Java Assignment Number 3

Stream Operations

PRN-240840128003
PRN-240840128021

--

ToyStore Assignment1 using stream

Toy.java

```
package day8.Assignment1;

import day6.Manufacture_Date;

public class Toy{
    protected int prod_id;
    protected String prod_name;
    protected double prod_price;
    protected Manufacture_Date purchase_date;
    protected String category;
    protected int age;
    public Toy(int prod_id, String prod_name, double prod_price,
Manufacture_Date purchase_date,String category ,int age)
    {
        this.prod_id=prod_id;
        this.prod_name=prod_name;
        this.prod_price=prod_price;
        this.purchase_date=purchase_date;
        this.category=category;
        this.age=age;
    }

    public int getProd_id() {
        return prod_id;
    }
```

```
public void setProd_id(int prod_id) {
    this.prod_id = prod_id;
}

public String getProd_name() {
    return prod_name;
}

public void setProd_name(String prod_name) {
    this.prod_name = prod_name;
}

public double getProd_price() {
    return prod_price;
}

public void setProd_price(double prod_price) {
    this.prod_price = prod_price;
}

public Manufacture_Date getPurchase_date() {
    return purchase_date;
}

public void setPurchase_date(Manufacture_Date purchase_date) {
    this.purchase_date = purchase_date;
}

public int getAge() {
    return age;
}

public void setAge(int age) {
    this.age = age;
}

public String getCategory() {
    return category;
}
```

```
    public void setCategory(String category) {
        this.category = category;
    }

    @Override
    public String toString() {
        return "Toy{" +
                " prod_name='" + prod_name + '\'' +
                ", prod_price=" + prod_price +
                ", category='" + category + '\'' +
                ", age=" + age +
                '}';
    }




}
```

Stock.java

```
package day8.Assignment1;

import day6.Manufacture_Date;

import java.util.*;
import java.util.function.Predicate;
import java.util.stream.Collectors;
import java.util.stream.Stream;

public class Stock {

    Stream<Toy> stream;
    Comparator<Toy> byPrice =
Comparator.comparing(Toy::getProd_price);
    // Printing Stock
    public void print_Stock(List <Toy> toyList)
    {
        stream = toyList.stream();
        stream.forEach(System.out::println);

    }

    // Filtering stock by catrgory and printing it....
    public void filter_by_category(List <Toy> toyList, String
category)
    {
        stream = toyList.stream();

stream.filter((toy)->toy.getCategory().equalsIgnoreCase(category)
).forEach(System.out::println);
    }

    //Displaying toys by price range
    public void toys_by_priceRange(List <Toy> toyList,double
maxPrice, double minPrice)
    {
        stream = toyList.stream();
        stream.filter((toy)->{                          // instead
of writing predicate separately we write it in stream
                if(toy.getProd_price()>=minPrice &&
toy.getProd_price()<=maxPrice)
                    return true;
                else
                    return false;
                })
                .forEach(System.out::println);

    }
    // sorting toy by category and then by price
    public void sort_toys_by_category_and_price_wise(List<Toy>
toyList)
    {

        Comparator<Toy> byCategory =
Comparator.comparing(Toy::getCategory).thenComparing(byPrice);
        stream = toyList.stream();
        stream.sorted(byCategory).forEach(System.out::println);
    }
    // using predicate interface for sorting the toys older than an
year
    public void older_stock_list(List<Toy> toyList)
    {
        Manufacture_Date current_date = new
Manufacture_Date(10,2024);
        stream = toyList.stream();
        Predicate<Toy> old_stock = (toy)->{

if(current_date.getYear()-toy.getPurchase_date().getYear()>1)
                return true;
            else if
(current_date.getYear()-toy.getPurchase_date().getYear()==1)
            {

if(current_date.getMonth()-toy.getPurchase_date().getMonth()>0)
                    return true;
                else
                    return false;
            }
            else
                return false;
        };
        stream.filter(old_stock).forEach(System.out::println);
    }
    // Collecting Toy object in a map by giving key as category and
object as their key
    public void count_toys_category_wise(List<Toy> toyList)
    {
        stream = toyList.stream();
        Map<String, List<Toy>> count_toys =
stream.collect(Collectors.groupingBy(Toy::getCategory));
        count_toys.forEach((toykey, toylist) ->
System.out.println(toykey + " : " + toylist));    // printing the map
        System.out.println("Count of Toys as per category : ");
        count_toys.forEach((toykey, toylist) ->
System.out.println(toykey + " : " + toylist.size()));  // printing
the key and the size of the list
    }
    // comparing toy by their price and printing the max valued toy
    public void most_expensive_toy(List<Toy> toyList)
    {
        stream= toyList.stream();
        Map<String, Optional<Toy>> count_toys =
stream.collect(Collectors.groupingBy(Toy::getCategory,Collectors.
maxBy(byPrice)));
        count_toys.forEach((key, value) -> System.out.println(key+" :
"+value));
    }
    // comparing toy by their price and printing the max valued toy
    public void least_expensive_toy(List<Toy> toyList)
    {
        stream= toyList.stream();
        Map<String, Optional<Toy>> count_toys =
stream.collect(Collectors.groupingBy(Toy::getCategory,Collectors.
minBy(byPrice)));
        count_toys.forEach((key, value) -> System.out.println(key+" :
"+value));
    }
    // Number of toys in an age group
    public void total_toys_age_group_wise(List<Toy> toyList,int age)
    {
        stream = toyList.stream();
        Long agecount =
stream.filter((toy)->(toy.getAge()>=age)).collect(Collectors.coun
ting());
        System.out.println("Age : "+age+", and number of toys in this
group : "+agecount);
    }

}
```

