

**1) Given a list of integers, find out all the even numbers that exist in the list using Stream functions?**

```
import java.util.*;
import java.util.stream.*;

public class EvenNumber{
    public static void main(String args[]) {
        List<Integer> list = Arrays.asList(10,15,8,49,25,98,32);
        list.stream()
            .filter(n -> n%2 == 0)
            .forEach(System.out::println);

        /* or can also try below method */

        Map<Boolean, List<Integer>> list = Arrays.stream(nums).boxed()
            .collect(Collectors.partitioningBy(num -> num % 2 == 0));
        System.out.println(list);
    }
}
```

*Output:*  
10, 8, 98, 32

**2) Given a list of integers, find out all the numbers starting with 1 using Stream functions?**

```
import java.util.*;
import java.util.stream.*;

public class NumberStartingWithOne{
    public static void main(String args[]) {
        List<Integer> myList = Arrays.asList(10,15,8,49,25,98,32);
        myList.stream()
            .map(s -> s + "") // Convert integer to String
            .filter(s -> s.startsWith("1"))
            .forEach(System.out::println);

        /* or can also try below method */

        List<String> list = Arrays.stream(arr).boxed()
            .map(s -> s + "")
            .filter(s -> s.startsWith("1"))
            .collect(Collectors.toList());

        System.out.println(list);
    }
}
```

```
}
```

*Output:*

*10, 15*

### 3) How to find duplicate elements in a given integers list in java using Stream functions?

```
import java.util.*;
import java.util.stream.*;

public class DuplicateElements {
    public static void main(String args[]) {
        List<Integer> myList = Arrays.asList(10,15,8,49,25,98,98,32,15);
        Set<Integer> set = new HashSet();
        myList.stream()
            .filter(n -> !set.add(n))
            .forEach(System.out::println);
    }
}
```

*Output:*

*98, 15*

// Or you can also try using distinct() keyword

```
public static void getDataWithoutDuplicates() {
    List<Integer> myList = Arrays.asList(1, 1, 85, 6, 2, 3, 65, 6, 45, 45, 5662, 2582, 2, 2, 266,
    666, 656);
    myList.stream().distinct().forEach(noDuplicateData ->
    System.out.println(noDuplicateData));
}
```

*Output : 1 85 6 2 3 65 45 5662 2582 266 666 656*

//Or you can also use below

```
public static void getDataWithoutDuplicates() {
    List<Integer> myList = Arrays.asList(1, 1, 85, 6, 2, 3, 65, 6, 45, 45, 5662, 2582, 2, 2, 266,
    666, 656);
    Set<Integer> set = new HashSet<>(myList);

    // Convert the set back to a list if needed
```

```

List<Integer> uniqueData = set.stream().collect(Collectors.toList());

// Print the unique elements
uniqueData.forEach(System.out::println);
}

```

*Output : 1 65 2 3 6 266 45 656 85 2582 666 5662*

```

/* or can also try below single line code */
List<Integer> list = Arrays.stream(arr).boxed().distinct().collect(Collectors.toList());

```

#### 4) Given the list of integers, find the first element of the list using Stream functions?

```

import java.util.*;
import java.util.stream.*;

public class FindFirstElement{
    public static void main(String args[]) {
        List<Integer> myList = Arrays.asList(10,15,8,49,25,98,98,32,15);
        myList.stream()
            .findFirst()
            .ifPresent(System.out::println);

        /* or can also try below single line code */
        Arrays.stream(arr).boxed().findFirst().ifPresent(System.out::print);
    }
}

```

*Output:*  
10

#### 5) Given a list of integers, find the total number of elements present in the list using Stream functions?

```

import java.util.*;
import java.util.stream.*;

public class FindTheTotalNumberOfElements{
    public static void main(String args[]) {
        List<Integer> myList = Arrays.asList(10,15,8,49,25,98,98,32,15);
        long count = myList.stream()
            .count();
        System.out.println(count);

        /* or can also try below line code */
        Arrays.stream(arr).boxed().count();
    }
}

```

Output:

9

**6) Given a list of integers, find the maximum value element present in it using Stream functions?**

```
import java.util.*;
import java.util.stream.*;

public class FindMaxElement{
    public static void main(String args[]) {
        List<Integer> myList = Arrays.asList(10,15,8,49,25,98,98,32,15);
        int max = myList.stream()
            .max(Integer::compare)
            .get();
        System.out.println(max);

        /* or we can try using below way */

        int maxdata = Arrays.stream(arr).boxed()
            .max(Comparator.naturalOrder()).get();

        System.out.println(maxdata);
    }
}
```

Output:

