**1: how stream api works**

**A screenshot of a computer

Description automatically generated**

**Intermediate operation :**  convert stream into another stream

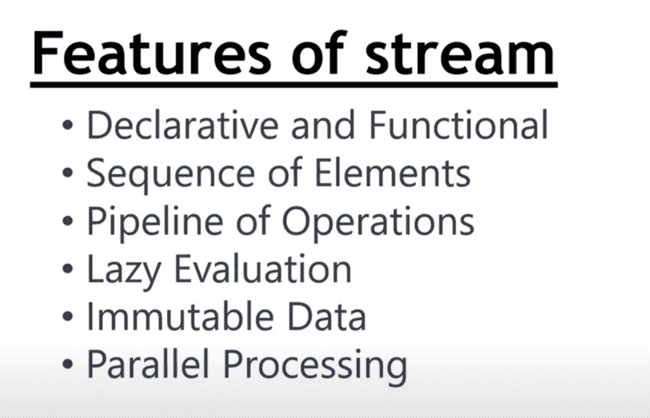
Terminal operation : bring the required result.

**Note** : when terminal operation called then only intermediate operation executed

A diagram of a program

Description automatically generated

**2- features of stream**

****

**Pipeline of operation :** method chaining will be there in stream

**Sequence of element** : it can be created from various sources collection, arrays etc.

Each element in stream of can be any object type

**Lazy evaluation** : stream operation will not be performed until the terminal operation is executed

**Immutable data** : whatever operation will perform they will create a new stream out of that.

Ensure that original data remain unchanged.

**Parallel stream** : original stream can be divided into smaller chunk and process it concurrently

**3 : create java 8 instance stream**

There are multiple ways of creating stream.

A screen shot of a computer program

Description automatically generated

A computer screen with white and orange text

Description automatically generated

**4 : java stream vs java collection**

**A white background with black text

Description automatically generated**

**How data**  is represented in stream and collection ?

**Collection** : group of object

Stream are just a abstraction layer and they do not modify source data

**Lazy evaluation** : all intermediate operation are not executed and help to process large data set.

Collection can be modify while stream is immutable.

We are able to do lot of work with few lines of code

**5 : java 8 collect method**

**Collect :**  terminal operation

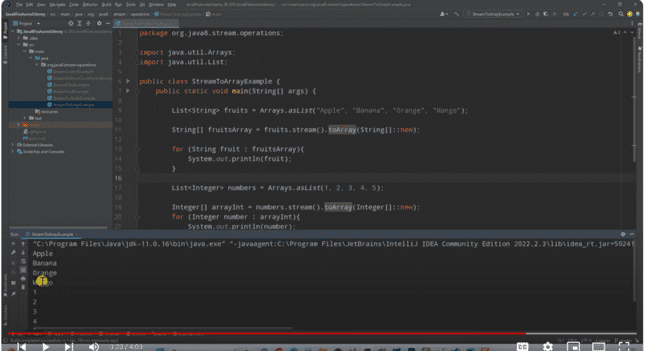
Collecters.joining : convert element into string

