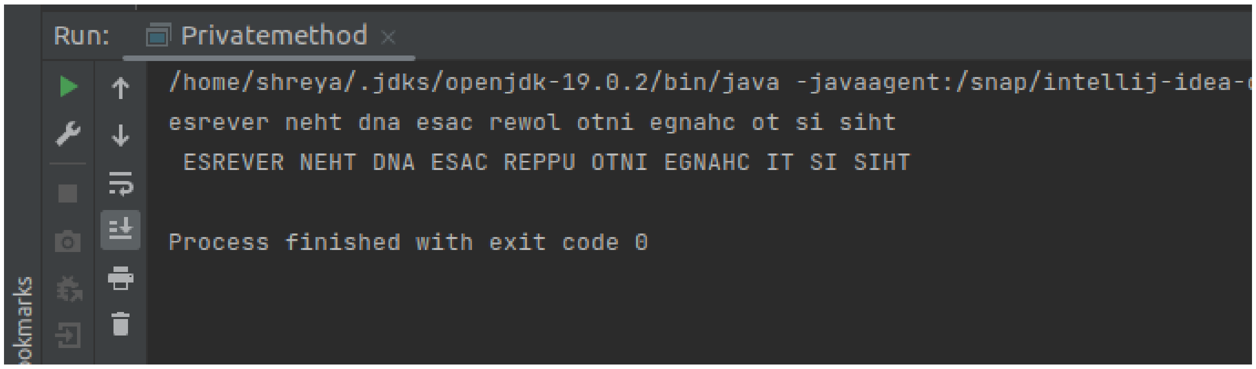
Java 9-17 Additions

Assignment

1. Demonstrate the use of private methods in interfaces

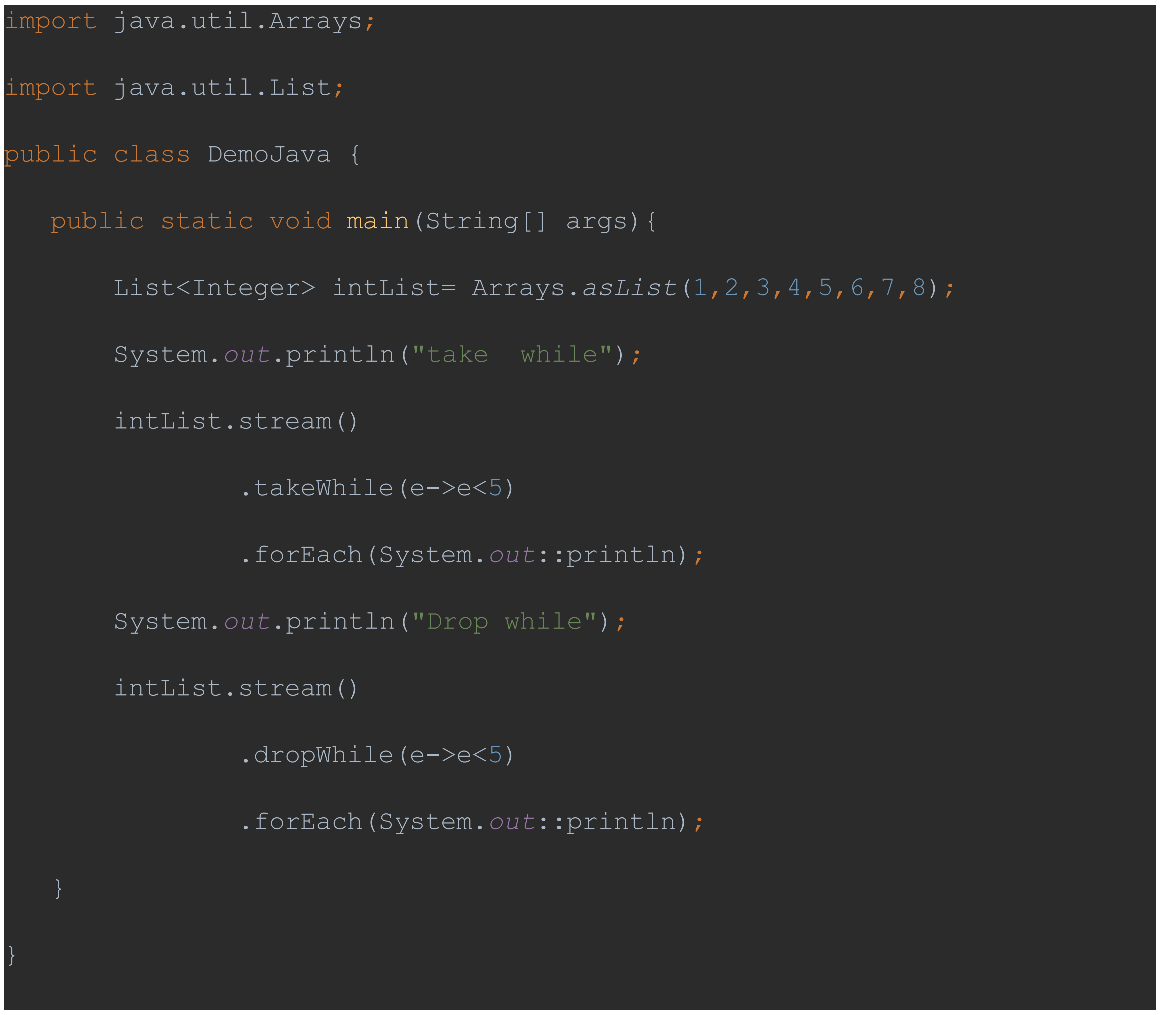
```java
public interface StringManipulationUtil {
   private static String reverseString(String s){
       return new StringBuilder(s).reverse().toString();
   }
   static String toUpperReverse(String s){
       String upperS=s.toUpperCase();
       return reverseString(upperS);
   }
   static String toLowerReverse(String s){
       String lowerS=s.toLowerCase();
       return reverseString(lowerS);
   }
}
```

```java
public class Privatemethod {
   public static void main(String[] args){
       System.out.println(StringManipulationUtil.toLowerReverse(" THIS IS TO CHANGE INTO LOWER CASE AND THEN REVERSE"));
       System.out.println(StringManipulationUtil.toUpperReverse("this is ti change into upper case and then reverse "));
   }
}
```

