Here is a comprehensive list of Java Streams practice problems and task types, combining exercises from leading practice resources and tutorials:

**Core Java Streams Practice Problems**

1. **Calculate the average of integers using streams**
   * Use a list of integers and the Stream API to compute the average[1](https://www.w3resource.com/java-exercises/stream/index.php)[4](https://stackify.com/streams-guide-java-8/).
2. **Convert strings to uppercase or lowercase using streams**
   * Transform each string in a list to uppercase or lowercase using map and collect the result[1](https://www.w3resource.com/java-exercises/stream/index.php)[2](https://codefinity.com/courses/v2/190d2568-3d25-44d0-832f-da03468004c9/c0bcd017-ff39-46ec-bc93-acd569f3497d/d56d7f54-9992-491a-a50f-64ceee652f04).
3. **Sum even and odd numbers in a list using streams**
   * Filter and sum all even numbers, then all odd numbers in a list[1](https://www.w3resource.com/java-exercises/stream/index.php).
4. **Remove duplicates from a list using streams**
   * Find and collect only unique elements from a list using distinct[1](https://www.w3resource.com/java-exercises/stream/index.php)[2](https://codefinity.com/courses/v2/190d2568-3d25-44d0-832f-da03468004c9/c0bcd017-ff39-46ec-bc93-acd569f3497d/d56d7f54-9992-491a-a50f-64ceee652f04).
5. **Count strings starting with a specific letter using streams**
   * Filter and count strings in a list that start with a given letter[1](https://www.w3resource.com/java-exercises/stream/index.php).
6. **Sort strings in alphabetical order (ascending and descending) using streams**
   * Sort a list of strings in ascending and descending order using sorted[1](https://www.w3resource.com/java-exercises/stream/index.php).
7. **Find maximum and minimum values in a list of integers using streams**
   * Use terminal operations to find the max and min values[1](https://www.w3resource.com/java-exercises/stream/index.php)[4](https://stackify.com/streams-guide-java-8/).
8. **Find the second smallest and largest elements in a list of integers using streams**
   * Sort and skip to retrieve the second smallest and largest elements[1](https://www.w3resource.com/java-exercises/stream/index.php).

**Additional Stream API Practice Tasks**

1. **Square each number in a list and collect the result**
   * Use map and toList to create a new list of squared values[2](https://codefinity.com/courses/v2/190d2568-3d25-44d0-832f-da03468004c9/c0bcd017-ff39-46ec-bc93-acd569f3497d/d56d7f54-9992-491a-a50f-64ceee652f04).
2. **Find the length of the longest name in a list**
   * Use map to get lengths, then max to find the longest[2](https://codefinity.com/courses/v2/190d2568-3d25-44d0-832f-da03468004c9/c0bcd017-ff39-46ec-bc93-acd569f3497d/d56d7f54-9992-491a-a50f-64ceee652f04).
3. **Count the total number of distinct words (case-insensitive) in all sentences**
   * Use flatMap to split sentences into words, distinct to remove duplicates, and count for the result[2](https://codefinity.com/courses/v2/190d2568-3d25-44d0-832f-da03468004c9/c0bcd017-ff39-46ec-bc93-acd569f3497d/d56d7f54-9992-491a-a50f-64ceee652f04).
4. **Filter elements based on a predicate and count them**
   * Example: Count employees with salary above a certain threshold[4](https://stackify.com/streams-guide-java-8/).
5. **Generate infinite streams and limit the output**
   * Use Stream.iterate or Stream.generate with limit to generate finite sequences[4](https://stackify.com/streams-guide-java-8/).
6. **Process files using streams**
   * Read lines from a file, filter or transform them, and collect results[4](https://stackify.com/streams-guide-java-8/).
7. **Find palindromes in a list or file**
   * Filter strings that are palindromes using stream operations[4](https://stackify.com/streams-guide-java-8/).

**Advanced Streams Practice**

1. **Use mapToInt, mapToDouble, or mapToLong for primitive streams**
   * Perform operations like summing, averaging, or finding max/min on primitive data[4](https://stackify.com/streams-guide-java-8/).
2. **Use short-circuiting operations (findFirst, anyMatch, allMatch, noneMatch)**
   * Early termination of stream processing based on conditions[4](https://stackify.com/streams-guide-java-8/).
3. **Combine streams using concat**
   * Merge two streams into one[4](https://stackify.com/streams-guide-java-8/).
4. **Use Java 9+ methods: takeWhile, dropWhile, ofNullable**
   * Conditionally take or drop elements, or handle nullable sources[4](https://stackify.com/streams-guide-java-8/).
5. **Group elements by a property using Collectors.groupingBy**
   * Group objects (e.g., employees by department)[4](https://stackify.com/streams-guide-java-8/).

**Sample Problem Table**

| **Problem Type** | **Stream Methods Used** |
| --- | --- |
| Average of integers | mapToInt, average |
| Convert strings to upper/lowercase | map, collect |
| Sum even/odd numbers | filter, sum |
| Remove duplicates | distinct, collect |
| Count strings by starting letter | filter, count |
| Sort strings | sorted, collect |
| Max/min in list | max, min |
| Second smallest/largest | sorted, skip, findFirst |
| Square each number | map, toList |
| Longest name length | map, max |
| Distinct word count (case-insensitive) | flatMap, distinct, count |
| Filter and count elements | filter, count |
| Infinite streams with limit | iterate, generate, limit |
| File processing | Files.lines, filter, collect |
| Palindromes | filter, collect |
| Primitive streams | mapToInt/mapToDouble/etc. |
| Short-circuiting operations | findFirst, anyMatch |
| Combine streams | concat |
| Java 9+ methods | takeWhile, dropWhile |
| Grouping by property | groupingBy |

This list covers a wide range of Java Streams practice problems, from basic to advanced, and provides a solid foundation for mastering the Stream API[1](https://www.w3resource.com/java-exercises/stream/index.php)[2](https://codefinity.com/courses/v2/190d2568-3d25-44d0-832f-da03468004c9/c0bcd017-ff39-46ec-bc93-acd569f3497d/d56d7f54-9992-491a-a50f-64ceee652f04)[4](https://stackify.com/streams-guide-java-8/).