

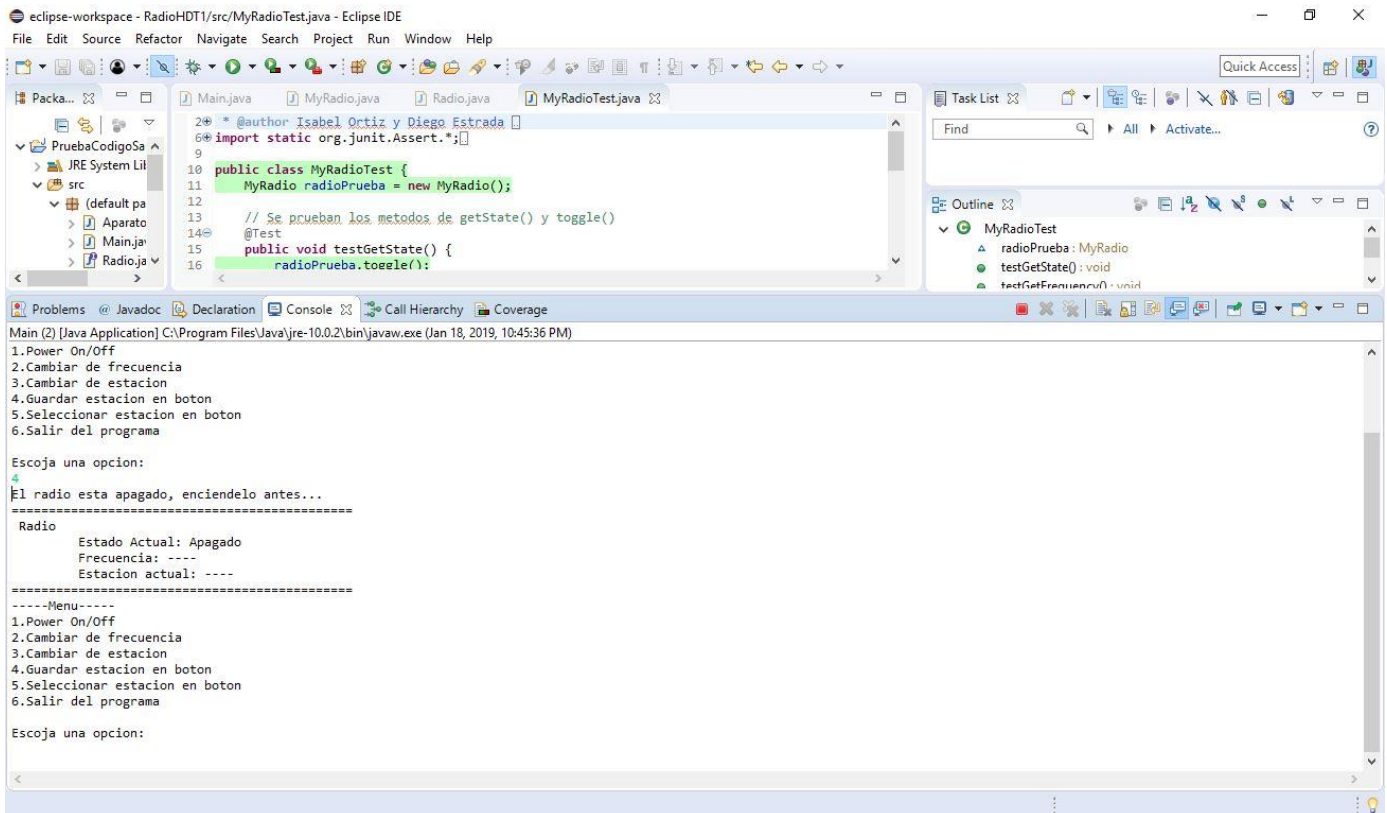
Diego Estrada – Carne 18540

María Isabel Ortiz Naranjo – Carne 18176

Hoja de trabajo 1

Funcionamiento del programa

Opción 1



The screenshot shows the Eclipse IDE interface. The main editor displays the `MyRadioTest.java` file with the following code:

```
20 * @author Isabel Ortiz y Diego Estrada
21
22 import static org.junit.Assert.*;
23
24 public class MyRadioTest {
25     MyRadio radioPrueba = new MyRadio();
26
27     // Se prueban los metodos de getState() y toggle()
28     @Test
29     public void testGetState() {
30         radioPrueba.toggle();
31     }
32 }
```

