

# *TECNOLOGIA EM INFORMÁTICA*

## ***Estrutura de Dados***

*Prof. Glauco da Silva*  
*contato: [glaucoslv@gmail.com](mailto:glaucoslv@gmail.com)*

# ***Roteiro***

- *Objetivos*
- *Filas*
- *Filas de prioridade*
- *Exercícios*

# ***Roteiro***

- ***Objetivos***
- *Filas*
- *Filas de prioridade*
- *Exercícios*

# *Objetivos*

- *Apresentar o conceito de Filas e Filas de prioridade*
- *Mostrar exemplos de utilização de filas*

# ***Roteiro***

- *Objetivos*
- ***Filas***
- *Filas de prioridade*
- *Exercícios*

# *Filas*

- *Estrutura parecida com a pilha, com exceção que o primeiro elemento inserido é o primeiro elemento a ser removido*
- *São utilizadas para se trabalhar com outras estruturas, por exemplo a pesquisa em um grafo*
- *Trabalham com a arquitetura FIFO (First In First Out)*

# *Filas*

- *Aplicações na computação*
  - *Software: Fila de processamento de tarefas, fila de armazenamento de dados digitados, fila de impressão*
  - *Rede: Fila de pacotes a serem transmitidos, fila de pacotes recebidos em um roteador*

# ***Filas***

- *Analogias*

- *Fila de banco;*

- *Carros em uma cabine de pedágio;*

- *Compra de ingressos no cinema;*

- *Caminhões preparados para descarregar a carga;*

- *etc.*

# *Filas*

- *Funções*

- *Inserir*

- *Adiciona um elemento no final da fila*

- *Remover*

- *Remove o elemento que está no início da fila*

- *Ver o início da fila (peek)*

- *Torna possível ver qual é o valor armazenado no início da fila*

# *Filas*

- *Funções*

- *Vazia*

- *Verifica se a fila está vazia, útil para quando se deseja remover um elemento*

- *Cheia*

- *Verifica se a fila está cheia, útil para quando se deseja inserir um elemento*

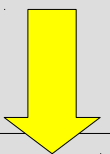
# Filas

Fila.java

```
public class Fila {
    private int tamanho, inicio, fim, itens;
    private long[] vetorFila;

    public Fila(int s){
        tamanho = s;
        vetorFila = new long[tamanho];
        inicio = 0;
        fim = -1;
        Itens = 0;
    }

    public void inserir(long i){
        if (fim == tamanho-1)
            fim = -1;
        vetorFila[++fim] = i;
        Itens++;
    }
}
```



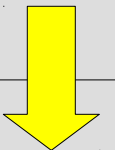
# Filas

Fila.java

```
public long remover(){
    long aux = vetorFila[inicio++];
    if (inicio == tamanho)
        inicio = 0;
    itens--;
    return aux;
}

public long peekFront(){
    return vetorFila[inicio];
}

public boolean estaVazia(){
    return (itens == 0);
}
```



# Filas

Fila.java

```
public boolean estaCheia(){
    return (itens == tamanho);
}

public static void main(String[] args){
    Fila fila = new Fila(10);

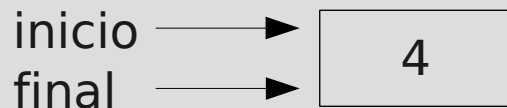
    fila.inserir(10); fila.inserir(20); fila.inserir(30);
    fila.inserir(40);
    fila.remover(); fila.remover(); fila.remover();
    fila.inserir(50); fila.inserir(60); fila.inserir(70);
    fila.inserir(80);

    while(!fila.estaVazia()){
        long n = fila.remover();
        System.out.print(n);
        System.out.print(" ");
    }
    System.out.println();
}
```

# Filas

- *Exemplo*

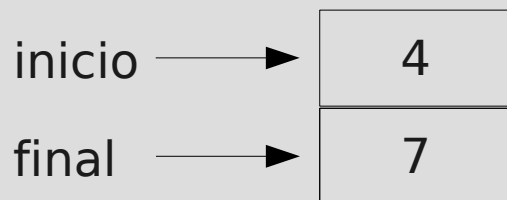
→ *Inserir os elementos 4, 7, 45, 23, 30 e 50 em uma fila e em seguida remover dois itens*