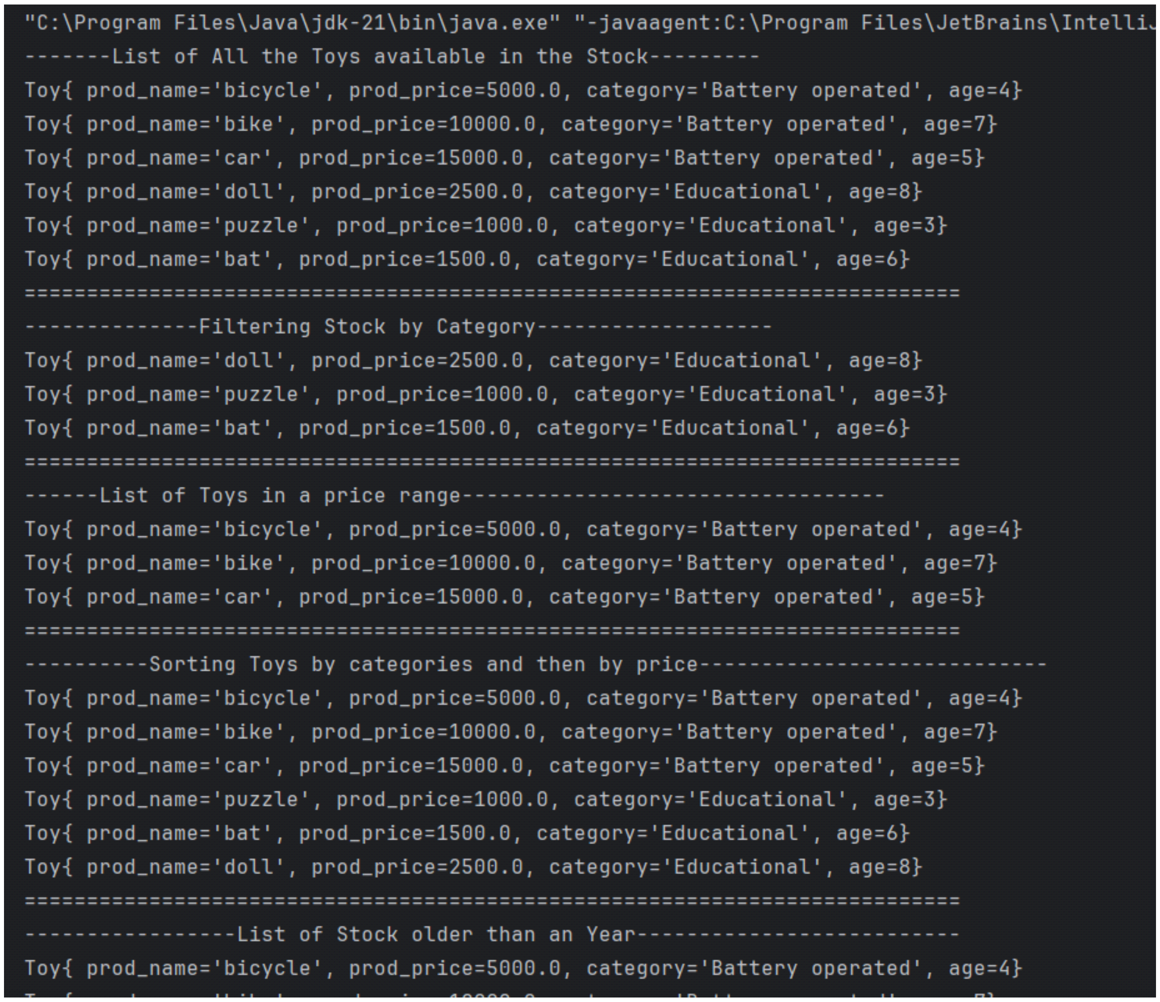
ToyStore.java

```
package day8.Assignment1;
```