A screenshot of a computer program

Description automatically generated

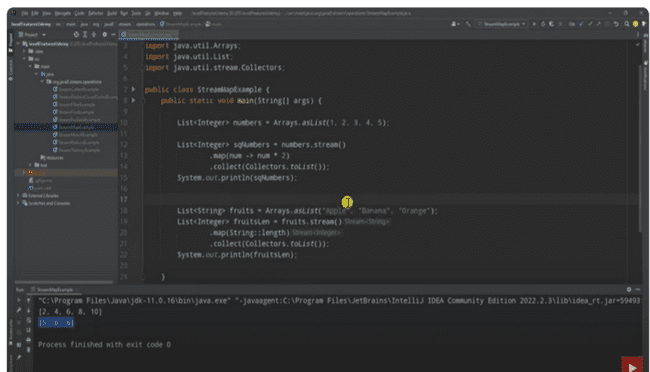
**6 : java 8 stream toArray()**

**It** convert stream to array



**7 : java 8 stream map method**

**Map : it**  transform each element into new element



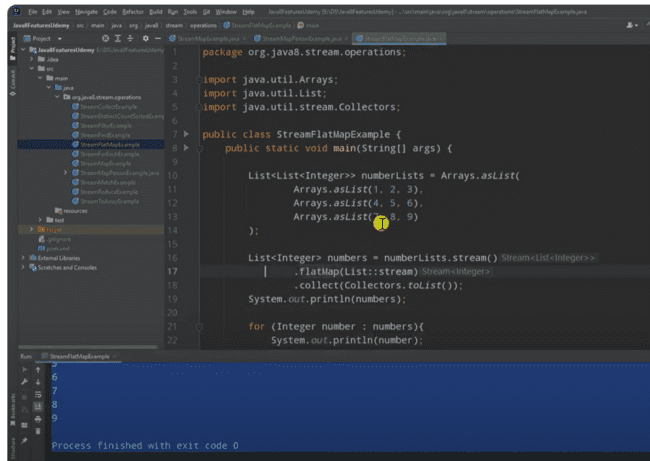
Rather than string and integer if these are some custom object like Employee then how we will use it.

**8 ) java 8 stream flatMap method**

**Question :**  when we should go for map and when we should go for flatMap.

**We should go for map** if we have given a list, set or map of element.

But if we have given list of list, set of set or map of map then we should go for **flatMap**



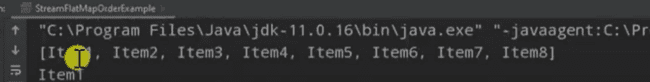
**9 :java 8 stream flatMap with custom object**

**A screen shot of a computer

Description automatically generated**



Output

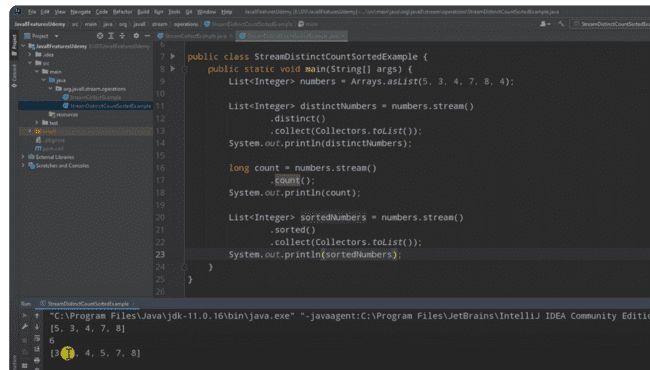


flatMap help in flattening the data structure

**10) java 8 distinct, count,sorted**

**Distinct()—**it will give unique values

Count : how many number of element present ---- it is terminal operation and return long



**11—stream filter method**

**Filter :**  intermediate operation

A screenshot of a computer program

Description automatically generated

**Note : filter is going to take predicate**

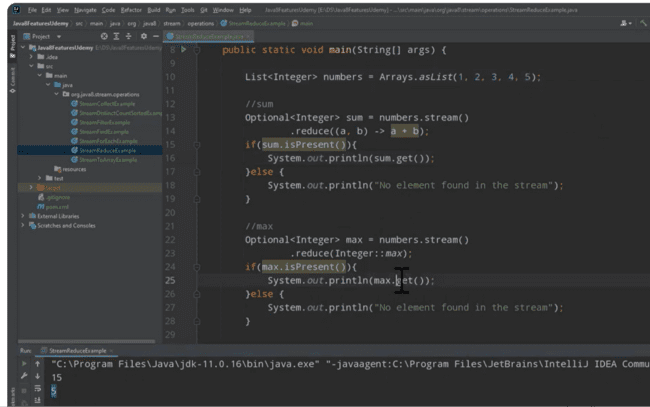
**12 : stream reduce method**

**Reduce :**

**we use this**  method when we have to reduce to certain value

example :-----sum of all number , multiply of all number

do a certain operation on all values and reduce to single values.