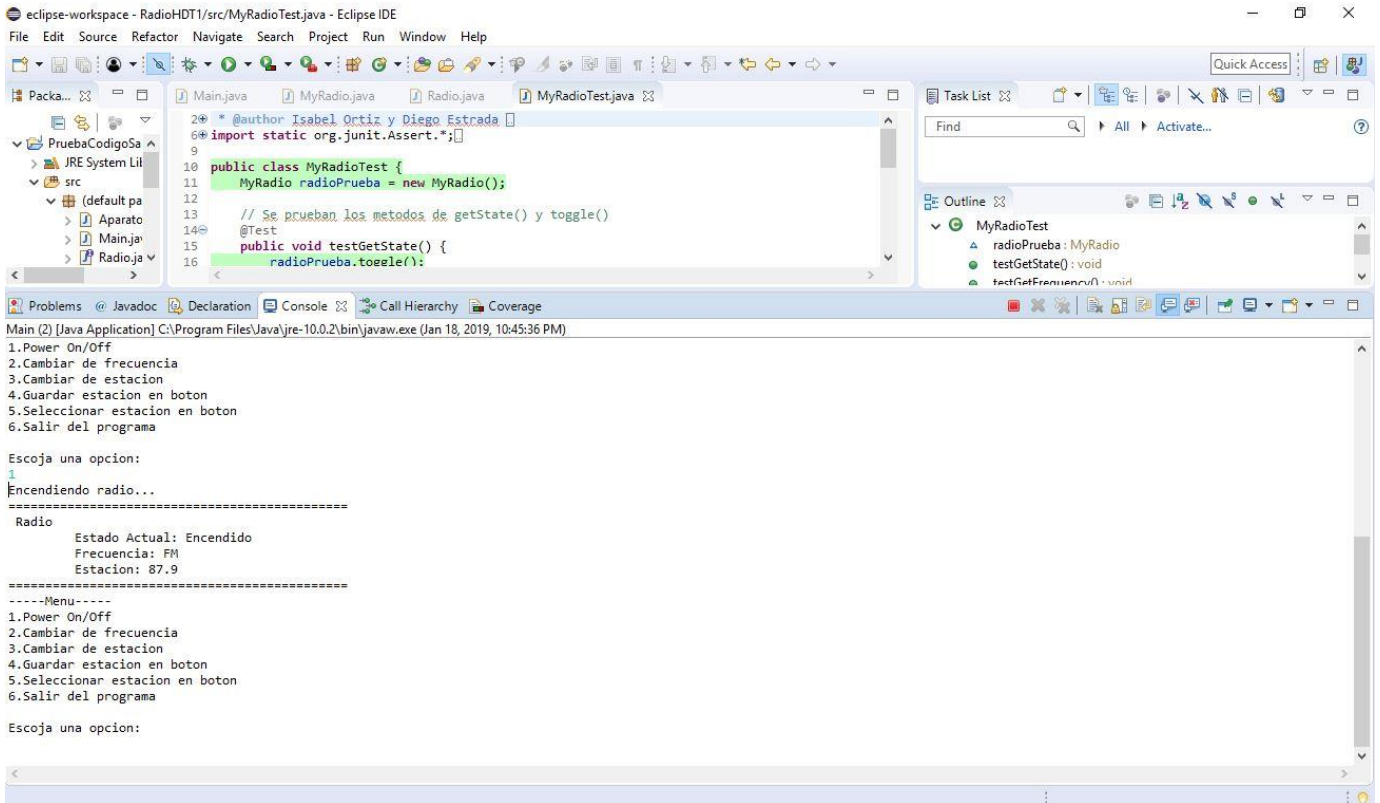
The console output shows the execution of the program:

```
Main (2) [Java Application] C:\Program Files\Java\jre-10.0.2\bin\javaw.exe (Jan 18, 2019, 10:45:36 PM)
1.Power On/Off
2.Cambiar de frecuencia
3.Cambiar de estacion
4.Guardar estacion en boton
5.Seleccionar estacion en boton
6.Salir del programa

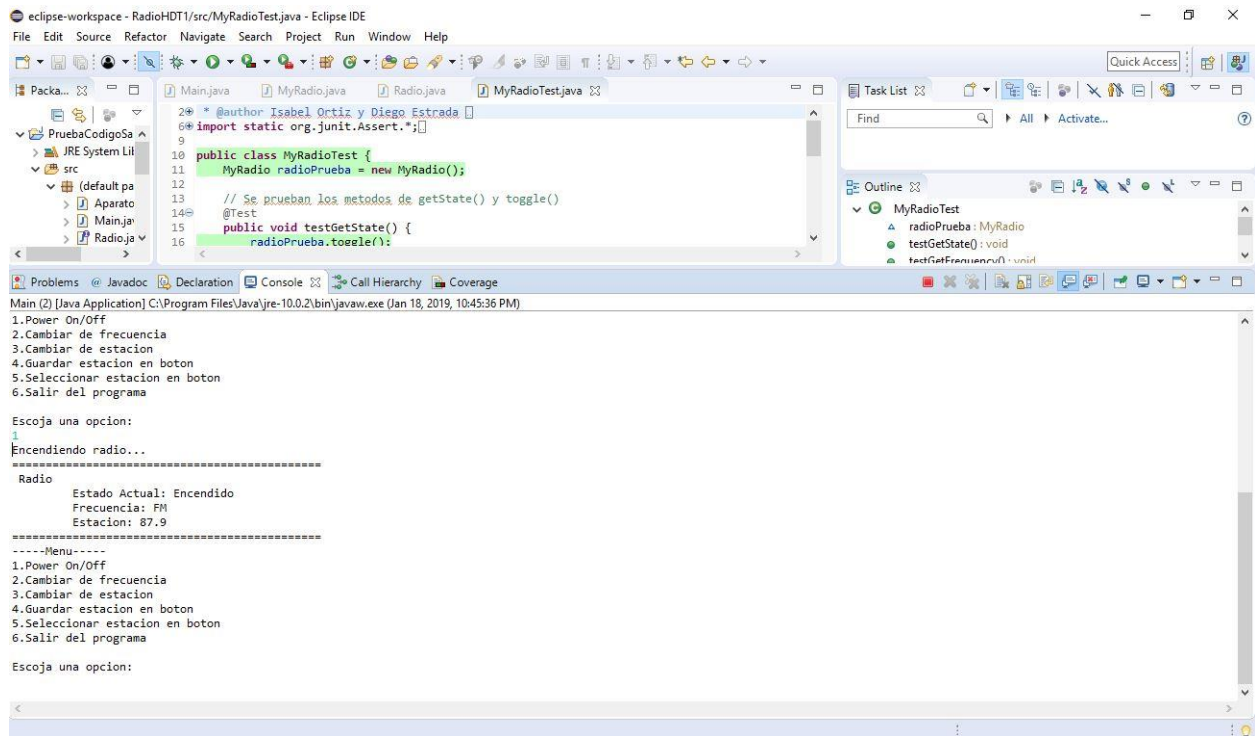
Escoja una opcion:
4
El radio esta apagado, enciendolo antes...
=====
Radio      Estado Actual: Apagado
           Frecuencia: ----
           Estacion actual: ----
=====
-----Menu-----
1.Power On/Off
2.Cambiar de frecuencia
3.Cambiar de estacion
4.Guardar estacion en boton
5.Seleccionar estacion en boton
6.Salir del programa

Escoja una opcion:
```

