Challenge #1

Description: Filter even numbers from a list and print them.

Solution:

```
java
CopyEdit
numbers.stream()
       .filter(n -> n % 2 == 0)
       .forEach(System.out::println);
```

Challenge #2

Description: Convert all names in a list to uppercase and collect them into a new list.

Solution:

```
java
CopyEdit
List<String> upperNames = names.stream()
                               .map(String::toUpperCase)
                               .collect(Collectors.toList());
```

Challenge #3

Description: Find the first city in a list that starts with "P".

Solution:

```
java
CopyEdit
Optional<String> city = cities.stream()
                              .filter(c -> c.startsWith("P"))
                              .findFirst();
```

Challenge #4

Description: Print numbers divisible by 5 but not by 10.

Solution:

```
java
CopyEdit
nums.stream()
    .filter(n -> n % 5 == 0)
    .filter(n -> n % 10 != 0)
    .forEach(System.out::println);
```

Challenge #5

Description: Flatten a nested list of names and print each name.

Solution:

```
java
CopyEdit
nestedNames.stream()
           .flatMap(List::stream)
           .forEach(System.out::println);
```

Challenge #6

Description: Collect the length of each fruit name into a list.

Solution:

```
java
CopyEdit
List<Integer> fruitLengths = fruits.stream()
                                   .map(String::length)
                                   .collect(Collectors.toList());
```

Challenge #7

Description: Filter names that are 5 characters long and start with 'A', then collect them into a list.

Solution:

```
java
CopyEdit
List<String> filteredNames = names.stream()
                                  .filter(n -> n.length() == 5)
                                  .filter(n -> n.startsWith("A"))
                                  .collect(Collectors.toList());
```

Challenge #8

Description: Group words by their first character into a map.

Solution:

```
java
CopyEdit
Map<Character, List<String>> groupedWords = words.stream()

.collect(Collectors.groupingBy(w -> w.charAt(0)));
```

Challenge #9

Description: Sum all numbers in a list.

Solution:

```
java
CopyEdit
int sum = numbers.stream()
                 .mapToInt(Integer::intValue)
                 .sum();
```

Challenge #10

Description: Find the product of all numbers using `reduce`.

Solution:

```
java
CopyEdit
int product = nums.stream()
                  .reduce(1, (acc, n) -> acc * n);
```

Challenge #11

Description: Find the longest word in a list using `reduce`.

Solution:

```
java
CopyEdit
Optional<String> longestWord = words.stream()
                                    .reduce((w1, w2) -> w1.length() >
w2.length() ? w1 : w2);
```

Challenge #12

Description: Concatenate all words into a single string, separated by commas.

Solution:

```
java
CopyEdit
String result = words.stream()
                     .reduce("", (w1, w2) -> w1.isEmpty() ? w2 : w1 + ", "
+ w2);
```

Challenge #13

Description: Filter out negative numbers, square them, sort, and collect into a list.

Solution:

```
java
CopyEdit
List<Integer> squaredNegatives = nums.stream()
                                     .filter(n -> n < 0)
                                     .map(n -> n * n)
                                     .sorted()
                                     .collect(Collectors.toList());
```

Challenge #14

Description: Group words by their length.

Solution:

```
java
CopyEdit
Map<Integer, List<String>> wordsByLength = words.stream()

.collect(Collectors.groupingBy(String::length));
```

Challenge #15

Description: Count the frequency of each word in a list.

Solution:

```
java
CopyEdit
Map<String, Long> wordCount = words.stream()
                                   .collect(Collectors.groupingBy(w -> w,
Collectors.counting()));
```

Challenge #16

Description: Partition numbers into even and odd.

Solution:

```
java
CopyEdit
Map<Boolean, List<Integer>> partitioned = numbers.stream()

.collect(Collectors.partitioningBy(n -> n % 2 == 0));
```

Challenge #17

Description: Convert a list of strings to their lengths using `map`.

Solution:

```
java
CopyEdit
List<Integer> lengths = strings.stream()
                               .map(String::length)
                               .collect(Collectors.toList());
```

Challenge #18

Description: Use `flatMap` to flatten a list of lists of integers.

Solution:

```
java
CopyEdit
List<Integer> flattened = nestedList.stream()
                                    .flatMap(List::stream)
                                    .collect(Collectors.toList());
```

Challenge #19

Description: Find the minimum integer in a list.

Solution:

```
java
CopyEdit
Optional<Integer> min = numbers.stream()
                               .min(Integer::compareTo);
```

Challenge #20

Description: Get a list of distinct names.

