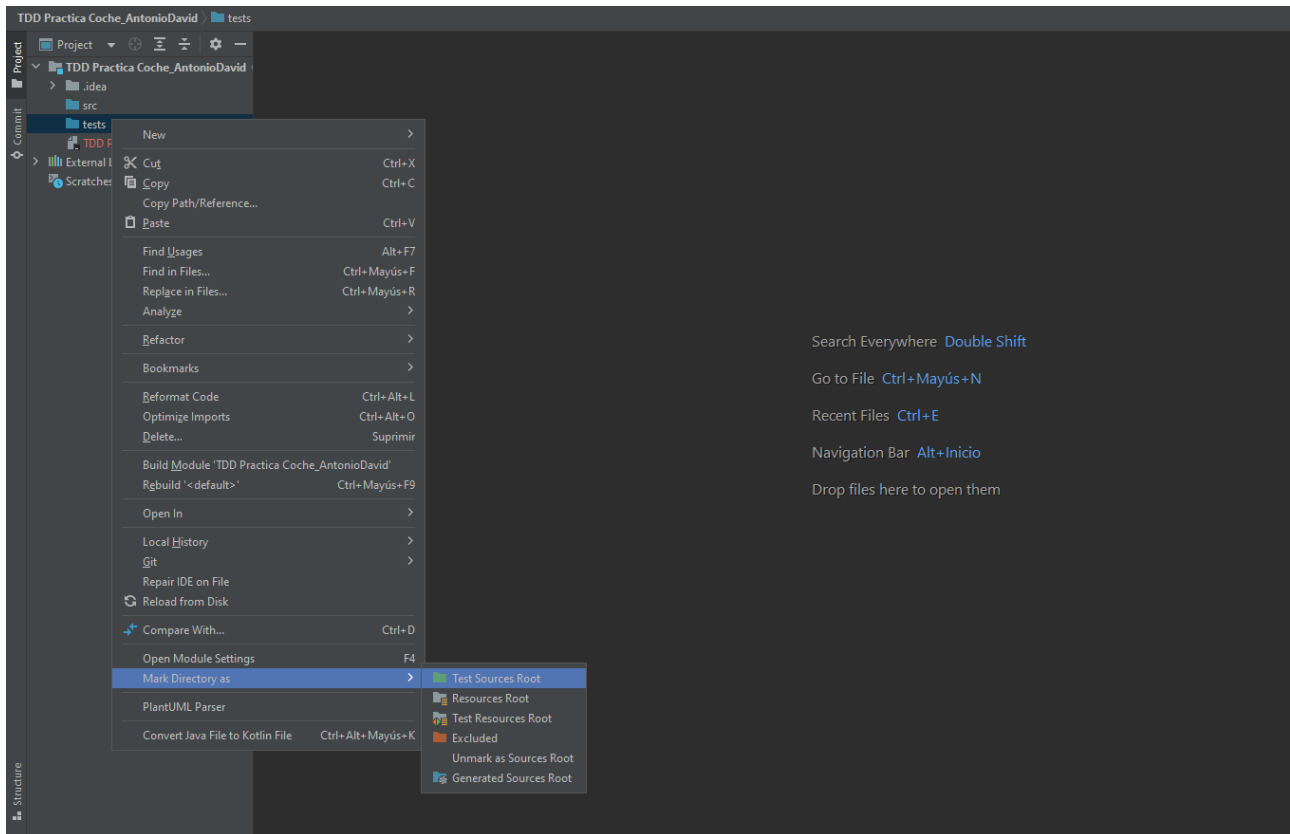
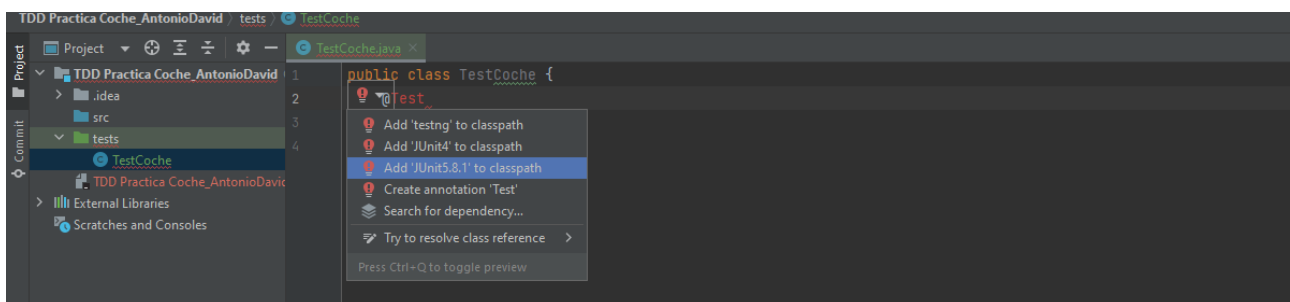


