Marisangila Alves, MSc
marisangila.alves@catolicasc.org.br
marisangila.com.br

Católica de Santa Catarina

2025/2

Estrutura de Dados

Estrutura de Dados
Fila

Sumário

1 Definição
2 Operações
3 Fila Sequencial
4 Fila Circular
5 Fila Encadeada

Definição

Fila

> *Um fila ou queue é simplesmente uma lista linear de informações, que é acessada na ordem primeiro a entrar, primeiro a sair, sendo chamada, alguma vezes de FIFO First In, First Out. Isto é, o primeiro item colocado na fila é o primeiro a ser retirado.*
>
> (Schildt, 1996)

Atenção

Essa é a única forma de armazenar e recuperar em uma fila; não é permitido acesso randômico a nenhum item específico.

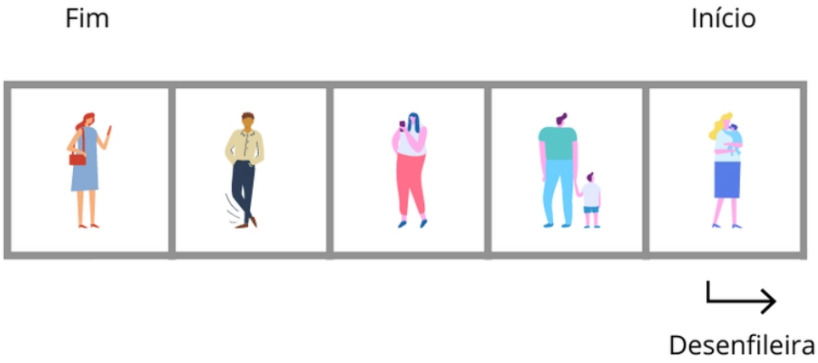
Filas são muito comuns no nosso dia-a-dia. Por exemplo, filas físicas em um caixa em restaurante.

- **Fila de impressão**: cada documento enviado à impressora é colocado na fila e processado na ordem de chegada (FIFO).
- **Atendimento em banco ou hospital**: clientes ou pacientes aguardam atendimento por ordem de chegada, com ou sem prioridade.
- **Sistema operacional**: utiliza filas para escalonamento de processos, gerenciamento de uso da memória, leitura e escrita em disco.

Exemplos II

- **Fila de pedidos em aplicativos de entrega**: os pedidos são enfileirados e atendidos conforme a ordem de chegada no sistema.
- **Buffer de teclado (em sistemas operacionais)**: as teclas digitadas rapidamente são armazenadas em uma fila até serem processadas.
- **Atendimento de suporte técnico**: os clientes entram em fila e são atendidos um por vez, seguindo a ordem de solicitação.
- **Distriuição de eventos**: como nos diagramas PERT e Gantt.

Exemplos III



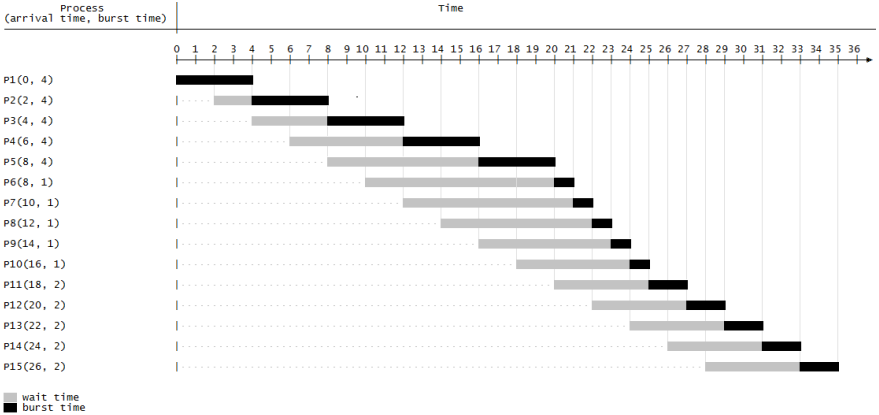


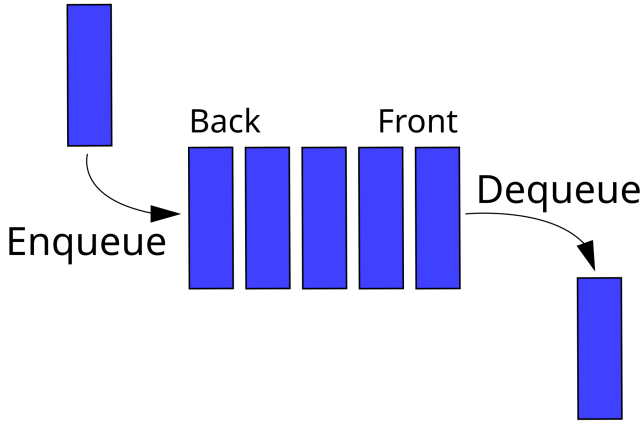
Figura 1: Exemplo de agendamento de tarefas em um sistema operacional.