Solution:

```
java
CopyEdit
List<String> distinctNames = names.stream()
                                  .distinct()
                                  .collect(Collectors.toList());
```

Challenge #21

Description: Sum all numbers in a list using `reduce`.

Solution:

```
java
CopyEdit
int sum = numbers.stream()
                 .reduce(0, Integer::sum);
```

Challenge #22

Description: Find the product of all numbers in a list using `reduce`.

Solution:

```
java
CopyEdit
int product = numbers.stream()
                     .reduce(1, (acc, n) -> acc * n);
```

Challenge #23

Description: Check if any word in the list starts with "A".

Solution:

```
java
CopyEdit
boolean anyStartsWithA = words.stream()
                              .anyMatch(w -> w.startsWith("A"));
```

Challenge #24

Description: Find the first number greater than 10.

Solution:

```
java
CopyEdit
Optional<Integer> firstGreaterThan10 = numbers.stream()
                                              .filter(n -> n > 10)
                                              .findFirst();
```

Challenge #25

Description: Collect words into a comma-separated single string using `Collectors.joining()`.

Solution:

```
java
CopyEdit
String joined = words.stream()
                     .collect(Collectors.joining(", "));
```

Challenge #26

Description: Sort words by length.

Solution:

```
java
CopyEdit
List<String> sortedByLength = words.stream()

.sorted(Comparator.comparingInt(String::length))
                                    .collect(Collectors.toList());
```

Challenge #27

Description: Remove duplicates and sort numbers.

Solution:

```
java
CopyEdit
List<Integer> distinctSorted = numbers.stream()
                                      .distinct()
                                      .sorted()
                                      .collect(Collectors.toList());
```

Challenge #28

Description: Count how many words have length greater than 5.

Solution:

```
java
CopyEdit
long count = words.stream()
                  .filter(w -> w.length() > 5)
                  .count();
```

Challenge #29

Description: Find max and min number in a list using `Collectors.summarizingInt`.

Solution:

```
java
CopyEdit
IntSummaryStatistics stats = numbers.stream()

.collect(Collectors.summarizingInt(Integer::intValue));
int max = stats.getMax();
int min = stats.getMin();
```

Challenge #30

Description: Convert a list of integers to a set to remove duplicates.

Solution:

```
java
CopyEdit
Set<Integer> uniqueNums = numbers.stream()
                                 .collect(Collectors.toSet());
```

Challenge #31

Description: Find the longest word in a list using `reduce`.

Solution:

```
java
CopyEdit
Optional<String> longestWord = words.stream()
                                    .reduce((w1, w2) -> w1.length() >
w2.length() ? w1 : w2);
```

Challenge #32

Description: Get a list of squares of all even numbers.

Solution:

```
java
CopyEdit
List<Integer> squares = numbers.stream()
                               .filter(n -> n % 2 == 0)
                               .map(n -> n * n)
                               .collect(Collectors.toList());
```

Challenge #33

Description: Group words by their length.

Solution:

```
java
CopyEdit
Map<Integer, List<String>> groupedByLength = words.stream()

.collect(Collectors.groupingBy(String::length));
```

Challenge #34

Description: Check if all numbers are positive.

Solution:

```
java
CopyEdit
boolean allPositive = numbers.stream()
                             .allMatch(n -> n > 0);
```

Challenge #35

Description: Convert list of strings to uppercase and collect as set.

Solution:

```
java
CopyEdit
Set<String> uppercaseSet = words.stream()
                                .map(String::toUpperCase)
                                .collect(Collectors.toSet());
```

Challenge #36

Description: Find any number divisible by 7.

Solution:

```
java
CopyEdit
Optional<Integer> anyDivisibleBy7 = numbers.stream()
                                           .filter(n -> n % 7 == 0)
                                           .findAny();
```

Challenge #37

Description: Count frequency of each word in a list.

Solution:

```
java
CopyEdit
```

```
Map<String, Long> frequency = words.stream()

.collect(Collectors.groupingBy(Function.identity(),
Collectors.counting()));
```

Challenge #38

Description: Sum of all lengths of the strings in a list.

Solution:

```
java
CopyEdit
int totalLength = words.stream()
                       .mapToInt(String::length)
                       .sum();
```

Challenge #39

Description: Create a map of first character to list of words starting with that character.

Solution:

```
java
CopyEdit
Map<Character, List<String>> map = words.stream()
                                        .collect(Collectors.groupingBy(w ->
w.charAt(0)));
```

Challenge #40

Description: Sort numbers in reverse order.