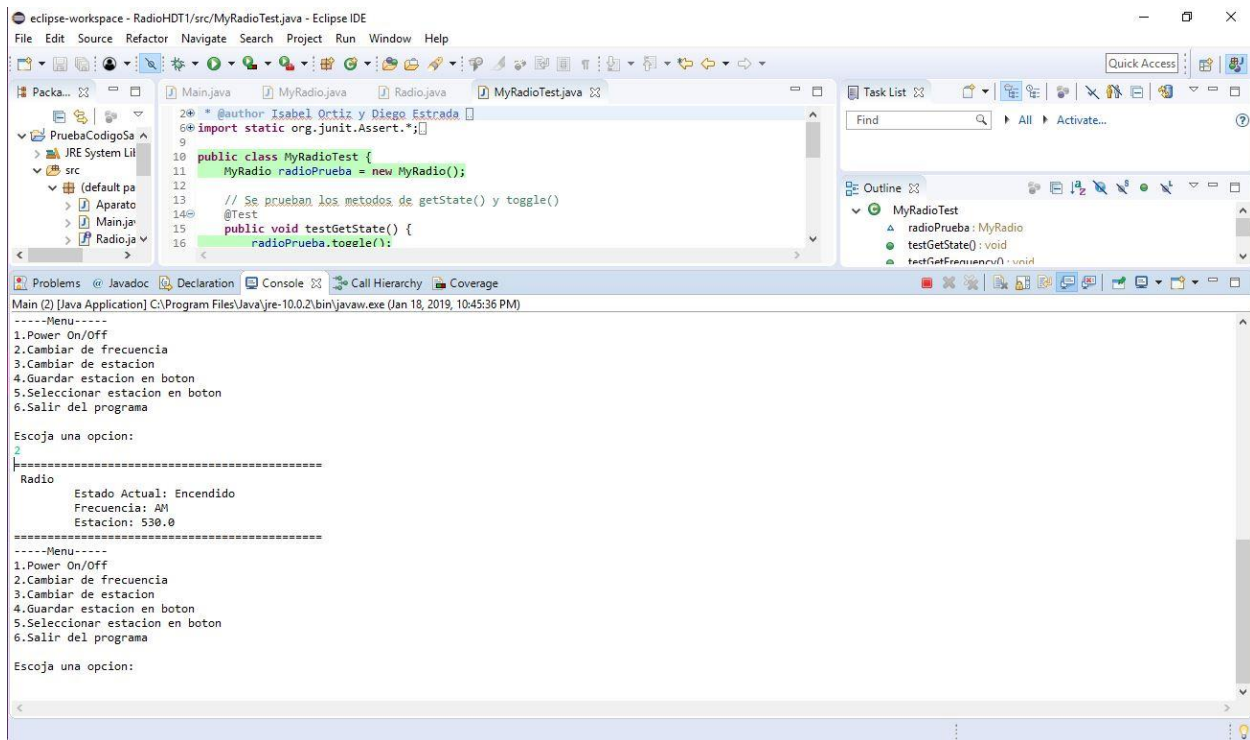
Opción 1



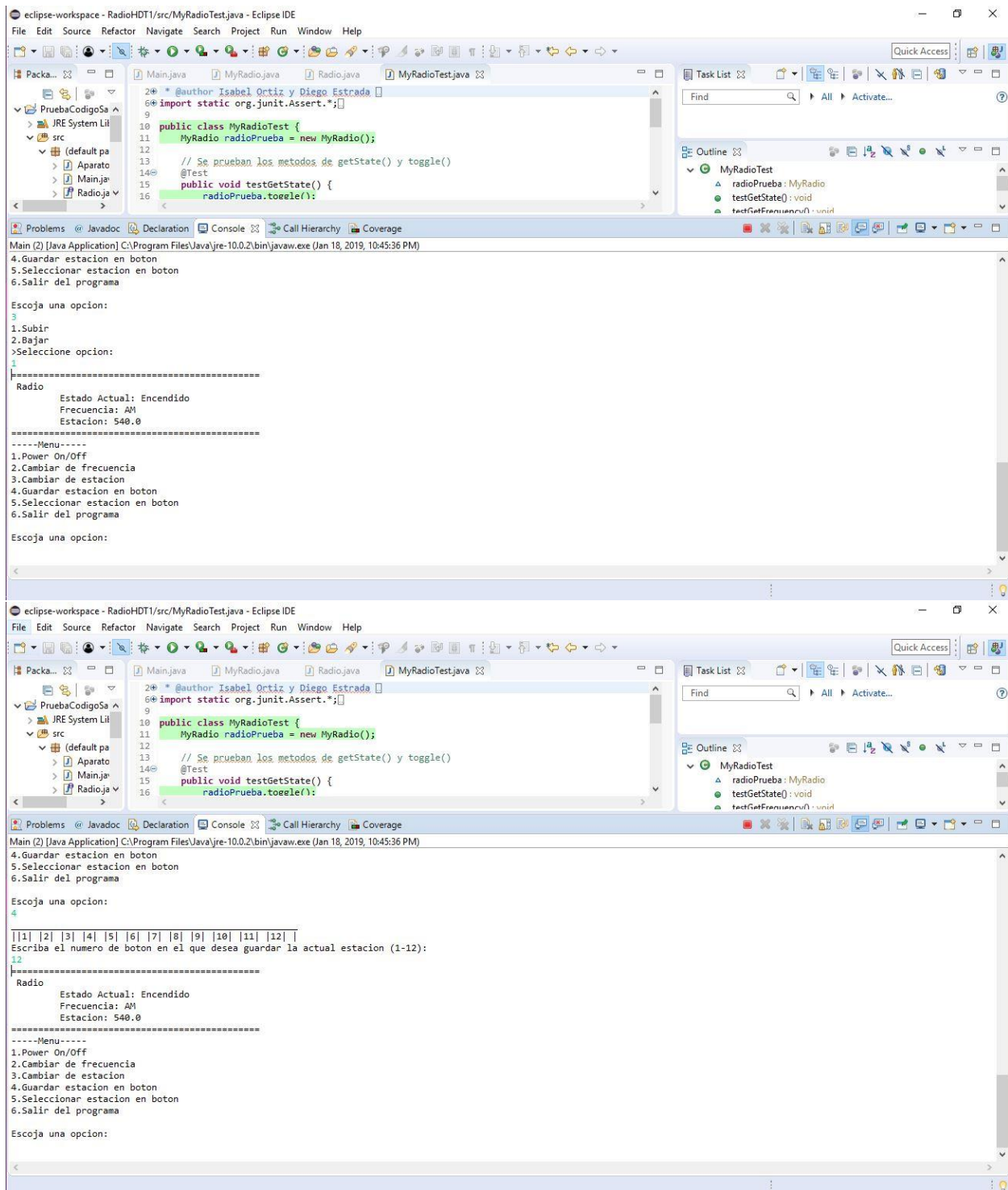
Opciones 1 y 2



Opciones 3 y 4



Opciones 3 y 4



Opciones 4 y 5

The screenshot shows the Eclipse IDE with the following components:

