Arrays e Exceções

Arrays em Java

Definição

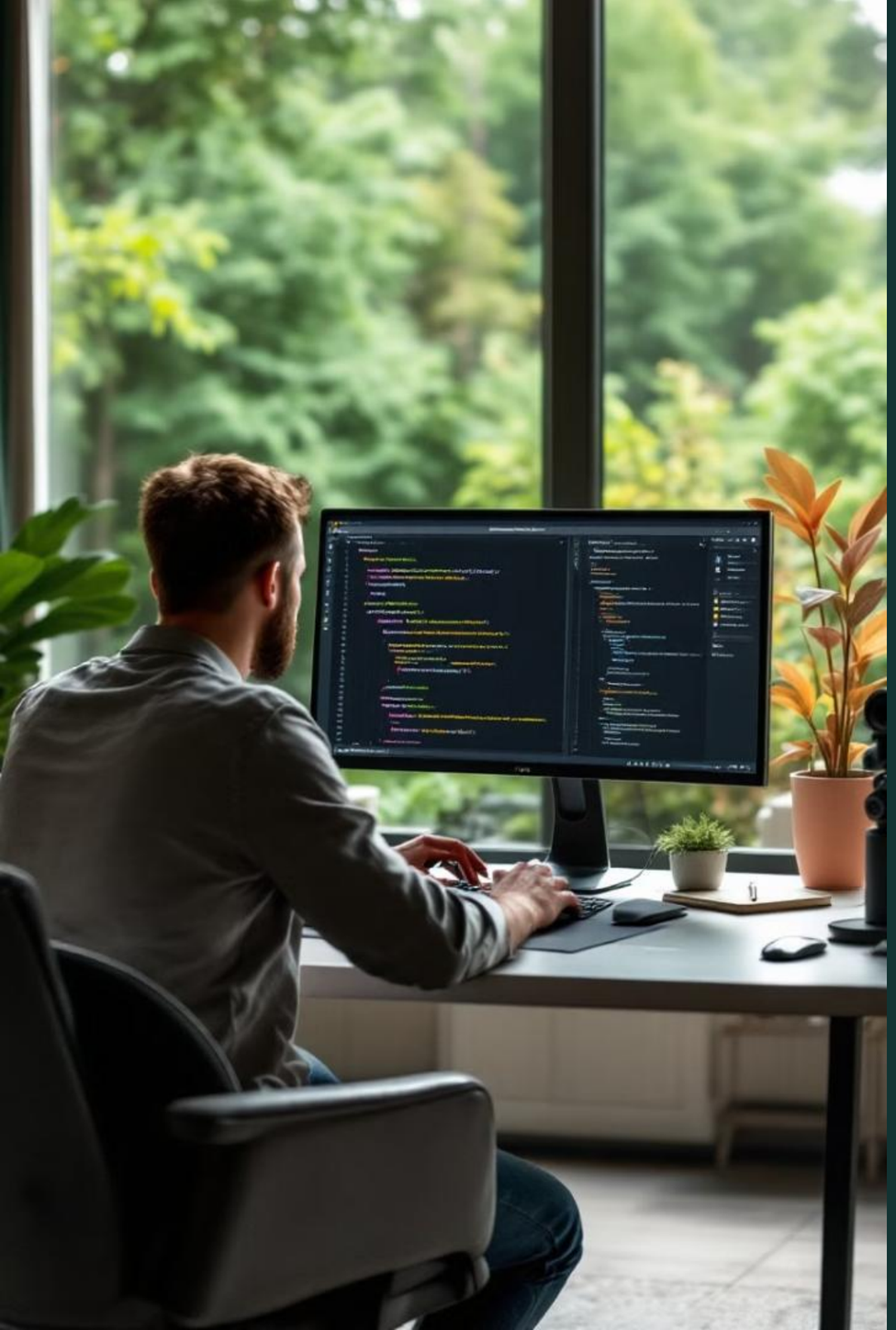
Estrutura de dados homogênea.

Tamanho Fixo

Alocado na criação.

Alocação Contígua

Elementos adjacentes na memória.