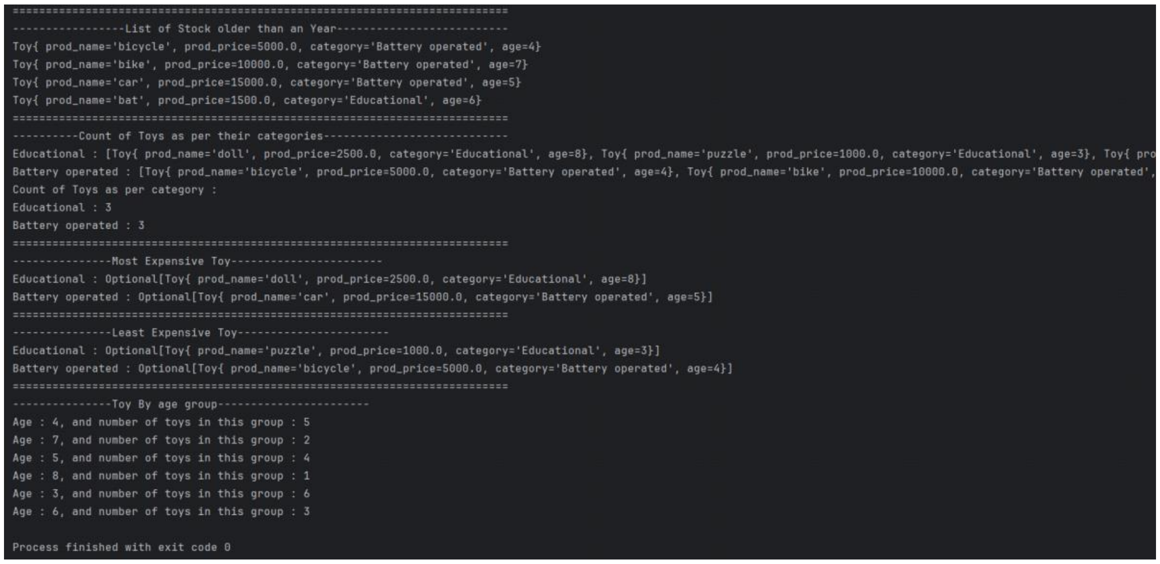
```
import day6.Manufacture_Date;
import java.util.ArrayList;
import java.util.List;

public class ToyStore {
    public static void main(String[] args) {
        Stock stock = new Stock();
        List<Toy> toyList = new ArrayList<>();
        toyList.add(new Toy(101,"bicycle",5000,new
Manufacture_Date(6,2022),"Battery operated", 4));
        toyList.add(new Toy(201,"bike",10000,new
Manufacture_Date(9,2023),"Battery operated", 7));
        toyList.add(new Toy(501,"car",15000,new
Manufacture_Date(1,2023),"Battery operated", 5));
        toyList.add(new Toy(601,"doll",2500,new
Manufacture_Date(10,2023),"Educational", 8));
        toyList.add(new Toy(401,"puzzle",1000,new
Manufacture_Date(8,2024),"Educational", 3));
        toyList.add(new Toy(301,"bat",1500,new
Manufacture_Date(11,2022),"Educational", 6));

        System.out.println("-------List of All the Toys available in
the Stock---------");
        stock.print_Stock(toyList);

System.out.println("==================================================
==============================");
        System.out.println("---------------Filtering Stock by
Category-------------------");
        stock.filter_by_category(toyList,"educaTional");

System.out.println("==================================================
==============================");
        System.out.println("------List of Toys in a price
range-------------------------------------");
        stock.toys_by_priceRange(toyList,15000,5000);

System.out.println("==================================================
==============================");
        System.out.println("----------Sorting Toys by categories and
then by price-----------------------------");
       stock.sort_toys_by_category_and_price_wise(toyList);

System.out.println("=============================================
==============================");
       System.out.println("-----------------List of Stock older than
an Year--------------------------");
       stock.older_stock_list(toyList);

System.out.println("=============================================
==============================");
       System.out.println("----------Count of Toys as per their
categories-----------------------------");
       stock.count_toys_category_wise(toyList);

System.out.println("=============================================
==============================");
       System.out.println("---------------Most Expensive
Toy------------------------");
       stock.most_expensive_toy(toyList);

System.out.println("=============================================
==============================");
       System.out.println("---------------Least Expensive
Toy------------------------");
       stock.least_expensive_toy(toyList);

System.out.println("=============================================
==============================");
       System.out.println("---------------Toy By age
group------------------------");
       for(Toy toy : toyList)
       {
          stock.total_toys_age_group_wise(toyList,toy.getAge());
       }


    }
}
```

==

Output :



```
"C:\Program Files\Java\jdk-21\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\Intelli
-------List of All the Toys available in the Stock---------
Toy{ prod_name='bicycle', prod_price=5000.0, category='Battery operated', age=4}
Toy{ prod_name='bike', prod_price=10000.0, category='Battery operated', age=7}
Toy{ prod_name='car', prod_price=15000.0, category='Battery operated', age=5}
Toy{ prod_name='doll', prod_price=2500.0, category='Educational', age=8}
Toy{ prod_name='puzzle', prod_price=1000.0, category='Educational', age=3}
Toy{ prod_name='bat', prod_price=1500.0, category='Educational', age=6}
===========================================================================
--------------Filtering Stock by Category-------------------
Toy{ prod_name='doll', prod_price=2500.0, category='Educational', age=8}
Toy{ prod_name='puzzle', prod_price=1000.0, category='Educational', age=3}
Toy{ prod_name='bat', prod_price=1500.0, category='Educational', age=6}
===========================================================================
------List of Toys in a price range---------------------------------
Toy{ prod_name='bicycle', prod_price=5000.0, category='Battery operated', age=4}
Toy{ prod_name='bike', prod_price=10000.0, category='Battery operated', age=7}
Toy{ prod_name='car', prod_price=15000.0, category='Battery operated', age=5}
===========================================================================
----------Sorting Toys by categories and then by price----------------------------
Toy{ prod_name='bicycle', prod_price=5000.0, category='Battery operated', age=4}
Toy{ prod_name='bike', prod_price=10000.0, category='Battery operated', age=7}
Toy{ prod_name='car', prod_price=15000.0, category='Battery operated', age=5}
Toy{ prod_name='puzzle', prod_price=1000.0, category='Educational', age=3}
Toy{ prod_name='bat', prod_price=1500.0, category='Educational', age=6}
Toy{ prod_name='doll', prod_price=2500.0, category='Educational', age=8}
===========================================================================
-----------------List of Stock older than an Year--------------------------
Toy{ prod_name='bicycle', prod_price=5000.0, category='Battery operated', age=4}
```



```
===========================================================================
-----------------List of Stock older than an Year--------------------------
Toy{ prod_name='bicycle', prod_price=5000.0, category='Battery operated', age=4}
Toy{ prod_name='bike', prod_price=10000.0, category='Battery operated', age=7}
Toy{ prod_name='car', prod_price=15000.0, category='Battery operated', age=5}
Toy{ prod_name='bat', prod_price=1500.0, category='Educational', age=6}
===========================================================================
----------Count of Toys as per their categories---------------------------
Educational : [Toy{ prod_name='doll', prod_price=2500.0, category='Educational', age=8}, Toy{ prod_name='puzzle', prod_price=1000.0, category='Educational', age=3}, Toy{ pro
Battery operated : [Toy{ prod_name='bicycle', prod_price=5000.0, category='Battery operated', age=4}, Toy{ prod_name='bike', prod_price=10000.0, category='Battery operated',
Count of Toys as per category :
Educational : 3
Battery operated : 3
===========================================================================
---------------Most Expensive Toy-----------------------
Educational : Optional[Toy{ prod_name='doll', prod_price=2500.0, category='Educational', age=8}]
Battery operated : Optional[Toy{ prod_name='car', prod_price=15000.0, category='Battery operated', age=5}]
===========================================================================
---------------Least Expensive Toy-----------------------
Educational : Optional[Toy{ prod_name='puzzle', prod_price=1000.0, category='Educational', age=3}]
Battery operated : Optional[Toy{ prod_name='bicycle', prod_price=5000.0, category='Battery operated', age=4}]
===========================================================================
---------------Toy By age group----------------------
Age : 4, and number of toys in this group : 5
Age : 7, and number of toys in this group : 2
Age : 5, and number of toys in this group : 4
Age : 8, and number of toys in this group : 1
Age : 3, and number of toys in this group : 6
Age : 6, and number of toys in this group : 3

Process finished with exit code 0
```

