Vai retorna o primeiro resultado quer for localizado.

Ideal para um resultado mais rápido.

**findAny**

Uma operação útil para trabalhar com threads é o método findAny em coleções de dados. Esse método é utilizado para encontrar qualquer elemento que satisfaça uma determinada condição em uma coleção. Ele retorna um Optional, que pode conter o elemento encontrado ou ser vazio caso nenhum elemento satisfaça a condição.

Ao utilizar o findAny em uma coleção com threads, cada thread pode buscar um elemento da coleção de forma paralela, tornando a busca mais rápida e eficiente. É importante ressaltar que o findAny **não** garante que sempre será retornado o mesmo elemento, pois a ordem de busca pode variar entre as threads.

Para utilizar o findAny com threads, você pode criar uma implementação de Predicate que define a condição que os elementos devem satisfazer. Em seguida, você pode utilizar o método parallelStream em uma coleção para criar uma stream paralela e chamar o findAny, passando o Predicate como argumento. Vejamos um exemplo:

**public** **class** **ExemploFindAnyParallelStream** {

**public** **static** **void** **main**(String[] args) {

List<Integer> numeros = **new** ArrayList<>();

**for** (int i = 1; i <= 100; i++) {

numeros.**add**(i);

}

// Utilizando parallelStream para encontrar um elemento qualquer em paralelo

Optional<Integer> numeroQualquer = numeros.parallelStream()

.filter(numero -> numero % 10 == 0) // Filtra os números que são múltiplos de 10

.findAny();

**if** (numeroQualquer.isPresent()) {

System.**out**.println("Encontrado: " + numeroQualquer.**get**());

} **else** {

System.**out**.println("Nenhum número encontrado.");

}

}

}

Neste exemplo, temos uma lista de números de 1 a 100. Utilizando parallelStream, filtramos os números que são múltiplos de 10 e, em seguida, utilizamos findAny para encontrar qualquer elemento que atenda a essa condição.

Como a busca é feita em paralelo, o resultado pode variar a cada execução do programa, pois diferentes threads podem encontrar elementos diferentes do stream. A saída no console pode ser, por exemplo:

**Encontrado: 70**

Ou

**Encontrado: 50**

ou qualquer outro número que seja múltiplo de 10 dentro da lista.

Lembrando que, para alguns tipos de stream, como streams de listas ou arrays, a ordem dos elementos é mantida. No entanto, em um stream paralelo, a busca será executada de forma concorrente, e o resultado pode ser qualquer elemento que satisfaça a condição de filtragem. Portanto, em streams paralelos, o método findAny pode ser mais útil para obter qualquer elemento sem a garantia de um resultado específico.