98

**7) Given a String, find the first non-repeated character in it using Stream functions?**

```
import java.util.*;
import java.util.stream.*;
import java.util.function.Function;

public class FirstNonRepeated{
    public static void main(String args[]) {
        String input = "Java articles are Awesome";

        Character result = input.chars() // Stream of String
            .mapToObj(s -> Character.toLowerCase(Character.valueOf((char) s))) // First convert to
            Character object and then to lowercase
            .collect(Collectors.groupingBy(Function.identity(), LinkedHashMap::new,
            Collectors.counting())) //Store the chars in map with count
            .entrySet()
            .stream()
            .filter(entry -> entry.getValue() == 1L)
            .map(entry -> entry.getKey())
```

```

        .findFirst()
        .get();
    System.out.println(result);

    /* or can also try using */

    input.chars().mapToObj(c -> (char) c)
        .filter(ch -> input.indexOf(ch) == input.lastIndexOf(ch))
        .findFirst().orElse(null);
    }
}

```

**Output:**

j

### 8) Given a String, find the first repeated character in it using Stream functions?

```

import java.util.*;
import java.util.stream.*;
import java.util.function.Function;

public class FirstRepeated{
    public static void main(String args[]) {
        String input = "Java Articles are Awesome";

        Character result = input.chars() // Stream of String
            .mapToObj(s -> Character.toLowerCase(Character.valueOf((char) s))) //
            First convert to Character object and then to lowercase
            .collect(Collectors.groupingBy(Function.identity(), LinkedHashMap::new,
            Collectors.counting())) //Store the chars in map with count
            .entrySet()
            .stream()
            .filter(entry -> entry.getValue() > 1L)
            .map(entry -> entry.getKey())
            .findFirst()
            .get();

        System.out.println(result);

        /* or can also try */

        Set<Character> seenCharacters = new HashSet<>();

        return input.chars()
            .mapToObj(c -> (char) c)
            .filter(c -> !seenCharacters.add(c))
            .findFirst()
            .orElse(null);
    }
}

```

```
}  
}
```

*Output:*

*a*

**9) Given a list of integers, sort all the values present in it using Stream functions?**

```
import java.util.*;  
import java.util.stream.*;  
import java.util.function.Function;  
  
public class SortValues{  
    public static void main(String args[]) {  
        List<Integer> myList = Arrays.asList(10,15,8,49,25,98,98,32,15);  
  
        myList.stream()  
            .sorted()  
            .forEach(System.out::println);  
  
        /* Or can also try below way */  
  
        Arrays.stream(arr).boxed().sorted().collect(Collectors.toList())  
    }  
}
```

*Output:*

*8  
10  
15  
15  
25  
32  
49  
98  
98*

**10) Given a list of integers, sort all the values present in it in descending order using Stream functions?**

```
import java.util.*;  
import java.util.stream.*;  
import java.util.function.Function;  
  
public class SortDescending{  
    public static void main(String args[]) {  
        List<Integer> myList = Arrays.asList(10,15,8,49,25,98,98,32,15);
```

```

        myList.stream()
            .sorted(Collections.reverseOrder())
            .forEach(System.out::println);
    }
}

```

*Output:*

```

98
98
49
32
25
15
15
10
8

```

**11) Given an integer array nums, return true if any value appears at least twice in the array, and return false if every element is distinct.**

```

public boolean containsDuplicate(int[] nums) {
    List<Integer> list = Arrays.stream(nums)
        .boxed()
        .collect(Collectors.toList());
    Set<Integer> set = new HashSet<>(list);
    if(set.size() == list.size()) {
        return false;
    }
    return true;

    /* or can also try below way */
    Set<Integer> setData = new HashSet<>();
    return Arrays.stream(nums)
        .anyMatch(num -> !setData.add(num));
}

```

*Input: nums = [1,2,3,1]*

*Output: true*

*Input: nums = [1,2,3,4]*

*Output: false*

**12) How will you get the current date and time using Java 8 Date and Time API?**

```

class Java8 {
    public static void main(String[] args) {
        System.out.println("Current Local Date: " + java.time.LocalDate.now());
    }
}

```

```

//Used LocalDate API to get the date
System.out.println("Current Local Time: " + java.time.LocalTime.now());
//Used LocalTime API to get the time
System.out.println("Current Local Date and Time: " + java.time.LocalDateTime.now());
//Used LocalDateTime API to get both date and time
}
}

```

### 13) Write a Java 8 program to concatenate two Streams?

```

import java.util.Arrays;
import java.util.List;
import java.util.stream.Stream;

public class Java8 {
    public static void main(String[] args) {

        List<String> list1 = Arrays.asList("Java", "8");
        List<String> list2 = Arrays.asList("explained", "through", "programs");

        Stream<String> concatStream = Stream.concat(list1.stream(), list2.stream());

        // Concatenated the list1 and list2 by converting them into Stream

        concatStream.forEach(str -> System.out.print(str + " "));

        // Printed the Concatenated Stream

    }
}