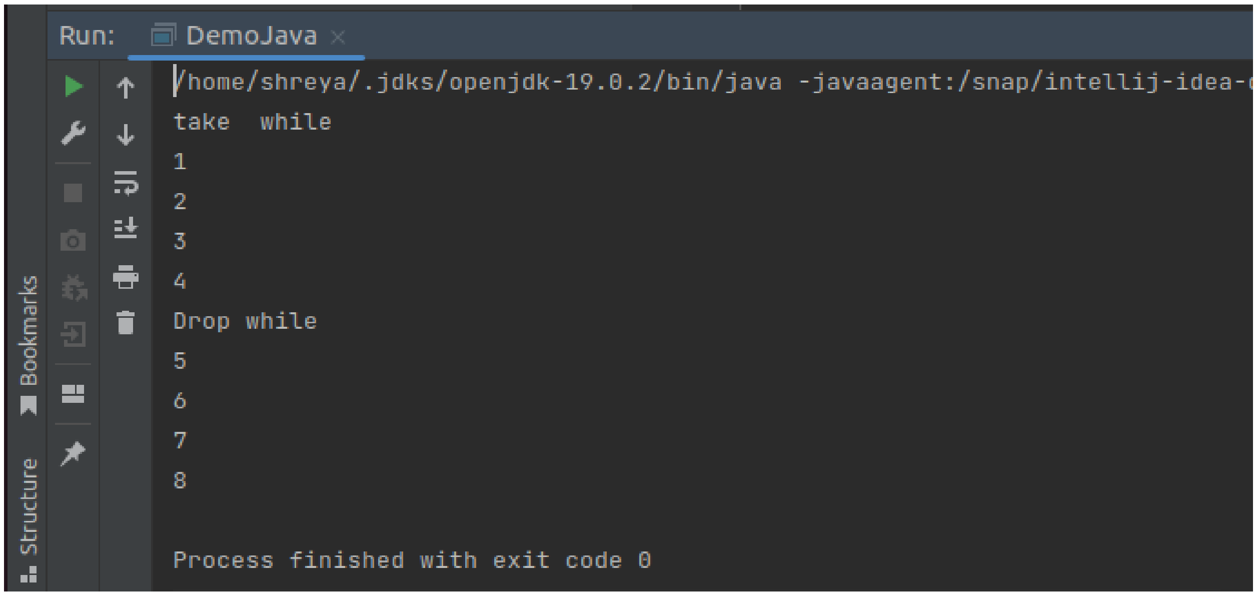


```
/home/shreya/.jdks/openjdk-19.0.2/bin/java -javaagent:/snap/intellij-idea-
esrever neht dna esac rewol otni egnahc ot si siht
 ESREVER NEHT DNA ESAC REPPU OTNI EGNAHC IT SI SIHT

Process finished with exit code 0
```

2. Perform takeWhile and dropWhile operations on stream



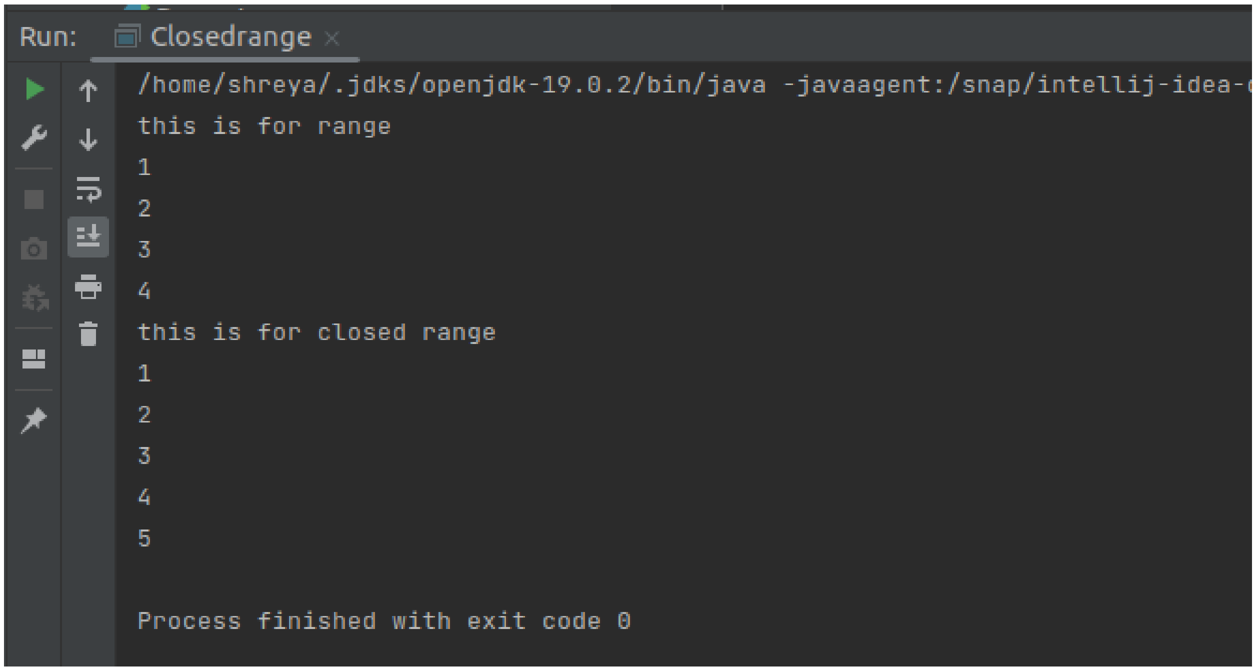
```
import java.util.Arrays;

import java.util.List;

public class DemoJava {

   public static void main(String[] args){

       List<Integer> intList= Arrays.asList(1,2,3,4,5,6,7,8);

       System.out.println("take  while");

       intList.stream()

               .takeWhile(e->e<5)

               .forEach(System.out::println);

       System.out.println("Drop while");

       intList.stream()

               .dropWhile(e->e<5)

               .forEach(System.out::println);

   }

}
```



```
/home/shreya/.jdks/openjdk-19.0.2/bin/java -javaagent:/snap/intellij-idea-
take  while
1
2
3
4
Drop while
5
6
7
8

Process finished with exit code 0
```