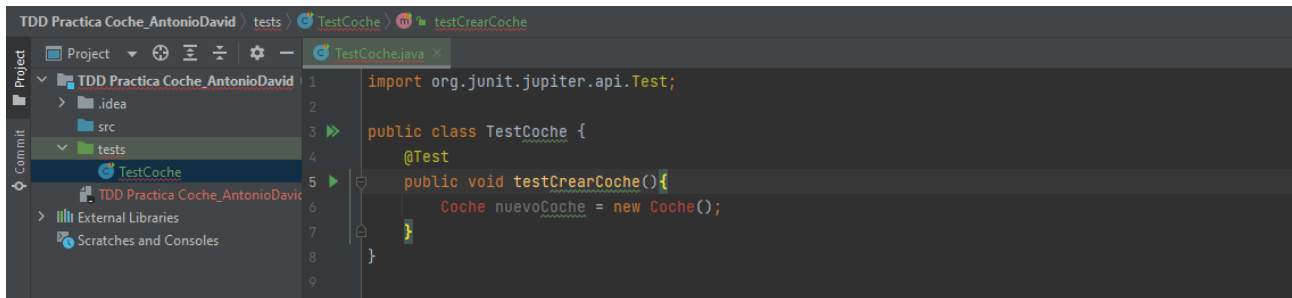
MEMORIA TDD EN INTELLIJ



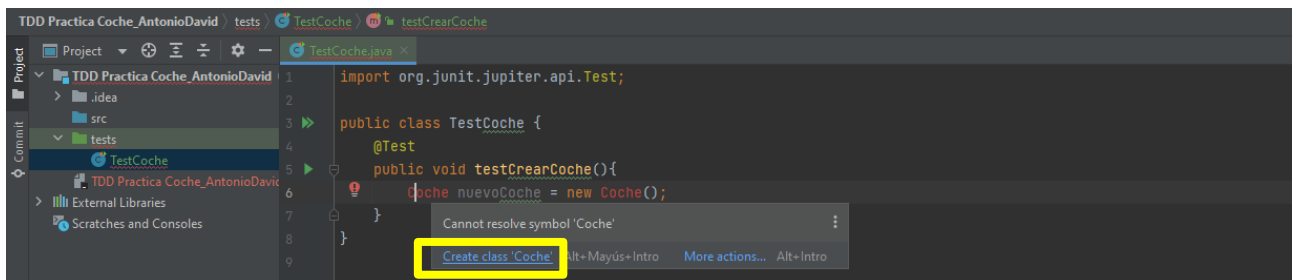
Creamos nuestro nuevo proyecto y creamos un directorio llamado “tests”, este directorio lo marcamos como “Test Sources Root”, ya que va a ser un directorio de pruebas.



