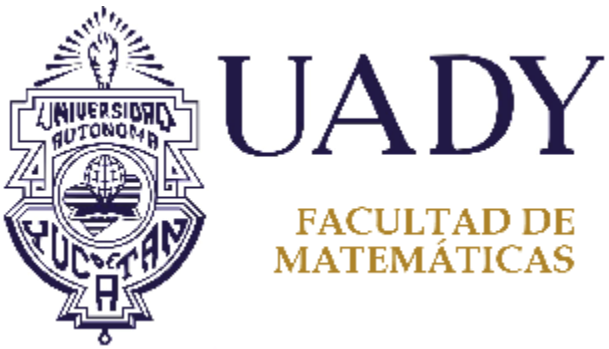
UNIVERSIDAD AUTÓNOMA DE YUCATÁN

FACULTAD DE MATEMÁTICAS

INGENIERÍA DE SOFTWARE



TEST DRIVEN DEVELOPMENT

DANIEL FERNANDO BAAS COLONIA

CARLOS BRAULIO BETANCOURT ESTRADA

RODRIGO CASTILLA LÓPEZ

RAFAEL RODRÍGUEZ GUZMÁN

ASEGURAMIENTO DE LA CALIDAD DE SOFTWARE

RENÉ HUMBERTO MORENO ACEVEDO

MÉRIDA, YUCATÁN, FEBRERO 2018

Práctica desarrollo conducido por pruebas

Utiliza TDD para automatizar el siguiente requerimiento del cálculo de pago de un empleado:

- La duración máxima de los turnos en la jornada será: ocho horas el turno diurno, siete horas el turno nocturno y siete horas y media el turno mixto.
- Las primeras nueve horas extras de trabajo son pagadas dobles y las excedentes triples. (Artículo 68 de la LFT).

De la descripción anterior se pueden obtener las siguientes características:

- Existen tres turnos: diurno, nocturno y mixto.
- El turno diurno tiene una duración de 8 horas mínimas diarias.
- El turno nocturno tiene una duración de 7 horas mínimas diarias.
- El turno mixto tiene una duración de 7.5 horas mínimas diarias.
- Cada empleado tiene un único turno.
- El pago de las horas de trabajo se realiza en el número de horas realizadas en 6 días de la semana (una semana laboral).
- Las primeras 9 horas extras son pagadas dobles y las excedentes triples.
- Se debe idear una forma de realizar el cálculo del total de horas a pagar al empleado.

La primera prueba que se realizó fue el cálculo de horas a pagar dadas las horas trabajadas y un turno. En esta primera prueba se tomó la decisión de generar una clase llamada "*CalculoHorasTrabajadas.java*" para realizar el cálculo. Dicha clase luce así:

```
package main;

public class CalculoHorasTrabajadas {

    public float calcularHorasAPagar(String turno, float horas) {
        return (float) 48.0;
    }

}
```

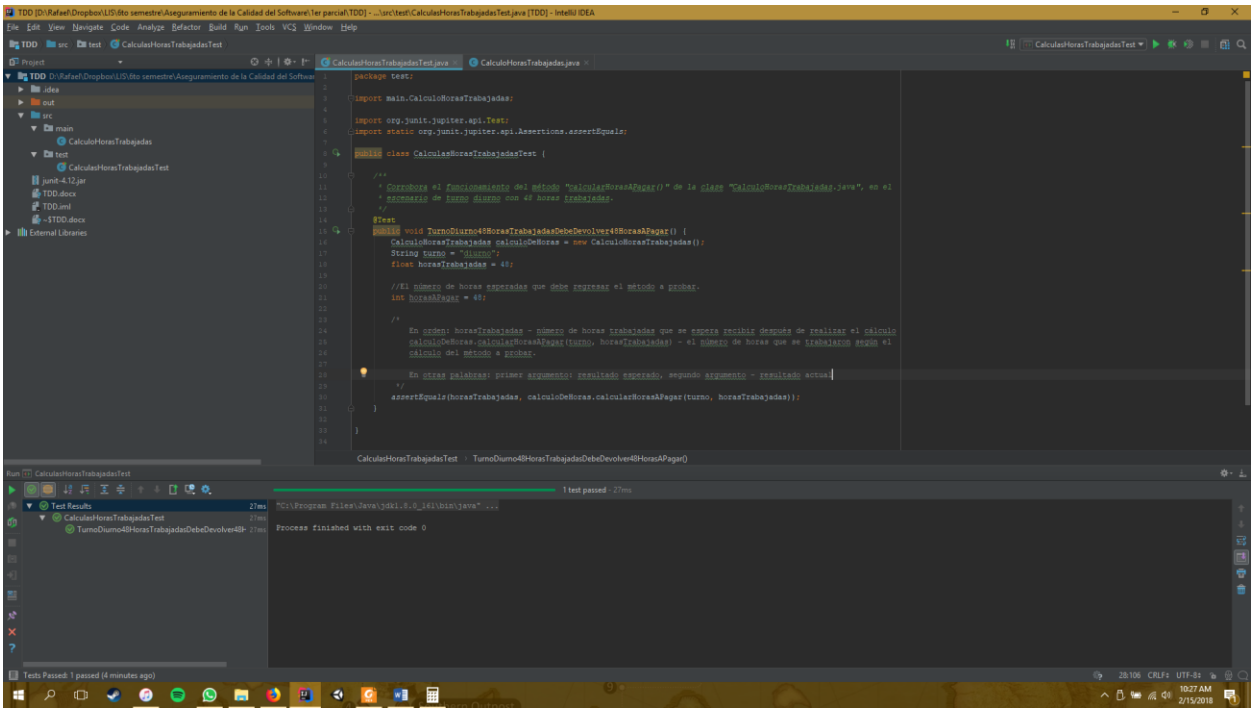
Como aún no realiza una implementación, el método "*calcularHorasAPagar()*" sólo regresa un valor cualquiera. La primera prueba que se desea realizar es para verificar que la prueba pasa y es correcta. Se tienen los siguientes valores de entrada:

- Horas trabajadas: 48
- Turno: diurno
- Horas esperadas de trabajo: 48

La primera prueba luce de esta manera:

```java
package test;

import main.CalculoHorasTrabajadas;

import org.junit.jupiter.api.Test;
import static org.junit.jupiter.api.Assertions.assertEquals;

public class CalculasHorasTrabajadasTest {

    /**
     * Corrobora el funcionamiento del método "calcularHorasAPagar()" de la clase
"CalculoHorasTrabajadas.java", en el
     * escenario de turno diurno con 48 horas trabajadas.
     */
    @Test
    public void TurnoDiurno48HorasTrabajadasDebeDevolver48HorasAPagar() {
        CalculoHorasTrabajadas calculoDeHoras = new CalculoHorasTrabajadas();
        String turno = "diurno";
        float horasTrabajadas = 48;

        //El número de horas esperadas que debe regresar el método a probar.
        int horasAPagar = 48;

        assertEquals(horasTrabajadas, calculoDeHoras.calcularHorasAPagar(turno,
horasTrabajadas));
    }

}
```

