colocado na fila é o primeiro a ser retirado.
- Também é conhecido como **QUEUE**.

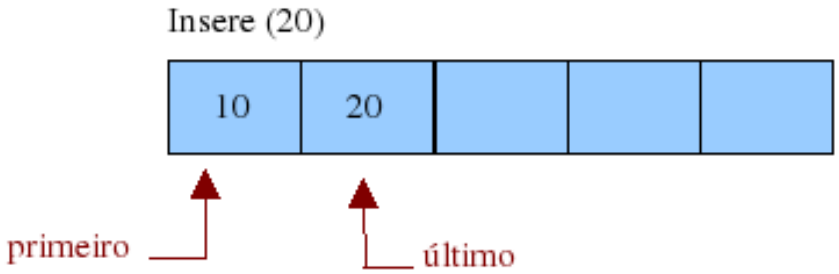
Filas

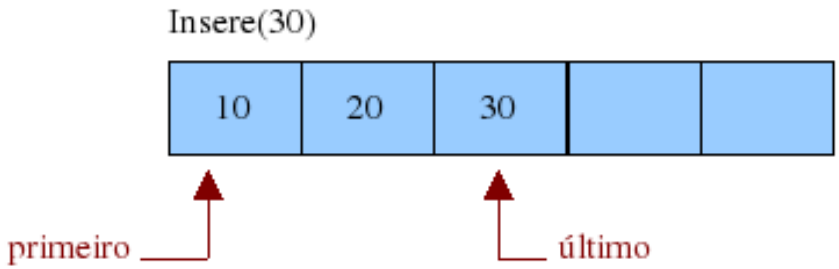
- Uma fila possui duas funções básicas:
  - **ENQUEUE**, que adiciona um elemento ao final da fila
  - **DEQUEUE**, que remove o elemento no início da fila.
- A operação **DEQUEUE** só pode ser aplicada se a fila não estiver vazia, causando um erro de fila vazia se esta operação for realizada nesta situação.

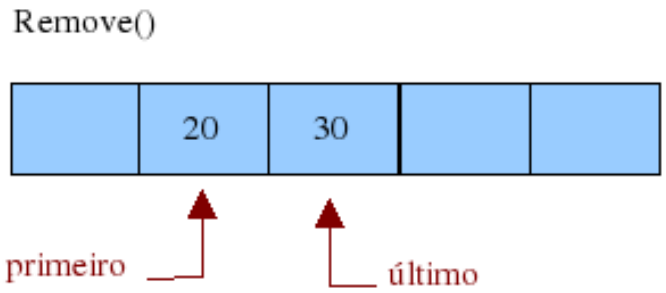
Anhanguera

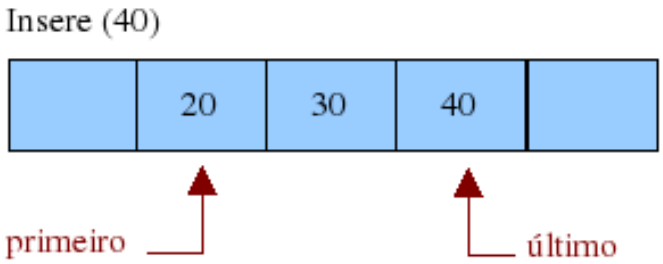
Filas

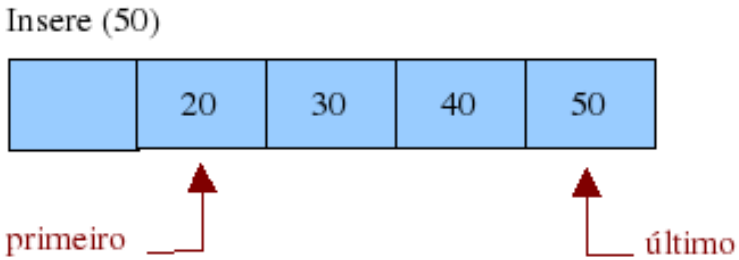












Filas com alocação dinâmica

```
typedef struct no
{
    int info;
    struct no* next;
} sNo;

typedef struct fila
{
    sNo* first;
    sNo* last;
} sFila;
```

Filas com alocação dinâmica

```
void criarFila(sFila* fila)
{
     fila->first = NULL;
     fila->last = NULL;
     printf("\nFila criada.");
}
```

