***Collect containing methods***

***toList***

public class CollectingToList {  
 public static void main(String[] args) {  
 // Collectors is a utility class  
 // provides a set of methods to create common collectors  
  
  
 // 1. Collecting to a List  
 List<String> names = Arrays.*asList*("Alice", "Bob", "Charlie");  
 List<String> res = names.stream()  
 .filter(name -> name.startsWith("A"))  
 .collect(Collectors.*toList*());  
 System.*out*.println(res);  
  
 }  
}

***[Alice]***

***toSet***

public class CollectingToSet {  
 public static void main(String[] args) {  
  
 // 2. Collecting to a Set  
 List<Integer> nums = Arrays.*asList*(1, 2, 2, 3, 4, 4, 5);  
 Set<Integer> set = nums.stream().collect(Collectors.*toSet*());  
 System.*out*.println(set);  
  
  
 }  
}

***[1, 2, 3, 4, 5]***

***Specified Collection***

public class CollectingToSpecificCollection {  
 public static void main(String[] args) {  
  
 // 3. Collecting to a Specific Collection  
 List<String> names = Arrays.*asList*("Alice", "Bob", "Charlie");  
 ArrayDeque<String> collect = names.stream().collect(Collectors.*toCollection*(() -> new ArrayDeque<>()));  
  
  
  
 }  
}

***GetStatistics***

public class GetStatistics {  
  
 public static void main(String[] args) {  
  
 List<Integer> numbers = Arrays.*asList*(2, 3, 5, 7, 11);  
 IntSummaryStatistics stats = numbers.stream().collect(Collectors.*summarizingInt*(x -> x));  
 System.*out*.println("Count: " + stats.getCount());  
 System.*out*.println("Sum: " + stats.getSum());  
 System.*out*.println("Min: " + stats.getMin());  
 System.*out*.println("Average: " + stats.getAverage());  
 System.*out*.println("Max: " + stats.getMax());  
 Double average = numbers.stream().collect(Collectors.*averagingInt*(x->x));  
 System.*out*.println("Average: " + average);  
  
 }  
}

***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=50899: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.GetStatistics***

***Count: 5***

***Sum: 28***

***Min: 2***

***Average: 5.6***

***Max: 11***

***Average: 5.6***

***Process finished with exit code 0***

***Grouping By***

public class GroupingOfElements {  
 public static void main(String[] args) {  
 List<String> words = Arrays.*asList*("hello", "world", "java", "streams", "collecting");  
 System.*out*.println(words.stream().collect(Collectors.*groupingBy*(String::length)));  
 System.*out*.println(words.stream().collect(Collectors.*groupingBy*(String::length, Collectors.*joining*(", "))));  
 System.*out*.println(words.stream().collect(Collectors.*groupingBy*(String::length, Collectors.*counting*())));  
 TreeMap<Integer, Long> treeMap = words.stream().collect(Collectors.*groupingBy*(String::length, TreeMap::new, Collectors.*counting*()));  
 System.*out*.println(treeMap);  
 }  
}

***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=50902: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.GroupingOfElements***

***{4=[java], 5=[hello, world], 7=[streams], 10=[collecting]}***

***{4=java, 5=hello, world, 7=streams, 10=collecting}***

***{4=1, 5=2, 7=1, 10=1}***

***{4=1, 5=2, 7=1, 10=1}***

***Process finished with exit code 0***

***PartitioningBy***

package Streams.StreamWithLambda.TerminalOpsPack.Collectors;  
  
import java.util.Arrays;  
import java.util.List;  
import java.util.stream.Collectors;  
  
public class PartitioningOfElements {  
 public static void main(String[] args) {  
 List<String> words = Arrays.*asList*("hello", "world", "java", "streams", "collecting");  
 // 9. Partitioning Elements  
 // Partitions elements into two groups (true and false) based on a predicate  
 System.*out*.println(words.stream().collect(Collectors.*partitioningBy*(x -> x.length() > 5)));  
 }  
}

***{false=[hello, world, java], true=[streams, collecting]}***

***JoiningOfStreams***

package Streams.StreamWithLambda.TerminalOpsPack.Collectors;  
  
import java.util.Arrays;  
import java.util.List;  
import java.util.stream.Collectors;  
  
public class ConcatinationOfStream {  
 public static void main(String[] args) {  
  
 // 4. Joining Strings  
 // Concatenates stream elements into a single String  
 List<String> names = Arrays.*asList*("Alice", "Bob", "Charlie");  
 System.*out*.println(names.stream().collect(Collectors.*joining*(", ")));  
  
  
 }  
}

***Alice, Bob, Charlie***

***Map related collect operations***

package Streams.StreamWithLambda.TerminalOpsPack.Collectors;  
  
import java.util.\*;  
import java.util.stream.Collectors;  
  
public class MapRelatedExample {  
 public static void main(String[] args) {  
  
 Map<String, Integer> items = new HashMap<>();  
 items.put("Apple", 10);  
 items.put("Banana", 20);  
 items.put("Orange", 15);  
 System.*out*.println(items.values().stream().reduce(Integer::*sum*));  
 System.*out*.println(items.values().stream().collect(Collectors.*summingInt*(x -> x)));  
  
  
 }  
}

***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=50909: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.MapRelatedExample***

***Optional[45]***

***45***

***Process finished with exit code 0***

***Map with merge functionality***

package Streams.StreamWithLambda.TerminalOpsPack.Collectors;  
  
import java.util.Arrays;  
import java.util.List;  
import java.util.stream.Collectors;  
  
public class CollectingToMapWithMerge {  
 public static void main(String[] args) {  
  
 List<String> words2 = Arrays.*asList*("apple", "banana", "apple", "orange", "banana", "apple");  
 System.*out*.println(words2.stream().collect(Collectors.*toMap*(k -> k, v -> 1, (x, y) -> x + y)));;  
  
 }  
}

***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=50911: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.CollectingToMapWithMerge***

***{orange=1, banana=2, apple=3}***

***Process finished with exit code 0***