**13 ) allMatch() , anyMatch(), noneMatch()**

**Note :**

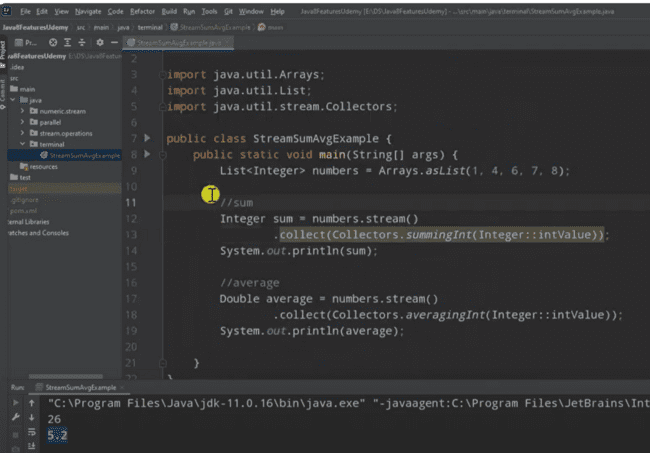
These method will take a **predicate**

A screenshot of a computer program

Description automatically generated

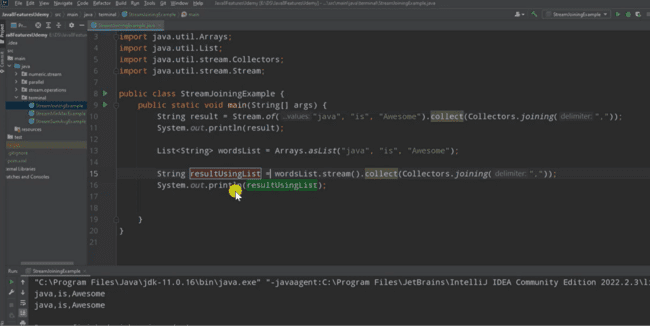
**14 :- sum() , avg() using collect**

**Method : summingInt**

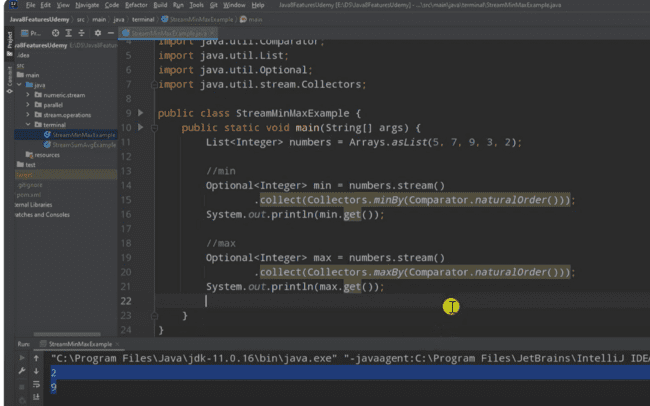
****

**15 stream joining method -- terminal method**

**Let**  say we have number of string and we want to join them on certain delimeter.

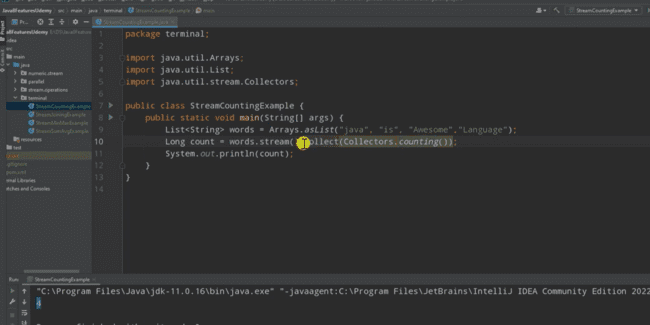


**16 minBy(), maxBy() – terminal operation**

****

**17 :- counting method : terminal operation**

**Number** of elements whether it is an array or list



**18 : groupingBy method :---- terminal operation**

**A screenshot of a computer

Description automatically generated**

**A screenshot of a computer

Description automatically generated**

**19 : groupingBy type 2**

**Objective :**  groupBy age and in the value we want to return person name.

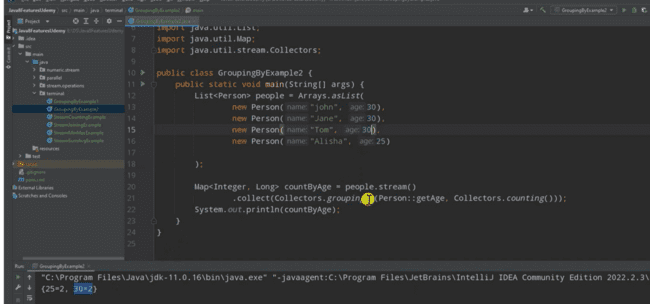
A screenshot of a computer

Description automatically generated

**20 ) groupingBy method on custom object**

**Objective: grou**pingBy on certain property and want to count how many element on respective group

We want hashmap value as count



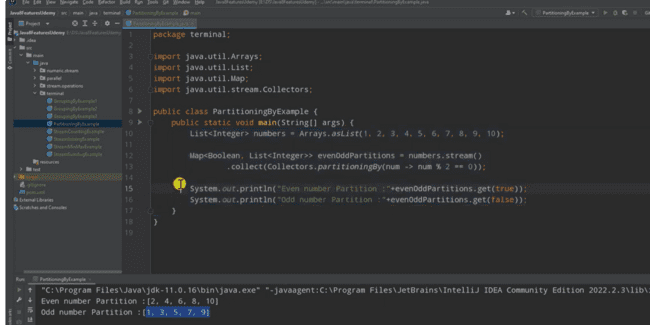
groupingBy always return map, first one is key and second one is value. If we don’t pass the value, it will pass entire object.