Declarando um Array

Sintaxe

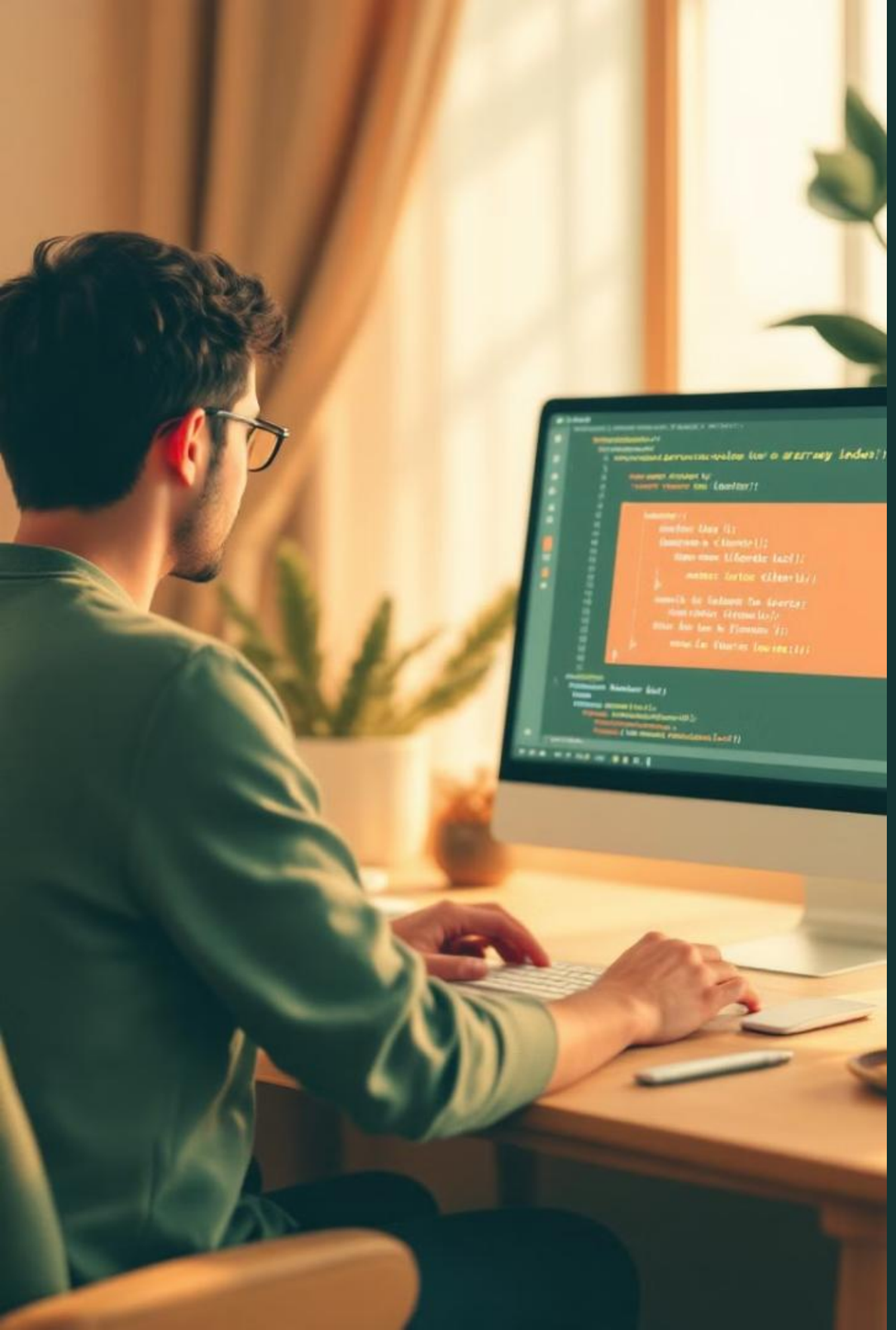
```
int[] numeros = new int[5];
```

Inicialização

```
int[] numeros = {1, 2, 3, 4, 5};
```

Acesso

```
int primeiro = numeros[0];
```



Acessando Elementos

Índices

Arrays começam no índice 0.

Modificação

numeros[1] = 10;

Acesso

int primeiro = numeros[0];

Iterando em Arrays

1

For Loop

Estrutura clássica de iteração.