- Editor:** Displays the `MyRadioTest.java` file. The code includes a package declaration, imports, and a `MyRadioTest` class with a `testGetState()` method.
- Outline:** Shows the structure of the `MyRadioTest` class, including the `radioPrueba` instance and the `testGetState()` method.
- Console:** Displays the output of the application. It shows the program running and the user selecting option 4.

```
20 * @author Isabel Ortiz y Diego Estrada
60 import static org.junit.Assert.*;

9
10 public class MyRadioTest {
11     MyRadio radioPrueba = new MyRadio();
12
13     // Se prueban los metodos de getState() y toggle()
14     @Test
15     public void testGetState() {
16         radioPrueba.toggle();
17     }
18 }
```

Main (2) [Java Application] C:\Program Files\Java\jre-10.0.2\bin\javaw.exe (Jan 18, 2019, 10:45:36 PM)

4.Guardar estacion en boton
5.Seleccionar estacion en boton
6.Salir del programa

Escoja una opcion:
4

|1| 2| 3| 4| 5| 6| 7| 8| 9| 10| 11| 12| |
Escriba el numero de boton en el que desea guardar la actual estacion (1-12):
10

Radio

Estado Actual: Encendido
Frecuencia: AM
Estacion: 540.0

-----Menu-----
1.Power On/Off
2.Cambiar de frecuencia
3.Cambiar de estacion
4.Guardar estacion en boton
5.Seleccionar estacion en boton
6.Salir del programa

Escoja una opcion:

The screenshot shows the Eclipse IDE with the following components:

- Editor:** Displays the `MyRadioTest.java` file. The code includes a package declaration, imports, and a `MyRadioTest` class with a `testGetState()` method.
- Outline:** Shows the structure of the `MyRadioTest` class, including the `radioPrueba` instance and the `testGetState()` method.
- Console:** Displays the output of the application. It shows the program running and the user selecting option 5.

```
20 * @author Isabel Ortiz y Diego Estrada
60 import static org.junit.Assert.*;

9
10 public class MyRadioTest {
11     MyRadio radioPrueba = new MyRadio();
12
13     // Se prueban los metodos de getState() y toggle()
14     @Test
15     public void testGetState() {
16         radioPrueba.toggle();
17     }
18 }
```

Main (2) [Java Application] C:\Program Files\Java\jre-10.0.2\bin\javaw.exe (Jan 18, 2019, 10:45:36 PM)

4.Guardar estacion en boton
5.Seleccionar estacion en boton
6.Salir del programa

Escoja una opcion:
5

|1| 2| 3| 4| 5| 6| 7| 8| 9| 10| 11| 12| |
Escriba el numero de boton con la estacion que desea sintonizar (1-12):
10

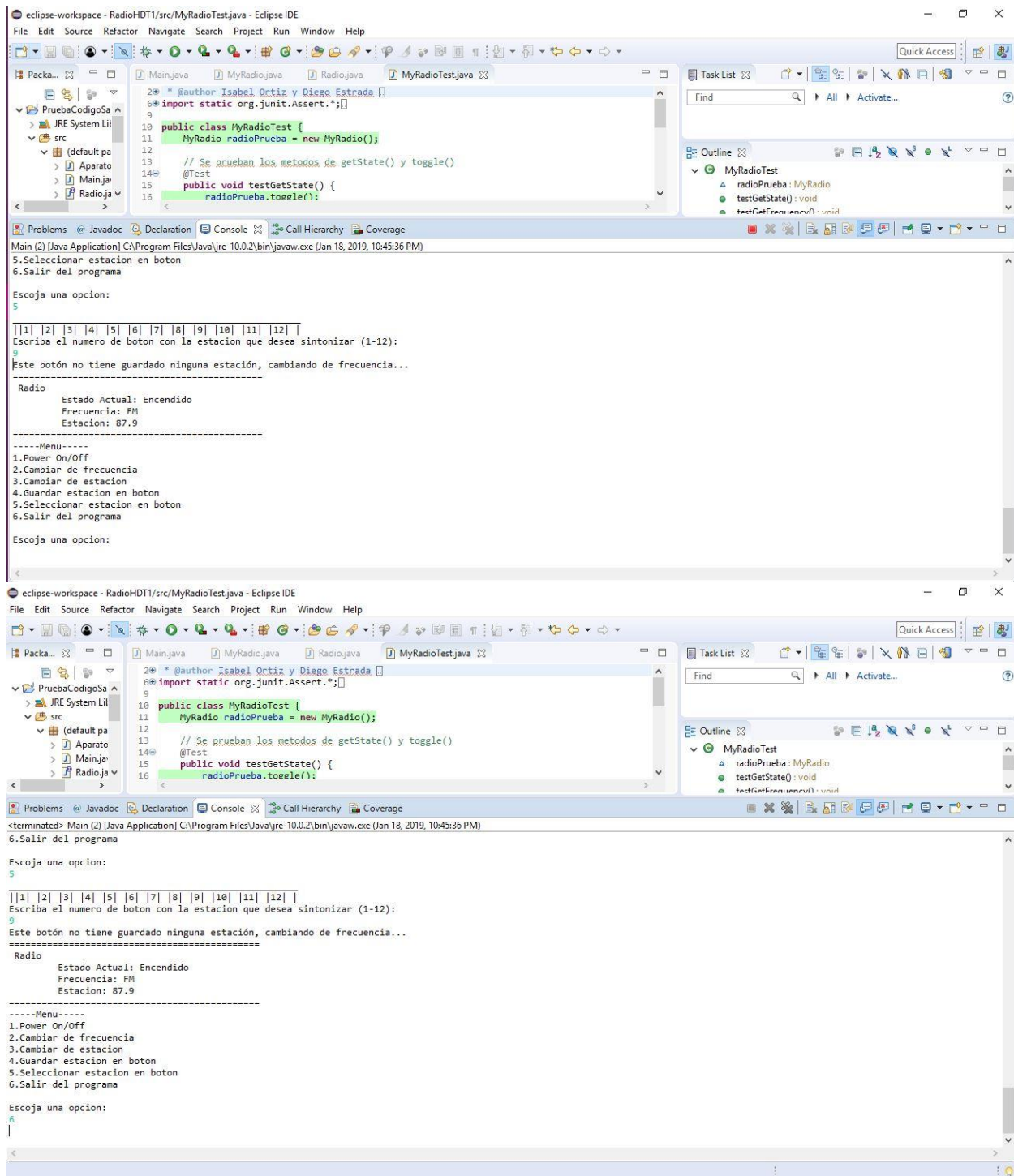
Radio

Estado Actual: Encendido
Frecuencia: AM
Estacion: 540.0

-----Menu-----
1.Power On/Off
2.Cambiar de frecuencia
3.Cambiar de estacion
4.Guardar estacion en boton
5.Seleccionar estacion en boton
6.Salir del programa

