Usually when an abstract method in an interface has to implement to different behavior we have to create two different class that implements the interface and override the method in each classes with different behavior .

The implementation behavior have been obtain by crating the decide class object .

But when using its interface that has the abstract method using lamda can simply cut down the work of creating two implementation of class and also coresponding to the class .

package day11;

interface Add{

int sum(int a , int b);

}

public class task1{

public static void main()

{

Add addnum = (a,b)-> a + b;

Add multiple = (a,b)-> a \* b;

System.***out***.println(addnum.sum(100, 10));

System.***out***.println(multiple.sum(100, 10));

}

}

package day11;

interface A{

int operation(int a , int b);

}

public class task2{

public static void main()

{

A sum = (a,b)->{

System.***out***.println(a+b);

System.***out***.println(a\*b);

return a;

};

sum.operation(5,5);

}

}

Stream :

package day11;

import java.util.Arrays;

import java.util.List;

import java.util.\*;

public class tak4 {

public static void main(String args[])

{

List<String> name = Arrays.*asList*("A","B","C","D");

name.stream().forEach(n -> System.***out***.print(n));

}

}

ABCD

Used to reduce the size of the code by using the stream and forEach method

Parallel Stream :

package day11;

import java.util.Arrays;

import java.util.List;

import java.util.\*;

public class task5 {

public static void main(String args[])

{

List<String> name = Arrays.*asList*("Abi","Bob","Cop","Don");

name.parallelStream().forEach(n -> System.***out***.print(n));

}

}

Cop Bob Abi Don

package day11;

import java.util.Arrays;

import java.util.List;

import java.util.\*;

public class task6 {

public static void main(String args[])

{

List<Integer> num = Arrays.*asList*(1,2,3,4,5,6);

num.stream().filter(n-> n % 2==0);

}

}

This code display nothing because there is no terminal operation :

Terminal operation : that produce the result

1.collect , 2.forEach , 3.reduce :

package day11;

import java.util.\*;

public class task7 {

public static void main(String args[])

{

Scanner s=new Scanner(System.***in***);

String str = s.nextLine();

String arr[] = str.split(" ");

int ar[]= new int[arr.length];

for(int i=0;i<arr.length;i++)

{

ar[i]=Integer.*parseInt*(arr[i]);

}

for(int i=0;i<ar.length;i++)

{

System.***out***.println(ar[i]);

}

}

}

METHODS :

package day11;

import java.util.Arrays;

import java.util.List;

import java.util.stream.Collectors;

public class task8 {

public static void main(String args[])

{

List<String> name = Arrays.*asList*("Abi","Bob","Cop","Don");

List<String> aslist = name.stream()

.filter(n->n.startsWith("D"))

.map(String::toUpperCase)

.sorted()

.collect(Collectors.*toList*());

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

List<Integer> in = name.stream()

.map(String::length)

.collect(Collectors.*toList*());

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

List<String> list = name.stream()

.filter(n->n.contains("A"))

.map(String::toUpperCase)

.sorted()

.collect(Collectors.*toList*());

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

String s = name.stream()

.filter(n->n.startsWith("A"))

.findFirst().orElse("No data Found");

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

String ss = name.stream()

.filter(n->n.startsWith("A"))

.findFirst().orElse("No data Found");

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

}

}

[DON]

[3, 3, 3, 3]

[ABI]

Abi