Written Output :

```
"C:\Program Files\Java\jdk-21\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA
Community Edition 2024.2.2\lib\idea_rt.jar=55077:C:\Program Files\JetBrains\IntelliJ IDEA Community
Edition 2024.2.2\bin" -Dfile.encoding=UTF-8 -Dsun.stdout.encoding=UTF-8 -Dsun.stderr.encoding=UTF-8
-classpath C:\Users\DAI.STUDENTSDC\Desktop\Abhsihek\Java\Day1\out\production\Java
day8.Assignment1.ToyStore

-------List of All the Toys available in the Stock---------

Toy{ prod_name='bicycle', prod_price=5000.0, category='Battery operated', age=4}

Toy{ prod_name='bike', prod_price=10000.0, category='Battery operated', age=7}

Toy{ prod_name='car', prod_price=15000.0, category='Battery operated', age=5}

Toy{ prod_name='doll', prod_price=2500.0, category='Educational', age=8}

Toy{ prod_name='puzzle', prod_price=1000.0, category='Educational', age=3}

Toy{ prod_name='bat', prod_price=1500.0, category='Educational', age=6}

==========================================================================

--------------Filtering Stock by Category-------------------

Toy{ prod_name='doll', prod_price=2500.0, category='Educational', age=8}

Toy{ prod_name='puzzle', prod_price=1000.0, category='Educational', age=3}

Toy{ prod_name='bat', prod_price=1500.0, category='Educational', age=6}

==========================================================================

------List of Toys in a price range----------------------------------

Toy{ prod_name='bicycle', prod_price=5000.0, category='Battery operated', age=4}

Toy{ prod_name='bike', prod_price=10000.0, category='Battery operated', age=7}

Toy{ prod_name='car', prod_price=15000.0, category='Battery operated', age=5}

==========================================================================

----------Sorting Toys by categories and then by price----------------------------

Toy{ prod_name='bicycle', prod_price=5000.0, category='Battery operated', age=4}

Toy{ prod_name='bike', prod_price=10000.0, category='Battery operated', age=7}
```

```
Toy{ prod_name='car', prod_price=15000.0, category='Battery operated', age=5}

Toy{ prod_name='puzzle', prod_price=1000.0, category='Educational', age=3}

Toy{ prod_name='bat', prod_price=1500.0, category='Educational', age=6}

Toy{ prod_name='doll', prod_price=2500.0, category='Educational', age=8}

==========================================================================

-----------------List of Stock older than an Year-------------------------

Toy{ prod_name='bicycle', prod_price=5000.0, category='Battery operated', age=4}

Toy{ prod_name='bike', prod_price=10000.0, category='Battery operated', age=7}

Toy{ prod_name='car', prod_price=15000.0, category='Battery operated', age=5}

Toy{ prod_name='bat', prod_price=1500.0, category='Educational', age=6}

==========================================================================

----------Count of Toys as per their categories---------------------------

Educational : [Toy{ prod_name='doll', prod_price=2500.0, category='Educational', age=8},
Toy{ prod_name='puzzle', prod_price=1000.0, category='Educational', age=3}, Toy{ prod_name='bat',
prod_price=1500.0, category='Educational', age=6}]

Battery operated : [Toy{ prod_name='bicycle', prod_price=5000.0, category='Battery operated', age=4},
Toy{ prod_name='bike', prod_price=10000.0, category='Battery operated', age=7},
Toy{ prod_name='car', prod_price=15000.0, category='Battery operated', age=5}]

Count of Toys as per category :

Educational : 3

Battery operated : 3

==========================================================================

---------------Most Expensive Toy-----------------------

Educational : Optional[Toy{ prod_name='doll', prod_price=2500.0, category='Educational', age=8}]

Battery operated : Optional[Toy{ prod_name='car', prod_price=15000.0, category='Battery operated',
age=5}]

==========================================================================

---------------Least Expensive Toy-----------------------

Educational : Optional[Toy{ prod_name='puzzle', prod_price=1000.0, category='Educational', age=3}]

Battery operated : Optional[Toy{ prod_name='bicycle', prod_price=5000.0, category='Battery operated',
age=4}]
==================================================================================
---------------Toy By age group-----------------------
Age : 4, and number of toys in this group : 5
Age : 7, and number of toys in this group : 2
Age : 5, and number of toys in this group : 4
Age : 8, and number of toys in this group : 1
Age : 3, and number of toys in this group : 6
Age : 6, and number of toys in this group : 3

Process finished with exit code 0
```