Después de ejecutar la prueba se tienen los siguientes resultados de ejecución:



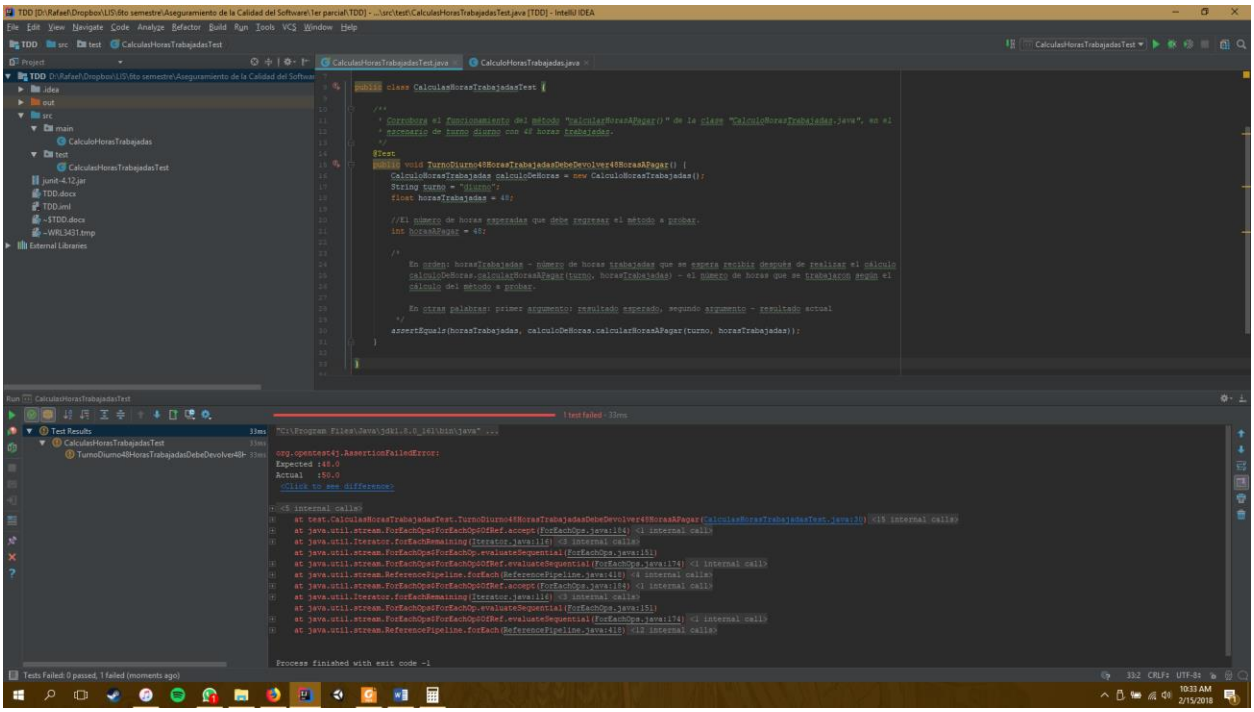
Podemos ver que el resultado esperado coincide con el actual. Ahora hay que verificar que la prueba falla. Los valores de entrada son los siguientes:

- Horas trabajadas 48
- Turno: diurno
- Horas esperadas de trabajo: 48

En este caso el método para calcular las horas de trabajo luce de la siguiente manera:

```
public float calcularHorasAPagar(String turno, float horas) {
    return (float) 50.0;
}
```

Dado que el número de horas que devuelve el método es diferente al número de horas que se esperan, la prueba debe fallar. Podemos verificarlo en el resultado de la ejecución de la prueba:



Por lo tanto, podemos verificar que la prueba funciona de la manera correcta.

Para aplicar una solución es necesario establecer los escenarios en los que el problema puede caer. En este caso se decidió hacer una primera división del problema por turnos:

- Turno diurno: 48 horas mínimas en la semana laboral.
- Turno nocturno: 42 horas mínimas en la semana laboral.
- Turno mixto: 45 horas mínimas en la semana laboral.

Para cada turno existe ciertos intervalos de horas trabajadas bajo los cuales se deben realizar los pagos. Esto es de acuerdo con el número de horas mínimas por turno, y afecta a partir de cuántas horas trabajadas se deberán pagar como dobles y triples. Los turnos tienen los siguientes intervalos:

Turno diurno:

- [0] horas
- (0, 48] horas
- [48] horas
- (48, 57) horas – horas dobles
- [57] horas – horas dobles
- (57, ∞) horas- horas triples

Turno nocturno:

- [0] horas
- (0, 42) horas
- [42] horas
- (42, 51) horas – horas dobles
- [51] horas – horas dobles
- (51, ∞) horas- horas triples

Turno mixto:

- [0] horas
- (0, 45) horas
- [45] horas
- (45, 54) horas – horas dobles
- [54] horas – horas dobles
- (54, ∞) horas- horas triples

Dada esta información es necesario implementar nuevas pruebas para los casos mostrados anteriormente. También se tomó la decisión de crear nuevos métodos en la clase "*CalculoHorasTrabajadas.java*" para realizar el cálculo de horas a pagar por turnos. Además, para mantener una organización mejor, las clases de prueba se decidieron separar en tres clases distintas, una para probar cada turno. Se tienen así las clases de prueba: "*CalcularHorasTrabajadasTurnoDiurno.java*", "*CalcularHorasTrabajadasTurnoNocturno.java*" y "*CalcularHorasTrabajadasTurnoMixto.java*".