# Filas

- *Exemplo*

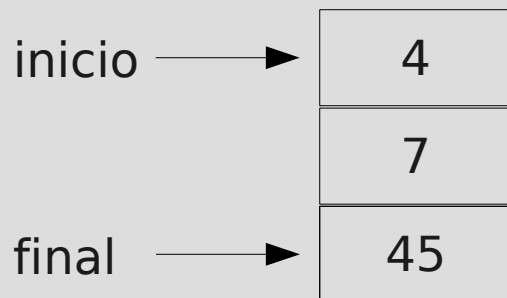
→ *Inserir os elementos 4, 7, 45, 23, 30 e 50 em uma fila e em seguida remover dois itens*



# Filas

- *Exemplo*

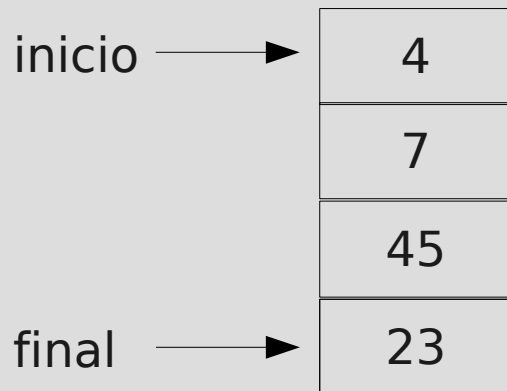
→ *Inserir os elementos 4, 7, 45, 23, 30 e 50 em uma fila e em seguida remover dois itens*



# Filas

- *Exemplo*

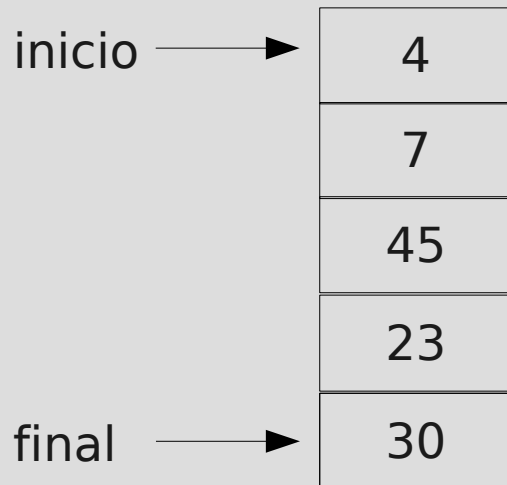
→ *Inserir os elementos 4, 7, 45, 23, 30 e 50 em uma fila e em seguida remover dois itens*



# Filas

- *Exemplo*

→ *Inserir os elementos 4, 7, 45, 23, 30 e 50 em uma fila e em seguida remover dois itens*



# Filas

- *Exemplo*

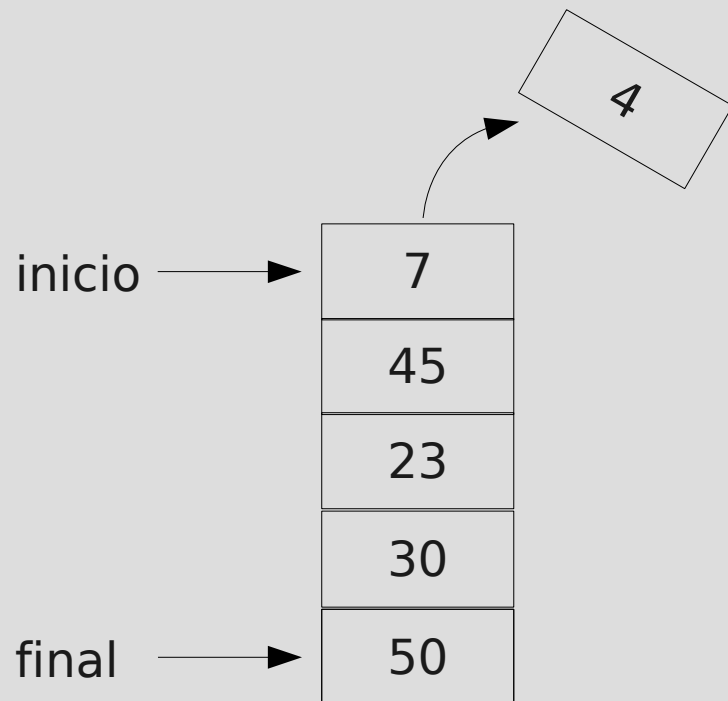
→ *Inserir os elementos 4, 7, 45, 23, 30 e 50 em uma fila e em seguida remover dois itens*