Solution:

```
java
CopyEdit
List<Integer> reverseSorted = numbers.stream()
                                     .sorted(Comparator.reverseOrder())
                                     .collect(Collectors.toList());
```

Challenge #41

Description: Find the first palindrome word in a list.

Solution:

```
java
CopyEdit
```

```
Optional<String> firstPalindrome = words.stream()
                                        .filter(w -> w.equals(new
StringBuilder(w).reverse().toString()))
                                        .findFirst();
```

Challenge #42

Description: Convert a list of integers to a comma-separated string.

Solution:

```
java
CopyEdit
String result = numbers.stream()
                       .map(String::valueOf)
                       .collect(Collectors.joining(", "));
```

Challenge #43

Description: Create a map of words and their lengths.

Solution:

```
java
CopyEdit
Map<String, Integer> wordLengths = words.stream()

.collect(Collectors.toMap(Function.identity(), String::length));
```

Challenge #44

Description: Get a list of distinct squares of numbers.

Solution:

```
java
CopyEdit
List<Integer> distinctSquares = numbers.stream()
                                       .map(n -> n * n)
                                       .distinct()
                                       .collect(Collectors.toList());
```

Challenge #45

Description: Partition numbers into even and odd.

Solution:

```
java
```

```
CopyEdit
Map<Boolean, List<Integer>> partitioned = numbers.stream()

.collect(Collectors.partitioningBy(n -> n % 2 == 0));
```

Challenge #46

Description: Find the maximum number in a list using `reduce`.

Solution:

```
java
CopyEdit
Optional<Integer> maxNumber = numbers.stream()
                                     .reduce(Integer::max);
```

Challenge #47

Description: Count how many words start with a vowel.

Solution:

```
java
CopyEdit
long count = words.stream()
                  .filter(w -> w.matches("^[AEIOUaeiou].*"))
                  .count();
```

Challenge #48

Description: Flatten a list of lists of integers and collect all into one list.

Solution:

```
java
CopyEdit
List<Integer> flatList = listOfLists.stream()
                                    .flatMap(List::stream)
                                    .collect(Collectors.toList());
```

Challenge #49

Description: Find the average length of words.

Solution:

```
java
CopyEdit
OptionalDouble avgLength = words.stream()
```

```
                                .mapToInt(String::length)
                                .average();
```

Challenge #50

Description: Skip the first 3 elements and collect the rest.

Solution:

```
java
CopyEdit
List<String> skipped = words.stream()
                            .skip(3)
                            .collect(Collectors.toList());
```

Challenge #51

Description: Limit the stream to only the first 5 elements.

Solution:

```
java
CopyEdit
List<String> limited = words.stream()
                            .limit(5)
                            .collect(Collectors.toList());
```

Challenge #52

Description: Find the sum of all even numbers in a list.

Solution:

```
java
CopyEdit
int sumEven = numbers.stream()
                     .filter(n -> n % 2 == 0)
                     .mapToInt(Integer::intValue)
                     .sum();
```

Challenge #53

Description: Create a map grouping words by their length.

Solution:

```
java
CopyEdit
Map<Integer, List<String>> groupedByLength = words.stream()
```

```
.collect(Collectors.groupingBy(String::length));
```

Challenge #54

Description: Check if all numbers are positive.

Solution:

```
java
CopyEdit
boolean allPositive = numbers.stream()
                             .allMatch(n -> n > 0);
```

Challenge #55

Description: Check if any word contains the substring "test".

Solution:

```
java
CopyEdit
boolean anyContainsTest = words.stream()
                               .anyMatch(w -> w.contains("test"));
```

Challenge #56

Description: Collect the distinct characters from all words into a Set.

Solution:

```
java
CopyEdit
Set<Character> distinctChars = words.stream()
                                    .flatMapToInt(String::chars)
                                    .mapToObj(c -> (char) c)
                                    .collect(Collectors.toSet());
```

Challenge #57

Description: Find the word with the shortest length.

Solution:

```
java
CopyEdit
Optional<String> shortestWord = words.stream()

.min(Comparator.comparingInt(String::length));
```

Challenge #58

Description: Sort a list of strings in reverse alphabetical order.

Solution:

```
java
CopyEdit
List<String> reverseSorted = words.stream()
                                  .sorted(Comparator.reverseOrder())
                                  .collect(Collectors.toList());
```

Challenge #59

Description: Create a string that concatenates all the words separated by a hyphen "-".