```

### 14) Java 8 program to perform cube on list elements and filter numbers greater than 50.

```

import java.util.*;

public class Main {
    public static void main(String[] args) {
        List<Integer> integerList = Arrays.asList(4,5,6,7,1,2,3);
        integerList.stream()
            .map(i -> i*i*i)
            .filter(i -> i>50)
            .forEach(System.out::println);
    }
}

```

Output:

64  
125



216

343

**15) Write a Java 8 program to sort an array and then convert the sorted array into Stream?**

```
import java.util.Arrays;

public class Java8 {

    public static void main(String[] args) {
        int arr[] = { 99, 55, 203, 99, 4, 91 };
        Arrays.parallelSort(arr);
        // Sorted the Array using parallelSort()

        Arrays.stream(arr).forEach(n > System.out.print(n + " "));
        /* Converted it into Stream and then
        printed using forEach */
    }
}
```

**16) How to use map to convert object into Uppercase in Java 8?**

```
public class Java8 {

    public static void main(String[] args) {
        List<String> nameLst = names.stream()
            .map(String::toUpperCase)
            .collect(Collectors.toList());
        System.out.println(nameLst);
    }
}
```

*output:*

*AA, BB, CC, DD*

**17) How to convert a List of objects into a Map by considering duplicated keys and store them in sorted order?**

```
public class TestNotes {

    public static void main(String[] args) {

        List<Notes> noteLst = new ArrayList<>();
        noteLst.add(new Notes(1, "note1", 11));
        noteLst.add(new Notes(2, "note2", 22));
        noteLst.add(new Notes(3, "note3", 33));
        noteLst.add(new Notes(4, "note4", 44));
        noteLst.add(new Notes(5, "note5", 55));
```

```

noteLst.add(new Notes(6, "note4", 66));

Map<String, Long> notesRecords = noteLst.stream()
    .sorted(Comparator
        .comparingLong(Notes::getTagId)
        .reversed()) // sorting is based on TagId 55,44,33,22,11
    .collect(Collectors.toMap
        (Notes::getTagName, Notes::getTagId,
        (oldValue, newValue) -> oldValue,LinkedHashMap::new));
// consider old value 44 for dupilcate key
// it keeps order
    System.out.println("Notes : " + notesRecords);
}
}

```

### 18) How to count each element/word from the String ArrayList in Java8?

```

public class TestNotes {

    public static void main(String[] args) {
        List<String> names = Arrays.asList("AA", "BB", "AA", "CC");
        Map<String,Long> namesCount = names
            .stream()
            .collect(
                Collectors.groupingBy(
                    Function.identity(), Collectors.counting()));
        System.out.println(namesCount);
    }
}

```

*Output:*

{CC=1, BB=1, AA=2}

### 19) How to find only duplicate elements with its count from the String ArrayList in Java8?

```

public class TestNotes {

    public static void main(String[] args)
    List<String> names = Arrays.asList("AA", "BB", "AA", "CC");
    Map<String,Long> namesCount = names
        .stream()
        .filter(x->Collections.frequency(names, x)>1)
        .collect(Collectors.groupingBy
            (Function.identity(), Collectors.counting()));
    System.out.println(namesCount);

    /*or you can also try using */
}

```

```

Map<String, Long> namesCount = names.stream()
    .collect(Collectors.groupingBy(Function.identity(), Collectors.counting()))
    .entrySet()
    .stream()
    .filter(entry -> entry.getValue() > 1)
    .collect(Collectors.toMap(Map.Entry::getKey, Map.Entry::getValue));
}
}

```

*Output:*

*{AA=2}*

## 20) How to check if list is empty in Java 8 using Optional, if not null iterate through the list and print the object?

```

Optional.ofNullable(noteLst)
    .orElseGet(Collections::emptyList) // creates empty immutable list: [] in case noteLst is
null
    .stream().filter(Objects::nonNull) //loop throgh each object and consider non null
objects
    .map(note -> Notes::getTagName) // method reference, consider only tag name
    .forEach(System.out::println); // it will print tag names

```

## 21) Write a Program to find the Maximum element in an array?

```

public static int findMaxElement(int[] arr) {
    return Arrays.stream(arr).max().getAsInt();
}

```

*Input: 12,19,20,88,00,9*

*output: 88*

## 22) Write a program to print the count of each character in a String?

```

public static void findCountOfChars(String s) {
    Map<String, Long> map = Arrays.stream(s.split(""))
        .map(String::toLowerCase)
        .collect(Collectors
            .groupingBy(str -> str,
                LinkedHashMap::new, Collectors.counting()));
}

```

*// or you can also try using Function.identity() instead of LinkedHashMap*

```

Map<String, Long> mapObject = Arrays.stream(s.split(""))
    .map(String::toLowerCase)
    .collect(Collectors.groupingBy(Function.identity(), Collectors.counting()));

```

}

*Input: String s = "string data to count each character";*

*Output: {s=1, t=5, r=3, i=1, n=2, g=1, =5, d=1, a=5, o=2, c=4, u=1, e=2, h=2}*