**Ex 1**

package Streams.StreamWithLambda;  
import java.util.Arrays;  
import java.util.Comparator;  
import java.util.List;  
import java.util.Optional;  
import java.util.stream.Collectors;  
import java.util.stream.Stream;  
public class StreamsExample {  
 public static void main(String[] args) {  
// Example: Filtering and Collecting Names  
 List<String> names = Arrays.*asList*("Anna", "Bob", "Charlie", "David");  
 System.*out*.println(names.stream().filter(x -> x.length() > 3).toList());  
  
 // Example: Squaring and Sorting Numbers  
 List<Integer> numbers = Arrays.*asList*(5, 2, 9, 1, 6);  
 System.*out*.println(numbers.stream().map(x -> x \* x).sorted().toList());  
  
 // Example: Summing Values  
 List<Integer> integers = Arrays.*asList*(1, 2, 3, 4, 5);  
 System.*out*.println(integers.stream().reduce(Integer::*sum*).get());  
  
 // Example: Counting Occurrences of a Character  
 String sentence = "Hello world";  
 System.*out*.println(sentence.chars().filter(x -> x == 'l').count());  
  
 // Example  
 // Streams cannot be reused after a terminal operation has been called  
 Stream<String> stream = names.stream();  
 stream.forEach(System.*out*::println);  
// List<String> list1 = stream.map(String::toUpperCase).toList(); // exception  
 }  
}

C:\Users\Roystan\.jdks\openjdk-21.0.2\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2023.3.3\lib\idea\_rt.jar=50915:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2023.3.3\bin" -Dfile.encoding=UTF-8 -Dsun.stdout.encoding=UTF-8 -Dsun.stderr.encoding=UTF-8 -classpath C:\Users\Roystan\IdeaProjects\JavaWorkspace\out\production\JavaWorkspace Streams.StreamWithLambda.StreamsExample

[Anna, Charlie, David]

[1, 4, 25, 36, 81]

15

3

Anna

Bob

Charlie

David

Process finished with exit code 0

**Ex 2**

package com.engineeringdigest.collectionframework.stream;  
  
import java.util.Arrays;  
import java.util.List;  
import java.util.stream.Collectors;  
import java.util.stream.Stream;  
  
public class LazyEvaluation {  
 public static void main(String[] args) {  
 List<String> names = Arrays.*asList*("Alice", "Bob", "Charlie", "David");  
  
 Stream<String> stream = names.stream()  
 .filter(name -> {  
 System.*out*.println("Filtering: " + name);  
 return name.length() > 3;  
 });  
  
 System.*out*.println("Before terminal operation");  
  
 List<String> result = stream.collect(Collectors.*toList*());  
  
 System.*out*.println("After terminal operation");  
 System.*out*.println(result);  
  
  
 }  
}

C:\Users\Roystan\.jdks\openjdk-21.0.2\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2023.3.3\lib\idea\_rt.jar=50917:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2023.3.3\bin" -Dfile.encoding=UTF-8 -Dsun.stdout.encoding=UTF-8 -Dsun.stderr.encoding=UTF-8 -classpath C:\Users\Roystan\IdeaProjects\JavaWorkspace\out\production\JavaWorkspace com.engineeringdigest.collectionframework.stream.LazyEvaluation

Before terminal operation

Filtering: Alice

Filtering: Bob

Filtering: Charlie

Filtering: David

After terminal operation

[Alice, Charlie, David]

Process finished with exit code 0

**Ex 3**

package Streams.StreamWithLambda;  
  
import java.util.Arrays;  
import java.util.List;  
import java.util.stream.Collectors;  
  
public class MethodAndConstructorReference {  
 public static void main(String[] args) {  
 // 1. Method Reference  
 List<String> list1 = Arrays.*asList*("A", "B", "c");  
// list1.forEach(x -> System.out.println(x));  
 list1.forEach(System.*out*::println);  
  
 // 2. Construtor reference  
 List<String> list2 = Arrays.*asList*("A", "B", "c");  
// List<Phone> phoneList=list1.stream().map(x-> new Phone(x)).collect(Collectors.toList());  
 List<Phone> phoneList=list2.stream().map(Phone::new).collect(Collectors.*toList*());  
  
 }  
}  
  
class Phone{  
 private String name;  
 public Phone(String name) {  
 this.name = name;  
 }  
}