Solution:

```
java
CopyEdit
String joined = words.stream()
                     .collect(Collectors.joining("-"));
```

Challenge #60

Description: Convert a list of strings to a map of first character to concatenated string of all words starting with that character.

Solution:

```
java
CopyEdit
Map<Character, String> map = words.stream()
                                  .collect(Collectors.groupingBy(w ->
w.charAt(0),

Collectors.mapping(Function.identity(),
                                       Collectors.joining())));
```

Challenge #61

Description: Get the average length of words in the list.

Solution:

```
java
CopyEdit
OptionalDouble averageLength = words.stream()
```

```
                                    .mapToInt(String::length)
                                    .average();
```

Challenge #62

Description: Find the maximum number in a list.

Solution:

```
java
CopyEdit
OptionalInt maxNumber = numbers.stream()
                               .mapToInt(Integer::intValue)
                               .max();
```

Challenge #63

Description: Count the number of words that start with the letter 'A'.

Solution:

```
java
CopyEdit
long countA = words.stream()
                   .filter(w -> w.startsWith("A"))
                   .count();
```

Challenge #64

Description: Remove duplicate numbers and collect the result in a list.

Solution:

```
java
CopyEdit
List<Integer> distinctNumbers = numbers.stream()
                                       .distinct()
                                       .collect(Collectors.toList());
```

Challenge #65

Description: Create a map of words to their lengths.

Solution:

```
java
CopyEdit
Map<String, Integer> wordLengths = words.stream()
```

```
.collect(Collectors.toMap(Function.identity(), String::length));
```

Challenge #66

Description: Group numbers by even or odd.

Solution:

```
java
CopyEdit
Map<Boolean, List<Integer>> evenOddGroups = numbers.stream()

.collect(Collectors.partitioningBy(n -> n % 2 == 0));
```

Challenge #67

Description: Convert a stream of words into a single uppercase string separated by commas.

Solution:

```
java
CopyEdit
String result = words.stream()
                     .map(String::toUpperCase)
                     .collect(Collectors.joining(", "));
```

Challenge #68

Description: Sort numbers in natural order and collect into a list.

Solution:

```
java
CopyEdit
List<Integer> sortedNumbers = numbers.stream()
                                     .sorted()
                                     .collect(Collectors.toList());
```

Challenge #69

Description: Get the first word in the list that contains 'cat', or return "No match" if none found.

Solution:

```
java
CopyEdit
String wordWithCat = words.stream()
```

```
                            .filter(w -> w.contains("cat"))
                            .findFirst()
                            .orElse("No match");
```

Challenge #70

Description: Create a stream of squared numbers and collect into a list.

Solution:

```
java
CopyEdit
List<Integer> squares = numbers.stream()
                                .map(n -> n * n)
                                .collect(Collectors.toList());
```

Challenge #71

Description: Use `reduce` to concatenate all words separated by a dash (`-`).

Solution:

```
java
CopyEdit
Optional<String> concatenated = words.stream()
                                     .reduce((w1, w2) -> w1 + "-" + w2);
```

Challenge #72

Description: Find the minimum number in a list using `reduce`.

Solution:

```
java
CopyEdit
Optional<Integer> minNumber = numbers.stream()
                                     .reduce(Integer::min);
```

Challenge #73

Description: Count how many numbers are divisible by 3.

Solution:

```
java
CopyEdit
long countDivBy3 = numbers.stream()
                            .filter(n -> n % 3 == 0)
                            .count();
```

Challenge #74

Description: Group words by their length.

Solution:

```
java
CopyEdit
Map<Integer, List<String>> groupedByLength = words.stream()

.collect(Collectors.groupingBy(String::length));
```

Challenge #75

Description: Create a map of word to its frequency (count of occurrences).

Solution:

```
java
CopyEdit
Map<String, Long> frequencyMap = words.stream()

.collect(Collectors.groupingBy(Function.identity(),
Collectors.counting()));
```

Challenge #76

Description: Filter words that have exactly 4 letters, convert them to uppercase, and collect to a list.

Solution:

```
java
CopyEdit
List<String> fourLetterWords = words.stream()
                                    .filter(w -> w.length() == 4)
                                    .map(String::toUpperCase)
                                    .collect(Collectors.toList());
```

Challenge #77

Description: Check if all numbers are positive.

Solution:

```
java
CopyEdit
boolean allPositive = numbers.stream()
```

```
                                .allMatch(n -> n > 0);
```

Challenge #78

Description: Check if any word contains "Java".

