Practice Question on Streams

1. Write a Java program to calculate the average of a list of integers using streams.

Ans.

import java.util.List;

import java.util.stream.Collectors;

public class Average {

public static void main(String[] args) {

List<Integer> numbers = List.*of*(1, 2, 3, 4, 5);

double average = numbers.stream()

.mapToInt(Integer::intValue) // Optional for primitive streams

.average()

.orElse(0.0); // Handle potential empty stream

System.***out***.println("Average: " + average);

}

}

2. Write a Java program to convert a list of strings to uppercase or lowercase using streams.

Ans.

import java.util.\*;

import java.util.stream.Collectors;

public class StringConverter {

public static void main(String[] args) {

List<String> strings = List.*of*("hello", "world", "java");

// To uppercase:

List<String> uppercase = strings.stream()

.map(String::toUpperCase)

.collect(Collectors.*toList*());

System.***out***.println("Uppercase: " + uppercase);

// To lowercase:

List<String> lowercase = strings.stream()

.map(String::toLowerCase)

.collect(Collectors.*toList*());

System.***out***.println("Lowercase: " + lowercase);

}

}

3. Write a Java program to calculate the sum of all even, odd numbers in a list using streams.

Ans.

import java.util.List;

public class EvenOddSumCalculator {

public static void main(String[] args) {

List<Integer> numbers = List.*of*(1, 2, 3, 4, 5);

int evenSum = numbers.stream()

.filter(n -> n % 2 == 0)

.mapToInt(Integer::intValue) // Optional for primitive streams

.sum();

System.***out***.println("Sum of even numbers: " + evenSum);

int oddSum = numbers.stream()

.filter(n -> n % 2 != 0)

.mapToInt(Integer::intValue) // Optional for primitive streams

.sum();

System.***out***.println("Sum of odd numbers: " + oddSum);

}

}

4. Write a Java program to remove all duplicate elements from a list using streams

Ans.

import java.util.List;

import java.util.stream.Collectors;

public class DuplicateRemover {

public static void main(String[] args) {

List<Integer> numbers = List.*of*(1, 2, 2, 3, 4, 4);

List<Integer> distinctNumbers = numbers.stream()

.distinct()

.collect(Collectors.*toList*());

System.***out***.println("Distinct numbers: " + distinctNumbers);

}

}

5. Write a Java program to count the number of strings in a list that start with a specific letter using streams.

Ans.

import java.util.List;

public class StringCounter {

public static void main(String[] args) {

List<String> strings = List.*of*("apple", "banana", "cherry", "apple", "orange");

long count = strings.stream()

.filter(s -> s.startsWith("a"))

.count();

System.***out***.println("Number of strings starting with 'a': " + count);

}

}

6. Optional Classes in JAVA 8.

Ans. Optional classes in Java 8:

* Optional is a container object used to represent the potential absence of a value.
* It helps avoid NullPointerExceptions by explicitly indicating when a value might be null.
* Common methods:
  + isPresent(): Checks if a value is present.
  + get(): Retrieves the value if present, throws NoSuchElementException if absent.
  + orElse(): Returns a default value if the Optional is empty.
  + orElseGet(): Returns a value generated by a supplier function if the Optional is empty.
  + map(): Transforms the value if present