Streams assignments

**Setup:**

Create the following classes:

Class Fruit { String name ; int calories ; int price; String color;}

Class News { int newsid; String postedByUser; String commentByUser; String comment;}

Class Trader { String name; String city}

Class Transaction { Trader trader; int year; int value;}

1. Display the fruit names of low calories fruits i.e., calories < 100 sorted in descending order of calories.
2. Display colour wise list of fruit names.
3. Display only RED colour fruits sorted as per their price in ascending order.

import java.util.ArrayList;

import java.util.Collections;

import java.util.List;

import java.util.stream.Collectors;

public class Fruits {

private String name;

private Integer calories;

private Integer price;

private String color;

public Fruits(String name, Integer calories, Integer price, String color) {

super();

this.name = name;

this.calories = calories;

this.price = price;

this.color = color;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public Integer getCalories() {

return calories;

}

public void setCalories(Integer calories) {

this.calories = calories;

}

public Integer getPrice() {

return price;

}

public void setPrice(Integer price) {

this.price = price;

}

public String getColor() {

return color;

}

public void setColor(String color) {

this.color = color;

}

@Override

public String toString() {

// TODO Auto-generated method stub

return "Fruits [Name=" + name + ", Calories=" + calories + ", Color=" + color + "]";

}

}

mport java.util.ArrayList;

import java.util.Collections;

import java.util.Comparator;

import java.util.List;

import java.util.stream.Collectors;

public class StreamTest {

public static void main(String[] args) {

// TODO Auto-generated method stub

List<Fruits> fruitsList = new ArrayList<>();

fruitsList.add(new Fruits("Mango", 100, 10, "Yellow"));

fruitsList.add(new Fruits("Banana", 100, 20, "Yellow"));

fruitsList.add(new Fruits("Apple", 80, 30, "Red"));

fruitsList.add(new Fruits("Guava", 70, 10, "Green"));

fruitsList.add(new Fruits("Watermelon", 100, 50, "Green"));

fruitsList.add(new Fruits("Pomegranate", 90, 30, "Green"));

System.out.println("Answer 1:\n");

//Display the fruit names of low calories fruits

List<String> lowCalories = fruitsList.stream().filter(f -> f.getCalories() < 100).map(t -> t.getName()).sorted(Collections.reverseOrder()).collect(Collectors.toList());

System.out.println("Desending order of calories" + lowCalories);

System.out.println("\nAnswer 2: \n");

//Display color wise fruit names

for (Fruits f : fruitsList) {

System.out.println(f.getColor() + " " + f.getName());

}

System.out.println("\nAnswer 3: \n");

//Display only RED color fruits sorted as per their price in Ascending Order

List<Fruits> colorList = fruitsList.stream().filter(e -> e.getColor().equalsIgnoreCase("Red")).sorted(Comparator.comparing(Fruits::getPrice)).collect(Collectors.toList());

System.out.println(colorList);

}

}

**Output:**  Answer 1:

Desending order of calories[Pomegranate, Guava, Apple]

Answer 2:

Yellow Mango

Yellow Banana

Red Apple

Green Guava

Green Watermelon

Green Pomegranate

Answer 3:

[Fruits [Name=Apple, Calories=80, Color=Red]]

1. Find out the newsid which has received maximum comments.
2. Find out how many times the word ‘budget’ arrived in user comments all news.
3. Find out which user has posted maximum comments.
4. Display commentByUser wise number of comments
5. Find all transaction in the year 2011 and sort them by value (small to high).
6. What are all the unique cities where the traders work?
7. Find all traders from Pune and sort them by name.
8. Return a string of all traders’ name sorted alphabetically.
9. Are any traders based in Indore?

**public** **class** Traders {

**private** String name;

**private** String city;

**public** Traders(String name, String city) {

**super**();

**this**.name = name;

**this**.city = city;

}

**public** String getName() {

**return** name;

}

**public** **void** setName(String name) {

**this**.name = name;

}

**public** String getCity() {

**return** city;

}

**public** **void** setCity(String city) {

**this**.city = city;

}

@Override

**public** String toString() {

// **TODO** Auto-generated method stub

**return** "Traders {" + "name='" + name + '\'' +", city='" + city + '\'' + '}';

}

}

**import** java.util.ArrayList;

**import** java.util.Comparator;

**import** java.util.List;

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

**public** **class** StreamTest2 {

**public** **static** **void** main(String[] args) {

// **TODO** Auto-generated method stub

List<Traders> traderList = **new** ArrayList<>();

traderList.add(**new** Traders("Rob", "Pune"));

traderList.add(**new** Traders("Jim", "Chennai"));

traderList.add(**new** Traders("Sam", "Mumbai"));

traderList.add(**new** Traders("Emily", "Indore"));

traderList.add(**new** Traders("Emma", "Kolkata"));

traderList.add(**new** Traders("Richad", "Delhi"));

System.***out***.println("Answer 9: \n");

traderList.stream().map(Traders::getCity).distinct().forEach(System.***out***::println);

System.***out***.println("\n Answer 10: \n");

//sort them by name

List<Traders> sortNames = traderList.stream().filter(e -> e.getCity().equalsIgnoreCase("pune")).sorted(Comparator.*comparing*(Traders::getName)).collect(Collectors.*toList*());

System.***out***.println(sortNames);

System.***out***.println("\nAnswer 11: \n");

List<String> stringList = traderList.stream().map(Traders::getName).sorted().collect(Collectors.*toList*());

stringList.forEach(System.***out***::println);

System.***out***.println("\nAnswer 12: \n");

traderList.stream().filter(e -> e.getCity().equalsIgnoreCase("indore")).forEach(System.***out***::println);

}

}

**Output:** Answer 9:

Pune

Chennai

Mumbai

Indore

Kolkata

Delhi

Answer 10:

[Traders {name='Rob', city='Pune'}]

Answer 11:

Emily

Emma

Jim

Richad

Rob

Sam

Answer 12:

Traders {name='Emily', city='Indore'}