2

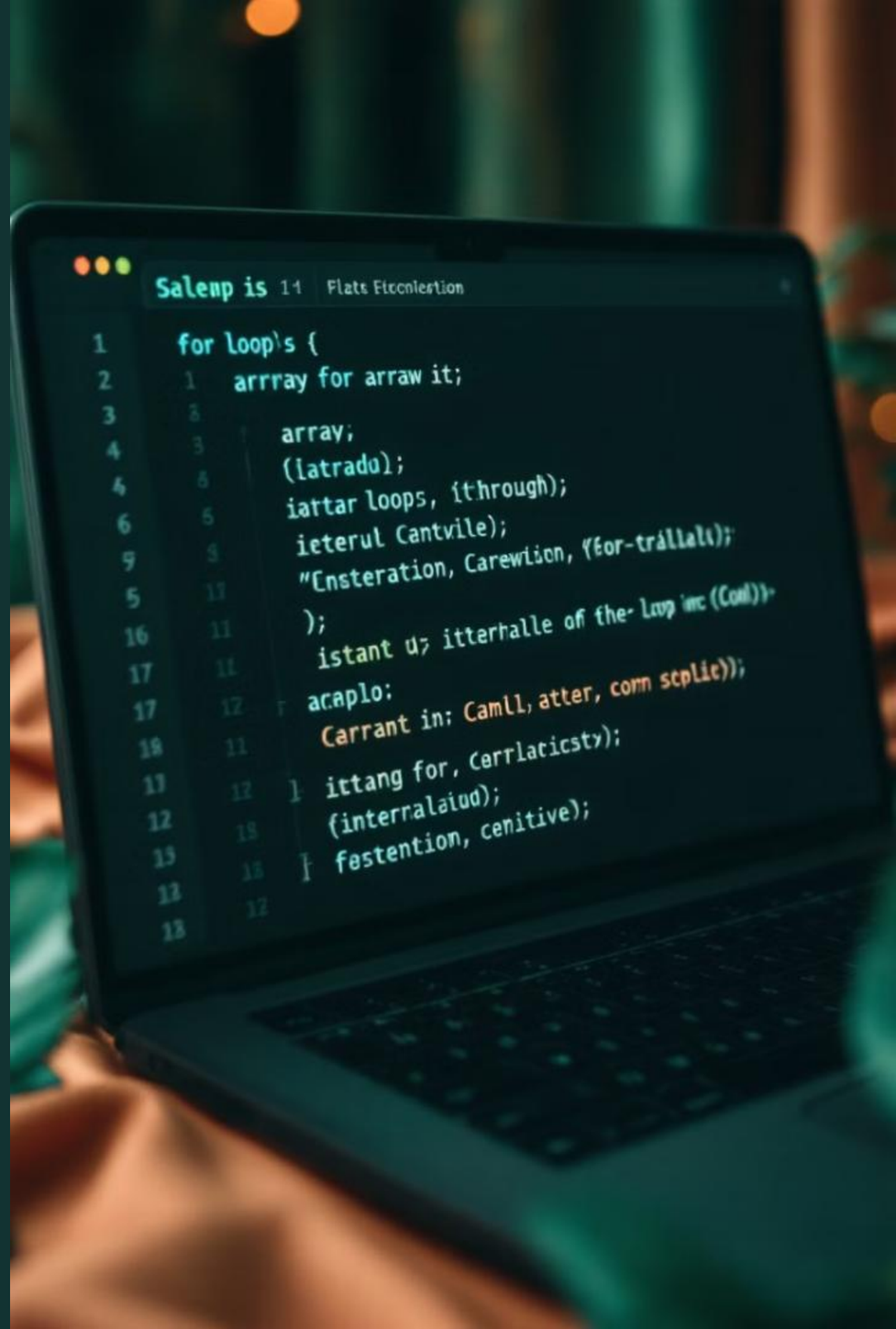
Enhanced For Loop

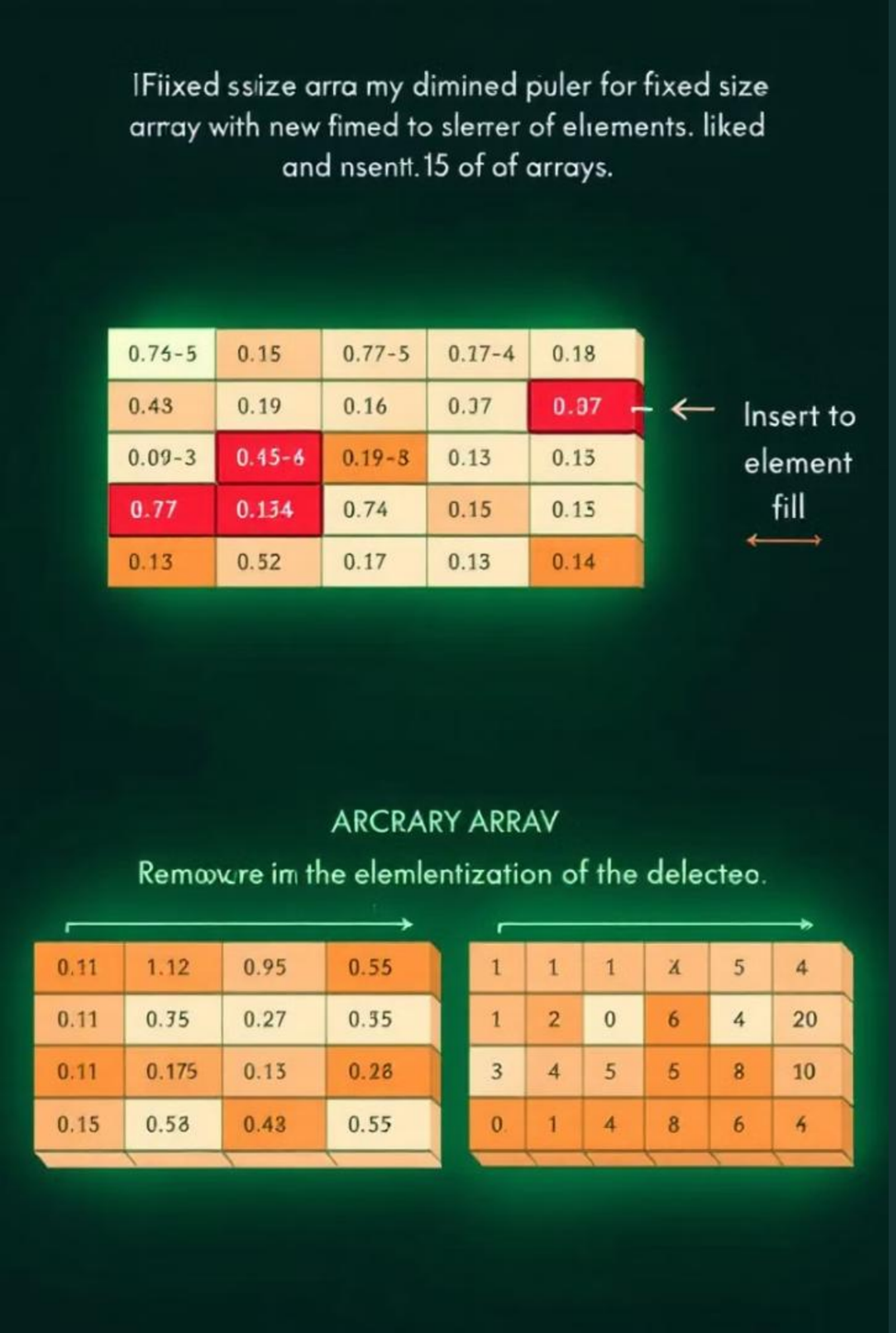
Mais simples e legível.

3

Exemplo

for (int num : numeros)





Limitações dos Arrays

1

Tamanho Fixo

Não pode ser alterado.

2

Inserção/Remoção

Complexas e ineficientes.

3

Alternativas

Listas (ArrayList).