# Filas

- *Exemplo*

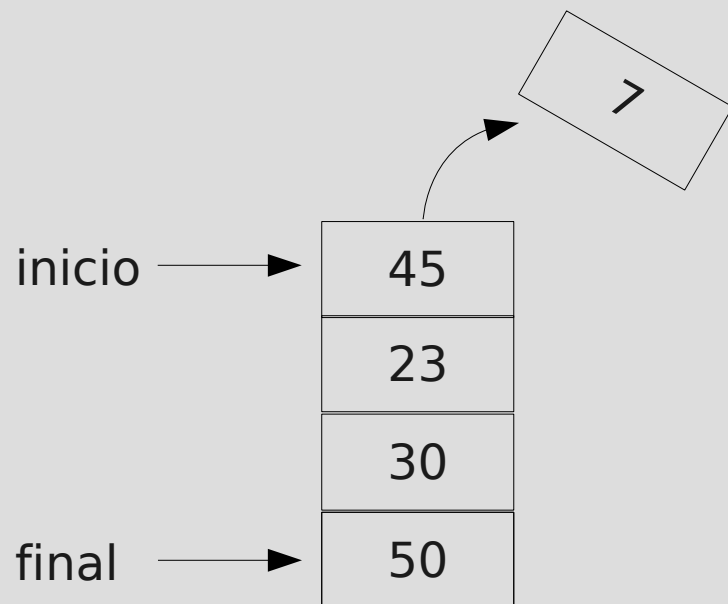
→ *Inserir os elementos 4, 7, 45, 23, 30 e 50 em uma fila e em seguida remover dois itens*



# Filas

- *Exemplo*

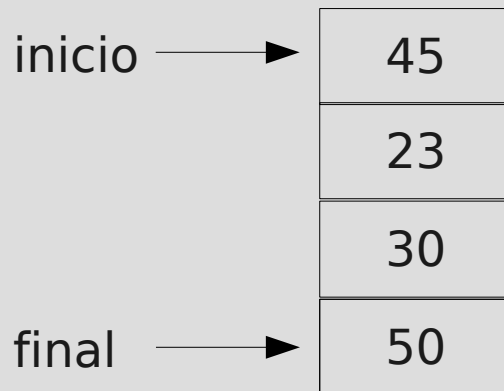
→ *Inserir os elementos 4, 7, 45, 23, 30 e 50 em uma fila e em seguida remover dois itens*



# Filas

- *Exemplo*

→ *Inserir os elementos 4, 7, 45, 23, 30 e 50 em uma fila e em seguida remover dois itens*



# ***Roteiro***

- *Objetivos*
- *Filas*
- ***Filas de prioridade***
- *Exercícios*

# ***Filas de Prioridade***

- *Estrutura mais especializada que uma pilha ou fila*
- *Possuem um final e um início e os itens são removidos do início, porém os elementos são ordenados por valor-chave*
- *Os elementos são inseridos na posição correta para manter a ordem*

# ***Filas de Prioridade***

- *Aplicações na computação*
  - *Software: Fila de processamento de tarefas em um sistema operacional*
  - *Estruturas de Dados: Encontrar uma árvore de extensão mínima para um grafo*

# ***Filas de Prioridade***

- *Analogias*

- *Propostas de custos para determinada tarefa;*

- *Cartas a serem entregues em uma rua;*

- *etc.*

# ***Filas de Prioridade***

- *Funções*

- *Inserir*

- *Adiciona um elemento na posição correta do item*

- *Remover*

- *Remove o elemento que está no início da fila*

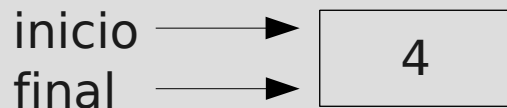
- *Ver o início da fila (peek)*

- *Torna possível ver qual é o valor armazenado no início da fila*

# *Filas de Prioridade*

- *Exemplo*

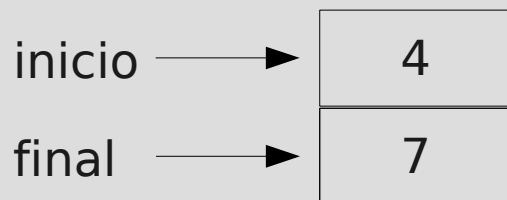
→ *Inserir os elementos 4, 7, 45, 23, 30 e 50 em uma fila e em seguida remover dois itens*