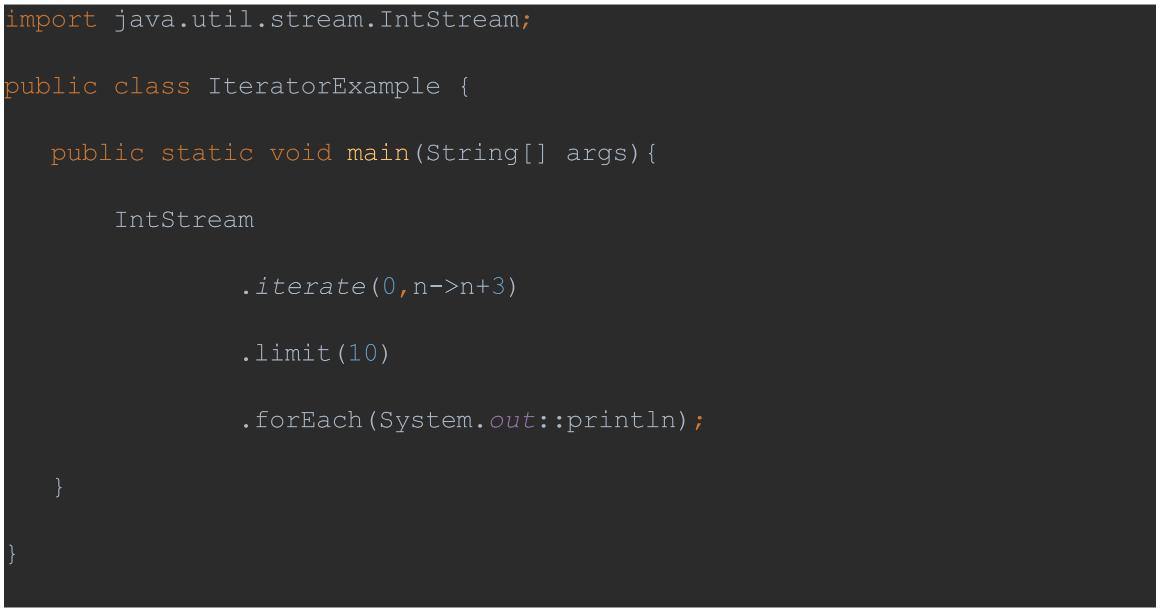
3. Use rangeClosed to create a Stream

```java
import java.util.stream.IntStream;

public class Closedrange {

   public static void main(String[] args){

       System.out.println("this is for range");

       IntStream.range(1,5).forEach(System.out::println);

       System.out.println("this is for closed range ");

       IntStream.rangeClosed(1,5).forEach(System.out::println);

   }

}
```

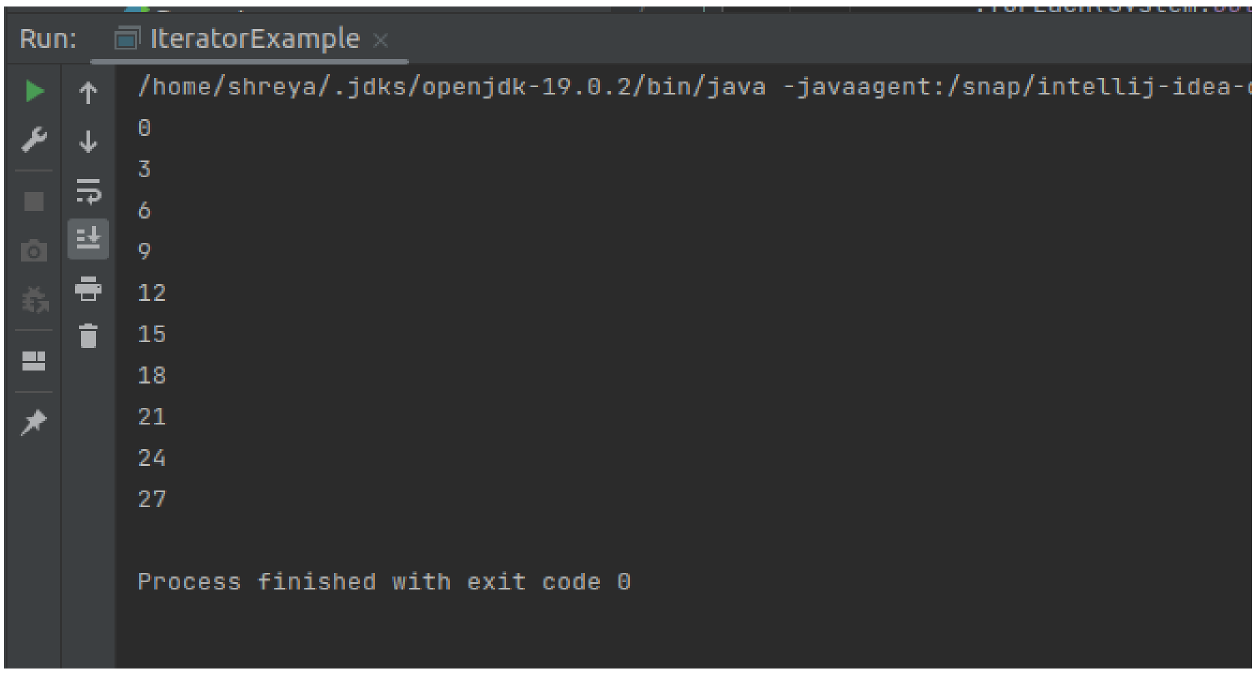


```
/home/shreya/.jdks/openjdk-19.0.2/bin/java -javaagent:/snap/intellij-idea-
this is for range
1
2
3
4
this is for closed range
1
2
3
4
5

Process finished with exit code 0
```