==

TweeterApp Assignment2

Tweet.java

```java
package day8.Assignment1;

import java.time.LocalDate;
import java.util.Set;

public class Tweet {
   private String subject;
   private LocalDate dateOfPost;
   private int views;
   Set<String> hashtags;

   public Tweet(String subject, LocalDate dateOfPost, int views,
Set<String> hashtags) {
       this.subject = subject;
```

```
        this.dateOfPost = dateOfPost;
        this.views = views;
        this.hashtags = hashtags;
    }

    public String getSubject() {
        return subject;
    }

    public void setSubject(String subject) {
        this.subject = subject;
    }

    public LocalDate getDateOfPost() {
        return dateOfPost;
    }

    public void setDateOfPost(LocalDate dateOfPost) {
        this.dateOfPost = dateOfPost;
    }

    public int getViews() {
        return views;
    }

    public void setViews(int views) {
        this.views = views;
    }

    public Set<String> getHashtags() {
        return hashtags;
    }

    public void setHashtags(Set<String> hashtags) {
        this.hashtags = hashtags;
    }

    @Override
    public String toString() {
        return "Tweet{" +
                "subject='" + subject + '\'' +
                ", dateOfPost=" + dateOfPost +
                ", views=" + views +
                ", hashtags=" + hashtags +
                '}';
    }
}
```

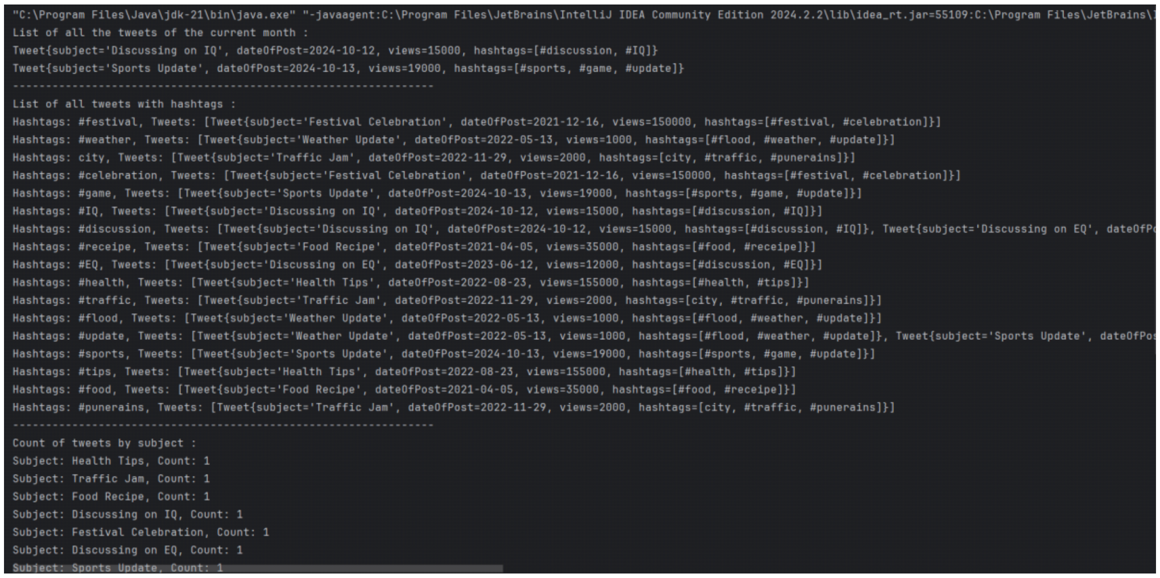
Tweeter.java

```
package day8.Assignment1;

import java.time.LocalDate;
import java.util.*;
import java.util.stream.Collectors;
import java.util.stream.Stream;

public class Tweeter {
    Stream<Tweet> stream;
    public void list_all_tweets_current_month(List<Tweet> tweets) {
        LocalDate today = LocalDate.now();
        stream = tweets.stream();
        stream.filter(t -> t.getDateOfPost().getMonthValue() ==
today.getMonthValue()).forEach(System.out::println);
    }

    public void list_all_tweets_for_hashtags(List<Tweet> tweets) {
        stream = tweets.stream();

        Map<String, List<Tweet>> hashtags = stream.flatMap(tweet ->
tweet.getHashtags().stream().map(hashtag -> Map.entry(hashtag,
tweet)))
                .collect(Collectors.groupingBy(Map.Entry::getKey,
Collectors.mapping(Map.Entry::getValue, Collectors.toList())));

        hashtags.forEach((k, v) -> System.out.println("Hashtags: " +
k + ", Tweets: " + v));
    }

    public void count_tweets_by_subject(List<Tweet> tweets){
        stream = tweets.stream();
        Map<String, Long> counts =
stream.collect(Collectors.groupingBy(Tweet::getSubject,
Collectors.counting()));
        counts.forEach((k, v) -> System.out.println("Subject: " + k
+ ", Count: " + v));
    }

    public void
```