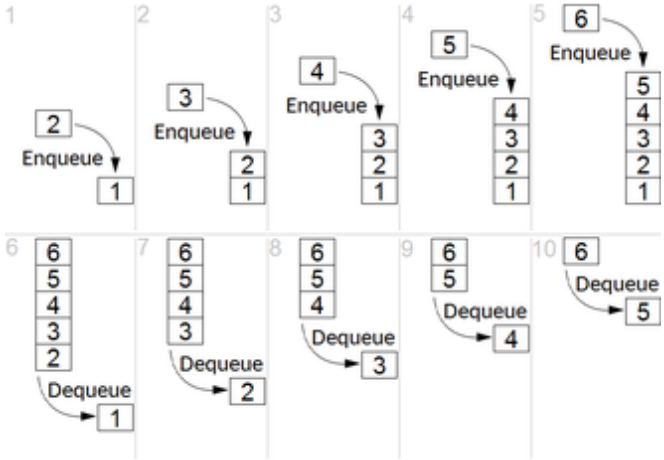
Operações

O funcionamento de uma fila se resume em duas funções ou operações:

- `qstore()`: Coloca um item no final da fila.
- `qretrieve()`: remove o primeiro item da fila e devolve seu valor.

Em uma operação que remove um item da fila, o item será destruído, se for necessário, o item deve ser armazenado.


Back
Front
Dequeue
Enqueue


1
2
Enqueue
1
2
3
Enqueue
2
1
3
4
Enqueue
3
2
1
4
5
Enqueue
4
3
2
1
5
6
Enqueue
5
4
3
2
1
6
6
5
4
3
2
Dequeue
1
7
6
5
4
3
Dequeue
2
8
6
5
4
Dequeue
3
9
6
5
Dequeue
4
10
6
Dequeue
5

Ação	**Conteúdo da fila**
qstore(A)	A
qstore(B)	A B
qstore(C)	A B C
qretrieve() devolve A	B C
qstore(D)	B C D
qretrieve() devolve B	C D
qretrieve() devolve C	D

Tabela 1: Operações em uma fila e seus estados

- `inicializar_fila(f)`: Cria uma fila vazia.
- `fila_vazia(f)`: Testa se a fila `f` está vazia.
- `fila_cheia(f)`: Testa se a fila `f` está cheia.
- `mostrar_fila(f)`: Função auxiliar para visualização dos valores de `f` na tela.

Fila Sequencial

```c
typedef struct
{
    int inicio, fim;
    int dados[CAPACIDADE_MAXIMA];
}Fila;
```

Código 1: Estrutura da fila.

```c
#define CAPACIDADE_MAXIMA 10
typedef struct
{
    int inicio, fim;
    int dados[CAPACIDADE_MAXIMA];
}Fila;
int inserir(int dado, Fila *fila);
int remover(Fila *fila);
Fila* inicializar();
int cheia(Fila *fila);
int vazia(Fila *fila);
void mostrar(Fila *fila);
```

Código 2: Exemplo de protótipos das funções.

```c
Fila* inicializar(){
    Fila *fila =  malloc(sizeof(Fila));
    fila->inicio = 0;
    fila->fim = 0;
    return fila;
}
```

Código 3: Exemplo de inicialização.

```c
int inserir(int dado, Fila *fila){
    if(cheia(fila))
        return 0;
    fila->dados[fila->fim] = dado;
    fila->fim++;
    return 1;
}
```

Código 4: Exemplo de inserção.

```c
int remover(Fila *fila){
    if(vazia(fila))
        return 0;
    int dado = fila->dados[fila->inicio];
    fila->inicio++;
    return dado;
}
```

Código 5: Exemplo de remoção.

```c
int cheia(Fila *fila){
    return fila->fim == CAPACIDADE_MAXIMA;
}
```

Código 6: Exemplo de verificação de fila cheia.

```c
int vazia(Fila *fila){
    return fila->inicio == fila->fim;
}
```

Código 7: Exemplo de verificação de fila vazia.

```c
int main(void)
{
    Fila *fila = inicializar();
    inserir(20, fila);
    mostrar(fila);
    inserir(10, fila);
    mostrar(fila);
    inserir(50, fila);
    mostrar(fila);
    remover(fila);
    mostrar(fila);
    inserir(5, fila);
    mostrar(fila);
    return 0;
}
```

Código 8: Exemplo de utilização da fila.

Considerações sobre a Fila Sequencial

- Com o tempo, o espaço disponível no vetor vai se esgotando, mesmo com remoções.
- Uma solução simples seria deslocar todos os elementos para o início do vetor após cada remoção.
- No entanto, essa abordagem é ineficiente, especialmente quando a fila armazena muitos elementos.

```c
int remover_deslocar(Fila *fila){
    if(vazia(fila))
        return 0;
    int dado = fila->dados[fila->inicio];
    for (int i = 0; i < fila->fim; i++)
        fila->dados[i] = fila->dados[i+1];
    fila->fim--;
    return dado;
}
```

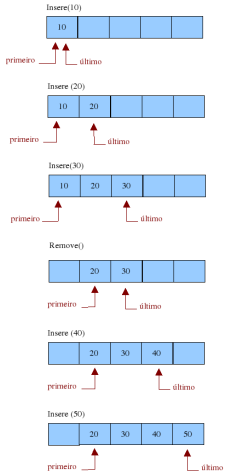
Código 9: Remoção com deslocamento dos elementos para o início do vetor.