4. Use iterator stream method to generate a stream



```java
import java.util.stream.IntStream;

public class IteratorExample {

   public static void main(String[] args){

       IntStream

               .iterate(0,n->n+3)

               .limit(10)

               .forEach(System.out::println);

   }

}
```

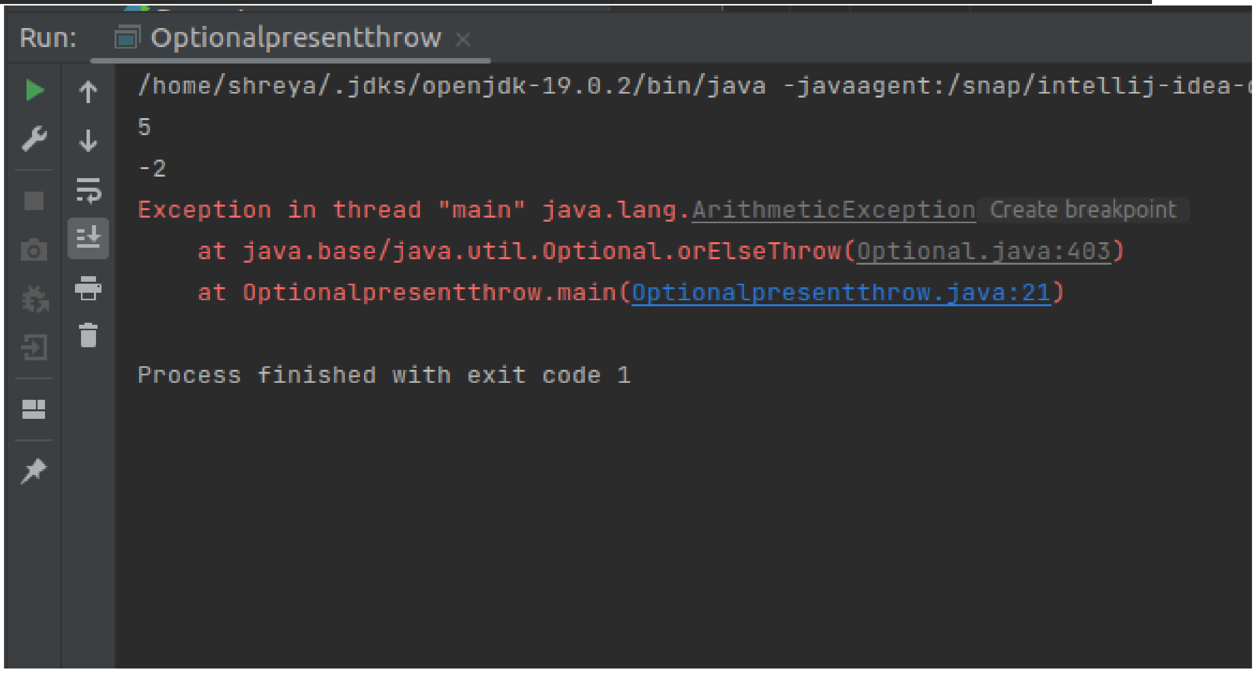


```
/home/shreya/.jdks/openjdk-19.0.2/bin/java -javaagent:/snap/intellij-idea-
0
3
6
9
12
15
18
21
24
27

Process finished with exit code 0
```

5. Use ifPresentOrElse, or, orElseThrow Operations with Optional

```
import java.util.Arrays;

import java.util.List;

import java.util.Optional;



public class Optionalpresentthrow {

   public static void main(String[] args){

       List<Integer> listy= Arrays.asList(1,2,3,4,5,6,7,8,9);

       listy

               .stream()

               .filter(e-> (e>4)).findFirst()

               .ifPresentOrElse(System.out::println,

                       ()-> System.out.println("this is for null
values"));

       listy

               .stream()
```

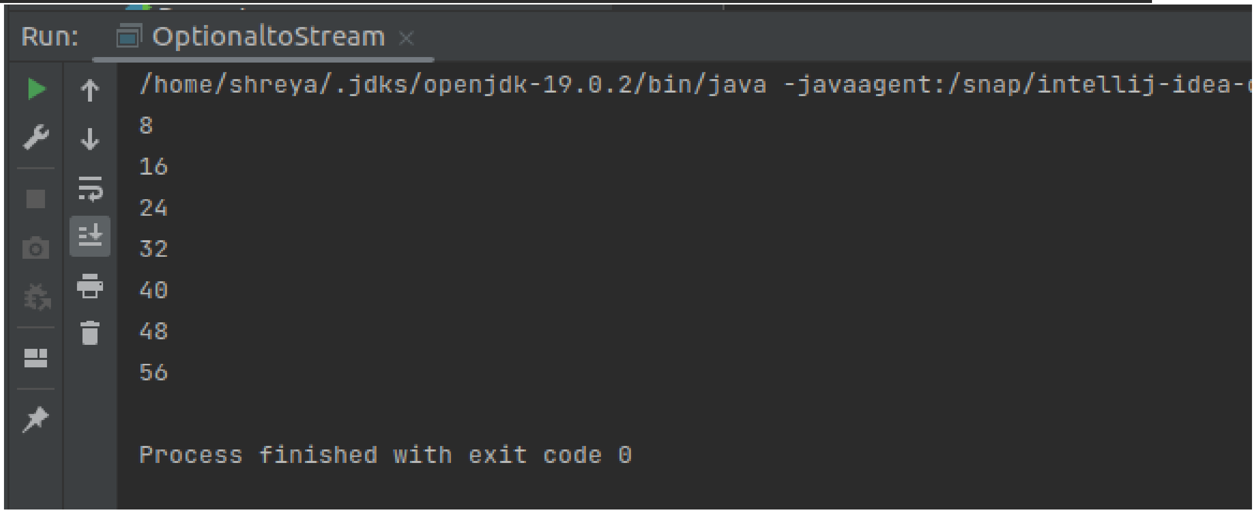
```
                        .filter(e->e>9).findFirst()

                        .or(()->
Optional.of(-2)).ifPresentOrElse(System.out::println,

                        ()->System.out.println("this is for null"));

       listy

               .stream()

               .filter(e->(e>11)).findFirst()

               .orElseThrow(ArithmeticException::new);

   }

}
```



```
Run:   Optionalpresentthrow ×
/home/shreya/.jdks/openjdk-19.0.2/bin/java -javaagent:/snap/intellij-idea-
5
-2
Exception in thread "main" java.lang.ArithmeticException Create breakpoint
    at java.base/java.util.Optional.orElseThrow(Optional.java:403)
    at Optionalpresentthrow.main(Optionalpresentthrow.java:21)

Process finished with exit code 1
```