Escoja una opcion:

Opciones 5 y 6



Prueba con Código de Saul Contreras

The screenshot shows the Eclipse IDE with the file `AparatoRadial.java` open. The code defines a `Radio` interface and a `AparatoRadial` class that implements it. The class has private fields for `estado`, `frequency`, `station`, `maxFM`, `minFM`, `maxAM`, and `minAM`. The console output shows the program running and displaying the current state of the radio: `Estado Actual: Encendido`, `Frecuencia: AM`, and `Estacion: 530.0`. A menu is displayed with options: 1. Power On/Off, 2. Cambiar de frecuencia, 3. Cambiar de estacion, 4. Guardar estacion en boton, 5. Seleccionar estacion en boton, and 6. Salir del programa. The user is prompted to choose an option.

```
1  /*
2  AparatoRadial.java
3  java version 1.8.0_191
4  18/01/2018
5  Saul Contreras Godoy 18409
6  Radio Hoja de trabajo 1
7  Algoritmos y estructura de datos
8  Clase aparato radial que implementa la Interfaz Radio
9  */
10
11 /**
12  * Esta clase implementa la interfaz Radio.
13  */
14 public class AparatoRadial implements Radio {
15
16     /**
17      * El estado de la radio. ON/OFF
18      */
19     private boolean estado;
20
21     /**
22      * El tipo de frecuencia en el que se encuentra AM/FM
23      */
24     private boolean frequency;
25
26     /**
27      * La estacion en la que se encuentra
28      */
29     private double station;
30
31     /**
32      * Es la estación máxima que puede alcanzar el FM
33      */
34     private double maxFM;
35
36     /**
37      * Es la estación mínima que puede alcanzar el FM
38      */
39     private double minFM;
40
41     /**
42      * Es la estación máxima que puede alcanzar el AM
43      */
44     private double maxAM;
45
46     /**
47      * Es la estación mínima que puede alcanzar el AM
48      */
49 }
```

Console Output:

```
Main (2) [Java Application] C:\Program Files\Java\jre-10.0.2\bin\javaw.exe (Jan 18, 2019, 10:35:15 PM)
6.Salir del programa

Escoja una opcion:
2
-----
Radio
Estado Actual: Encendido
Frecuencia: AM
Estacion: 530.0
-----
-----Menu-----
1.Power On/Off
2.Cambiar de frecuencia
3.Cambiar de estacion
4.Guardar estacion en boton
5.Seleccionar estacion en boton
6.Salir del programa

Escoja una opcion:
```

The screenshot shows the Eclipse IDE with the file `AparatoRadial.java` open. The code is the same as in the previous screenshot. The console output shows the program running and displaying the current state of the radio: `Estado Actual: Encendido`, `Frecuencia: FM`, and `Estacion: 87.9`. A menu is displayed with options: 1. Power On/Off, 2. Cambiar de frecuencia, 3. Cambiar de estacion, 4. Guardar estacion en boton, 5. Seleccionar estacion en boton, and 6. Salir del programa. The user is prompted to choose an option. The user enters '5', and the program displays a list of 12 stations for selection. The user enters '12', and the program displays the current state of the radio: `Estado Actual: Encendido`, `Frecuencia: FM`, and `Estacion: 87.9`. A menu is displayed with options: 1. Power On/Off, 2. Cambiar de frecuencia, 3. Cambiar de estacion, 4. Guardar estacion en boton, 5. Seleccionar estacion en boton, and 6. Salir del programa. The user is prompted to choose an option.