**21 : partitioningBy method --**  it takes predicate

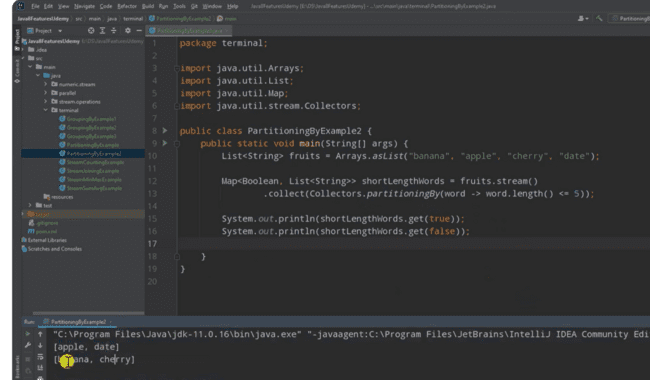
**Note : part**itioningBy create partition based on our logic

Will take list of integer and will create even and odd partition

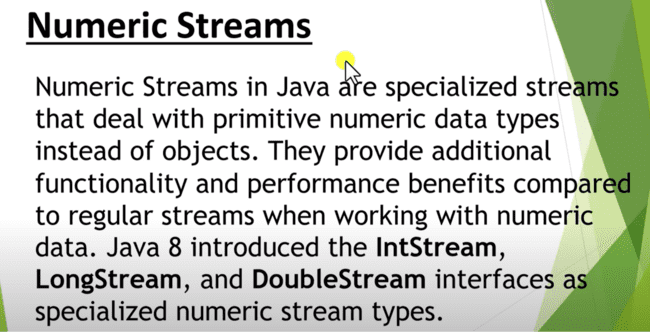
**It return a map and key is always a boolean**

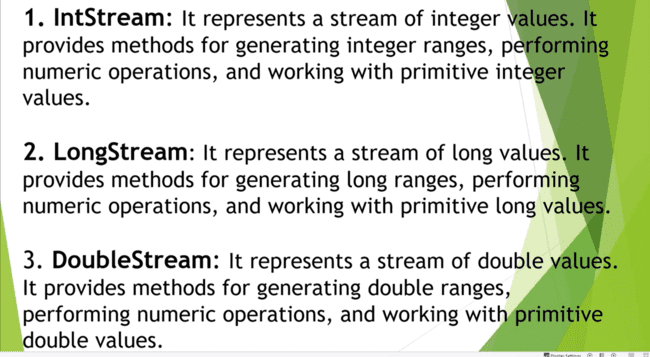
****

**22 ) partitioningBy type2 with string data**

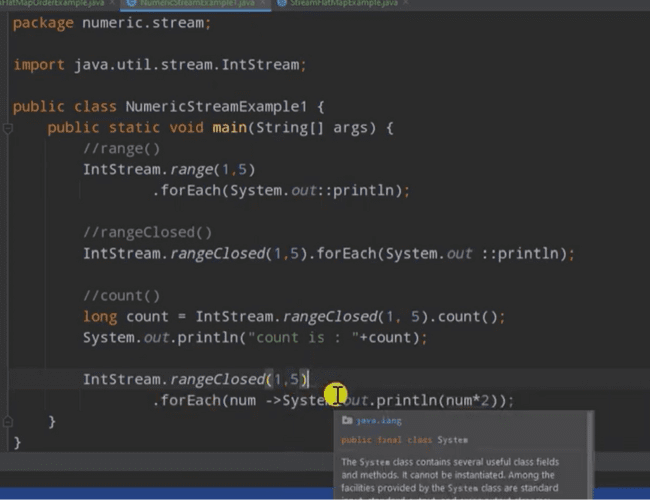
****

**23 : numeric streams**

****

****

**24 : numeric stream, range(), rangeClosed() , count(), forEach()**

****

**25: numeric stream sum,max,min avg**

**A screenshot of a computer program

Description automatically generated**

**Map : it transform from one stream to other stream**

**26 : numeric stream boxing, unboxing , mapToInt**

**Objective :** convert a stream into intStream and do the operation

Boxing is the process of converting a primitive datatype into an object wrapper datatype, and unboxing is the process of converting a value from an object wrapper type back to the native primitive value

MapToInt : convert stream into intStream ( since it is dealing with values so we need to box it).