6. Convert an Optional type into Stream

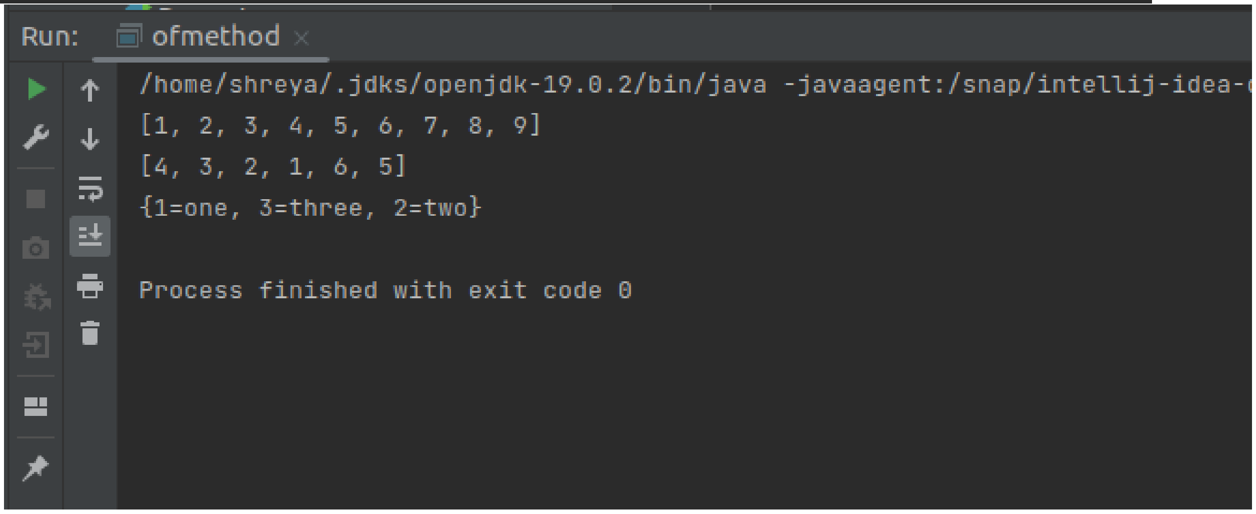
```
import java.util.Arrays;

import java.util.List;

import java.util.stream.IntStream;



public class OptionaltoStream {
   public static void main(String[] args){

       List<Integer> ilist= Arrays.asList(1,2,3,4,5,6,7,8,9);

       ilist

               .stream()

               .filter(e->(e>7)).findFirst().stream()

       .mapMulti((number,consumer)-> IntStream.rangeClosed(1,7)

               .forEach(e->consumer.accept(e*number)))

               .forEach(System.out::println);

   }

}
```



```
/home/shreya/.jdks/openjdk-19.0.2/bin/java -javaagent:/snap/intellij-idea-
8
16
24
32
40
48
56

Process finished with exit code 0
```

7. Use Of method to create List, Set and Map

```
import java.util.*;

public class ofmethod {

   public static void main(String[] args){

       System.out.println(List.of(1,2,3,4,5,6,7,8,9));

       System.out.println(Set.of(1,2,3,4,5,6));

       System.out.println(Map.of(1,"one",2,"two",3,"three"));

   }
}
```

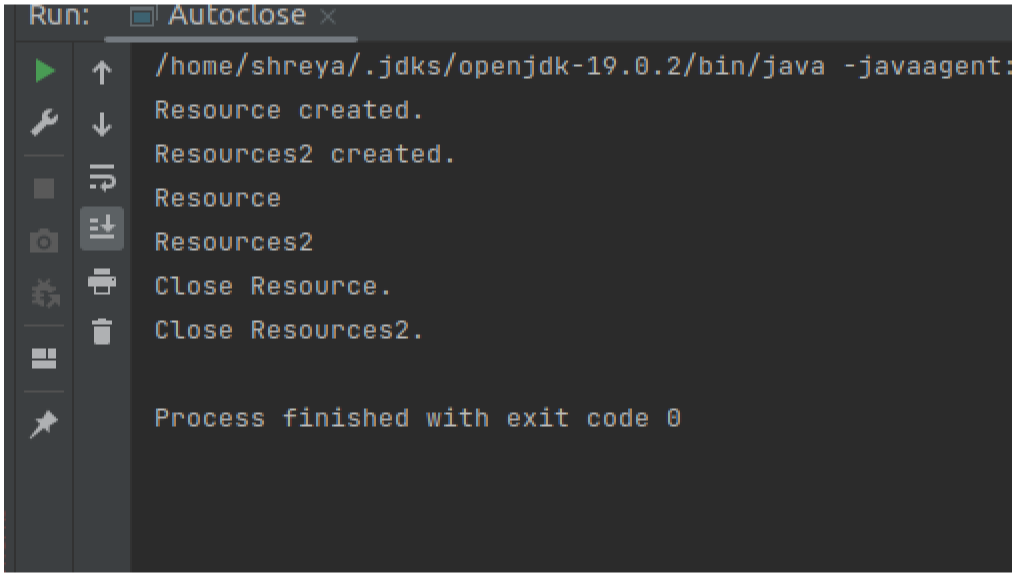


Run: ofmethod

```
/home/shreya/.jdks/openjdk-19.0.2/bin/java -javaagent:/snap/intellij-idea-
[1, 2, 3, 4, 5, 6, 7, 8, 9]
[4, 3, 2, 1, 6, 5]
{1=one, 3=three, 2=two}

Process finished with exit code 0
```