Solution:

```
java
CopyEdit
boolean anyJava = words.stream()
                       .anyMatch(w -> w.contains("Java"));
```

Challenge #79

Description: Collect numbers into a `Set` instead of a list to remove duplicates.

Solution:

```
java
CopyEdit
Set<Integer> numberSet = numbers.stream()
                                .collect(Collectors.toSet());
```

Challenge #80

Description: Create a stream of the lengths of the words and find the sum.

Solution:

```
java
CopyEdit
int sumLengths = words.stream()
                      .mapToInt(String::length)
                      .sum();
```

Challenge #81

Description: Use `partitioningBy` to separate numbers into even and odd groups.

Solution:

```
java
CopyEdit
Map<Boolean, List<Integer>> partitioned = numbers.stream()

.collect(Collectors.partitioningBy(n -> n % 2 == 0));
```

Challenge #82

Description: Find the longest word in the list using `max` and a comparator.

Solution:

```
java
CopyEdit
Optional<String> longestWord = words.stream()

.max(Comparator.comparingInt(String::length));
```

Challenge #83

Description: Create a comma-separated string of all words.

Solution:

```
java
CopyEdit
String joinedWords = words.stream()
                          .collect(Collectors.joining(", "));
```

Challenge #84

Description: Square each number and collect into a list.

Solution:

```
java
CopyEdit
List<Integer> squared = numbers.stream()
                               .map(n -> n * n)
                               .collect(Collectors.toList());
```

Challenge #85

Description: Filter out null values from a list and collect remaining values.

Solution:

```
java
CopyEdit
List<String> nonNullWords = words.stream()
                                 .filter(Objects::nonNull)
                                 .collect(Collectors.toList());
```

Challenge #86

Description: Create a list of distinct lengths of the words.

Solution:

```
java
CopyEdit
List<Integer> distinctLengths = words.stream()
                                     .map(String::length)
                                     .distinct()
                                     .collect(Collectors.toList());
```

Challenge #87

Description: Calculate the average length of words.

Solution:

```
java
CopyEdit
OptionalDouble avgLength = words.stream()
                                .mapToInt(String::length)
                                .average();
```

Challenge #88

Description: Convert a stream of integers to a comma-separated string.

Solution:

```
java
CopyEdit
String result = numbers.stream()
                       .map(String::valueOf)
                       .collect(Collectors.joining(","));
```

Challenge #89

Description: Find the second largest number in the list.

Solution:

```
java
CopyEdit
Optional<Integer> secondLargest = numbers.stream()
                                         .sorted(Comparator.reverseOrder())
                                         .skip(1)
                                         .findFirst();
```

Challenge #90

Description: Create a list of first letters of each word.

Solution:

```
java
CopyEdit
List<Character> firstLetters = words.stream()
                                    .map(w -> w.charAt(0))
                                    .collect(Collectors.toList());
```

Challenge #91

Description: Count how many words start with the letter 'A'.

Solution:

```
java
CopyEdit
long count = words.stream()
                  .filter(w -> w.startsWith("A"))
                  .count();
```

Challenge #92

Description: Group words by their length.

Solution:

```
java
CopyEdit
Map<Integer, List<String>> groupedByLength = words.stream()

.collect(Collectors.groupingBy(String::length));
```

Challenge #93

Description: Find the shortest word in the list.

Solution:

```
java
CopyEdit
Optional<String> shortestWord = words.stream()

.min(Comparator.comparingInt(String::length));
```

Challenge #94

Description: Check if all numbers are positive.

Solution:

```
java
CopyEdit
boolean allPositive = numbers.stream()
                             .allMatch(n -> n > 0);
```

Challenge #95

Description: Check if any word contains the substring "test".

Solution:

```
java
CopyEdit
boolean anyContainsTest = words.stream()
                               .anyMatch(w -> w.contains("test"));
```

Challenge #96

Description: Filter and collect numbers divisible by both 2 and 3.

Solution:

```
java
CopyEdit
List<Integer> divisibleBySix = numbers.stream()
                                      .filter(n -> n % 2 == 0 && n % 3 == 0)
                                      .collect(Collectors.toList());
```

Challenge #97

Description: Find the total length of all words combined.

Solution:

```
java
CopyEdit
int totalLength = words.stream()
                       .mapToInt(String::length)
                       .sum();
```

Challenge #98

Description: Convert a list of strings to a list of their reversed versions.

Solution:

```
java
```

```
CopyEdit
List<String> reversedWords = words.stream()
                                  .map(w -> new
StringBuilder(w).reverse().toString())
                                  .collect(Collectors.toList());
```

Challenge #99

Description: Sort the list of numbers in descending order.