A screenshot of a computer program

Description automatically generated

**27 : numeric stream mapToObj(), mapToDouble(), mapToLong()**

**Objective :**  how to convert IntStream into string stream

This we converted into stream of string.

A computer screen with white text

Description automatically generated

A computer screen shot of a program code

Description automatically generated

**28 : introduction to java 8 parallel stream**

A text on a white background

Description automatically generated

A screenshot of a computer

Description automatically generated

**29) how java 8 parallel stream works**

**A close-up of a white background

Description automatically generated**

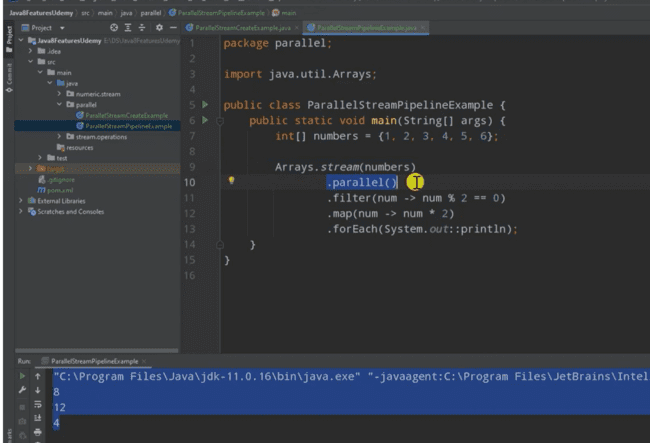
**30 ) create java 8 parallel stream**

**In parallel stream**  there is no sequenceA screenshot of a computer program

Description automatically generated

**31 : build a java 8 stream pipeline**

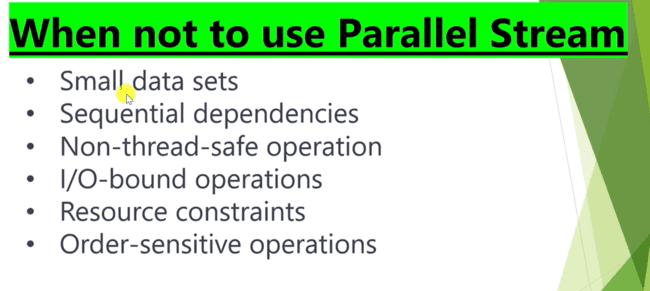
**Note :**  on parallel stream also we can chain method calls



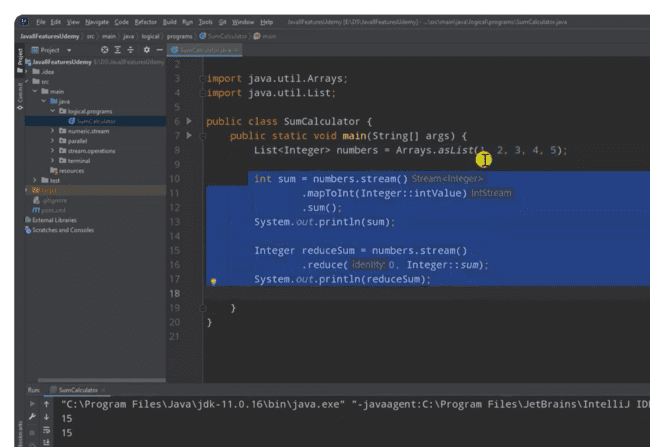
**32 : comparing performance of java 8 sequential vs parallel stream**