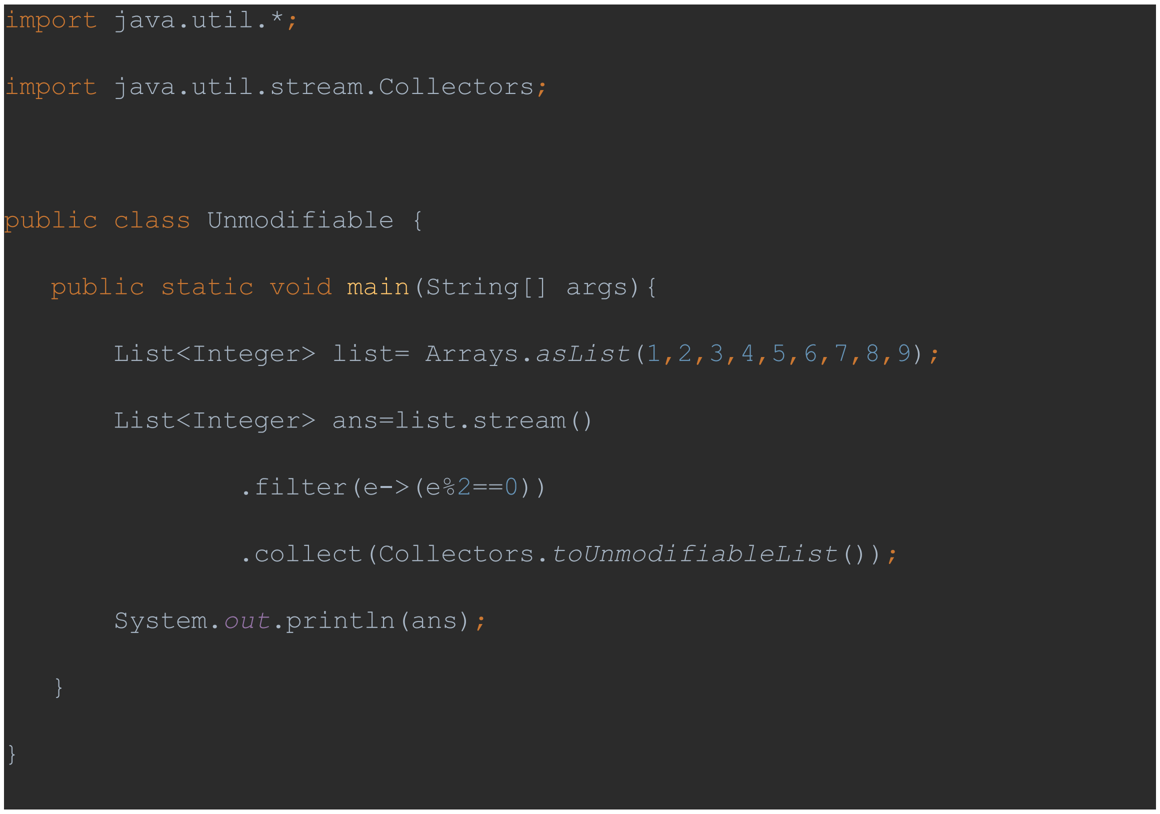
8. Demonstrate the use AutoCloseable

```java
public class Resource implements AutoCloseable {

   public Resource(){

       System.out.println("Resource created.");

   }

   public void display(){

       System.out.println("Resource");

   }


   @Override

   public void close() throws Exception {

       System.out.println("Close Resource.");

   }

}

public class Resources2 implements AutoCloseable{

   public Resources2(){

       System.out.println("Resources2 created.");
   }

   public void display(){

       System.out.println("Resources2");

   }

   @Override

   public void close() throws Exception {

       System.out.println("Close Resources2.");

   }

}


public class Autoclose {

public static void main(String[] args){

   Resource res=new Resource();

   Resources2 res2=new Resources2();

   try(res2;res){

       res.display();

       res2.display();

   } catch (Exception e) {

       throw new RuntimeException(e);

   }

}

}
```

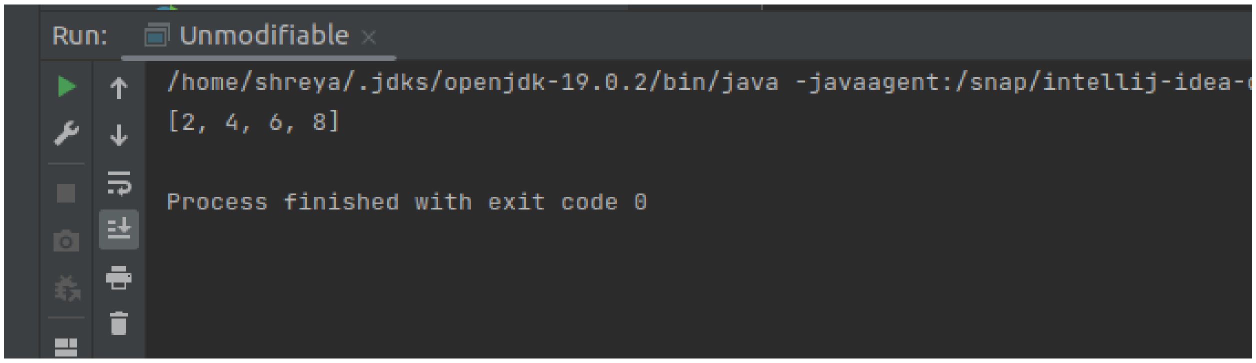


```
/home/shreya/.jdks/openjdk-19.0.2/bin/java -javaagent:
Resource created.
Resources2 created.
Resource
Resources2
Close Resource.
Close Resources2.

Process finished with exit code 0
```