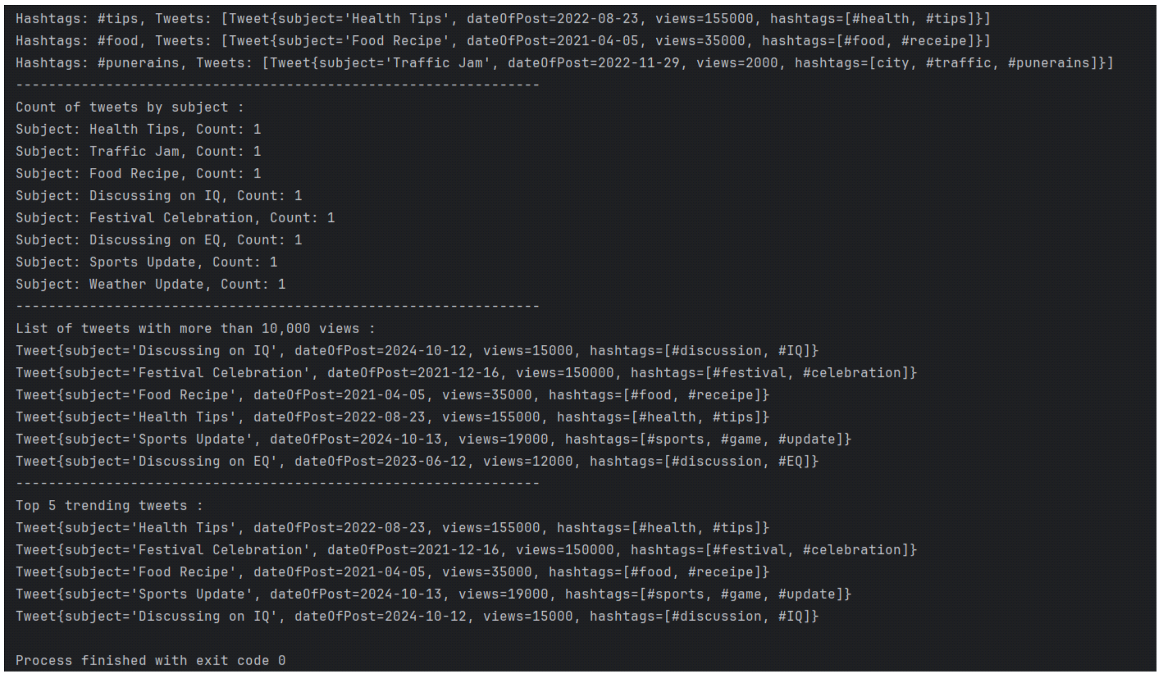
```
list_all_tweets_with_more_than_10000_views(List<Tweet> tweets) {
        stream = tweets.stream();
        stream.filter(t -> t.getViews() >
10000).forEach(System.out::println);
    }

    public void list_top5_trending_tweets(List<Tweet> tweets) {
        stream = tweets.stream();

stream.sorted(Comparator.comparingInt(Tweet::getViews).reversed()
).limit(5).forEach(System.out::println);
    }

    public static void main(String[] args) {
        Tweeter t = new Tweeter();

        List<Tweet> tweets = new ArrayList<>();
        tweets.add(new Tweet("Discussing on IQ", LocalDate.of(2024,
10, 12), 15000, Set.of("#discussion", "#IQ")));
        tweets.add(new Tweet("Weather Update", LocalDate.of(2022, 5,
13), 1000, Set.of("#weather", "#flood", "#update")));
        tweets.add(new Tweet("Festival Celebration",
LocalDate.of(2021, 12, 16), 150000, Set.of("#festival",
"#celebration")));
        tweets.add(new Tweet("Traffic Jam", LocalDate.of(2022, 11,
29), 2000, Set.of("#traffic", "#punerains", "city")));
        tweets.add(new Tweet("Food Recipe", LocalDate.of(2021, 4, 5),
35000, Set.of("#food", "#receipe")));
        tweets.add(new Tweet("Health Tips", LocalDate.of(2022, 8, 23),
155000, Set.of("#health", "#tips")));
        tweets.add(new Tweet("Sports Update", LocalDate.of(2024, 10,
13), 19000, Set.of("#sports","#game", "#update")));
        tweets.add(new Tweet("Discussing on EQ", LocalDate.of(2023,
6, 12), 12000, Set.of("#discussion", "#EQ")));

        System.out.println("List of all the tweets of the current
month :");
        t.list_all_tweets_current_month(tweets);

System.out.println("---------------------------------------------
-------------------");
        System.out.println("List of all tweets with hashtags :");
        t.list_all_tweets_for_hashtags(tweets);

System.out.println("---------------------------------------------
-------------------");
        System.out.println("Count of tweets by subject :");
        t.count_tweets_by_subject(tweets);

System.out.println("---------------------------------------------
-------------------");
        System.out.println("List of tweets with more than 10,000
views :");
        t.list_all_tweets_with_more_than_10000_views(tweets);

System.out.println("---------------------------------------------
-------------------");
        System.out.println("Top 5 trending tweets :");
        t.list_top5_trending_tweets(tweets);
    }
}
```