**Will** see what size of data sequential stream perform better and what size of data parallel stream perform better.

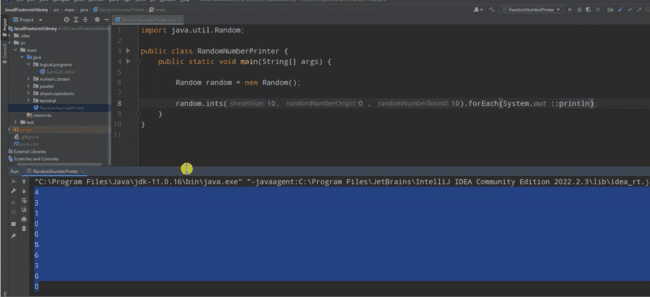
**33: when not to use parallel stream**

****

**34 : find sum of all element of arraylist**

****

**35 : Random number generator**

****

**36 : find duplicates from list/array of numbers**

**A screenshot of a computer

Description automatically generated**

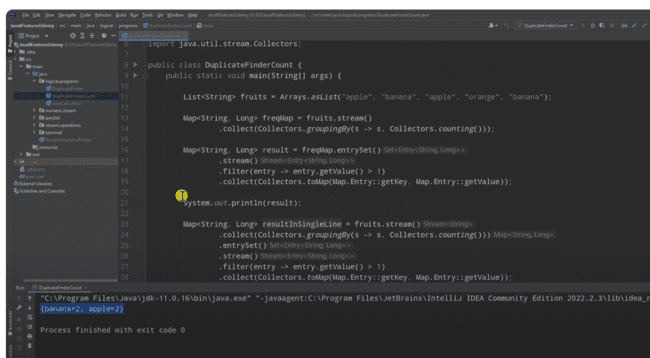
**A screenshot of a computer program

Description automatically generated**

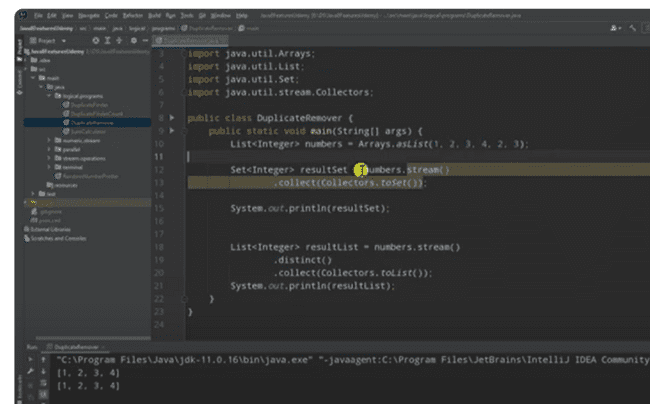
**37 find duplicates from list/array in custom object**

**Objective:**

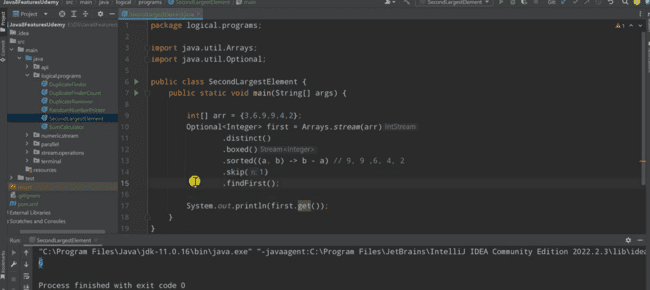
Print all duplicate elements wit key and value.

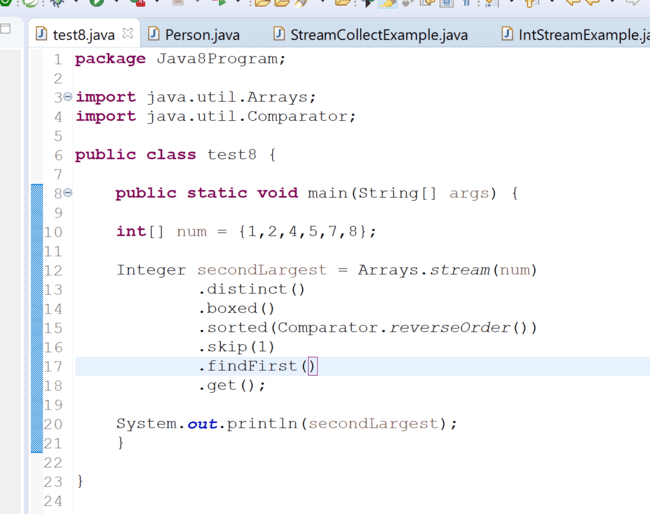


**38 : remove duplicates from list/array of numbers**

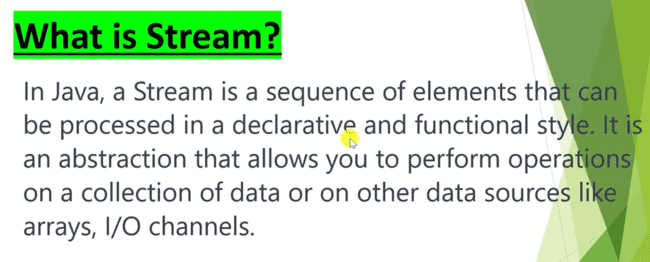
****

**39 : find second largest element in list/array**

****

****

**40: what is java 8 stream**

****