Solution:

```
java
CopyEdit
List<Integer> sortedDesc = numbers.stream()
                                .sorted(Comparator.reverseOrder())
                                .collect(Collectors.toList());
```

Challenge #100

Description: Remove duplicates from a list of strings.

Solution:

```
java
CopyEdit
List<String> distinctWords = words.stream()
                                .distinct()
                                .collect(Collectors.toList());
```

Challenge #101

Description: Find the first word in the list that ends with the letter 'e'.
Solution:

```
java
CopyEdit
Optional<String> firstEndsWithE = words.stream()
                                     .filter(w -> w.endsWith("e"))
                                     .findFirst();
```

Challenge #102

Description: Create a comma-separated string of all the numbers.
Solution:

```
java
CopyEdit
String joinedNumbers = numbers.stream()
```

```
                                .map(String::valueOf)
                                .collect(Collectors.joining(", "));
```

Challenge #103

Description: Check if no words are empty strings.
Solution:

```
java
CopyEdit
boolean noneEmpty = words.stream()
                         .noneMatch(String::isEmpty);
```

Challenge #104

Description: Calculate the average length of the words.
Solution:

```
java
CopyEdit
OptionalDouble averageLength = words.stream()
                                    .mapToInt(String::length)
                                    .average();
```

Challenge #105

Description: Partition the numbers into even and odd.
Solution:

```
java
CopyEdit
Map<Boolean, List<Integer>> partitioned = numbers.stream()

.collect(Collectors.partitioningBy(n -> n % 2 == 0));
```

Challenge #106

Description: Find the word with the maximum number of vowels.
Solution:

```
java
CopyEdit
Optional<String> maxVowelsWord = words.stream()
    .max(Comparator.comparingInt(w -> (int) w.chars()
                                            .filter(c ->
"aeiouAEIOU".indexOf(c) != -1)
                                            .count()));
```

Challenge #107

Description: Group numbers by whether they are prime or not (true for prime).
Solution:

```
java
CopyEdit
Map<Boolean, List<Integer>> primesPartition = numbers.stream()
    .collect(Collectors.partitioningBy(MyClass::isPrime)); // You'll need
to implement isPrime(int n)
```

Challenge #108

Description: Convert list of words to their lengths, but keep only lengths > 3.
Solution:

```
java
CopyEdit
List<Integer> filteredLengths = words.stream()
                                     .map(String::length)
                                     .filter(len -> len > 3)
                                     .collect(Collectors.toList());
```

Challenge #109

Description: Sort words by their last character.
Solution:

```
java
CopyEdit
List<String> sortedByLastChar = words.stream()
                                     .sorted(Comparator.comparing(w ->
w.charAt(w.length() - 1)))
                                     .collect(Collectors.toList());
```

Challenge #110

Description: Sum all numbers, but multiply the sum by 2 at the end.
Solution:

```
java
CopyEdit
int result = numbers.stream()
                    .reduce(0, Integer::sum) * 2;
```

Challenge #111

Description: Find all distinct characters used in a list of words.
Solution:

```
java
CopyEdit
Set<Character> distinctChars = words.stream()
    .flatMapToInt(String::chars)
    .mapToObj(c -> (char) c)
    .collect(Collectors.toSet());
```

Challenge #112

Description: Check if all numbers are positive.
Solution:

```
java
CopyEdit
boolean allPositive = numbers.stream()
                             .allMatch(n -> n > 0);
```

Challenge #113

Description: Count how many words contain the letter ‘a’.
Solution:

```
java
CopyEdit
long countA = words.stream()
                   .filter(w -> w.contains("a"))
                   .count();
```

Challenge #114

Description: Create a Map of words grouped by their length.
Solution:

```
java
CopyEdit
Map<Integer, List<String>> groupedByLength = words.stream()

.collect(Collectors.groupingBy(String::length));
```

Challenge #115

Description: Find the longest word in the list.
Solution:

```
java
CopyEdit
Optional<String> longestWord = words.stream()

.max(Comparator.comparingInt(String::length));
```

Challenge #116

Description: Get a list of squares of unique numbers.
Solution:

```
java
CopyEdit
List<Integer> squares = numbers.stream()
                              .distinct()
                              .map(n -> n * n)
                              .collect(Collectors.toList());
```

Challenge #117

Description: Filter out null or empty strings from a list.
Solution:

```
java
CopyEdit
List<String> filtered = words.stream()
                             .filter(w -> w != null && !w.isEmpty())
                             .collect(Collectors.toList());
```