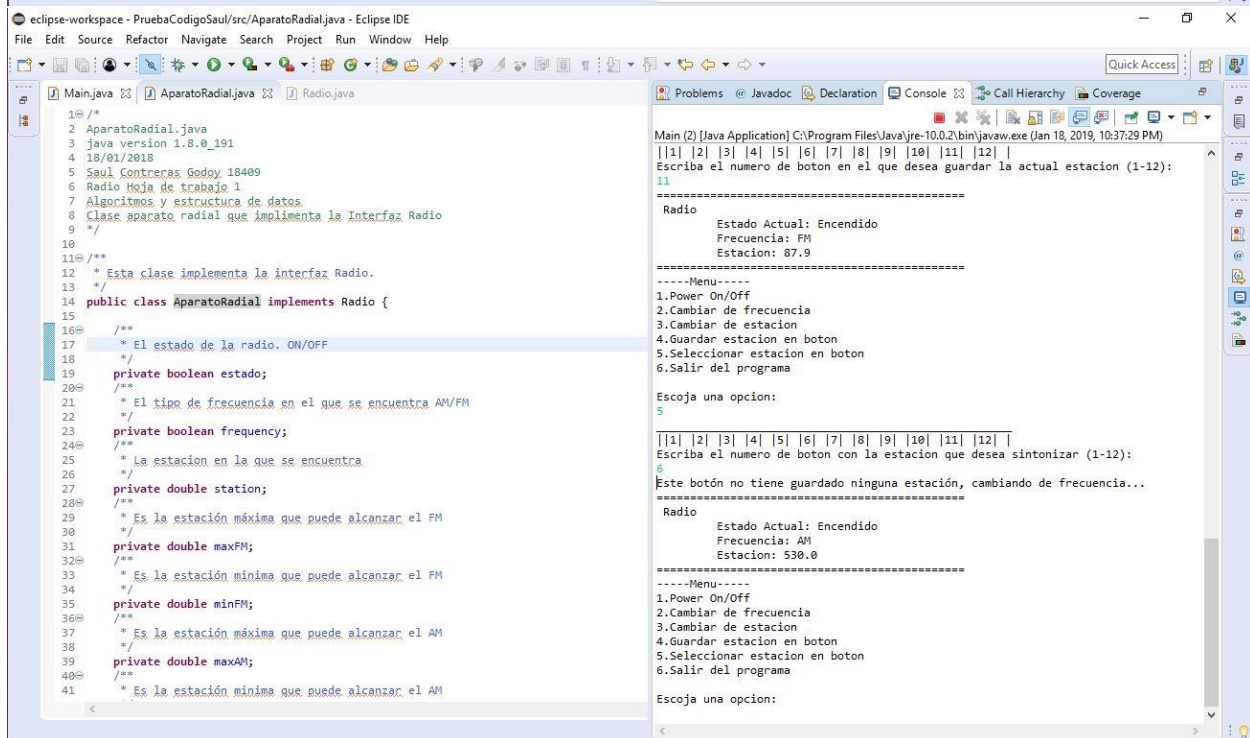
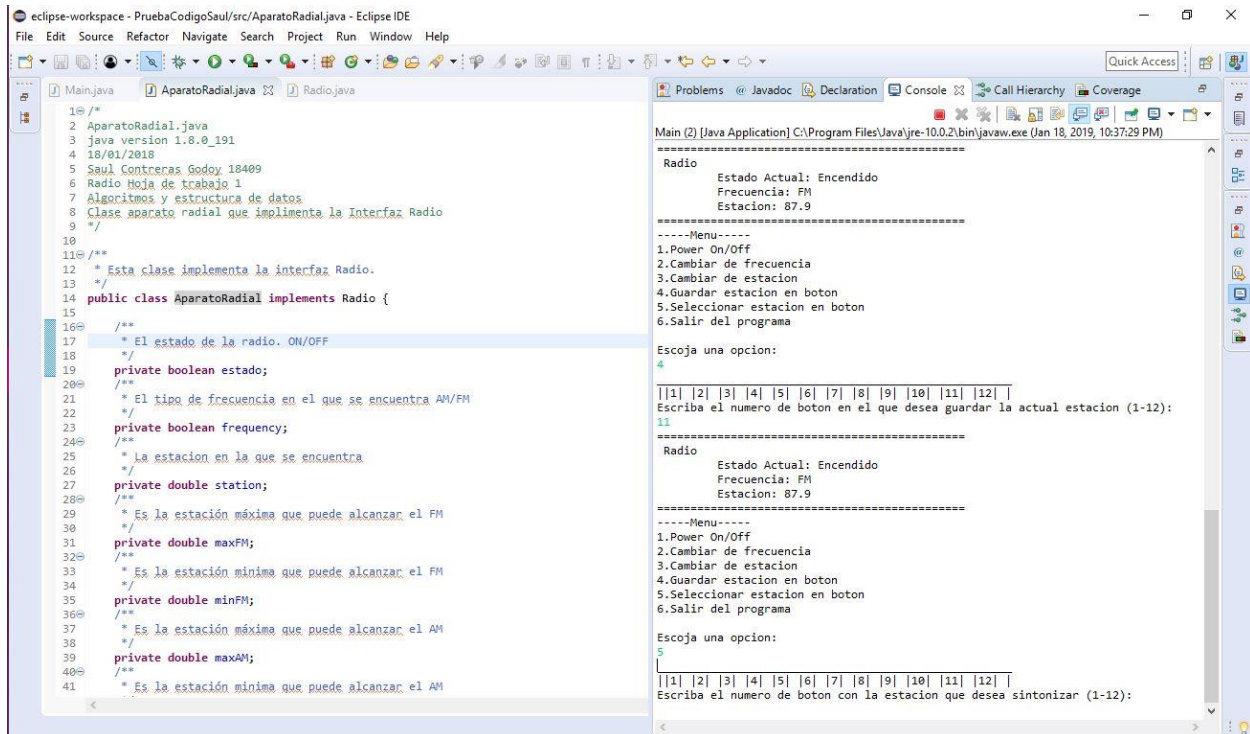
```
1  /*
2  AparatoRadial.java
3  java version 1.8.0_191
4  18/01/2018
5  Saul Contreras Godoy 18409
6  Radio Hoja de trabajo 1
7  Algoritmos y estructura de datos
8  Clase aparato radial que implementa la Interfaz Radio
9  */
10
11 /**
12  * Esta clase implementa la interfaz Radio.
13  */
14 public class AparatoRadial implements Radio {
15
16     /**
17      * El estado de la radio. ON/OFF
18      */
19     private boolean estado;
20
21     /**
22      * El tipo de frecuencia en el que se encuentra AM/FM
23      */
24     private boolean frequency;
25
26     /**
27      * La estacion en la que se encuentra
28      */
29     private double station;
30
31     /**
32      * Es la estación máxima que puede alcanzar el FM
33      */
34     private double maxFM;
35
36     /**
37      * Es la estación mínima que puede alcanzar el FM
38      */
39     private double minFM;
40
41     /**
42      * Es la estación máxima que puede alcanzar el AM
43      */
44     private double maxAM;
45
46     /**
47      * Es la estación mínima que puede alcanzar el AM
48      */
49 }
```

Console Output:

```
Main (2) [Java Application] C:\Program Files\Java\jre-10.0.2\bin\javaw.exe (Jan 18, 2019, 10:37:29 PM)
11 | 12 |
Escriba el numero de boton en el que desea guardar la actual estacion (1-12):
12
-----
Radio
Estado Actual: Encendido
Frecuencia: FM
Estacion: 87.9
-----
-----Menu-----
1.Power On/Off
2.Cambiar de frecuencia
3.Cambiar de estacion
4.Guardar estacion en boton
5.Seleccionar estacion en boton
6.Salir del programa

Escoja una opcion:
5
11 | 12 |
Escriba el numero de boton con la estacion que desea sintonizar (1-12):
12
-----
Radio
Estado Actual: Encendido
Frecuencia: FM
Estacion: 87.9
-----
-----Menu-----
1.Power On/Off
2.Cambiar de frecuencia
3.Cambiar de estacion
4.Guardar estacion en boton
5.Seleccionar estacion en boton
6.Salir del programa

Escoja una opcion:
```

eclipse-workspace - PruebaCodigoSaul/src/AparatoRadial.java - Eclipse IDE

File Edit Source Refactor Navigate Search Project Run Window Help

Main.java AparatoRadial.java Radio.java

```
10 /*
11 2 AparatoRadial.java
12 java version 1.8.0_191
13 18/01/2018
14 Saul Contreras Godoy 18409
15 Radio Moja de trabajo 1
16 Algoritmos y estructura de datos
17 Clase aparato radial que implementa la Interfaz Radio
18 */
19
20 /**
21  * Esta clase implementa la interfaz Radio.
22  */
23 public class AparatoRadial implements Radio {
24
25     /**
26      * El estado de la radio. ON/OFF
27      */
28     private boolean estado;
29
30     /**
31      * El tipo de frecuencia en el que se encuentra AM/FM
32      */
33     private boolean frequency;
34
35     /**
36      * La estación en la que se encuentra
37      */
38     private double station;
39
40     /**
41      * Es la estación máxima que puede alcanzar el FM
42      */
43     private double maxFM;
44
45     /**
46      * Es la estación mínima que puede alcanzar el FM
47      */
48     private double minFM;
49
50     /**
51      * Es la estación máxima que puede alcanzar el AM
52      */
53     private double minAM;
54
55     /**
56      * Es la estación mínima que puede alcanzar el AM
57      */
58     private double maxAM;
59 }
```

Problems Javadoc Declaration Console Call Hierarchy Coverage

Main (2) [Java Application] C:\Program Files\Java\jre-10.0.2\bin\javaw.exe (Jan 18, 2019, 10:39:58 PM)

6.Salir del programa

Escoja una opcion:

El radio esta apagado, enciendolo antes...

