**Lambda Expressions Assignment**

1. **public** **class** QuestionOne

{

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

{

Addition addfunction = (a, b) -> a+b;

System.***out***.println(addfunction.add(100, 200));

Subtraction subfunction = (c, d) -> c-d;

System.***out***.println(subfunction.sub(20, 120));

Multiplication multifunction = (e, f) -> e\*f;

System.***out***.println(multifunction.multi(40, 0));

Division dividefunction = (g, h) -> g/h;

System.***out***.println(dividefunction.div(20, 10));

}

}

**interface** Addition

{

**int** add(**int** a, **int** b);

}

**interface** Subtraction

{

**int** sub(**int** c, **int** d);

}

**interface** Multiplication

{

**int** multi(**int** e, **int** f);

}

**interface** Division

{

**int** div(**int** g, **int** h);

}

1. **Orderclass.java**

public class Orderclass {

int price;

String status;

public Orderclass(int price, String status) {

super();

this.price = price;

this.status = status;

}

public int getPrice() {

return price;

}

public void setPrice(int price) {

this.price = price;

}

public String getStatus() {

return status;

}

public void setStatus(String status) {

this.status = status;

}

@Override

public String toString() {

return "Orderclass [price=" + price + ", status=" + status + "]";

}

}

**Orders.java**

import java.util.Arrays;

import java.util.List;

import java.util.function.Consumer;

import java.util.function.Predicate;

public class Orders {

public static void main(String[] args) {

// TODO Auto-generated method stub

List<Orderclass> orderdetails =Arrays.asList(

new Orderclass(15000,"ACCEPTED"),

new Orderclass(9000,"ACCEPTED"),

new Orderclass(20000,"COMPLETED"),

new Orderclass(18000,"COMPLETED"),

new Orderclass(5000,"ACCEPTED")

);

//printing orders price more than 10000 and their status

System.out.println("--------Printing Orders----------");

performconditionaly(orderdetails,o->o.getPrice()>10000,o->System.out.println("Order Price : "+o.getPrice()+" Order Status : "+o.getStatus()));

}

private static void performconditionaly(List<Orderclass> orderdetails,Predicate<Orderclass> predicate,Consumer<Orderclass> consumer) {

// TODO Auto-generated method stub

for(Orderclass o: orderdetails) {

if(predicate.test(o)) {

consumer.accept(o);

}

}

}

}

1. import java.util.function.Consumer;

import java.util.function.Function;

import java.util.function.Predicate;

import java.util.function.Supplier;

public class LambdaInterfaces {

public static void main(String[] args) {

//consumer functional interface

String str = "Consumer Interface";

Consumer<String>displayConsumer = a->System.out.println(a);

displayConsumer.accept(str.toUpperCase());

//Predicate functional interface

Predicate<String> displaypredicate= p ->str.length() > 10;

System.out.println("Predicate functional interface: "+displaypredicate.test(str));

//Function functional interface

Function<Integer,Double>val = a ->a / 5.0;

System.out.println("Function functional interface: "+val.apply(37));

//Supplier functional interface

Supplier<Float>suppval = () ->Math.max(18.99f, 19.9f);

System.out.println("Supplier functional interface: "+suppval.get());

}

}

1. import java.util.ArrayList;

public class Oddlength {

public static void main(String[] args) {

// TODO Auto-generated method stub

ArrayList<String> words =new ArrayList<String>();

words.add("Hello");

words.add("Welcome");

words.add("Computer");

words.add("System");

words.add("Assignment");

words.add("Collections");

words.removeIf(w->(w.length()%2==0));

words.forEach(System.out::println);

}

}

1. import java.util.Arrays;

import java.util.List;

import java.util.function.Consumer;

public class AppendResult {

public static void main(String[] args) {

// TODO Auto-generated method stub

StringBuilder str=new StringBuilder();

List<String> names =Arrays.asList(

new String("What"), new String("Else"),

new String("Look"), new String("Like"),

new String("Dust"), new String("Or"),

new String("Not"),new String("Edible")

);

for(String n : names) {

str.append(n.charAt(0));

}

//prints the first letters of all the string in the list

printstring(str,c->System.out.println(str));

}

private static void printstring(StringBuilder str,Consumer consumer) {

// TODO Auto-generated method stub

if(str!=null) {

consumer.accept(str);

}

}

}

1. import java.util.Arrays;

import java.util.List;

import java.util.function.UnaryOperator;

class replace implements UnaryOperator<String>{

public String apply(String str) {

return str.toUpperCase();

}

}

public class Unaryopt {

public static void main(String[] args) {

// TODO Auto-generated method stub

List<String> names =Arrays.asList(

new String("What"), new String("Else"),

new String("Look"), new String("Like"),

new String("Dust"), new String("Or"),

new String("Not"),new String("Edible")

);

System.out.println("list before replace operation: "+names);

names.replaceAll(new replace());

System.out.println("Contents of the list after replace operation: \n"+names);

}

}

1. import java.util.Map;

import java.util.TreeMap;

public class Convert {

public static void main(String[] args) {

// TODO Auto-generated method stub

StringBuilder str=new StringBuilder();

Map<Integer,String> map = new TreeMap<>();

map.put(1, "Hello");

map.put(2, "Guys");

map.put(3, "How");

map.put(4, "Are");

map.put(5, "You");

map.put(6, "Doing");

map.put(7, "In");

map.put(8, "Training");

for(Map.Entry<Integer,String>entry:map.entrySet()) {

Integer key = entry.getKey();

String c = entry.getValue();

str.append(key + c);

}

//print result string

System.out.println(str);

}

}

1. import java.util.ArrayList;

import java.util.List;

public class Threadlist {

public static void main(String[] args) {

// TODO Auto-generated method stub

List<Integer> num=new ArrayList<Integer>(){{

add(11);

add(55);

add(37);

add(95);

add(99);

}};

Thread mylambda = new Thread(()->System.out.println(num));

mylambda.run();

}

}