Challenge #118

Description: Collect words into a single string separated by semicolons.
Solution:

```
java
CopyEdit
String result = words.stream()
                     .collect(Collectors.joining(";"));
```

Challenge #119

Description: Find the sum of all even numbers.
Solution:

```
java
CopyEdit
int sumEven = numbers.stream()
                     .filter(n -> n % 2 == 0)
                     .mapToInt(Integer::intValue)
                     .sum();
```

Challenge #120

Description: Convert a list of integers to a list of strings representing their binary form.
Solution:

```
java
CopyEdit
List<String> binaryStrings = numbers.stream()
                                    .map(Integer::toBinaryString)
                                    .collect(Collectors.toList());
```

Challenge #121

Description: Group words by their first letter and count how many words in each group.
Solution:

```
java
CopyEdit
Map<Character, Long> counts = words.stream()
                                   .collect(Collectors.groupingBy(w ->
w.charAt(0), Collectors.counting()));
```

Challenge #122

Description: Create a list of the lengths of all words, sorted descending.
Solution:

```
java
CopyEdit
List<Integer> lengthsDesc = words.stream()
                                 .map(String::length)
                                 .sorted(Comparator.reverseOrder())
                                 .collect(Collectors.toList());
```

Challenge #123

Description: Check if any word ends with “ing”.
Solution:

```
java
CopyEdit
boolean anyEndsWithIng = words.stream()
                              .anyMatch(w -> w.endsWith("ing"));
```

Challenge #124

Description: Get the product of all numbers using reduce.
Solution:

```
java
CopyEdit
int product = numbers.stream()
                     .reduce(1, (a, b) -> a * b);
```

Challenge #125

Description: Find the shortest word in the list.
Solution:

```
java
CopyEdit
Optional<String> shortestWord = words.stream()

.min(Comparator.comparingInt(String::length));
```

Challenge #126

Description: Filter out duplicate words ignoring case.
Solution:

```
java
CopyEdit
List<String> distinctIgnoreCase = words.stream()
    .map(String::toLowerCase)
    .distinct()
    .collect(Collectors.toList());
```

Challenge #127

Description: Count the total number of characters in all words combined.
Solution:

```
java
CopyEdit
int totalChars = words.stream()
                      .mapToInt(String::length)
                      .sum();
```

Challenge #128

Description: Create a Map of word lengths to a comma-separated string of words of that length.
Solution:

```
java
CopyEdit
Map<Integer, String> map = words.stream()
                                .collect(Collectors.groupingBy(
                                    String::length,
                                    Collectors.mapping(Function.identity(),
Collectors.joining(", "))
                                ));
```

Challenge #129

Description: Check if all numbers are even.
Solution:

```
java
CopyEdit
boolean allEven = numbers.stream()
                         .allMatch(n -> n % 2 == 0);
```

Challenge #130

Description: Create a list of distinct words sorted ignoring case.
Solution:

```
java
CopyEdit
List<String> sortedDistinctIgnoreCase = words.stream()
                                             .distinct()

.sorted(String.CASE_INSENSITIVE_ORDER)
                                             .collect(Collectors.toList());
```

Challenge #131

Description: Collect words that start and end with the same letter into a list.
Solution:

```
java
CopyEdit
List<String> sameStartEnd = words.stream()
                                 .filter(w -> w.charAt(0) ==
w.charAt(w.length() - 1))
                                 .collect(Collectors.toList());
```

Challenge #132

Description: Find the maximum number in a stream of integers.
Solution:

```
java
CopyEdit
OptionalInt maxNumber = numbers.stream()
                               .mapToInt(Integer::intValue)
                               .max();
```

Challenge #133

Description: Convert a stream of words to a map with the word as key and its length as value.
Solution:

```
java
CopyEdit
Map<String, Integer> wordLengthMap = words.stream()

.collect(Collectors.toMap(Function.identity(), String::length));
```

Challenge #134

Description: Find the first word that contains the letter ‘z’.
Solution:

```
java
CopyEdit
Optional<String> firstWithZ = words.stream()
                                   .filter(w -> w.contains("z"))
                                   .findFirst();
```

Challenge #135

Description: Count how many words have length greater than 5.
Solution:

```
java
CopyEdit
long countLongWords = words.stream()
                           .filter(w -> w.length() > 5)
                           .count();
```

Challenge #136

Description: Create a string of all words joined with a hyphen.
Solution:

```
java
CopyEdit
String joined = words.stream()
                     .collect(Collectors.joining("-"));
```