Así, tenemos las siguientes pruebas para la clase "*CalcularHorasTrabajadasTurnoDiurno.java*":

```
package test;

import main.CalculoHorasTrabajadas;

import org.junit.jupiter.api.Test;
import static org.junit.jupiter.api.Assertions.assertEquals;
```

```java
public class CalculasHorasTrabajadasTurnoDiurnoTest {

    @Test
    public void TurnoDiurnoCeroHorasTrabajadasDebeDevolverCeroHorasAPagar() {
        CalculoHorasTrabajadas calculoDeHoras = new CalculoHorasTrabajadas();
        float horasTrabajadas = 0;

        //Primer argumento - horas esperadas, segundo argumento - horas actuales
trabajadas calculadas
        //Se esperan 0 horas a pagar y se trabajaron 0 horas
        assertEquals(horasTrabajadas,
calculoDeHoras.calcularHorasAPagarTurnoDiurno(horasTrabajadas));
    }

    @Test
    public void TurnoDiurno1HoraTrabajadaDebeDevolver1HoraAPagar() {
        CalculoHorasTrabajadas calculoDeHoras = new CalculoHorasTrabajadas();
        float horasTrabajadas = 1;

        //Se esperan 1 hora a pagar y se trabajo 1 hora
        assertEquals(horasTrabajadas,
calculoDeHoras.calcularHorasAPagarTurnoDiurno(horasTrabajadas));
    }

    @Test
    public void TurnoDiurno47HorasTrabajadasDebeDevolver47HorasAPagar() {
        CalculoHorasTrabajadas calculoDeHoras = new CalculoHorasTrabajadas();
        float horasTrabajadas = 47;

        //Se esperan 47 horas a pagar y se trabajaron 47 horas
        assertEquals(horasTrabajadas,
calculoDeHoras.calcularHorasAPagarTurnoDiurno(horasTrabajadas));
    }

    @Test
    public void TurnoDiurno48HorasTrabajadasDebeDevolver48HorasAPagar() {
        CalculoHorasTrabajadas calculoDeHoras = new CalculoHorasTrabajadas();
        float horasTrabajadas = 48;

        //Se esperan 48 horas a pagar y se trabajaron 48 horas
        assertEquals(horasTrabajadas,
calculoDeHoras.calcularHorasAPagarTurnoDiurno(horasTrabajadas));
    }

    @Test
    public void TurnoDiurno49HorasTrabajadasDebeDevolver50HorasAPagar() {
        CalculoHorasTrabajadas calculoDeHoras = new CalculoHorasTrabajadas();
        float horasTrabajadasEsperadas = 50; //50 horas a pagar
        float horasTrabajadasActuales = 49; //1 hora doble

        //Se esperan 50 horas a pagar y se trabajaron 48 horas normales y 1 hora doble
        assertEquals(horasTrabajadasEsperadas,
calculoDeHoras.calcularHorasAPagarTurnoDiurno(horasTrabajadasActuales));
    }

    @Test
    public void TurnoDiurno56HorasTrabajadasDebeDevolver64HorasAPagar() {
        CalculoHorasTrabajadas calculoDeHoras = new CalculoHorasTrabajadas();
        float horasTrabajadasEsperadas = 64; //64 horas a pagar
        float horasTrabajadasActuales = 56; //8 horas dobles

        //Se esperan 64 horas a pagar y se trabajaron 48 horas normales y 8 horas
```

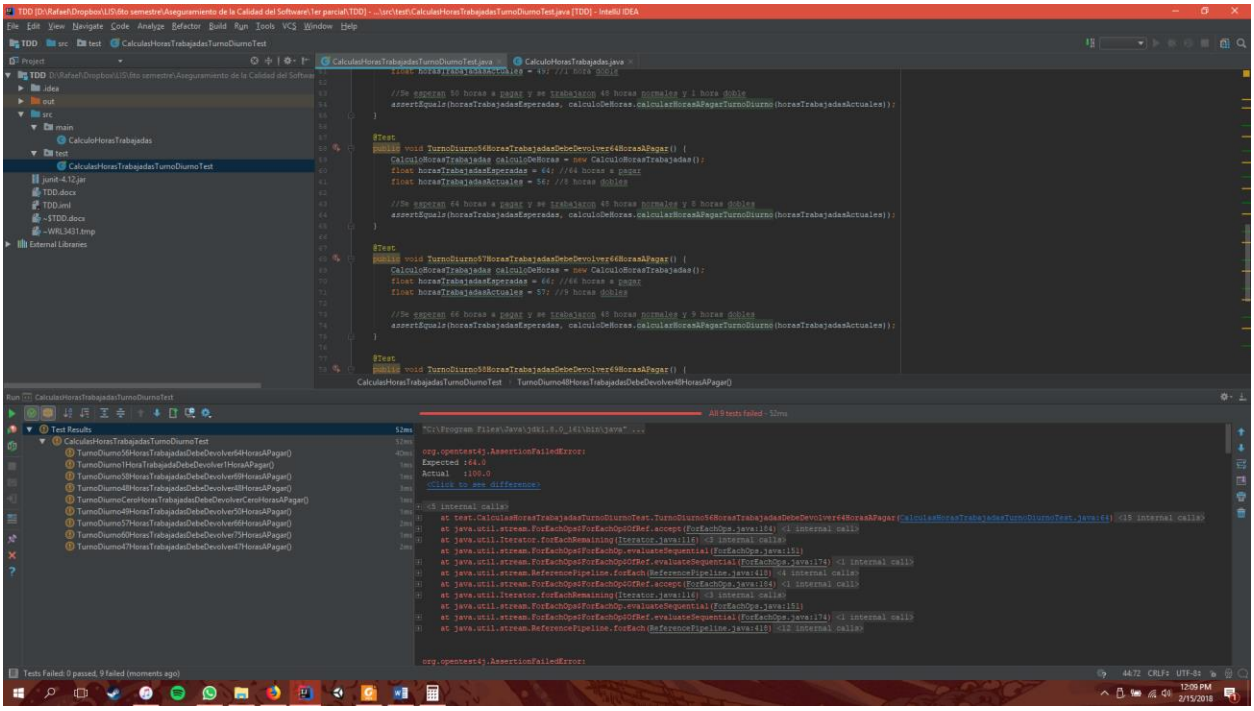
```
dobles
        assertEquals(horasTrabajadasEsperadas,
calculoDeHoras.calcularHorasAPagarTurnoDiurno(horasTrabajadasActuales));
    }

    @Test
    public void TurnoDiurno57HorasTrabajadasDebeDevolver66HorasAPagar() {
        CalculoHorasTrabajadas calculoDeHoras = new CalculoHorasTrabajadas();
        float horasTrabajadasEsperadas = 66; //66 horas a pagar
        float horasTrabajadasActuales = 57; //9 horas dobles

        //Se esperan 66 horas a pagar y se trabajaron 48 horas normales y 9 horas
dobles
        assertEquals(horasTrabajadasEsperadas,
calculoDeHoras.calcularHorasAPagarTurnoDiurno(horasTrabajadasActuales));
    }

    @Test
    public void TurnoDiurno58HorasTrabajadasDebeDevolver69HorasAPagar() {
        CalculoHorasTrabajadas calculoDeHoras = new CalculoHorasTrabajadas();
        float horasTrabajadasEsperadas = 69; //69 horas a pagar
        float horasTrabajadasActuales = 58; //9 horas dobles, 1 hora triple

        //Se esperan 69 horas a pagar y se trabajaron 48 horas normales, 9 horas
dobles y 1 hora triple
        assertEquals(horasTrabajadasEsperadas,
calculoDeHoras.calcularHorasAPagarTurnoDiurno(horasTrabajadasActuales));
    }

    @Test
    public void TurnoDiurno60HorasTrabajadasDebeDevolver75HorasAPagar() {
        CalculoHorasTrabajadas calculoDeHoras = new CalculoHorasTrabajadas();
        float horasTrabajadasEsperadas = 75; //75 horas a pagar
        float horasTrabajadasActuales = 60; //9 horas dobles, 3 horas triples

        //Se esperan 75 horas a pagar y se trabajaron 48 horas normales, 9 horas
dobles y 3 horas triples
        assertEquals(horasTrabajadasEsperadas,
calculoDeHoras.calcularHorasAPagarTurnoDiurno(horasTrabajadasActuales));
    }

}
```