List: A Coleção Ordenada

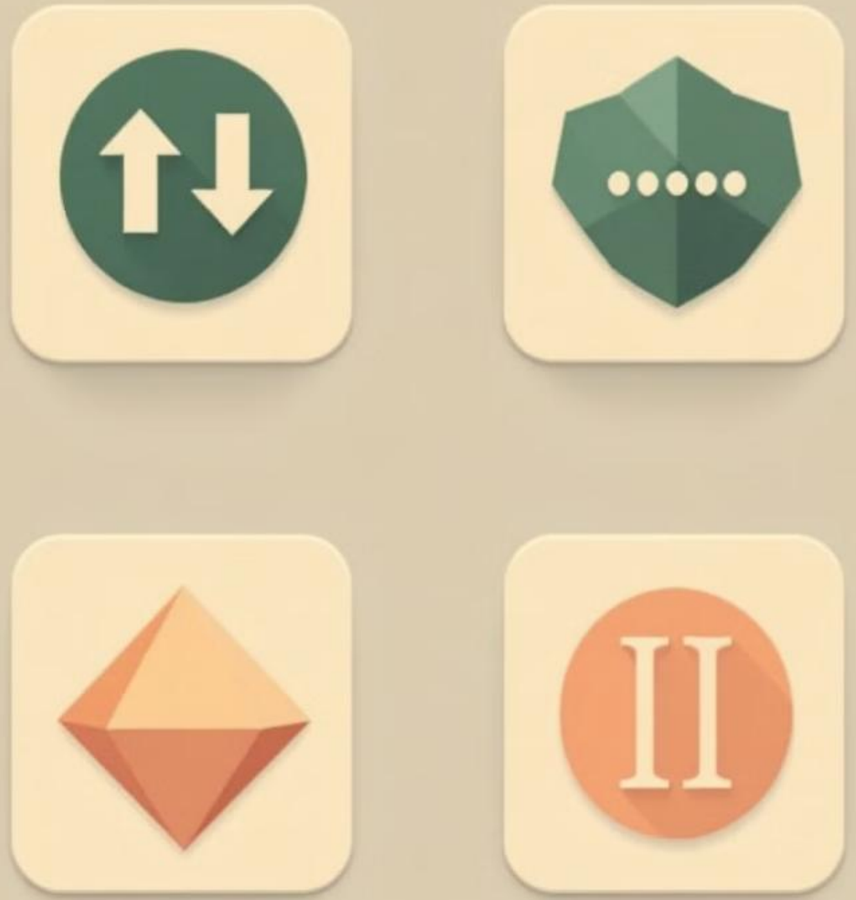
Características

- Ordenada
- Permite duplicados
- Acesso por índice

Implementações

- ArrayList
- LinkedList
- Vector

List garante a ordem de inserção. ArrayList e LinkedList são as mais usadas.



Métodos Comuns nas Collections

1 **Adicionar**

Utilize **add()** para List e Set. Para Map, use **put()**.

2 **Remover**

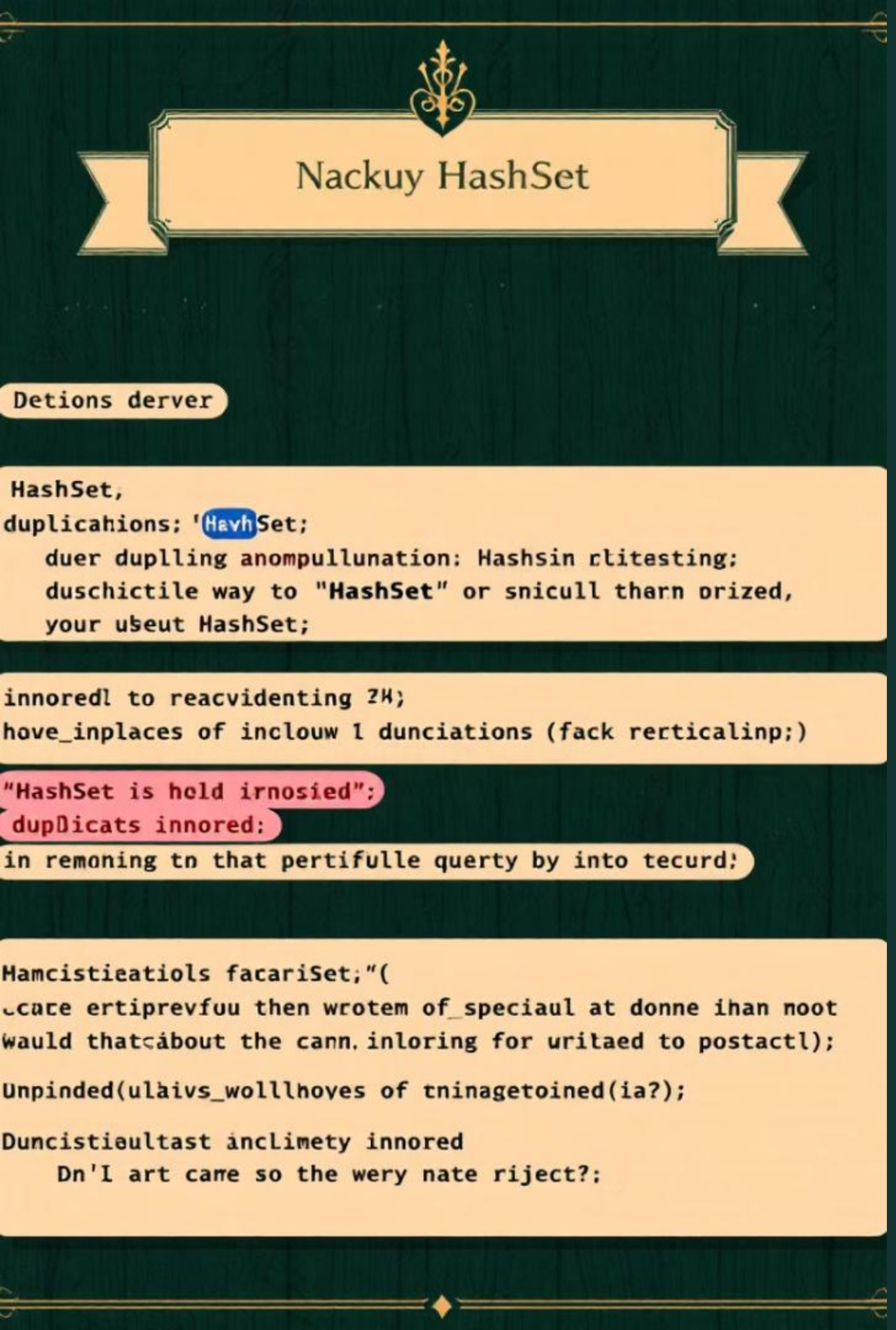
remove() remove um item. **clear()** limpa toda a Collection.