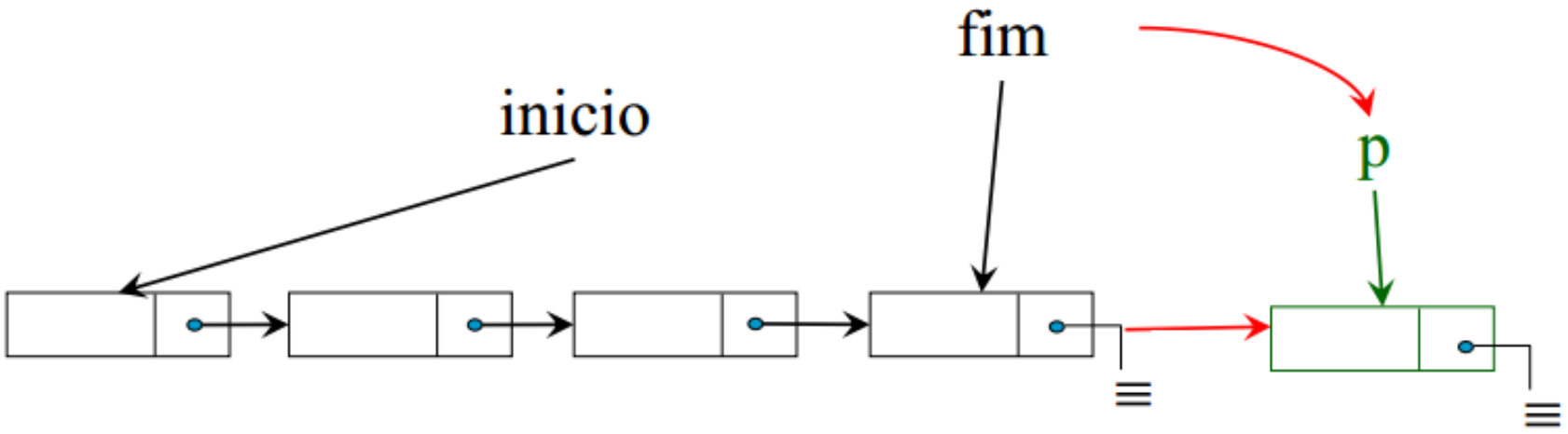
```
sNo* criarNo(int valor)
{
   sNo* p = (sNo*)malloc(sizeof(sNo));
   p->info = valor;
   p->next = NULL;
   return p;
}
```

Filas com alocação dinâmica

```
int vazia(sFila* fila)
{
    return fila->first == NULL;
}
```

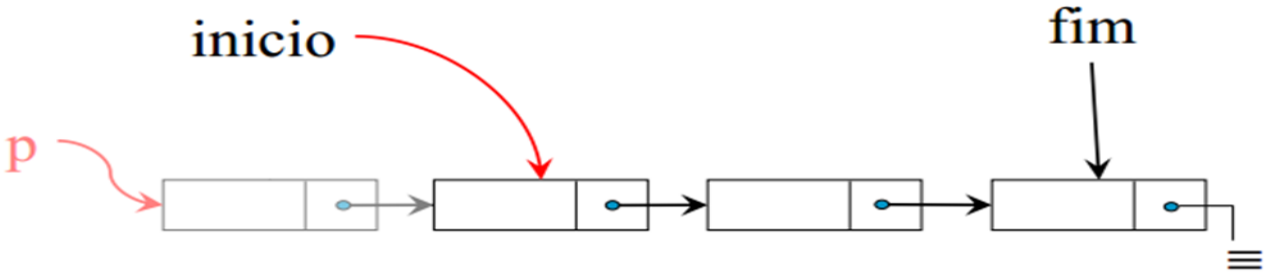
Filas com alocação dinâmica

```
void enqueue(sFila* fila, int valor)
{
   sNo* p = criarNo(valor);
   if (vazia(fila)) fila->first = p;
   else fila->last->next = p;
   fila->last = p;
   printf("\nValor inserido com sucesso: %d.", valor);
}
```



Filas com alocação dinâmica

```
int dequeue(sFila* fila)
{
     sNo* p; int num;
     if (vazia(fila)){
       printf("\nFila esta vazia.");
       return NULL;
     }
     p = fila->first;
     num = p->info;
     fila->first = p->next;
     if (fila->first == NULL) fila->last = NULL;
     free(p);
     return num;
}
```



Filas com alocação dinâmica

```
void imprimir(sFila* fila)
{
     sNo* p;
     if(vazia(fila))
       printf ("\nFila esta vazia.");
     else
          for (p = fila->first; p != NULL; p = p->next)
            printf("\nInfo = %d", p->info);
}
```

Filas com alocação dinâmica

```
void destruir(sFila* fila){
     sNo* p; sNo* d;
     if(!vazia(fila))
     {
          p = fila->first;
          while (p->next != NULL){
               d = p;
               p = p->next;
               free(d);
          }
          printf ("\nFila destruida.");
          criarFila(fila);
     } else printf ("\nFila vazia.");
}
```

Filas com alocação dinâmica

```
int main(int argc, char *argv[]) {
    int opcao, num; sFila fila; sFila* pfila = &fila;
    do
    {
        printf("\n");
        printf("1 - Inicializar fila\n");
        printf("2 - Enfileirar\n");
        printf("3 - Desenfileirar\n");
        printf("4 - Imprimir\n");
        printf("5 - Destruir\n");
        printf("0 - Sair\n");
        printf("\nEntre com a opcao: "); scanf("%d", &opcao); printf("\n");
        switch (opcao)
        {
            case 1:
                criarFila(pfila);
                break;
            case 2:
                printf ("\nEntre com o numero para enfileirar: "); scanf ("%d", &num); printf("\n");
                enqueue(pfila, num);
                break;
```

Filas com alocação dinâmica

```
            case 3:
                num = dequeue(pfila);
                if (num != NULL) printf("\nNumero desemfileirado %d.", num); printf("\n");
                break;
            case 4:
                printf("\nImpressao da fila:");
                imprimir(pfila);
                break;
            case 5:
                destruir(pfila);
                break;
        }
        fflush(stdin);
    } while (opcao != 0);
    return 0;
}
```

Finalização