



1.

package stream;

import java.util.ArrayList;

import java.util.Arrays;

import java.util.List;

import java.util.function.Predicate;

class Fruit{

String name;

int calories;

int price;

String color;

public Fruit(String name, int calories, int 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 int getCalories() {

return calories;

}

public void setCalories(int calories) {

this.calories = calories;

}

public int getPrice() {

return price;

}

public void setPrice(int price) {

this.price = price;

}

public String getColor() {

return color;

}

public void setColor(String color) {

this.color = color;

}

}

public class ExampleOfStream {

public static void main(String[] args) {

//create a list of fruit

List<Fruit> fruits = new ArrayList<Fruit>();

//adding fruits

fruits.add(new Fruit("Banana",85,15,"Yellow"));

fruits.add(new Fruit("Apple",95,15,"Red"));

fruits.add(new Fruit("Orange",355,15,"Orange"));

fruits.add(new Fruit("Guava",200,15,"Green"));

fruits.add(new Fruit("Grapes",101,15,"Light Green"));

List<String>fruitsName = fruits.stream()

.filter(p-> p.calories<100)

.map(p -> p.name)

.collect(Collectors.toList()); }

System.out.println("Calories less than 100: " + fruitsName);

}

}

2.

package stream;

import java.util.ArrayList;

import java.util.Arrays;

import java.util.List;

import java.util.Map;

import java.util.function.Predicate;

import java.util.stream.Collectors;

class Fruit{

String name;

int calories;

int price;

String color;

public Fruit(String name, int calories, int 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 int getCalories() {

return calories;

}

public void setCalories(int calories) {

this.calories = calories;

}

public int getPrice() {

return price;

}

public void setPrice(int price) {

this.price = price;

}

public String getColor() {

return color;

}

public void setColor(String color) {

this.color = color;

}

}

public class ExampleOfStream {

public static void main(String[] args) {

//create a list of fruit

List<Fruit> fruits = new ArrayList<Fruit>();

//adding fruits

fruits.add(new Fruit("Banana",85,15,"Yellow"));

fruits.add(new Fruit("Apple",95,15,"Red"));

fruits.add(new Fruit("Orange",355,15,"Orange"));

fruits.add(new Fruit("Guava",200,15,"Green"));

fruits.add(new Fruit("Grapes",101,15,"Light Green"));

Map<String, String> fruitsName = fruits.stream()

.collect(Collectors.toMap(p->p.color, p->p.name));

System.out.println(fruitsName);

}

9.

package stream;

import java.util.ArrayList;

import java.util.Arrays;

import java.util.List;

import java.util.stream.Collector;

import java.util.stream.Collectors;

class Trader{

String name;

String city;

public Trader(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;

}

}

class Transaction{

Trader trader;

int year;

int value;

public Transaction(Trader trader, int year, int value) {

super();

this.trader = trader;

this.year = year;

this.value = value;

}

public Trader getTrader() {

return trader;

}

public void setTrader(Trader trader) {

this.trader = trader;

}

public int getYear() {

return year;

}

public void setYear(int year) {

this.year = year;

}

public int getValue() {

return value;

}

public void setValue(int value) {

this.value = value;

}

}

public class transactions {

public static void main(String[] args) {

//adding Traders

Trader sawan = new Trader("Sawan Kundu","Indore");

Trader ana = new Trader("Ananya Biswas","Pune");

Trader rudy = new Trader("Rudrarup Das","Kolkata");

Trader madhu = new Trader("Madhurika Dutta","Pune");

Trader anirban = new Trader("Anirban Dey","Bangalore");

Trader anik = new Trader("Anik Das","Indore");

//adding transactions

List<Transaction> t1 = Arrays.asList(

new Transaction(sawan, 2011, 10),

new Transaction(ana, 1999, 85),

new Transaction(madhu, 2020, 25),

new Transaction(rudy, 2006, 45),

new Transaction(anik, 2021, 10),

new Transaction(anirban, 2025, 20),

new Transaction(madhu, 2003, 75),

new Transaction(ana, 1999, 85)

);

List<String> condition = t1.stream()

.map(t -> t.getTrader().getCity())

.distinct()

.collect(Collectors.toList());

System.out.println(condition);

}

}

11.

package stream;

import java.util.\*;

import java.util.Arrays;

import java.util.Comparator;

import java.util.List;

import java.util.stream.Collectors;

class Trader{

String name;

String city;

public Trader(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;

}

}

class Transaction{

Trader trader;

int year;

int value;

public Transaction(Trader trader, int year, int value) {

super();

this.trader = trader;

this.year = year;

this.value = value;

}

public Trader getTrader() {

return trader;

}

public void setTrader(Trader trader) {

this.trader = trader;

}

public int getYear() {

return year;

}

public void setYear(int year) {

this.year = year;

}

public int getValue() {

return value;

}

public void setValue(int value) {

this.value = value;

}

}

public class transactions {

public static void main(String[] args) {

//adding Traders

Trader sawan = new Trader("Sawan Kundu","Indore");

Trader ana = new Trader("Ananya Biswas","Pune");

Trader rudy = new Trader("Rudrarup Das","Kolkata");

Trader madhu = new Trader("Madhurika Dutta","Pune");

Trader anirban = new Trader("Anirban Dey","Bangalore");

Trader anik = new Trader("Anik Das","Indore");