Radio

Estado Actual: Apagado
Frecuencia: ----
Estacion actual: ----

-----Menu-----

1.Power On/Off
2.Cambiar de frecuencia
3.Cambiar de estacion
4.Guardar estacion en boton
5.Seleccionar estacion en boton
6.Salir del programa

Escoja una opcion:

2

El radio esta apagado, enciendolo antes...

Radio

Estado Actual: Apagado
Frecuencia: ----
Estacion actual: ----

-----Menu-----

1.Power On/Off
2.Cambiar de frecuencia
3.Cambiar de estacion
4.Guardar estacion en boton
5.Seleccionar estacion en boton
6.Salir del programa

Escoja una opcion:

1

eclipse-workspace - PruebaCodigoSaul/src/AparatoRadial.java - Eclipse IDE

File Edit Source Refactor Navigate Search Project Run Window Help

Main.java AparatoRadial.java Radio.java

```
10 /*
11 2 AparatoRadial.java
12 java version 1.8.0_191
13 18/01/2018
14 Saul Contreras Godoy 18409
15 Radio Moja de trabajo 1
16 Algoritmos y estructura de datos
17 Clase aparato radial que implementa la Interfaz Radio
18 */
19
20 /**
21  * Esta clase implementa la interfaz Radio.
22  */
23 public class AparatoRadial implements Radio {
24
25     /**
26      * El estado de la radio. ON/OFF
27      */
28     private boolean estado;
29
30     /**
31      * El tipo de frecuencia en el que se encuentra AM/FM
32      */
33     private boolean frequency;
34
35     /**
36      * La estación en la que se encuentra
37      */
38     private double station;
39
40     /**
41      * Es la estación máxima que puede alcanzar el FM
42      */
43     private double maxFM;
44
45     /**
46      * Es la estación mínima que puede alcanzar el FM
47      */
48     private double minFM;
49
50     /**
51      * Es la estación máxima que puede alcanzar el AM
52      */
53     private double minAM;
54
55     /**
56      * Es la estación mínima que puede alcanzar el AM
57      */
58     private double maxAM;
59 }
```

Problems Javadoc Declaration Console Call Hierarchy Coverage

Main (2) [Java Application] C:\Program Files\Java\jre-10.0.2\bin\javaw.exe (Jan 18, 2019, 10:39:58 PM)

Encendiendo radio...

Radio

Estado Actual: Encendido
Frecuencia: FM
Estacion: 87.9

-----Menu-----

1.Power On/Off
2.Cambiar de frecuencia
3.Cambiar de estacion
4.Guardar estacion en boton
5.Seleccionar estacion en boton
6.Salir del programa

Escoja una opcion:

5

|1| 2| 3| 4| 5| 6| 7| 8| 9| 10| 11| 12| |

Escriba el numero de boton con la estacion que desea sintonizar (1-12):

12

Este boton no tiene guardado ninguna estación, cambiando de frecuencia...