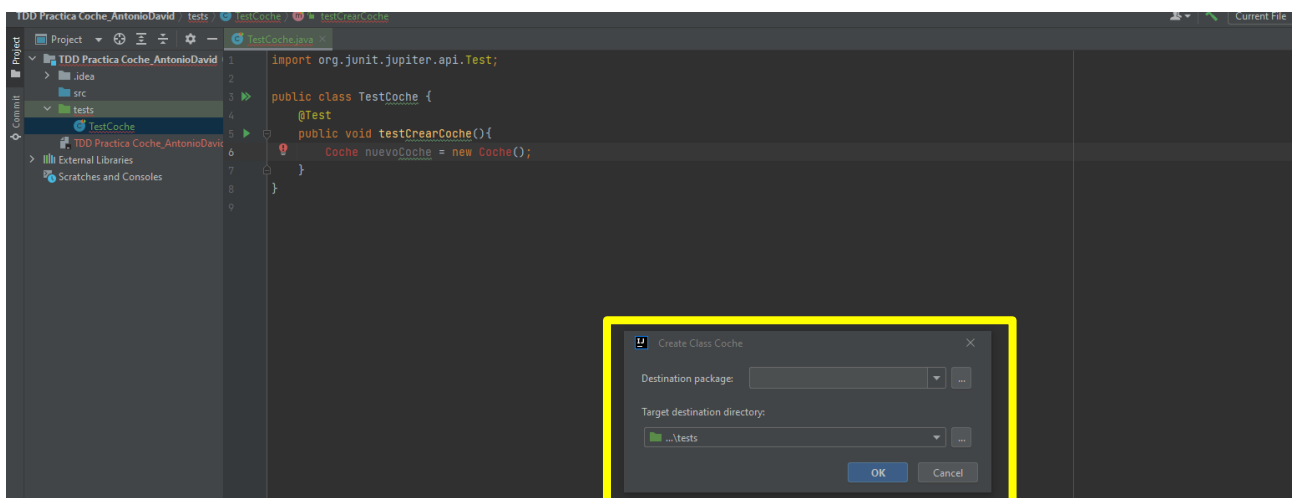
Una vez que lo hayamos marcado como un directorio de tests, creamos una nueva clase Java, llamada TestCoche, escribimos @Test, y clicamos en la bombilla que aparece a la izquierda la opción de Junit 5.



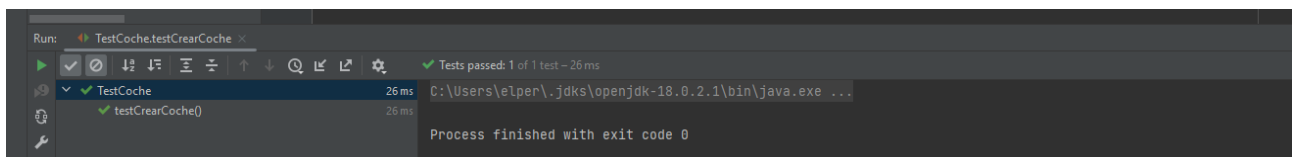
Creamos nuestro primer test y como vemos el mismo IntelliJ nos indica un error de compilación, ya que la clase Coche no existe.



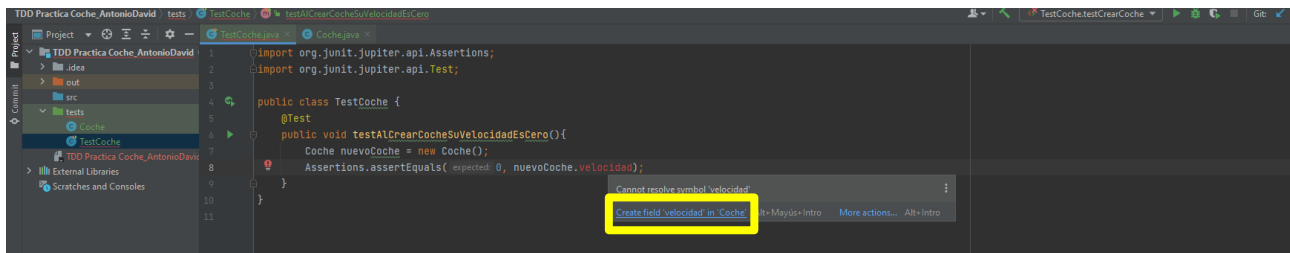
Para solucionar esto de una forma rápida ponemos el cursor encima de “Coche” e IntelliJ ya nos ofrece la solución, clicamos en ella y se nos creará la clase Coche.



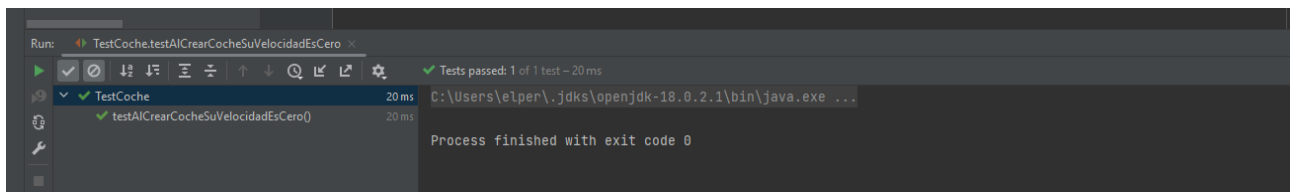
Nos saldrá esta ventana, clicamos en “Ok” y continuamos.