==

Output:



```
"C:\Program Files\Java\jdk-21\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2024.2.2\lib\idea_rt.jar=55109:C:\Program Files\JetBrains\I
List of all the tweets of the current month :
Tweet{subject='Discussing on IQ', dateOfPost=2024-10-12, views=15000, hashtags=[#discussion, #IQ]}
Tweet{subject='Sports Update', dateOfPost=2024-10-13, views=19000, hashtags=[#sports, #game, #update]}
----------------------------------------------------------------
List of all tweets with hashtags :
Hashtags: #festival, Tweets: [Tweet{subject='Festival Celebration', dateOfPost=2021-12-16, views=150000, hashtags=[#festival, #celebration]}]
Hashtags: #weather, Tweets: [Tweet{subject='Weather Update', dateOfPost=2022-05-13, views=1000, hashtags=[#flood, #weather, #update]}]
Hashtags: city, Tweets: [Tweet{subject='Traffic Jam', dateOfPost=2022-11-29, views=2000, hashtags=[city, #traffic, #punerains]}]
Hashtags: #celebration, Tweets: [Tweet{subject='Festival Celebration', dateOfPost=2021-12-16, views=150000, hashtags=[#festival, #celebration]}]
Hashtags: #game, Tweets: [Tweet{subject='Sports Update', dateOfPost=2024-10-13, views=19000, hashtags=[#sports, #game, #update]}]
Hashtags: #IQ, Tweets: [Tweet{subject='Discussing on IQ', dateOfPost=2024-10-12, views=15000, hashtags=[#discussion, #IQ]}]
Hashtags: #discussion, Tweets: [Tweet{subject='Discussing on IQ', dateOfPost=2024-10-12, views=15000, hashtags=[#discussion, #IQ]}, Tweet{subject='Discussing on EQ', dateOfP
Hashtags: #receipe, Tweets: [Tweet{subject='Food Recipe', dateOfPost=2021-04-05, views=35000, hashtags=[#food, #receipe]}]
Hashtags: #EQ, Tweets: [Tweet{subject='Discussing on EQ', dateOfPost=2023-06-12, views=12000, hashtags=[#discussion, #EQ]}]
Hashtags: #health, Tweets: [Tweet{subject='Health Tips', dateOfPost=2022-08-23, views=155000, hashtags=[#health, #tips]}]
Hashtags: #traffic, Tweets: [Tweet{subject='Traffic Jam', dateOfPost=2022-11-29, views=2000, hashtags=[city, #traffic, #punerains]}]
Hashtags: #flood, Tweets: [Tweet{subject='Weather Update', dateOfPost=2022-05-13, views=1000, hashtags=[#flood, #weather, #update]}]
Hashtags: #update, Tweets: [Tweet{subject='Weather Update', dateOfPost=2022-05-13, views=1000, hashtags=[#flood, #weather, #update]}, Tweet{subject='Sports Update', dateOfPos
Hashtags: #sports, Tweets: [Tweet{subject='Sports Update', dateOfPost=2024-10-13, views=19000, hashtags=[#sports, #game, #update]}]
Hashtags: #tips, Tweets: [Tweet{subject='Health Tips', dateOfPost=2022-08-23, views=155000, hashtags=[#health, #tips]}]
Hashtags: #food, Tweets: [Tweet{subject='Food Recipe', dateOfPost=2021-04-05, views=35000, hashtags=[#food, #receipe]}]
Hashtags: #punerains, Tweets: [Tweet{subject='Traffic Jam', dateOfPost=2022-11-29, views=2000, hashtags=[city, #traffic, #punerains]}]
----------------------------------------------------------------
Count of tweets by subject :
Subject: Health Tips, Count: 1
Subject: Traffic Jam, Count: 1
Subject: Food Recipe, Count: 1
Subject: Discussing on IQ, Count: 1
Subject: Festival Celebration, Count: 1
Subject: Discussing on EQ, Count: 1
Subject: Sports Update, Count: 1
```


```
Hashtags: #tips, Tweets: [Tweet{subject='Health Tips', dateOfPost=2022-08-23, views=155000, hashtags=[#health, #tips]}]
Hashtags: #food, Tweets: [Tweet{subject='Food Recipe', dateOfPost=2021-04-05, views=35000, hashtags=[#food, #receipe]}]
Hashtags: #punerains, Tweets: [Tweet{subject='Traffic Jam', dateOfPost=2022-11-29, views=2000, hashtags=[city, #traffic, #punerains]}]
---------------------------------------------------------------
Count of tweets by subject :
Subject: Health Tips, Count: 1
Subject: Traffic Jam, Count: 1
Subject: Food Recipe, Count: 1
Subject: Discussing on IQ, Count: 1
Subject: Festival Celebration, Count: 1
Subject: Discussing on EQ, Count: 1
Subject: Sports Update, Count: 1
Subject: Weather Update, Count: 1
---------------------------------------------------------------
List of tweets with more than 10,000 views :
Tweet{subject='Discussing on IQ', dateOfPost=2024-10-12, views=15000, hashtags=[#discussion, #IQ]}
Tweet{subject='Festival Celebration', dateOfPost=2021-12-16, views=150000, hashtags=[#festival, #celebration]}
Tweet{subject='Food Recipe', dateOfPost=2021-04-05, views=35000, hashtags=[#food, #receipe]}
Tweet{subject='Health Tips', dateOfPost=2022-08-23, views=155000, hashtags=[#health, #tips]}
Tweet{subject='Sports Update', dateOfPost=2024-10-13, views=19000, hashtags=[#sports, #game, #update]}
Tweet{subject='Discussing on EQ', dateOfPost=2023-06-12, views=12000, hashtags=[#discussion, #EQ]}
---------------------------------------------------------------
Top 5 trending tweets :
Tweet{subject='Health Tips', dateOfPost=2022-08-23, views=155000, hashtags=[#health, #tips]}
Tweet{subject='Festival Celebration', dateOfPost=2021-12-16, views=150000, hashtags=[#festival, #celebration]}
Tweet{subject='Food Recipe', dateOfPost=2021-04-05, views=35000, hashtags=[#food, #receipe]}
Tweet{subject='Sports Update', dateOfPost=2024-10-13, views=19000, hashtags=[#sports, #game, #update]}
Tweet{subject='Discussing on IQ', dateOfPost=2024-10-12, views=15000, hashtags=[#discussion, #IQ]}

Process finished with exit code 0
```