9. Create Unmodifiable List from a Steam



```java
import java.util.*;

import java.util.stream.Collectors;


public class Unmodifiable {

   public static void main(String[] args){

       List<Integer> list= Arrays.asList(1,2,3,4,5,6,7,8,9);

       List<Integer> ans=list.stream()

               .filter(e->(e%2==0))

               .collect(Collectors.toUnmodifiableList());

       System.out.println(ans);

   }

}
```

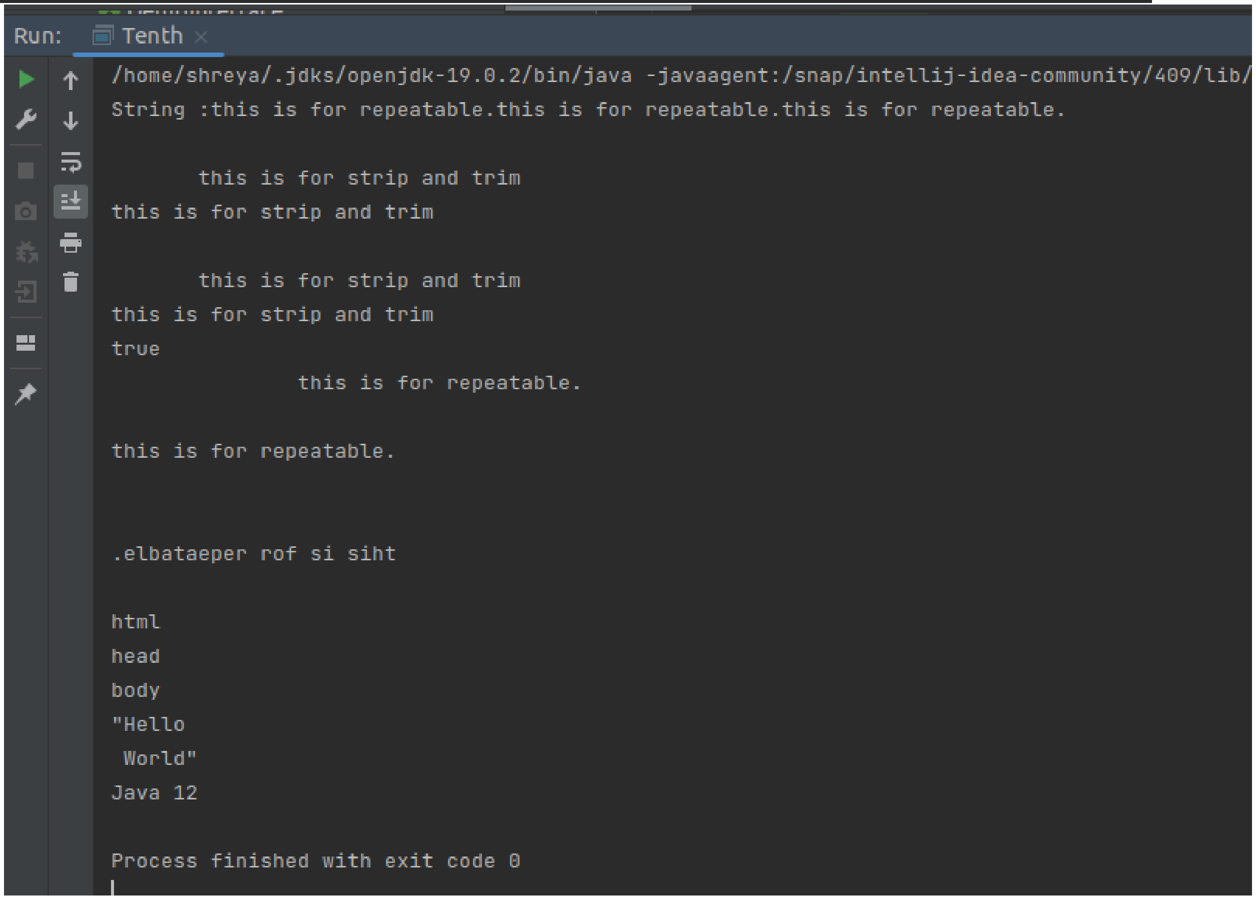


```
/home/shreya/.jdks/openjdk-19.0.2/bin/java -javaagent:/snap/intellij-idea-
[2, 4, 6, 8]

Process finished with exit code 0
```

10. Demonstrate the use of repeat, strip, isBlank, indent, transform, stripIndent, translateEscapes, formatted String methods.

```
public class Tenth {

   public static void main(String[] args){

       String str="this is for repeatable.";

       System.out.println("String :"+str.repeat(3));

       String st="\n\t   this is for strip and trim      \u2005";

       System.out.println(st);

       System.out.println(st.trim());

       System.out.println(st);

       System.out.println(st.strip());

       String blank="\n\t   ";

       System.out.println(blank.isBlank());

       str=str.indent(15);

       System.out.println(str);

       str=str.indent(-20);

       System.out.println(str);

       String reversestr=str.transform(string->new StringBuilder(string)

               .reverse().toString());

       System.out.println(reversestr);
```

```
        String html="\n\t\thtml\t"+"\n\t\thead\t"+"\n\t\tbody\t";

        System.out.println(html.stripIndent());

        String stri="\"Hello \\n World\"";

        System.out.println(stri.translateEscapes());

        System.out.println("Java %s".formatted("12"));

    }

}
```