La clase "CalculoHorasTrabajadas.java" debe ser modificada para realizar las evaluaciones de acuerdo con el nuevo paso de argumentos. Así que es necesario realizar pruebas para verificar que el código de pruebas funciona correctamente (falla y pasa las pruebas con valores definidos fijos).

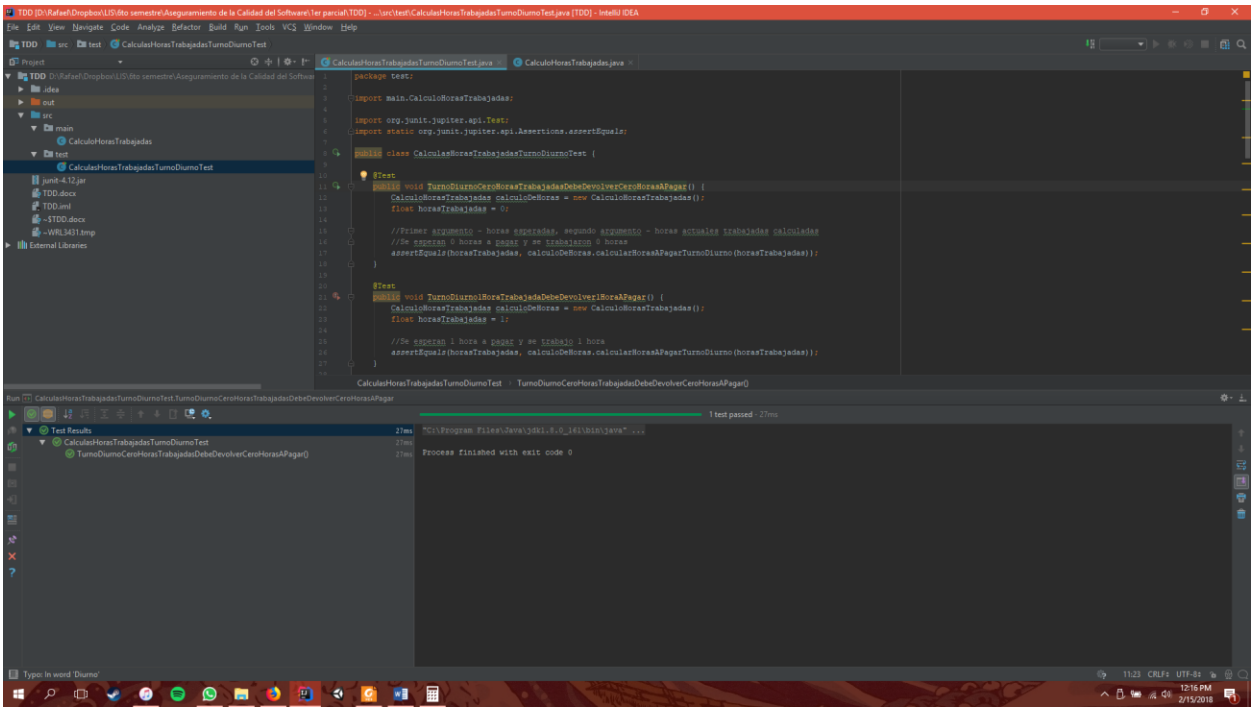
```
package main;

public class CalculoHorasTrabajadas {

    /**
     * Método para realizar el cálculo de horas a pagar. Para la primera prueba sólo
se desea verificar que la prueba
     * aprueba o falla, comprobando así que la prueba es correcta.
     * @param turno El turno del empleado bajo el cual se hace el cálculo.
     * @param horas Número de otras trabajadas en la semana laboral.
     * @return Número cualquiera. Sólo se usará para verificar el funcionamiento de la
prueba.
```

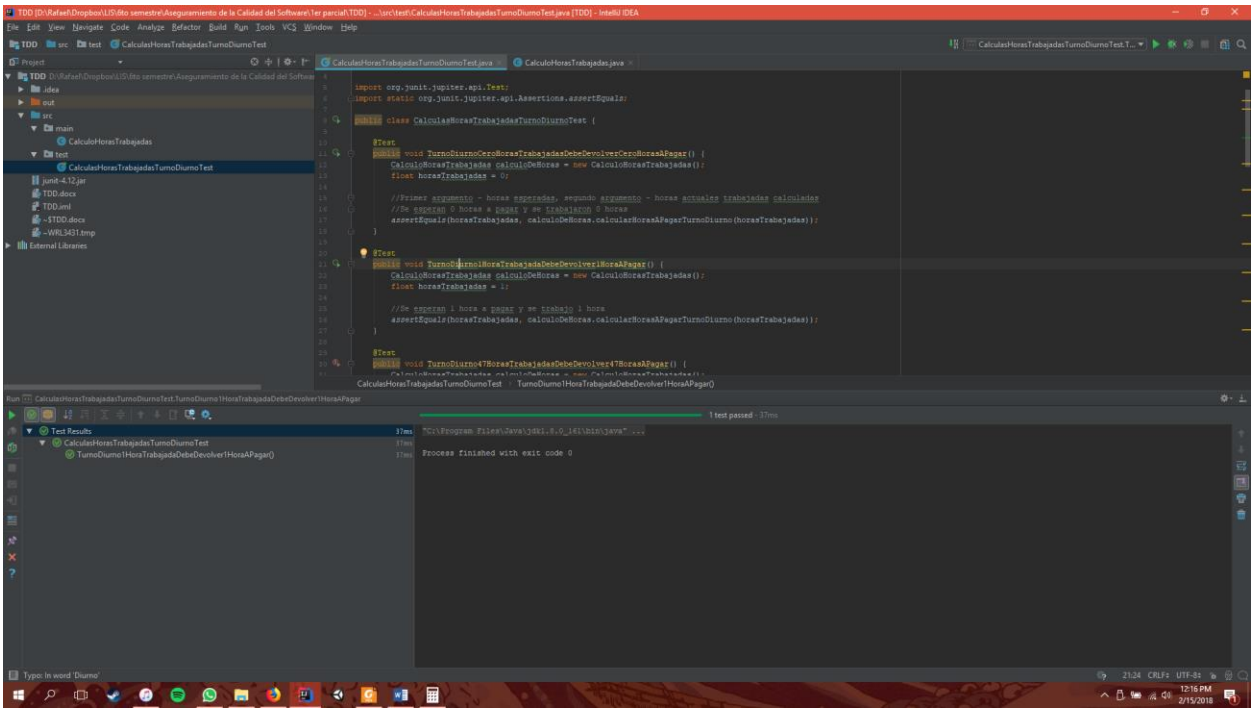
```
     */
    public float calcularHorasAPagar(String turno, float horas) {
        return (float) 50.0;
    }

    public float calcularHorasAPagarTurnoDiurno(float horas) {
        float horasAPagar = 100;

        return horasAPagar;
    }

    public float calcularHorasAPagarTurnoNocturno(float horas) {
        float horasAPagar = 100;

        return horasAPagar;
    }

    public float calcularHorasAPagarTurnoMixto(float horas) {
        float horasAPagar = 100;

        return horasAPagar;
    }

}
```