Written Output :

```
"C:\Program Files\Java\jdk-21\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2024.2.2\lib\idea_rt.jar=55109:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2024.2.2\bin" -Dfile.encoding=UTF-8 -Dsun.stdout.encoding=UTF-8 -Dsun.stderr.encoding=UTF-8 -classpath C:\Users\DAI.STUDENTSDC\Desktop\Abhsihek\Java\Day1\out\production\Java day8.Assignment1.Tweeter

List of all the tweets of the current month :

Tweet{subject='Discussing on IQ', dateOfPost=2024-10-12, views=15000, hashtags=[#discussion, #IQ]}

Tweet{subject='Sports Update', dateOfPost=2024-10-13, views=19000, hashtags=[#sports, #game, #update]}

---------------------------------------------------------------

List of all tweets with hashtags :

Hashtags: #festival, Tweets: [Tweet{subject='Festival Celebration', dateOfPost=2021-12-16, views=150000, hashtags=[#festival, #celebration]}]

Hashtags: #weather, Tweets: [Tweet{subject='Weather Update', dateOfPost=2022-05-13,
views=1000, hashtags=[#flood, #weather, #update]}]

Hashtags: city, Tweets: [Tweet{subject='Traffic Jam', dateOfPost=2022-11-29, views=2000,
hashtags=[city, #traffic, #punerains]}]

Hashtags: #celebration, Tweets: [Tweet{subject='Festival Celebration', dateOfPost=2021-12-16,
views=150000, hashtags=[#festival, #celebration]}]

Hashtags: #game, Tweets: [Tweet{subject='Sports Update', dateOfPost=2024-10-13,
views=19000, hashtags=[#sports, #game, #update]}]

Hashtags: #IQ, Tweets: [Tweet{subject='Discussing on IQ', dateOfPost=2024-10-12,
views=15000, hashtags=[#discussion, #IQ]}]

Hashtags: #discussion, Tweets: [Tweet{subject='Discussing on IQ', dateOfPost=2024-10-12,
views=15000, hashtags=[#discussion, #IQ]}, Tweet{subject='Discussing on EQ',
dateOfPost=2023-06-12, views=12000, hashtags=[#discussion, #EQ]}]

Hashtags: #receipe, Tweets: [Tweet{subject='Food Recipe', dateOfPost=2021-04-05,
views=35000, hashtags=[#food, #receipe]}]

Hashtags: #EQ, Tweets: [Tweet{subject='Discussing on EQ', dateOfPost=2023-06-12,
views=12000, hashtags=[#discussion, #EQ]}]

Hashtags: #health, Tweets: [Tweet{subject='Health Tips', dateOfPost=2022-08-23,
views=155000, hashtags=[#health, #tips]}]

Hashtags: #traffic, Tweets: [Tweet{subject='Traffic Jam', dateOfPost=2022-11-29, views=2000,
hashtags=[city, #traffic, #punerains]}]

Hashtags: #flood, Tweets: [Tweet{subject='Weather Update', dateOfPost=2022-05-13,
views=1000, hashtags=[#flood, #weather, #update]}]

Hashtags: #update, Tweets: [Tweet{subject='Weather Update', dateOfPost=2022-05-13,
views=1000, hashtags=[#flood, #weather, #update]}, Tweet{subject='Sports Update',
dateOfPost=2024-10-13, views=19000, hashtags=[#sports, #game, #update]}]

Hashtags: #sports, Tweets: [Tweet{subject='Sports Update', dateOfPost=2024-10-13,
views=19000, hashtags=[#sports, #game, #update]}]

Hashtags: #tips, Tweets: [Tweet{subject='Health Tips', dateOfPost=2022-08-23, views=155000,
hashtags=[#health, #tips]}]

Hashtags: #food, Tweets: [Tweet{subject='Food Recipe', dateOfPost=2021-04-05, views=35000,
hashtags=[#food, #receipe]}]

Hashtags: #punerains, Tweets: [Tweet{subject='Traffic Jam', dateOfPost=2022-11-29,
views=2000, hashtags=[city, #traffic, #punerains]}]
```

```
-------------------------------------------------------------

Count of tweets by subject :

Subject: Health Tips, Count: 1

Subject: Traffic Jam, Count: 1

Subject: Food Recipe, Count: 1

Subject: Discussing on IQ, Count: 1

Subject: Festival Celebration, Count: 1

Subject: Discussing on EQ, Count: 1

Subject: Sports Update, Count: 1

Subject: Weather Update, Count: 1

-------------------------------------------------------------

List of tweets with more than 10,000 views :

Tweet{subject='Discussing on IQ', dateOfPost=2024-10-12, views=15000, hashtags=[#discussion,
#IQ]}

Tweet{subject='Festival Celebration', dateOfPost=2021-12-16, views=150000,
hashtags=[#festival, #celebration]}

Tweet{subject='Food Recipe', dateOfPost=2021-04-05, views=35000, hashtags=[#food,
#receipe]}

Tweet{subject='Health Tips', dateOfPost=2022-08-23, views=155000, hashtags=[#health, #tips]}

Tweet{subject='Sports Update', dateOfPost=2024-10-13, views=19000, hashtags=[#sports,
#game, #update]}

Tweet{subject='Discussing on EQ', dateOfPost=2023-06-12, views=12000,
hashtags=[#discussion, #EQ]}

-------------------------------------------------------------

Top 5 trending tweets :

Tweet{subject='Health Tips', dateOfPost=2022-08-23, views=155000, hashtags=[#health, #tips]}

Tweet{subject='Festival Celebration', dateOfPost=2021-12-16, views=150000,
hashtags=[#festival, #celebration]}

Tweet{subject='Food Recipe', dateOfPost=2021-04-05, views=35000, hashtags=[#food,
#receipe]}

Tweet{subject='Sports Update', dateOfPost=2024-10-13, views=19000, hashtags=[#sports,
#game, #update]}

Tweet{subject='Discussing on IQ', dateOfPost=2024-10-12, views=15000, hashtags=[#discussion,
#IQ]}


Process finished with exit code 0
```