//adding transactions

List<Transaction> t1 = Arrays.asList(

new Transaction(sawan, 2011, 10),

new Transaction(ana, 1999, 85),

new Transaction(madhu, 2011, 25),

new Transaction(rudy, 2006, 45),

new Transaction(anik, 2011, 35),

new Transaction(anirban, 2011, 20),

new Transaction(madhu, 2003, 75),

new Transaction(ana, 1999, 85)

);

List<String> condition = t1.stream()

.map(t -> t.getTrader().getName())

.distinct()

.sorted()

.collect(Collectors.toList());

System.out.println(condition);

//condition.forEach(System.out:: println);

}

}

12.

package stream;

import java.util.\*;

import java.util.Arrays;

import java.util.Comparator;

import java.util.List;

import java.util.stream.Collectors;

class Trader{

String name;

String city;

public Trader(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;

}

}

class Transaction{

Trader trader;

int year;

int value;

public Transaction(Trader trader, int year, int value) {

super();

this.trader = trader;

this.year = year;

this.value = value;

}

public Trader getTrader() {

return trader;

}

public void setTrader(Trader trader) {

this.trader = trader;

}

public int getYear() {

return year;

}

public void setYear(int year) {

this.year = year;

}

public int getValue() {

return value;

}

public void setValue(int value) {

this.value = value;

}

}

public class transactions {

public static void main(String[] args) {

//adding Traders

Trader sawan = new Trader("Sawan Kundu","Indore");

Trader ana = new Trader("Ananya Biswas","Pune");

Trader rudy = new Trader("Rudrarup Das","Kolkata");

Trader madhu = new Trader("Madhurika Dutta","Pune");

Trader anirban = new Trader("Anirban Dey","Bangalore");

Trader anik = new Trader("Anik Das","Indore");

//adding transactions

List<Transaction> t1 = Arrays.asList(

new Transaction(sawan, 2011, 10),

new Transaction(ana, 1999, 85),

new Transaction(madhu, 2011, 25),

new Transaction(rudy, 2006, 45),

new Transaction(anik, 2011, 35),

new Transaction(anirban, 2011, 20),

new Transaction(madhu, 2003, 75),

new Transaction(ana, 1999, 85)

);

boolean condition = t1.stream()

.anyMatch(t -> t.getTrader().getCity().equals("Indore"));

System.out.println(condition);

}

}

14.

package stream;

import java.util.\*;

import java.util.Arrays;

import java.util.Comparator;

import java.util.List;

import java.util.stream.Collectors;

class Trader{

String name;

String city;

public Trader(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;

}

}

class Transaction{

Trader trader;

int year;

int value;

public Transaction(Trader trader, int year, int value) {

super();

this.trader = trader;

this.year = year;

this.value = value;

}

public Trader getTrader() {

return trader;

}

public void setTrader(Trader trader) {

this.trader = trader;

}

public int getYear() {

return year;

}

public void setYear(int year) {

this.year = year;

}

public int getValue() {

return value;

}

public void setValue(int value) {

this.value = value;

}

}

public class transactions {

public static void main(String[] args) {

//adding Traders

Trader sawan = new Trader("Sawan Kundu","Indore");

Trader ana = new Trader("Ananya Biswas","Delhi");

Trader rudy = new Trader("Rudrarup Das","Kolkata");

Trader madhu = new Trader("Madhurika Dutta","Pune");

Trader anirban = new Trader("Anirban Dey","Delhi");

Trader anik = new Trader("Anik Das","Indore");

//adding transactions

List<Transaction> t1 = Arrays.asList(

new Transaction(sawan, 2011, 10),

new Transaction(ana, 1999, 85),

new Transaction(madhu, 2011, 25),

new Transaction(rudy, 2006, 45),

new Transaction(anik, 2011, 35),

new Transaction(anirban, 2011, 20),

new Transaction(madhu, 2003, 75),

new Transaction(ana, 1999, 85)

);

Optional<Integer> condition = t1.stream()

.map(Transaction:: getValue)

.max(Integer:: compare);

System.out.println(condition);

}

}

15.

package stream;

import java.util.\*;

import java.util.stream.Collectors;

class Trader{

String name;

String city;

public Trader(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;

}

}

class Transaction{

Trader trader;

int year;

int value;

public Transaction(Trader trader, int year, int value) {

super();

this.trader = trader;

this.year = year;

this.value = value;

}

public Trader getTrader() {

return trader;

}

public void setTrader(Trader trader) {

this.trader = trader;

}

public int getYear() {

return year;

}

public void setYear(int year) {

this.year = year;

}

public int getValue() {

return value;

}

public void setValue(int value) {

this.value = value;

}

}

public class transactions {

public static void main(String[] args) {

//adding Traders

Trader sawan = new Trader("Sawan Kundu","Indore");

Trader ana = new Trader("Ananya Biswas","Delhi");

Trader rudy = new Trader("Rudrarup Das","Kolkata");

Trader madhu = new Trader("Madhurika Dutta","Pune");

Trader anirban = new Trader("Anirban Dey","Delhi");

Trader anik = new Trader("Anik Das","Indore");

//adding transactions

List<Transaction> t1 = Arrays.asList(

new Transaction(sawan, 2011, 10),

new Transaction(ana, 1999, 85),

new Transaction(madhu, 2011, 25),

new Transaction(rudy, 2006, 45),

new Transaction(anik, 2011, 35),

new Transaction(anirban, 2011, 20),

new Transaction(madhu, 2003, 75),

new Transaction(ana, 1999, 85)

);

Optional<Integer> condition = t1.stream()

.map(Transaction:: getValue)

.max(Comparator.reverseOrder());

System.out.println(condition);

}

}