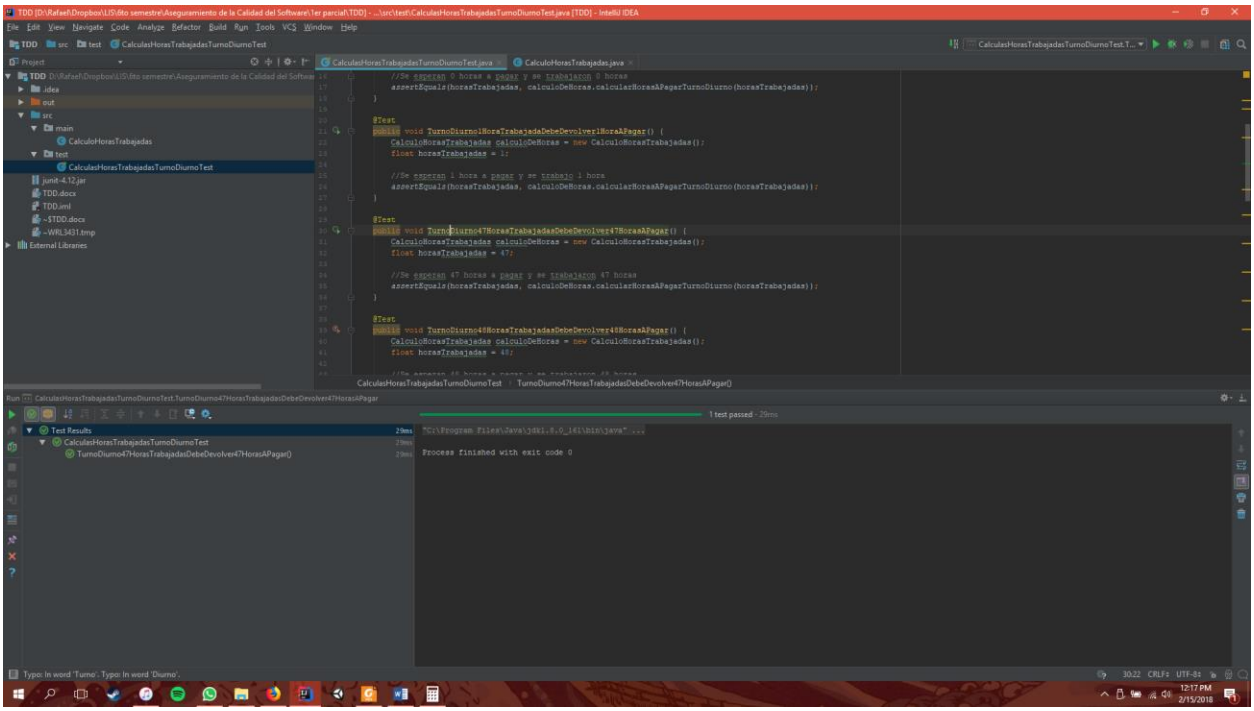
Se ejecutaron las pruebas para verificar que el código de pruebas falle.

