* **Stream API**

**Employee(id:1,name:"ABC",salary:10000,age:22,location: “ahmd”,Designation : “Java”);**

*Note :- Please Create multiple employees for better execution of your program*

=> Find out all employee where employee have a salary > 10000 and location is ahmd.

=> Collect employee where employee name start with “A”.

=> Find out no. of location where location is unique(same).

=> Find out location where location is unique.

=> Sort list based on Employee Name(ASC and DESC order).  
=> Sum of all employee salary.

=> Average of all employee salary.

=> List out all employee where designation is “Java”.

=> List out all employee where designation is “Java” and salary is 10000 and age is more than 20.

=> Select employee where employee have a Max salary.

=> Select employee where employee have a Min salary.

=> Find second elements from list.

### => https://www.java67.com/2014/04/java-8-stream-examples-and-tutorial.html

### Counting Empty String

### Count String whose length is more than three

### Count number of String which starts with "a"

### Remove all empty Strings from List

### Create a List with String more than 2 characters

### Convert String to uppercase and Join them with coma

### Create a List of the square of all distinct numbers

### Get count, min, max, sum, and the average for numbers

**=> https://blog.devgenius.io/15-practical-exercises-help-you-master-java-stream-api-3f9c86b1cf82**

## Obtain a list of products belongs to category “Books” with price > 100

## Obtain a list of order with products belong to category “Baby”

## Obtain a list of product with category = “Toys” and then apply 10% discount

## Obtain a list of products ordered by customer of tier 2 between 01-Feb-2021 and 01-Apr-2021

## Get the cheapest products of “Books” category

## Get the 3 most recent placed order

## Get a list of orders which were ordered on 15-Mar-2021, log the order records to the console and then return its product list

## Calculate total lump sum of all orders placed in Feb 2021

## Calculate order average payment placed on 14-Mar-2021

## Obtain a collection of statistic figures (i.e. sum, average, max, min, count) for all products of category “Books”

## Obtain a data map with order id and order’s product count

## Produce a data map with order records grouped by customer

## Produce a data map with order record and product total sum

## Obtain a data map with list of product name by category

## Get the most expensive product by category

**=> https://github.com/shekhargulati/java8-the-missing-tutorial/blob/master/03-streams.md**

import java.time.LocalDate;

import java.util.\*;

public class Task {

private final String id;

private final String title;

private final TaskType type;

private final LocalDate createdOn;

private boolean done = false;

private Set<String> tags = new HashSet<>();

private LocalDate dueOn;

// removed constructor, getter, and setter for brevity

}

Task task1 = new Task("Read Version Control with Git book", TaskType.READING, LocalDate.of(2015, Month.JULY, 1)).addTag("git").addTag("reading").addTag("books");

Task task2 = new Task("Read Java 8 Lambdas book", TaskType.READING, LocalDate.of(2015, Month.JULY, 2)).addTag("java8").addTag("reading").addTag("books");

Task task3 = new Task("Write a mobile application to store my tasks", TaskType.CODING, LocalDate.of(2015, Month.JULY, 3)).addTag("coding").addTag("mobile");

Task task4 = new Task("Write a blog on Java 8 Streams", TaskType.WRITING, LocalDate.of(2015, Month.JULY, 4)).addTag("blogging").addTag("writing").addTag("streams");

Task task5 = new Task("Read Domain Driven Design book", TaskType.READING, LocalDate.of(2015, Month.JULY, 5)).addTag("ddd").addTag("books").addTag("reading");

List<Task> tasks = Arrays.asList(task1, task2, task3, task4, task5);

### Find all reading task titles sorted by their creation date

### Find distinct tasks

### Find top 5 reading tasks sorted by creation date

### Count all reading tasks

### Find all unique tags from all tasks

### Check if all reading tasks have tag books

### Creating a summary of all titles

* **Local Date Time**

**=> <https://javarevisited.blogspot.com/2015/03/20-examples-of-date-and-time-api-from-Java8.html>**

### How to get today's date in Java 8

### How to get a current day, month, and year in Java 8

### How to get a particular date in Java 8

### How to check if two dates are equal in Java 8

### How to check for recurring events e.g. birthday in Java 8

### How to get current Time in Java 8

### How to add hours in time

### How to find Date after 1 week

### Date before and after 1 year

### How to see if a date is before or after another date in Java

### Dealing with time zones in Java 8

### How to check Leap Year in Java 8

### How many days, the month between two dates

### Date and Time with timezone offset

### How to parse the date in Java using custom formatting

### Conversion String to locaDateTime and localDateTime to String

### Difference between two dates.

Difference between to times in HH:mm:ss format

* **Base 64**

Encode String

Decode Encoded String