# *Filas de Prioridade*

- *Exemplo*

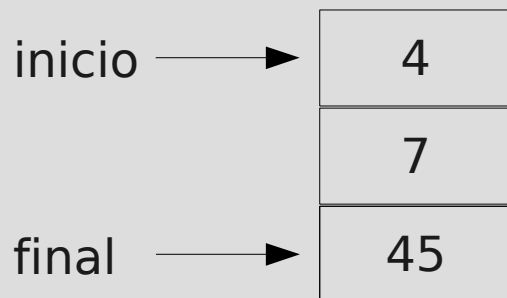
→ *Inserir os elementos 4, 7, 45, 23, 30 e 50 em uma fila e em seguida remover dois itens*



# *Filas de Prioridade*

- *Exemplo*

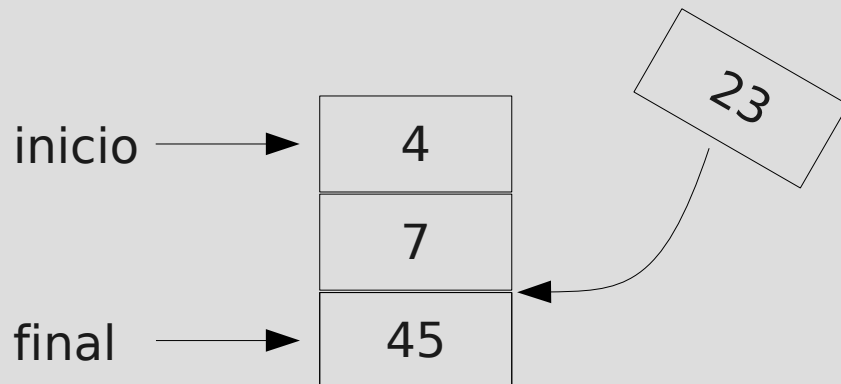
→ *Inserir os elementos 4, 7, 45, 23, 30 e 50 em uma fila e em seguida remover dois itens*



# Filas de Prioridade

- *Exemplo*

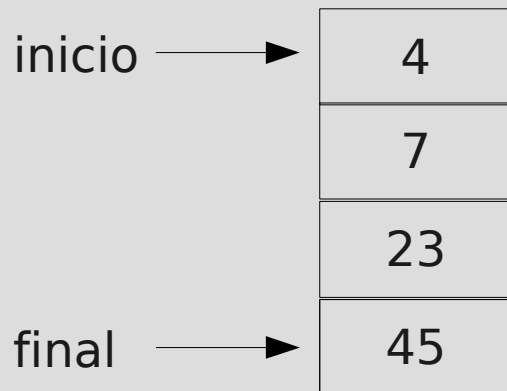
→ *Inserir os elementos 4, 7, 45, 23, 30 e 50 em uma fila e em seguida remover dois itens*



# *Filas de Prioridade*

- *Exemplo*

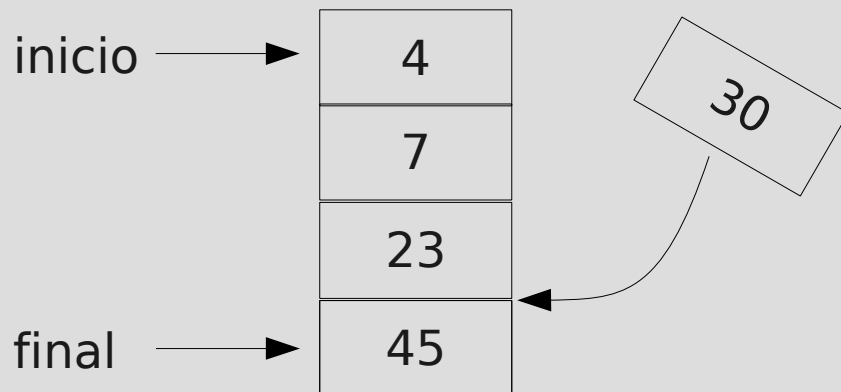
→ *Inserir os elementos 4, 7, 45, 23, 30 e 50 em uma fila e em seguida remover dois itens*