Challenge #137

Description: Partition numbers into primes and non-primes. (Hint: Implement your own isPrime method)
Solution:

```
java
CopyEdit
Map<Boolean, List<Integer>> primesPartition = numbers.stream()
    .collect(Collectors.partitioningBy(n -> isPrime(n)));

private static boolean isPrime(int number) {
```

```
    if (number <= 1) return false;
    for (int i = 2; i <= Math.sqrt(number); i++) {
        if (number % i == 0) return false;
    }
    return true;
}
```

Challenge #138

Description: Get the distinct first letters from all words, sorted.
Solution:

```
java
CopyEdit
List<Character> firstLetters = words.stream()
                                    .map(w -> w.charAt(0))
                                    .distinct()
                                    .sorted()
                                    .collect(Collectors.toList());
```

Challenge #139

Description: Create a map from word length to the count of words with that length.
Solution:

```
java
CopyEdit
Map<Integer, Long> lengthCount = words.stream()

.collect(Collectors.groupingBy(String::length, Collectors.counting()));
```

Challenge #140

Description: Filter numbers that are multiples of 3 or 5, then sum them.
Solution:

```
java
CopyEdit
int sum = numbers.stream()
                 .filter(n -> n % 3 == 0 || n % 5 == 0)
                 .mapToInt(Integer::intValue)
                 .sum();
```

Challenge #141

Description: Get the top 3 longest words from the list.
Solution:

```
java
CopyEdit
```

```
List<String> top3Longest = words.stream()
                                .sorted((w1, w2) ->
Integer.compare(w2.length(), w1.length()))
                                .limit(3)
                                .collect(Collectors.toList());
```

Challenge #142

Description: Get all words with only lowercase letters.
Solution:

```
java
CopyEdit
List<String> lowercaseWords = words.stream()
                                   .filter(w -> w.equals(w.toLowerCase()))
                                   .collect(Collectors.toList());
```

Challenge #143

Description: Find if any word is a palindrome (same forwards and backwards).
Solution:

```
java
CopyEdit
boolean hasPalindrome = words.stream()
                             .anyMatch(w -> new
StringBuilder(w).reverse().toString().equals(w));
```

Challenge #144

Description: Group integers by even and odd and find the average in each group.
Solution:

```
java
CopyEdit
Map<Boolean, Double> avgEvenOdd = numbers.stream()
    .collect(Collectors.partitioningBy(n -> n % 2 == 0,
             Collectors.averagingInt(Integer::intValue)));
```

Challenge #145

Description: Return a list of distinct characters used in all words.
Solution:

```
java
CopyEdit
List<Character> distinctChars = words.stream()
    .flatMapToInt(String::chars)
    .mapToObj(c -> (char) c)
    .distinct()
```

```
    .collect(Collectors.toList());
```

Challenge #146

Description: Get the most frequent word in a list.
Solution:

```
java
CopyEdit
String mostFrequent = words.stream()
    .collect(Collectors.groupingBy(Function.identity(),
Collectors.counting()))
    .entrySet().stream()
    .max(Map.Entry.comparingByValue())
    .map(Map.Entry::getKey)
    .orElse(null);
```

Challenge #147

Description: Convert all strings in the list to kebab-case (`hello-world-style`).
Solution:

```
java
CopyEdit
List<String> kebabWords = words.stream()
    .map(w -> w.toLowerCase().replace(" ", "-"))
    .collect(Collectors.toList());
```

Challenge #148

Description: Calculate total length of all words starting with ‘s’.
Solution:

```
java
CopyEdit
int totalLength = words.stream()
                       .filter(w -> w.startsWith("s"))
                       .mapToInt(String::length)
                       .sum();
```

Challenge #149

Description: Convert the stream of words into a `Set` of words that appear only once.
Solution:

```
java
CopyEdit
Set<String> uniqueWords = words.stream()
    .collect(Collectors.groupingBy(Function.identity(),
Collectors.counting()))
```

```
    .entrySet().stream()
    .filter(e -> e.getValue() == 1)
    .map(Map.Entry::getKey)
    .collect(Collectors.toSet());
```

Challenge #150

Description: Count the number of vowels in each word and return a map.
Solution:

```
java
CopyEdit
Map<String, Long> wordVowelCount = words.stream()
    .collect(Collectors.toMap(
        Function.identity(),
        w -> w.chars().filter(c -> "aeiouAEIOU".indexOf(c) >= 0).count()
    ));
```