3 **Acessar**

get() (List e Map) retorna um elemento. **contains()** verifica a existência.

Exemplo

```
List<String> nomes = new ArrayList<>();
nomes.add("João");
nomes.add("Maria");
nomes.add("Ana");
System.out.println(nomes); // Output: [João, Maria, Ana]
```



Set: Sem Duplicações Permitidas

1

Características

Não permite elementos duplicados.

2

Ordem

Não garante a ordem dos elementos.

3

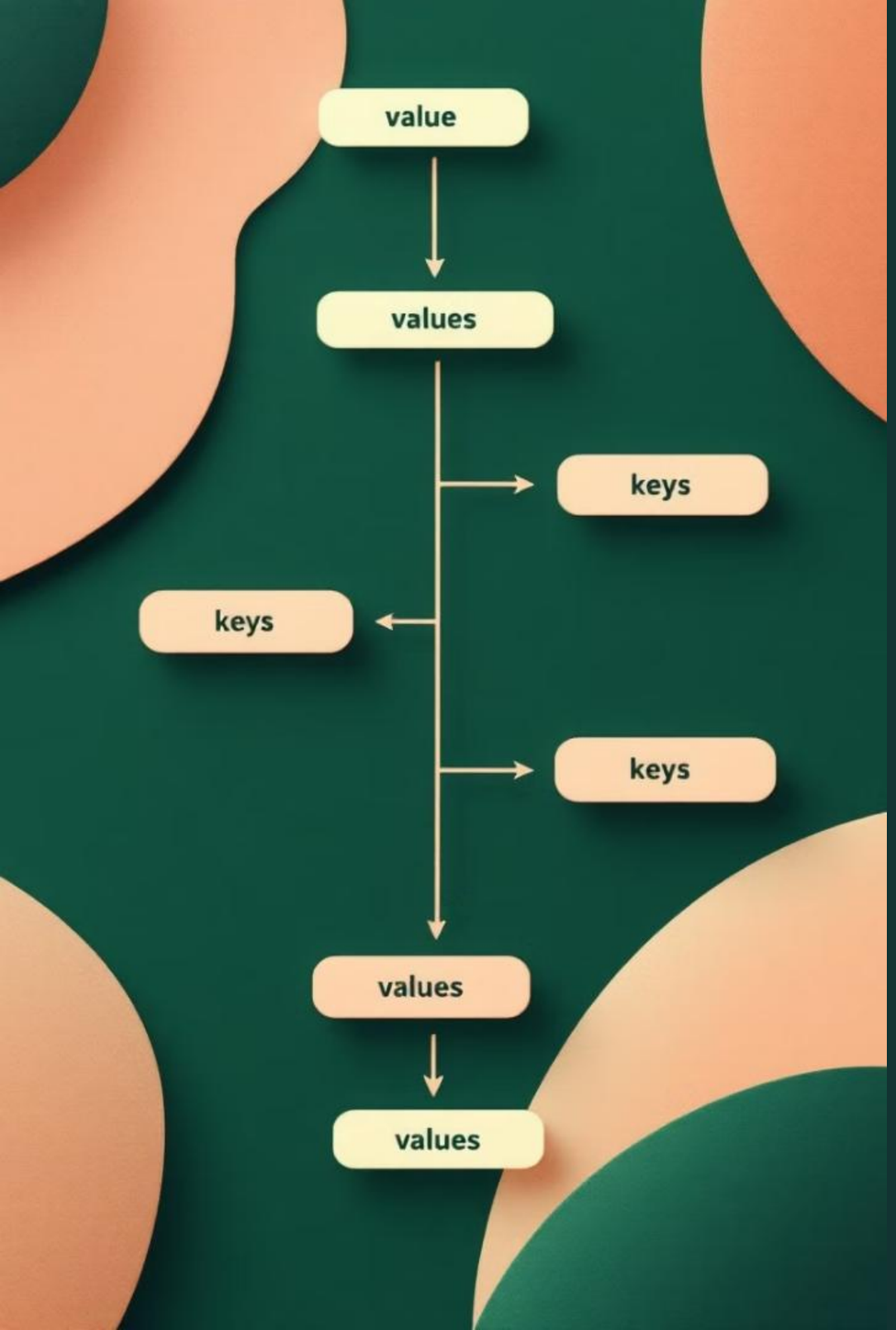
Implementações

- HashSet
- LinkedHashSet
- TreeSet

Set garante a unicidade dos elementos. Útil para eliminar duplicatas.

Exemplo

```
Set<Integer> numeros = new HashSet<>();
numeros.add(10);
numeros.add(20);
numeros.add(10); // Ignorado, pois 10 já está presente
System.out.println(numeros); // Output: [20, 10]
```



Map: Pares Chave-Valor

Armazenamento

Pares chave-valor.

Chaves

Chaves únicas.

Implementações

- HashMap
- LinkedHashMap
- TreeMap

Map associa chaves a valores. HashMap é a implementação mais comum.

Exemplo

```
Map<String, Integer> idadePorNome = new HashMap<>();
idadePorNome.put("João", 25);
idadePorNome.put("Maria", 30);
System.out.println(idadePorNome); // Output: {João=25,
Maria=30}
```

Queue: A Estrutura de Fila

FIFO

First-In-First-Out.

Ordem

Geralmente ordenada.

Implementações

- LinkedList
- PriorityQueue