Fila Circular

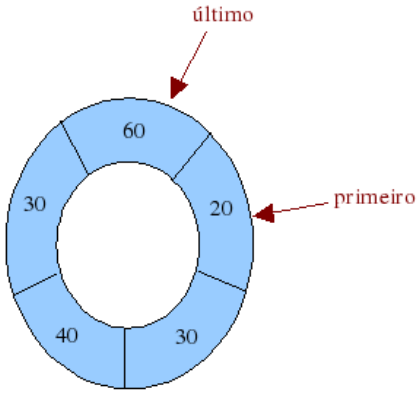
Fila Circular: Solução para Fila Sequencial I

- A fila circular utiliza um vetor fixo para armazenar elementos, mas conecta o final ao início formando um "círculo".
- Dois ponteiros são usados: **início** (head) e **fim** (tail), que "dão a volta"no vetor.
- Com isso, o espaço é reutilizado eficientemente sem necessidade de deslocar elementos.

Fila Circular: Solução para Fila Sequencial II



Fila Circular I



Fila Encadeada

- **Fila Sequencial com `realloc`**
  - Requer alocação de blocos contíguos de memória, o que pode ser difícil com o tempo devido à fragmentação.
  - Inserções e remoções podem exigir realocação ou deslocamento de elementos para manter a sequência.
  - Capacidade fixa ou necessidade de redimensionamento complexo (*realloc*), que pode ser custoso em tempo.
  - Pode desperdiçar espaço se a fila não estiver cheia (espaço pré-alocado não usado).
- **Fila Encadeada**
  - Usa nós dinamicamente alocados ligados por ponteiros, aproveitando melhor a memória disponível.
  - Inserções e remoções são mais eficientes, pois não exigem deslocamento de elementos.
  - Não necessita de bloco contíguo de memória.

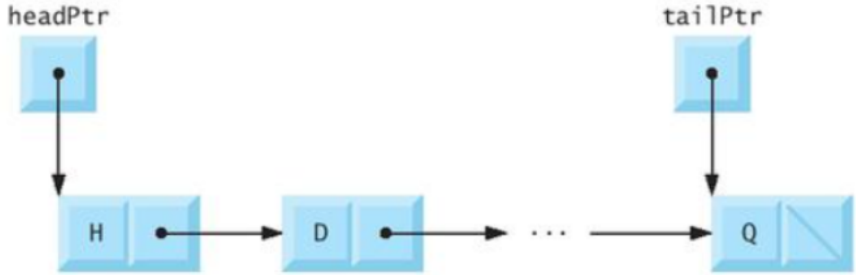


Figura 2: Operação enqueue (Deitel; Deitel, 2011).

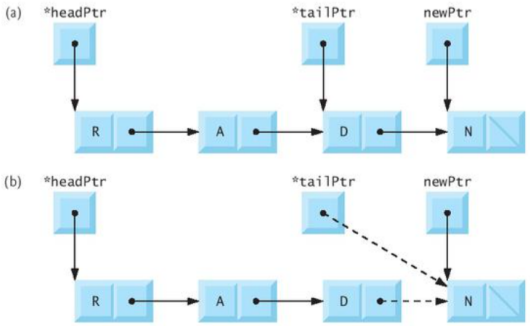


Figura 3: Operação enqueue (Deitel; Deitel, 2011).

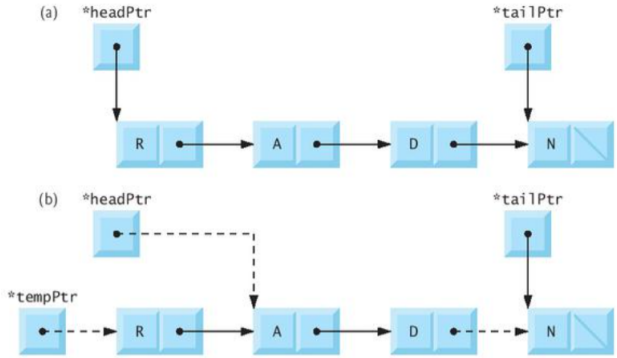


Figura 4: Operação dequeue (Deitel; Deitel, 2011).

(Deitel; Deitel, 2011) - Capítulo 12

DEITEL, Paul; DEITEL, Harvey. **C: Como Programar**. 6. ed. São Paulo: Pearson Universidades, 2011.

SCHILDT, Herbert. **C Completo e Total: O Guia Definitivo para Programação em C**. 3. ed. São Paulo: Makron Books, 1996. ISBN 978-8534606928.

Estes slides estão protegidos por uma licença Creative Commons

Este modelo foi adaptado de Maxime Chupin.

Marisangila Alves, MSc
marisangila.alves@catolicasc.org.br
marisangila.com.br

Católica de Santa Catarina

2025/2

Estrutura de Dados

Estrutura de Dados
Fila