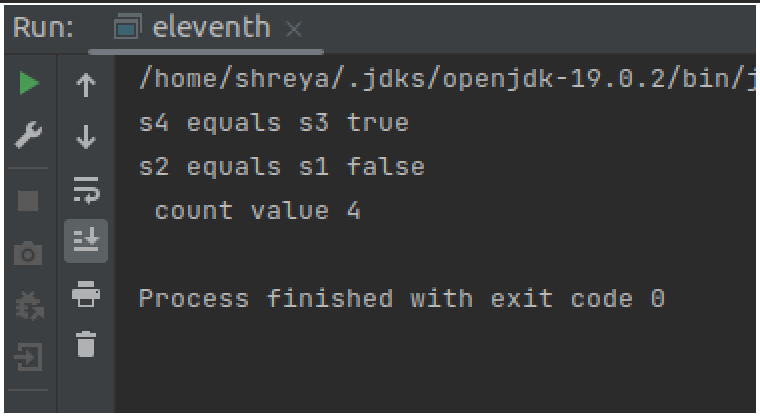


```
/home/shreya/.jdks/openjdk-19.0.2/bin/java -javaagent:/snap/intellij-idea-community/409/lib/
String :this is for repeatable.this is for repeatable.this is for repeatable.

       this is for strip and trim
this is for strip and trim

       this is for strip and trim
this is for strip and trim
true
               this is for repeatable.


this is for repeatable.



.elbataeper rof si siht


html
head
body
"Hello
 World"
Java 12

Process finished with exit code 0
```

11. Use record to create an immutable represent of student(name, id, age) and use its constructor for initialization, equals to compare 2 students methods. Also keep a static counter to keep the count of objects created.

```
import java.util.Objects;

public class eleventh {

    record Student(String Name,int id,int age) {
       static int count = 0;

       Student(String Name, int id, int age) {

           this.Name = Name;

           this.id = id;

           this.age = age;

           count++;

       }

       public static int getCount() {

           return count;

       }


       public boolean equals(Object obj) {

           if (this == obj) {

               return true;

           }

           if (!(obj instanceof Student)) {

               return false;

           }

           Student other = (Student) obj;

           return Name.equals(other.Name) && Objects.equals(id, other.id)
&& Objects.equals(age, other.age);

       }

   }

       public static void main(String[] args){

           Student s1=new Student("Shreya",34,12);

           Student s2=new Student("Shruti",45,55);
```

```java
        Student s3=new Student("Aarush",43,34);

        Student s4=new Student("Aarush",43,34);

        System.out.println("s4 equals s3 "+s4.equals(s3));

        System.out.println("s2 equals s1 "+s2.equals(s1));

        System.out.println(" count value "+ Student.getCount());

    }

}
```



```
/home/shreya/.jdks/openjdk-19.0.2/bin/j
s4 equals s3 true
s2 equals s1 false
 count value 4

Process finished with exit code 0
```

12. Demonstrate the use of Sealed Classes.

```java
public sealed class Shape permits Parallelogram,Rectangle,Circle {

    public double getArea() {

        return 0;

    }

}
```

```java
public final class Circle extends Shape{

    private final float radius;

    public Circle(float radius){

        this.radius=radius;

    }
```

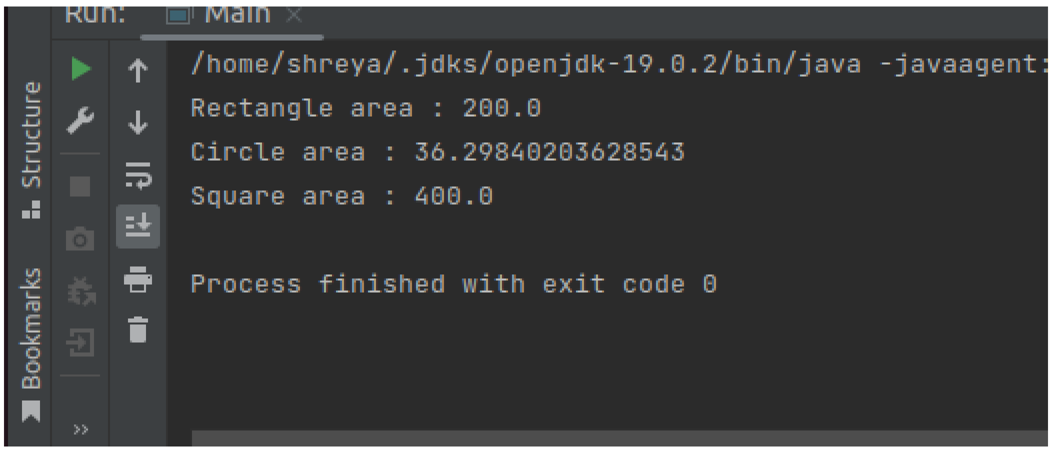
```
   @Override
   public double getArea(){
       return(3.14*radius*radius);
   }
}
```

```
public non-sealed class Rectangle extends Shape {
   private final float length,breadth;
   public Rectangle(float length,float breadth){
       this.length=length;
       this.breadth=breadth;
   }
   @Override
   public double getArea(){
       return(length*breadth);
   }
}
```

```
public sealed class Parallelogram extends Shape permits Square {


   @Override
   public double getArea(){
       return 0;
   }
}
```

```java
public non-sealed class Square extends Parallelogram{

   private final float length;

   public Square(float length){

       this.length=length;

   }

   @Override

   public double getArea(){

       return(length*length);

   }

}
```

```java
public class Main {

   public static void main(String[] args) {

       Circle s=new Circle(3.4F);

       Rectangle r=new Rectangle(20,10);

       Square sq=new Square(20);

       System.out.println("Rectangle area : "+r.getArea());

       System.out.println("Circle area : "+s.getArea());

       System.out.println("Square area : "+sq.getArea());

   }

}
```



```
/home/shreya/.jdks/openjdk-19.0.2/bin/java -javaagent:
Rectangle area : 200.0
Circle area : 36.29840203628543
Square area : 400.0

Process finished with exit code 0
```