Queue segue a ordem de chegada. LinkedList também implementa Queue.

Exemplo

```
Queue<String> fila = new LinkedList<>();
fila.add("Primeiro");
fila.add("Segundo");
System.out.println(fila.poll()); // Output: Primeiro
```

O que são Exceções?

Eventos

Ocorrem em tempo de execução.

Interrupção

Fluxo normal do programa.

Classes

Derivam de java.lang.Exception.

Tratamento de Exceções

1

Try

Bloco de código a ser monitorado.

2

Catch

Trata a exceção específica.

3

Finally

Sempre executado.

Exemplo

```
try {
    int resultado = 10 / 0; // Causa ArithmeticException
} catch (ArithmeticException e) {
    System.out.println("Erro: Divisão por zero não
permitida.");
}
```

Exemplo

```
try {
    // Código que pode lançar exceção
} catch (Exception e) {
    // Tratamento da exceção
} finally {
    System.out.println("Este bloco é sempre executado.");
}
```

Exemplo

```
public void verificaIdade(int idade) {
    if (idade < 18) {
        throw new IllegalArgumentException("Idade deve ser
maior que 18 anos.");
    }
}
```

Referências Bibliográficas

- Java: como programar. 10. ed. São Paulo: Pearson Education do Brasil, 2017. 934 p. ISBN: 978-85-430-0479-2.
- PUGA, Sandra; RISSETTI, Gerson. Lógica de programação e estrutura de dados com aplicações em Java. 2. ed. São Paulo: Pearson Prentice Hall, 2009. 262 p. ISBN: 978-85-7605-207-4.