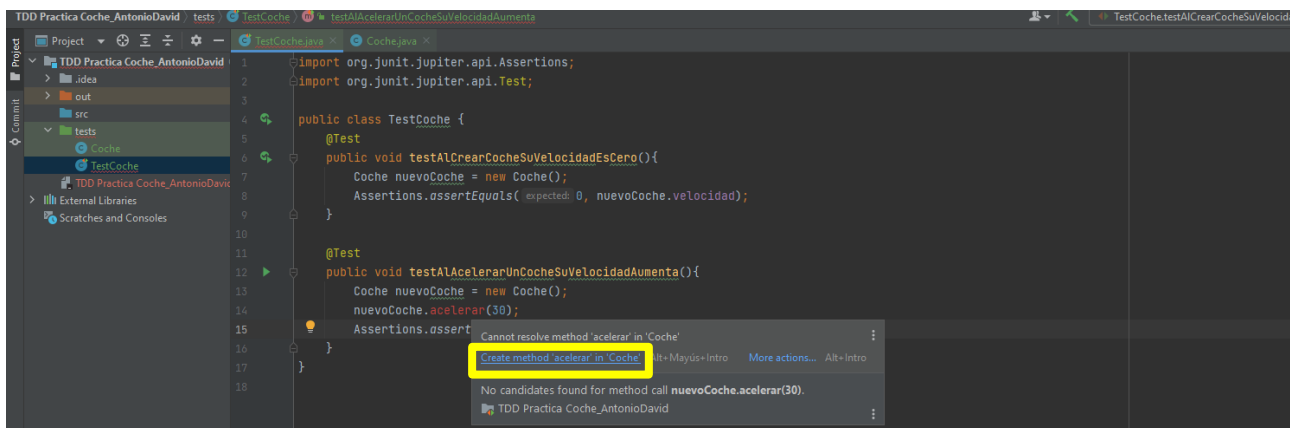
Ejecutamos nuestro test, y vemos que todo funciona correctamente.



Ahora vamos a hacer algunas pruebas un poco más elaboradas, con esta nueva línea de código le indicamos que cuando creamos un coche la velocidad de este debe de ser igual a cero. El programa ya nos está indicando que el atributo velocidad no se encuentra en la clase Coche, hacemos lo mismo que antes, para que se cree automáticamente.



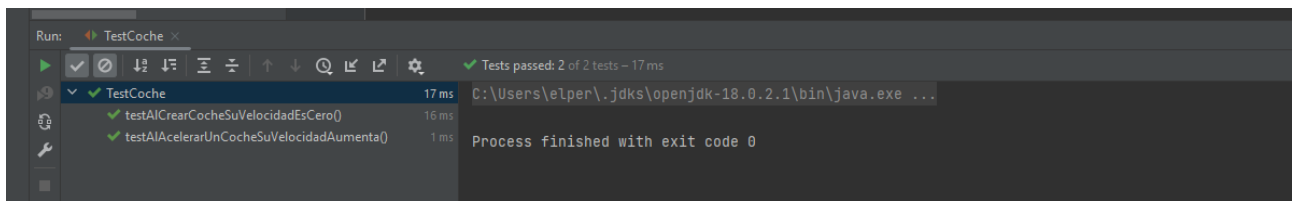
Ejecutamos nuestro test y vemos que funciona correctamente, ya que la velocidad es 0.