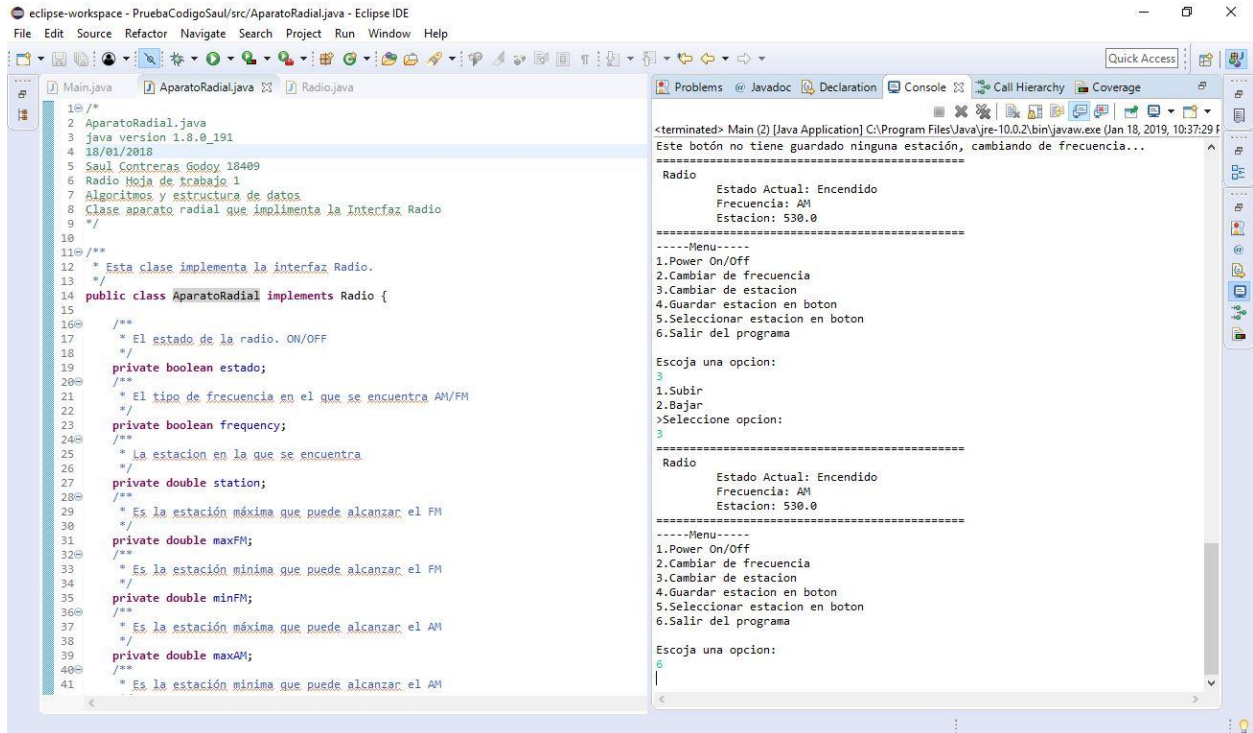
Radio

Estado Actual: Encendido
Frecuencia: AM
Estacion: 530.0

-----Menu-----

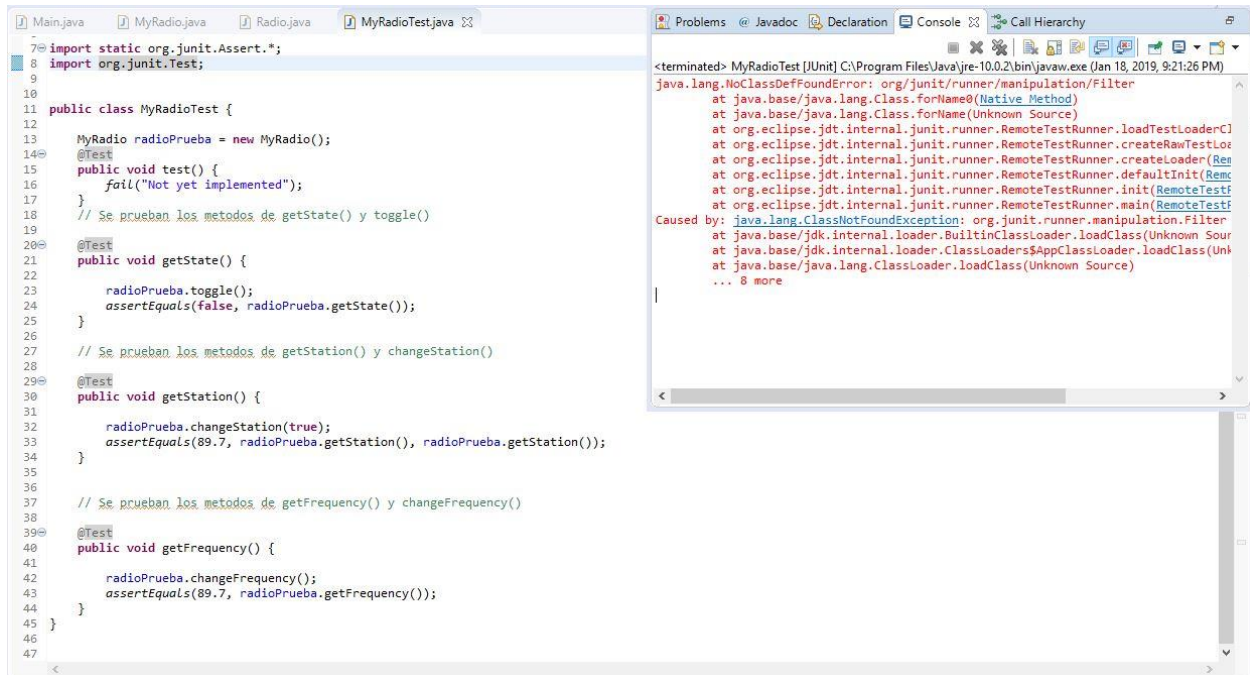
1.Power On/Off
2.Cambiar de frecuencia
3.Cambiar de estacion
4.Guardar estacion en boton
5.Seleccionar estacion en boton
6.Salir del programa

Escoja una opcion:



Pruebas con Junit – 3 pruebas

Prueba que fallo



The screenshot shows the Eclipse IDE with two panes. The left pane displays the source code of `MyRadioTest.java`, and the right pane shows the console output of a failed JUnit test.

Source Code (MyRadioTest.java):

```
7 import static org.junit.Assert.*;
8 import org.junit.Test;
9
10 public class MyRadioTest {
11     MyRadio radioPrueba = new MyRadio();
12
13     @Test
14     public void test() {
15         fail("Not yet implemented");
16     }
17
18     // Se prueban los metodos de getState() y toggle()
19
20     @Test
21     public void getState() {
22         radioPrueba.toggle();
23         assertEquals(false, radioPrueba.getState());
24     }
25
26     // Se prueban los metodos de getStation() y changeStation()
27
28
29     @Test
30     public void getStation() {
31         radioPrueba.changeStation(true);
32         assertEquals(89.7, radioPrueba.getStation(), radioPrueba.getStation());
33     }
34
35     // Se prueban los metodos de getFrequency() y changeFrequency()
36
37
38     @Test
39     public void getFrequency() {
40         radioPrueba.changeFrequency();
41         assertEquals(89.7, radioPrueba.getFrequency());
42     }
43
44 }
45
46
47
```

Console Output:

```
<terminated> MyRadioTest [JUnit] C:\Program Files\Java\jre-10.0.2\bin\javaw.exe (Jan 18, 2019, 9:21:26 PM)
java.lang.NoClassDefFoundError: org/junit/runner/manipulation/Filter
    at java.base/java.lang.Class.forName0(Native Method)
    at java.base/java.lang.Class.forName(Unknown Source)
    at org.eclipse.jdt.internal.junit.runner.RemoteTestRunner.loadTestLoaderClass()
    at org.eclipse.jdt.internal.junit.runner.RemoteTestRunner.createRawTestLoader()
    at org.eclipse.jdt.internal.junit.runner.RemoteTestRunner.createLoader(RemoteTestRunner.java:100)
    at org.eclipse.jdt.internal.junit.runner.RemoteTestRunner.defaultInit(RemoteTestRunner.java:100)
    at org.eclipse.jdt.internal.junit.runner.RemoteTestRunner.init(RemoteTestRunner.java:100)
    at org.eclipse.jdt.internal.junit.runner.RemoteTestRunner.main(RemoteTestRunner.java:100)
Caused by: java.lang.ClassNotFoundException: org.junit.runner.manipulation.Filter
    at java.base/jdk.internal.loader.BuiltinClassLoader.loadClass(Unknown Source)
    at java.base/jdk.internal.loader.ClassLoaders$AppClassLoader.loadClass(Unknown Source)
    at java.base/java.lang.ClassLoader.loadClass(Unknown Source)
    ... 8 more
```