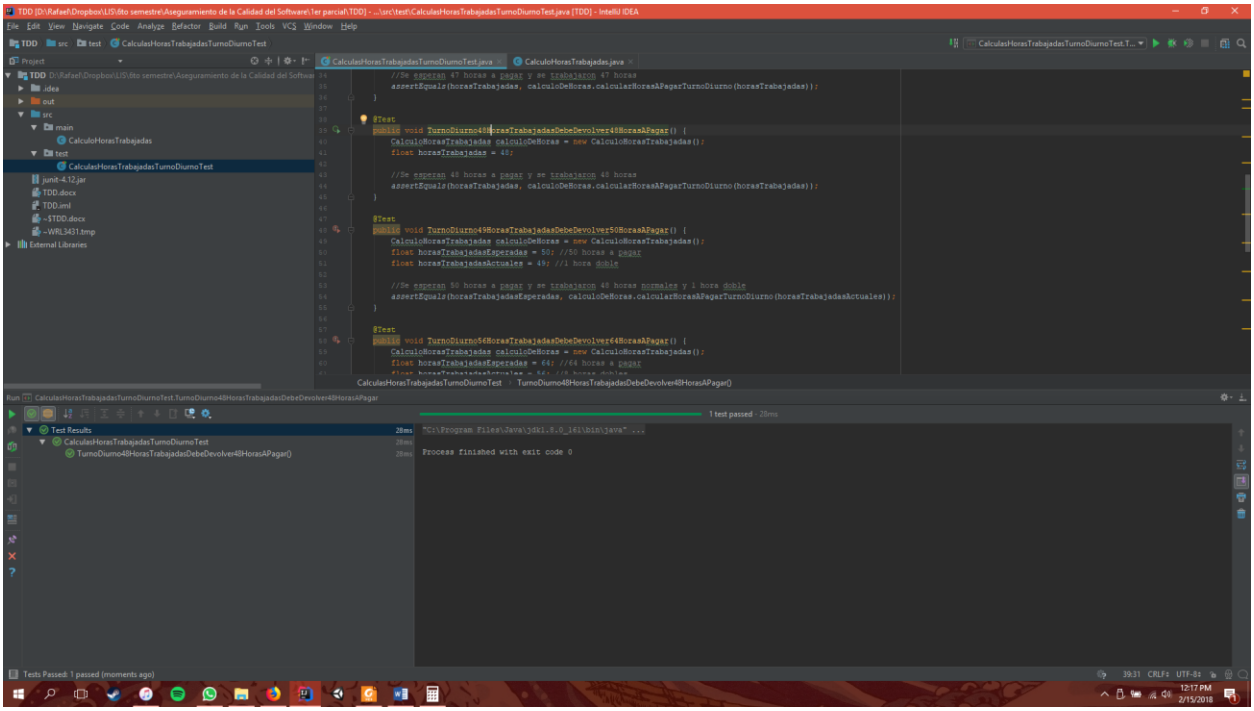


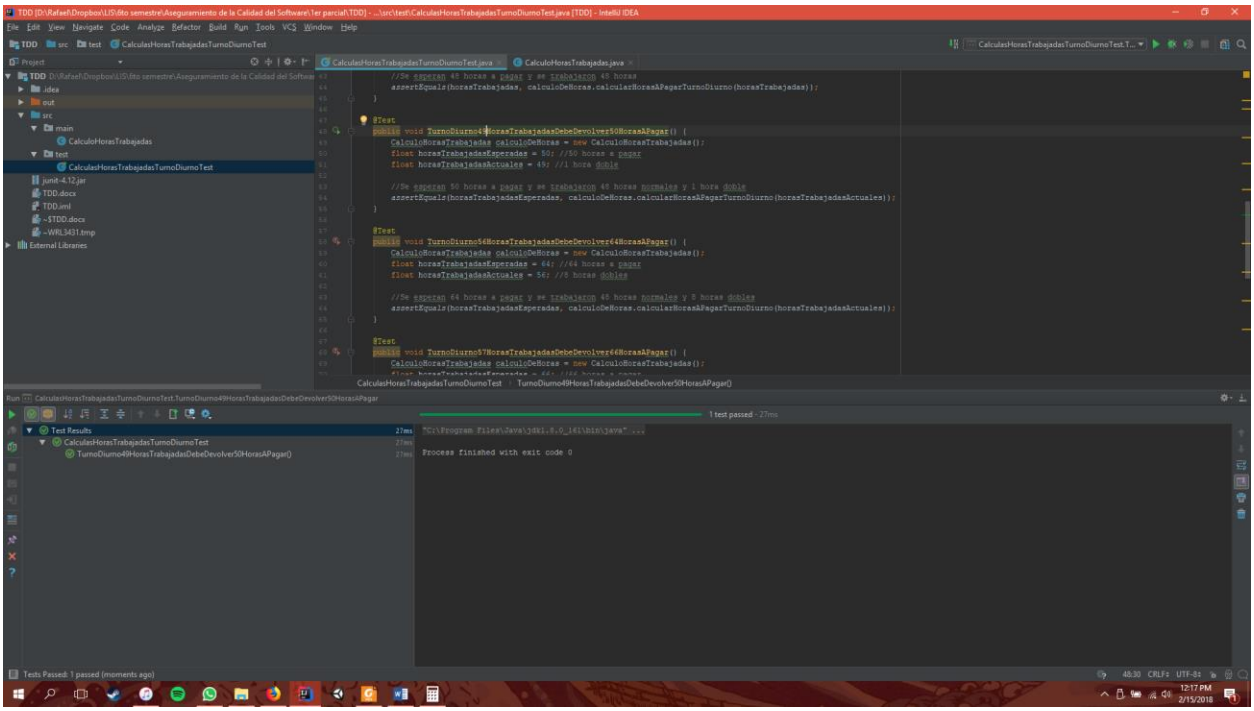
Después se ejecutaron las pruebas retornando los valores esperados dentro del método "*calcularHorasAPagarTurnoDiurno()*". El valor de la variable "*horasAPagar*" se fue cambiando para cada prueba con el objetivo de que coincida con el valor esperado. Así dicha variable tomó los valores de 0, 1, 47, 48, 50, 64, 66, 69 y 75, coincidiendo con los valores esperados de cada prueba respectivo al orden de estas.

