1. **Write a Java program that takes a list of integers, print out the even numbers.**

```
List<Integer> numbers = Arrays.asList(1, 2, 3, 4, 5, 6, 7, 8, 9, 10);
numbers.stream()
    .filter(n -> n % 2 == 0)
    .forEach(System.out::println);
```

2. **Given a list of integers, filter out the even numbers and collect the remaining odd numbers into a new list.**

```
List<Integer> numbers = Arrays.asList(1, 2, 3, 4, 5, 6, 7, 8, 9, 10);
List<Integer> oddNumbers = numbers.stream()
                    .filter(n -> n % 2 != 0)
                    .collect(Collectors.toList());
```

3. **Given a list of strings, convert all strings to uppercase and collect them into a new list.**

```
        List<String> words = Arrays.asList("apple", "banana", "cherry");

        List<String> upperCaseWords = words.stream()

                        .map(String::toUpperCase)

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

4. **Given a list of integers, find the maximum value using the reduce method.**

```
List<Integer> numbers = Arrays.asList(3, 5, 7, 2, 8, 10);
Optional<Integer> maxNumber = numbers.stream()
                    .reduce(Integer::max);
```

5. **Create a method that takes a list of integers and find the product of all elements in the list.**

```
List<Integer> numbers = Arrays.asList(1, 2, 3, 4, 5);

int product = numbers.stream()

            .reduce(1, (a, b) -> a * b);
```

6. **Write a Java function that takes a list of strings, converts each string to its length, and then returns the sum of all lengths using mapToInt and sum.**

```
List<String> words = Arrays.asList("apple", "banana", "cherry");
int totalLength = words.stream()
            .mapToInt(String::length)
            .sum();
```

7. **Write a method that takes a list of strings and returns a list of strings that start with the letter "A" using filter and collect(Collectors.toList()).**

```
List<String> words = Arrays.asList("apple", "banana", "avocado", "apricot");

List<String> wordsStartingWithA = words.stream()
```

```
                        .filter(word -> word.startsWith("A"))
.collect(Collectors.toList());
```

8. **Create a function that takes a list of doubles representing temperatures in Celsius and returns the highest temperature using mapToDouble and max.**

```
List<Double> temperatures = Arrays.asList(32.5, 36.7, 29.8, 40.2);
OptionalDouble maxTemperature = temperatures.stream()
                        .mapToDouble(Double::doubleValue)
                        .max();
```

9. **Write a method that takes a list of Customer objects and return list of customers.**

```
  customers.stream().collect(Collectors.toList()
```

10. **Write a function that takes a list of Customer objects and returns list of customers who have more than 1000 reward points.**

```
List<Customer> customers = getCustomerList();
List<Customer> filteredCustomers = customers.stream()
                        .filter(customer -> customer.getRewardPoints() > 1000)
                        .collect(Collectors.toList());
```