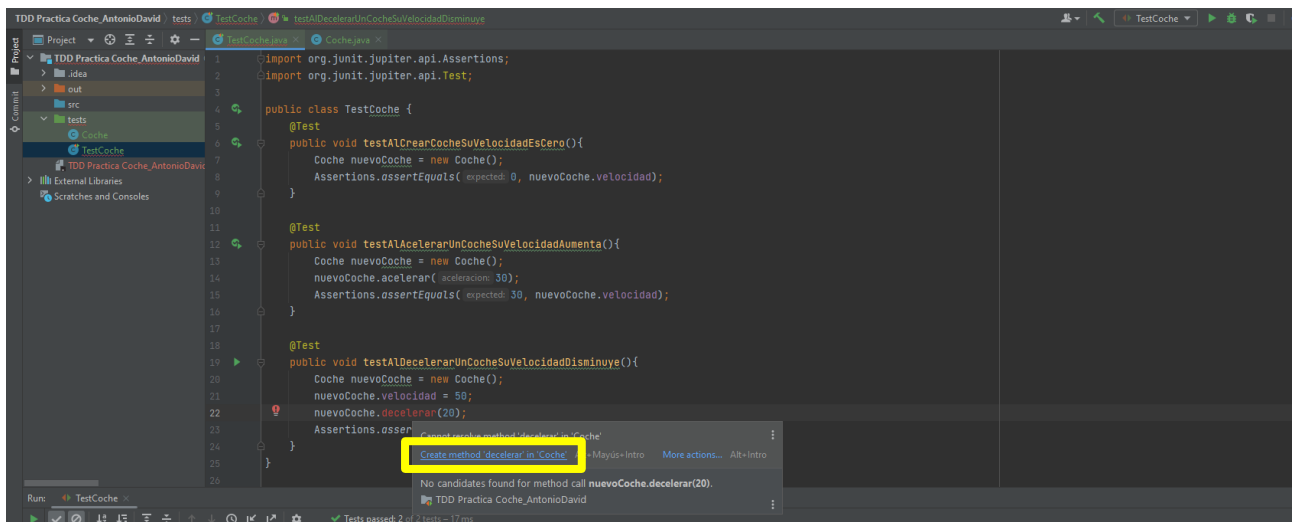
A continuación creamos otro método, en este le indicamos que el coche acelera 30 km/h, por tanto la velocidad debe de ser igual a 30, es decir `Assertions.assertEquals(30, nuevoCoche.velocidad);`. Como acelerar es un método que no está en la clase Coche lo creamos con la ayuda de Inteli.



Lo modificamos con un parámetro de entrada que refleje claramente la acción que va a hacer, y a velocidad le sumamos la aceleración.



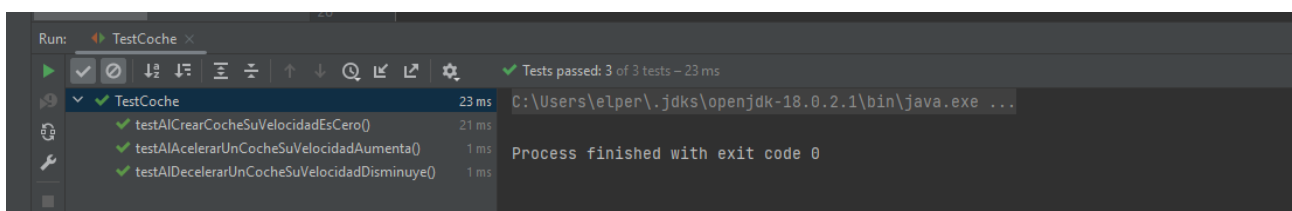
Ejecutamos nuestro test, y vemos que funciona correctamente.



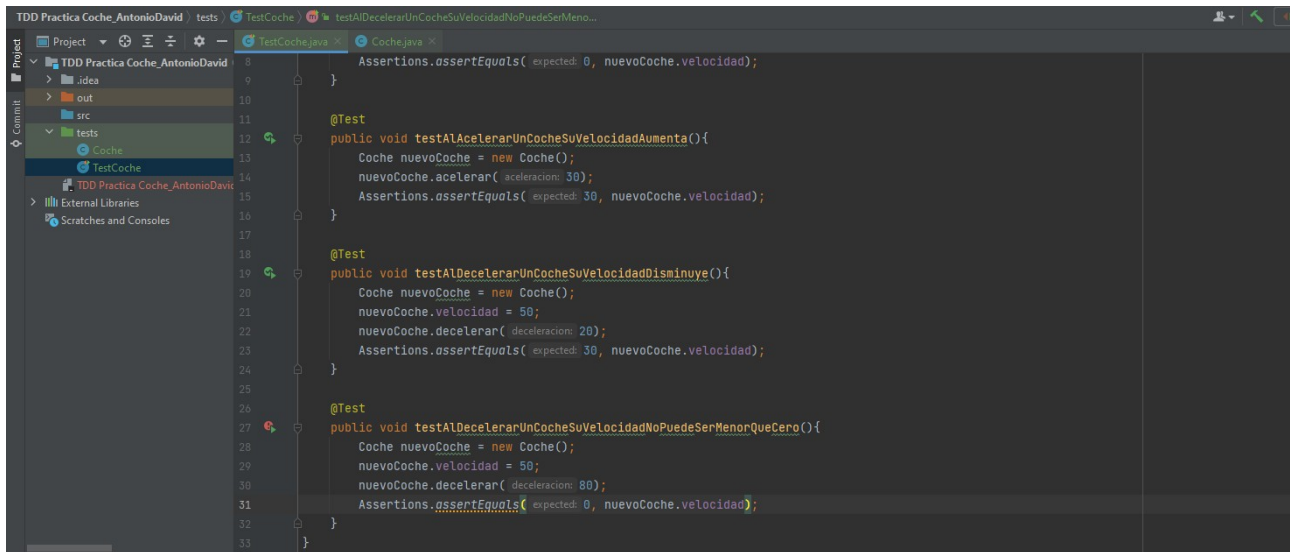
Ahora creamos otro test que en vez de acelerar va a decelerar, es decir va a reducir la velocidad, para ello primero declaramos una velocidad base mayor a la que le vamos a quitar, en este caso 50, después deceleramos llamando al método decelerar que crearemos con ayuda de Intelli, y por último le decimos que la velocidad debe de ser igual a 30, haciendo uso de `Assertions.assertEquals(30, nuevoCoche.velocidad)`.