# Filas de Prioridade

- *Exemplo*

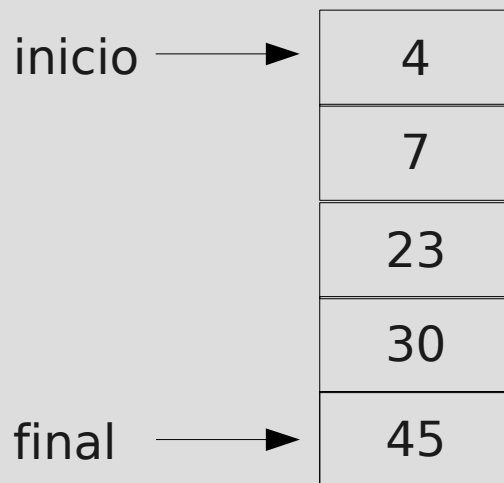
→ *Inserir os elementos 4, 7, 45, 23, 30 e 50 em uma fila e em seguida remover dois itens*



# *Filas de Prioridade*

- *Exemplo*

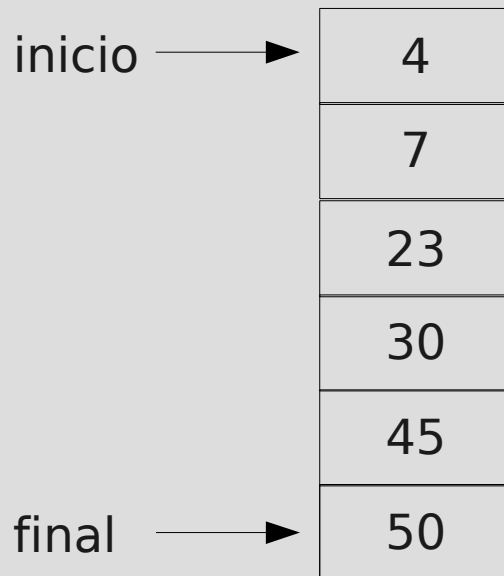
→ *Inserir os elementos 4, 7, 45, 23, 30 e 50 em uma fila e em seguida remover dois itens*



# *Filas de Prioridade*

- *Exemplo*

→ *Inserir os elementos 4, 7, 45, 23, 30 e 50 em uma fila e em seguida remover dois itens*



# *Filas de Prioridade*

FilaPrioridade.java

```
public class FilaPrioridade {
    private int tamanho;
    private long[] vetorFila;
    private int itens;

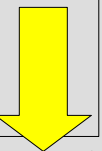
    public FilaPrioridade(int s){
        tamanho = s;
        vetorFila = new long[tamanho];
        itens = 0;
    }

    public long remover(){ return vetorFila[--itens]; }

    public long peek(){ return vetorFila[itens-1]; }

    public boolean estaVazia(){ return (itens==0); }

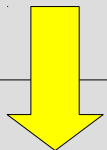
    public boolean estaCheia(){ return (itens==tamanho); }
```



# *Filas de Prioridade*

FilaPrioridade.java

```
public void inserir(long elemento){
    int i;
    if (itens==0)
        vetorFila[itens++] = elemento;
    else{
        for (i=itens-1; i>=0; i--){
            if (elemento > vetorFila[i])
                vetorFila[i+1] = vetorFila[i];
            else
                break;
        }
        vetorFila[i+1] = elemento;
        itens++;
    }
}
```



# *Filas de Prioridade*

FilaPrioridade.java

```
public static void main(String[] args){
    FilaPrioridade fila = new FilaPrioridade(5);
    fila.inserir(30);
    fila.inserir(50);
    fila.inserir(10);
    fila.inserir(40);
    fila.inserir(20);

    while (!fila.estaVazia()){
        long item = fila.remover();
        System.out.print(item + " ");
    }
    System.out.println();
}
```

# ***Roteiro***

- *Objetivos*
- *Filas*
- *Filas de prioridade*
- ***Exercícios***

# ***Exercícios***

- *O que LIFO e FIFO significam?*
- *Suponha que você insira 15, 25, 35 e 45 em uma fila, em seguida remove três itens. Qual deles é deixado?*
- *Mostre como implementar uma fila usando duas pilhas.*
- *Mostre como implementar uma pilha usando duas filas.*
- *Explique como implementar duas pilhas em um único arranjo  $A[1, \dots, n]$  de tal modo que nenhuma das pilhas sofra um estouro.*