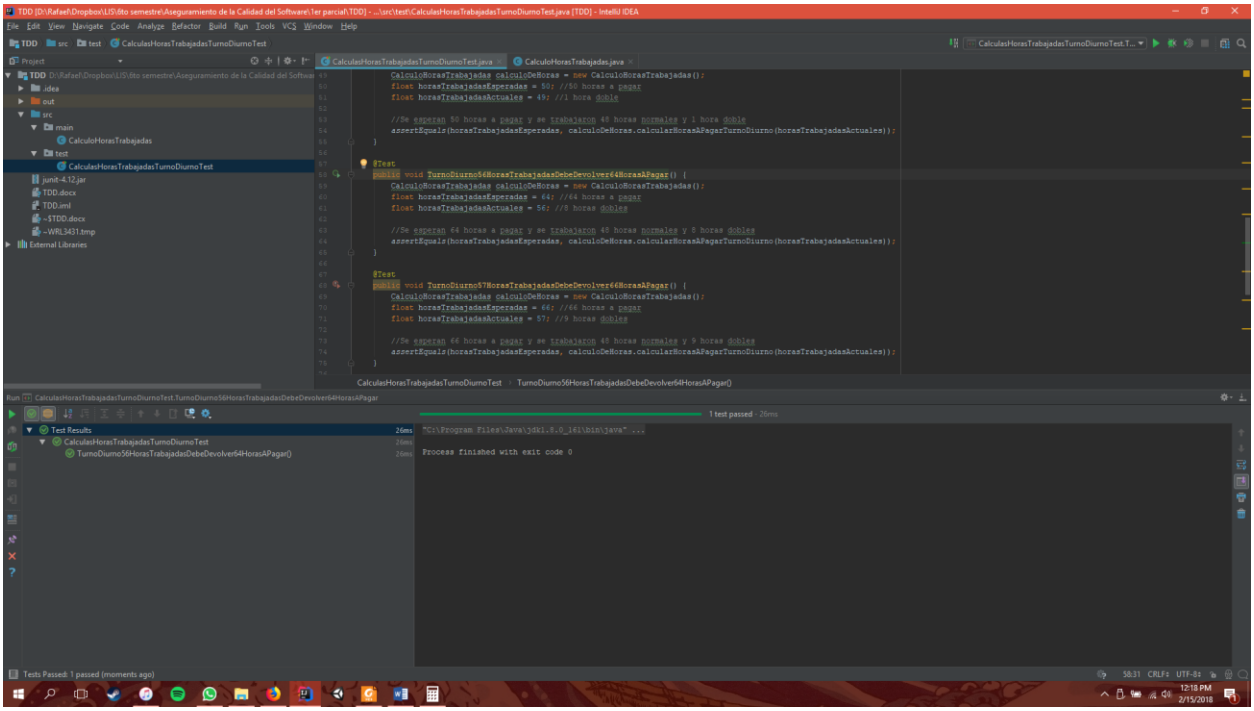

TDD [D:\Rafael\Dropbox\LIS\6to semestre\Aseguramiento de la Calidad del Software\1er parcial\TDD] - ...\src\test\CalculasHorasTrabajadasTurnoDiurnoTest.java [TDD] - IntelliJ IDEA
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
CalculasHorasTrabajadasTurnoDiurnoTest.java
CalculoHorasTrabajadas.java
package test;
import main.CalculoHorasTrabajadas;
import org.junit.jupiter.api.Test;
import static org.junit.jupiter.api.Assertions.assertEquals;
public class CalculasHorasTrabajadasTurnoDiurnoTest {
@Test
public void TurnoDiurnoCeroHorasTrabajadasDebeDevolverCeroHorasAPagar() {
CalculoHorasTrabajadas calculoDeHoras = new CalculoHorasTrabajadas();
float horasTrabajadas = 0;
//Primer argumento - horas esperadas, segundo argumento - horas actuales trabajadas calculadas
//Se esperan 0 horas a pagar y se trabajaron 0 horas
assertEquals(horasTrabajadas, calculoDeHoras.calcularHorasAPagarTurnoDiurno(horasTrabajadas));
}
@Test
public void TurnoDiurnoHoraTrabajadaDebeDevolver1HoraAPagar() {
CalculoHorasTrabajadas calculoDeHoras = new CalculoHorasTrabajadas();
float horasTrabajadas = 1;
//Se esperan 1 hora a pagar y se trabajo 1 hora
assertEquals(horasTrabajadas, calculoDeHoras.calcularHorasAPagarTurnoDiurno(horasTrabajadas));
}
CalculasHorasTrabajadasTurnoDiurnoTest
TurnoDiurnoCeroHorasTrabajadasDebeDevolverCeroHorasAPagar()
1 test passed - 27ms
Test Results
CalculasHorasTrabajadasTurnoDiurnoTest
TurnoDiurnoCeroHorasTrabajadasDebeDevolverCeroHorasAPagar()
"C:\Program Files\Java\jdk1.8.0_161\bin\java" ...
Process finished with exit code 0