Creamos el método decelerar y lo modificamos de tal modo que a velocidad se le reste el valor que se le pasa por parámetro al método, en este caso deceleración.

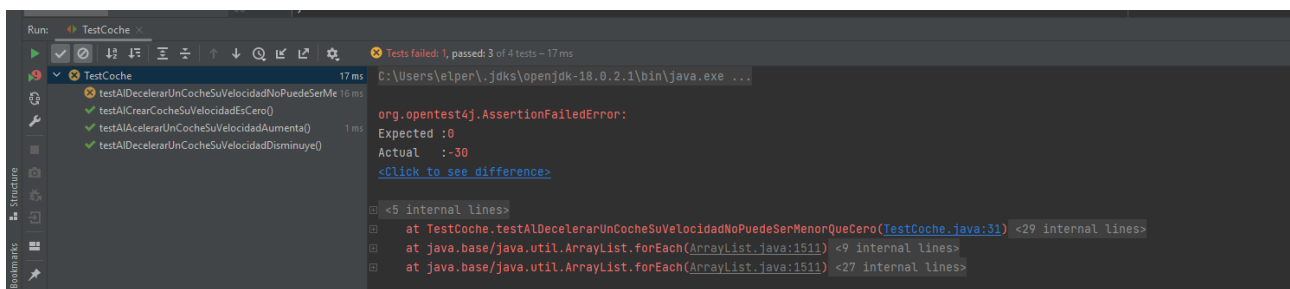


Ejecutamos nuestros tests y vemos que todo sigue funcionando correctamente.



```
8      Assertions.assertEquals( expected: 0, nuevoCoche.velocidad);
9  }
10
11
12  @Test
13  public void testAlAcelerarUnCocheSuVelocidadAumenta(){
14      Coche nuevoCoche = new Coche();
15      nuevoCoche.acelerar( aceleracion: 30);
16      Assertions.assertEquals( expected: 30, nuevoCoche.velocidad);
17  }
18
19  @Test
20  public void testAlDecelerarUnCocheSuVelocidadDisminuye(){
21      Coche nuevoCoche = new Coche();
22      nuevoCoche.velocidad = 50;
23      nuevoCoche.decelerar( deceleracion: 20);
24      Assertions.assertEquals( expected: 30, nuevoCoche.velocidad);
25  }
26
27  @Test
28  public void testAlDecelerarUnCocheSuVelocidadNoPuedeSerMenorQueCero(){
29      Coche nuevoCoche = new Coche();
30      nuevoCoche.velocidad = 50;
31      nuevoCoche.decelerar( deceleracion: 80);
32      Assertions.assertEquals( expected: 0, nuevoCoche.velocidad);
33  }
```

Ahora vamos a crear a posta un método que va a dar un error según tenemos nuestro programa hasta el momento, como vemos le estamos diciendo que el coche va a una velocidad de 50 y nosotros deceleramos 80, por lo tanto el coche debería de parar completamente, es decir que su velocidad sea 0.