C:\Users\Roystan\.jdks\openjdk-21.0.2\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2023.3.3\lib\idea\_rt.jar=50926:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2023.3.3\bin" -Dfile.encoding=UTF-8 -Dsun.stdout.encoding=UTF-8 -Dsun.stderr.encoding=UTF-8 -classpath C:\Users\Roystan\IdeaProjects\JavaWorkspace\out\production\JavaWorkspace Streams.StreamWithLambda.MethodAndConstructorReference

A

B

c

Process finished with exit code 0

EX4

package Streams.StreamWithLambda.TerminalOpsPack.Collectors;  
  
import java.util.Arrays;  
import java.util.List;  
import java.util.stream.Collectors;  
  
public class CollectorsExample {  
 public static void main(String[] args) {  
  
  
 // Example 1: Collecting Names by Length  
 List<String> l1 = Arrays.*asList*("Anna", "Bob", "Alexander", "Brian", "Alice");  
 System.*out*.println(l1.stream().collect(Collectors.*groupingBy*(String::length)));  
  
 // Example 2: Counting Word Occurrences  
 String sentence = "hello world hello java world";  
 System.*out*.println(Arrays.*stream*(sentence.split(" ")).collect(Collectors.*groupingBy*(x -> x, Collectors.*counting*())));  
  
 // Example 3: Partitioning Even and Odd Numbers  
 List<Integer> l2 = Arrays.*asList*(1, 2, 3, 4, 5, 6);  
 System.*out*.println(l2.stream().collect(Collectors.*partitioningBy*(x -> x % 2 == 0)));  
 System.*out*.println(l2.stream().collect(Collectors.*partitioningBy*(x -> x % 2 == 0)));  
  
  
 }  
}

C:\Users\Roystan\.jdks\openjdk-21.0.2\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2023.3.3\lib\idea\_rt.jar=50931:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2023.3.3\bin" -Dfile.encoding=UTF-8 -Dsun.stdout.encoding=UTF-8 -Dsun.stderr.encoding=UTF-8 -classpath C:\Users\Roystan\IdeaProjects\JavaWorkspace\out\production\JavaWorkspace Streams.StreamWithLambda.TerminalOpsPack.Collectors.CollectorsExample

{3=[Bob], 4=[Anna], 5=[Brian, Alice], 9=[Alexander]}

{java=1, world=2, hello=2}

{false=[1, 3, 5], true=[2, 4, 6]}

Process finished with exit code 0

***Ex5***

package Streams.StreamWithLambda;  
  
import java.util.Arrays;  
import java.util.Random;  
import java.util.stream.Collectors;  
import java.util.stream.DoubleStream;  
import java.util.stream.IntStream;  
  
public class PrimitiveStreams {  
 public static void main(String[] args) {  
 int[] numbers = {1, 2, 3, 4, 5};  
 IntStream stream = Arrays.*stream*(numbers);  
  
 System.*out*.println(IntStream.*range*(1, 5).boxed().collect(Collectors.*toList*()));  
 System.*out*.println(IntStream.*rangeClosed*(1, 5).boxed().collect(Collectors.*toList*()));  
  
 IntStream.*of*(1, 2, 3);  
  
 DoubleStream doubles = new Random().doubles(5);  
// System.out.println("sum"+doubles.sum());  
// System.out.println("min"+doubles.min());  
// System.out.println("max"+doubles.max());  
// System.out.println("average"+doubles.average());  
// doubles.summaryStatistics();  
// System.out.println(doubles.mapToInt(x -> (int) (x + 1)));  
// System.out.println(doubles.boxed().toList());  
  
 IntStream intStream = new Random().ints(5);  
 System.*out*.println(intStream.boxed().toList());  
 }  
}

C:\Users\Roystan\.jdks\openjdk-21.0.2\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2023.3.3\lib\idea\_rt.jar=50938:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2023.3.3\bin" -Dfile.encoding=UTF-8 -Dsun.stdout.encoding=UTF-8 -Dsun.stderr.encoding=UTF-8 -classpath C:\Users\Roystan\IdeaProjects\JavaWorkspace\out\production\JavaWorkspace Streams.StreamWithLambda.PrimitiveStreams

[1, 2, 3, 4]

[1, 2, 3, 4, 5]

[1912354064, -888458096, -1105124393, -1886050550, -1908277690]

Process finished with exit code 0