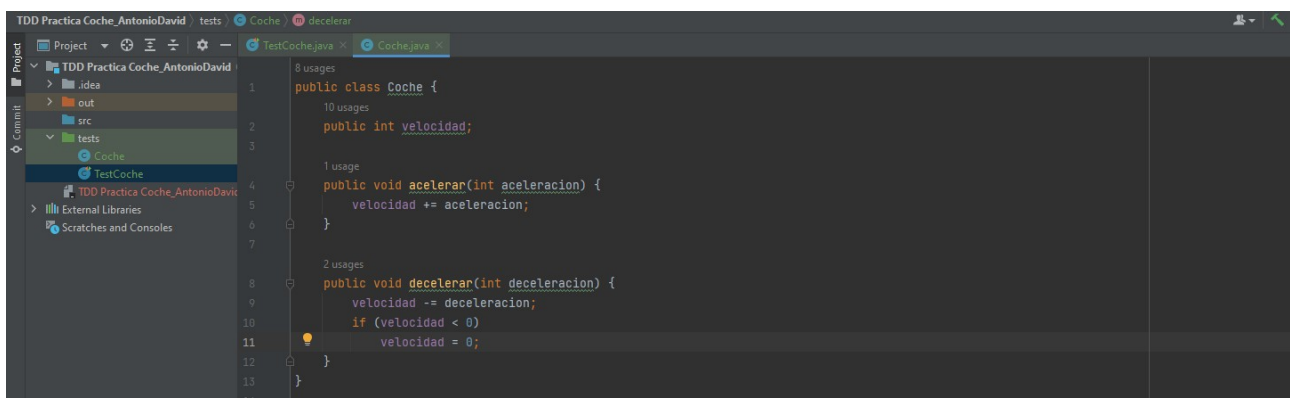


```
Run: TestCoche
Tests failed: 1, passed: 3 of 4 tests - 17 ms
C:\Users\elper\.jdk\openjdk-18.0.2.1\bin\java.exe ...

org.opentest4j.AssertionFailedError:
Expected: 0
Actual: -30
<Click to see difference>

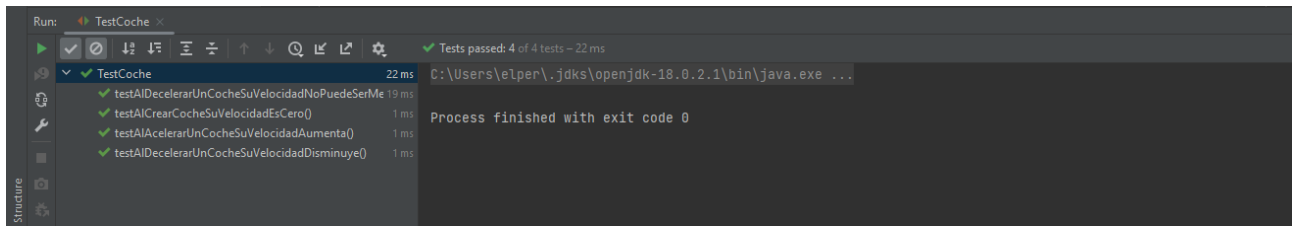
<5 internal lines>
at TestCoche.testAlDecelerarUnCocheSuVelocidadNoPuedeSerMenorQueCero(TestCoche.java:31) <29 internal lines>
at java.base/java.util.ArrayList.forEach(ArrayList.java:1511) <9 internal lines>
at java.base/java.util.ArrayList.forEach(ArrayList.java:1511) <27 internal lines>
```

Si ejecutamos esto vemos que, el programa compila pero el test falla, ya que le hemos indicado que la velocidad debía de ser igual a cero porque el coche paraba completamente, pero realmente la velocidad actual es -30, porque $50 - 80$, es -30.



```
1 public class Coche {
2     public int velocidad;
3
4     public void acelerar(int aceleracion) {
5         velocidad += aceleracion;
6     }
7
8     public void decelerar(int deceleracion) {
9         velocidad -= deceleracion;
10        if (velocidad < 0)
11            velocidad = 0;
12    }
13 }
14
```

Para solucionar esto debemos de tener una condición en nuestro método “decelerar” en el que velocidad será cero en caso de que velocidad sea menor que 0.



Ahora ya podemos ejecutar el test y vemos que todo funciona correctamente.