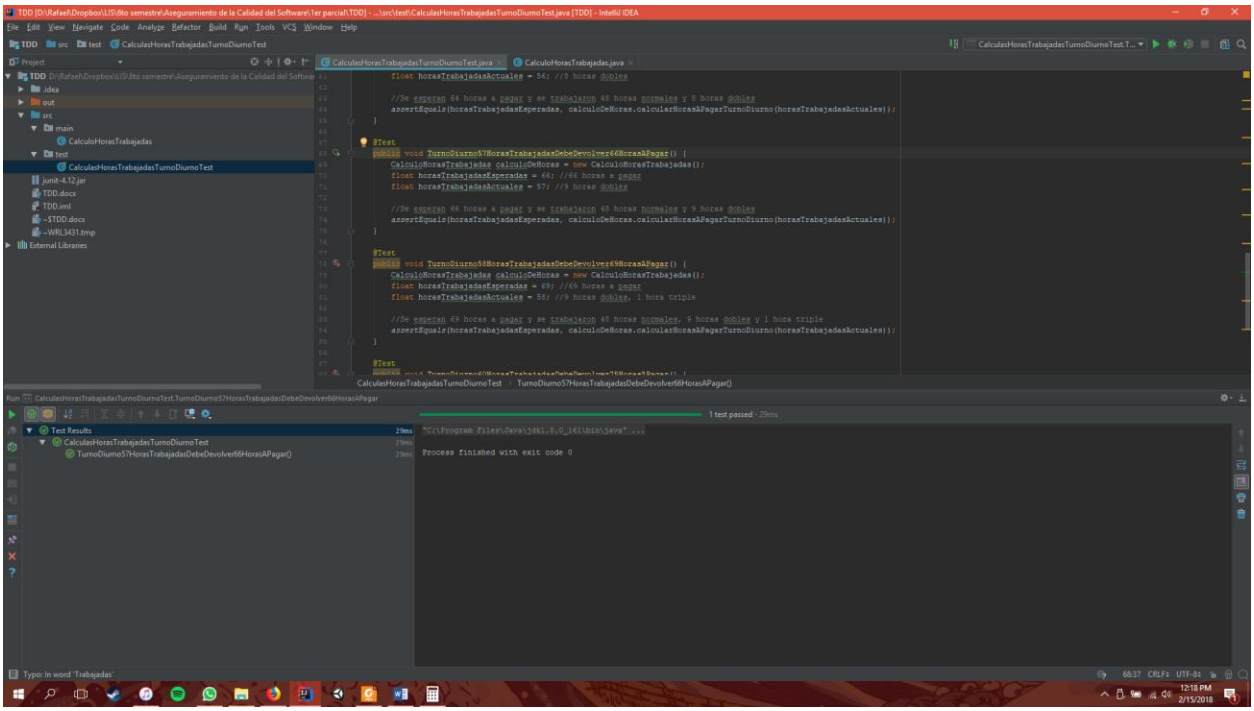

TDD [D:\Rafael\Dropbox\LIS\6to semestre\Aseguramiento de la Calidad del Software\1er parcial\TDD] - ...\src\test\CalculasHorasTrabajadasTurnoDiurnoTest.java [TDD] - IntelliJ IDEA
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
CalculasHorasTrabajadasTurnoDiurnoTest.java
CalculoHorasTrabajadas.java
import org.junit.jupiter.api.Test;
import static org.junit.jupiter.api.Assertions.assertEquals;
public class CalculasHorasTrabajadasTurnoDiurnoTest {
@Test
public void TurnoDiurnoCeroHorasTrabajadasDebeDevolverCeroHorasAPagar() {
CalculoHorasTrabajadas calculoDeHoras = new CalculoHorasTrabajadas();
float horasTrabajadas = 0;
//Primer argumento - horas esperadas, segundo argumento - horas actuales trabajadas calculadas
//Se esperan 0 horas a pagar y se trabajaron 0 horas
assertEquals(horasTrabajadas, calculoDeHoras.calcularHorasAPagarTurnoDiurno(horasTrabajadas));
}
@Test
public void TurnoDiurnoHoraTrabajadaDebeDevolver1HoraAPagar() {
CalculoHorasTrabajadas calculoDeHoras = new CalculoHorasTrabajadas();
float horasTrabajadas = 1;
//Se esperan 1 hora a pagar y se trabajo 1 hora
assertEquals(horasTrabajadas, calculoDeHoras.calcularHorasAPagarTurnoDiurno(horasTrabajadas));
}
@Test
public void TurnoDiurno47HorasTrabajadasDebeDevolver47HorasAPagar() {
CalculasHorasTrabajadasTurnoDiurnoTest
TurnoDiurno1HoraTrabajadaDebeDevolver1HoraAPagar()
1 test passed - 37ms
Test Results
CalculasHorasTrabajadasTurnoDiurnoTest
TurnoDiurno1HoraTrabajadaDebeDevolver1HoraAPagar()
"C:\Program Files\Java\jdk1.8.0_161\bin\java" ...
Process finished with exit code 0

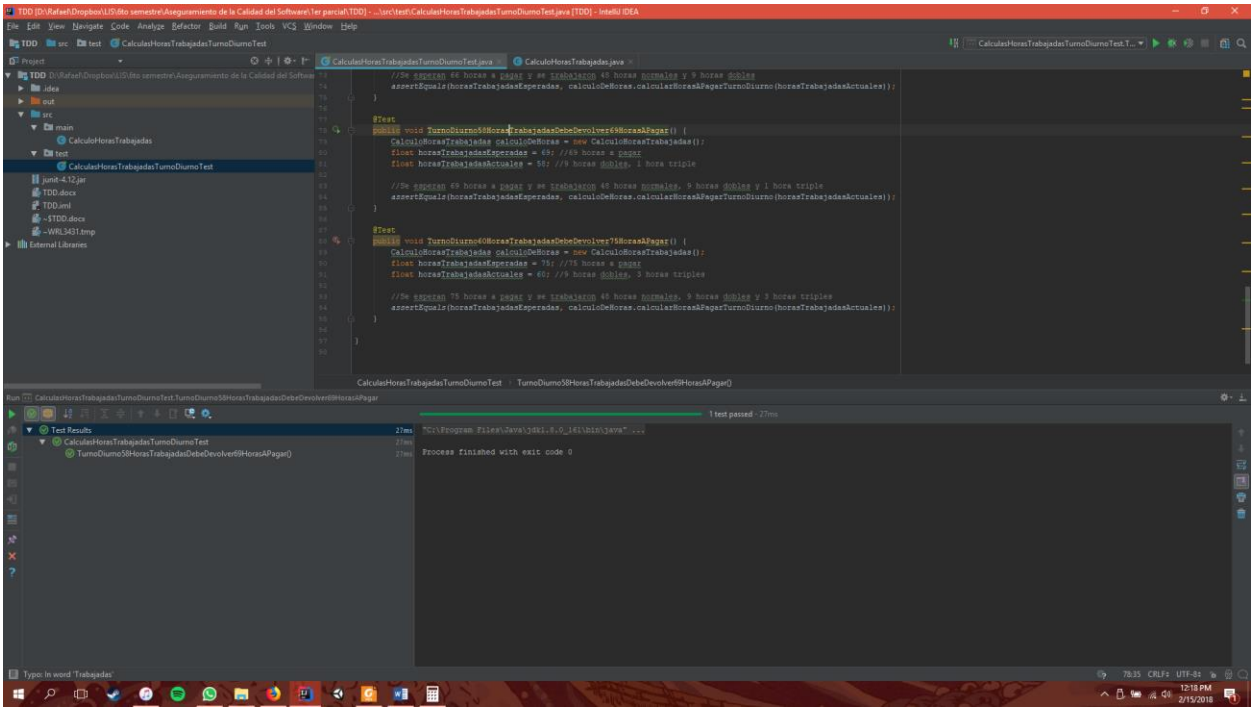

TDD [D:\Rafael\Dropbox\LIS\6to semestre\Aseguramiento de la Calidad del Software\1er parcial\TDD] - ...\src\test\CalculasHorasTrabajadasTurnoDiurnoTest.java [TDD] - IntelliJ IDEA
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
CalculasHorasTrabajadasTurnoDiurnoTest
CalculoHorasTrabajadas.java
//Se esperan 0 horas a pagar y se trabajaron 0 horas
assertEquals(horasTrabajadas, calculoDeHoras.calcularHorasAPagarTurnoDiurno(horasTrabajadas));
@Test
public void TurnoDiurno1HoraTrabajadaDebeDevolver1HoraAPagar() {
CalculoHorasTrabajadas calculoDeHoras = new CalculoHorasTrabajadas();
float horasTrabajadas = 1;
//Se esperan 1 hora a pagar y se trabajo 1 hora
assertEquals(horasTrabajadas, calculoDeHoras.calcularHorasAPagarTurnoDiurno(horasTrabajadas));
@Test
public void TurnoDiurno47HorasTrabajadasDebeDevolver47HorasAPagar() {
CalculoHorasTrabajadas calculoDeHoras = new CalculoHorasTrabajadas();
float horasTrabajadas = 47;
//Se esperan 47 horas a pagar y se trabajaron 47 horas
assertEquals(horasTrabajadas, calculoDeHoras.calcularHorasAPagarTurnoDiurno(horasTrabajadas));
@Test
public void TurnoDiurno48HorasTrabajadasDebeDevolver48HorasAPagar() {
CalculoHorasTrabajadas calculoDeHoras = new CalculoHorasTrabajadas();
float horasTrabajadas = 48;
CalculasHorasTrabajadasTurnoDiurnoTest
TurnoDiurno47HorasTrabajadasDebeDevolver47HorasAPagar()
Run CalculasHorasTrabajadasTurnoDiurnoTest.TurnoDiurno47HorasTrabajadasDebeDevolver47HorasAPagar
1 test passed - 29ms
Test Results 29ms
CalculasHorasTrabajadasTurnoDiurnoTest 29ms
TurnoDiurno47HorasTrabajadasDebeDevolver47HorasAPagar() 29ms
"C:\Program Files\Java\jdk1.8.0_161\bin\java" ...
Process finished with exit code 0
Typo: In word 'Turno'. Typo: In word 'Diurno'.
30:22 CRLF: UTF-8:
12:17 PM
2/15/2018

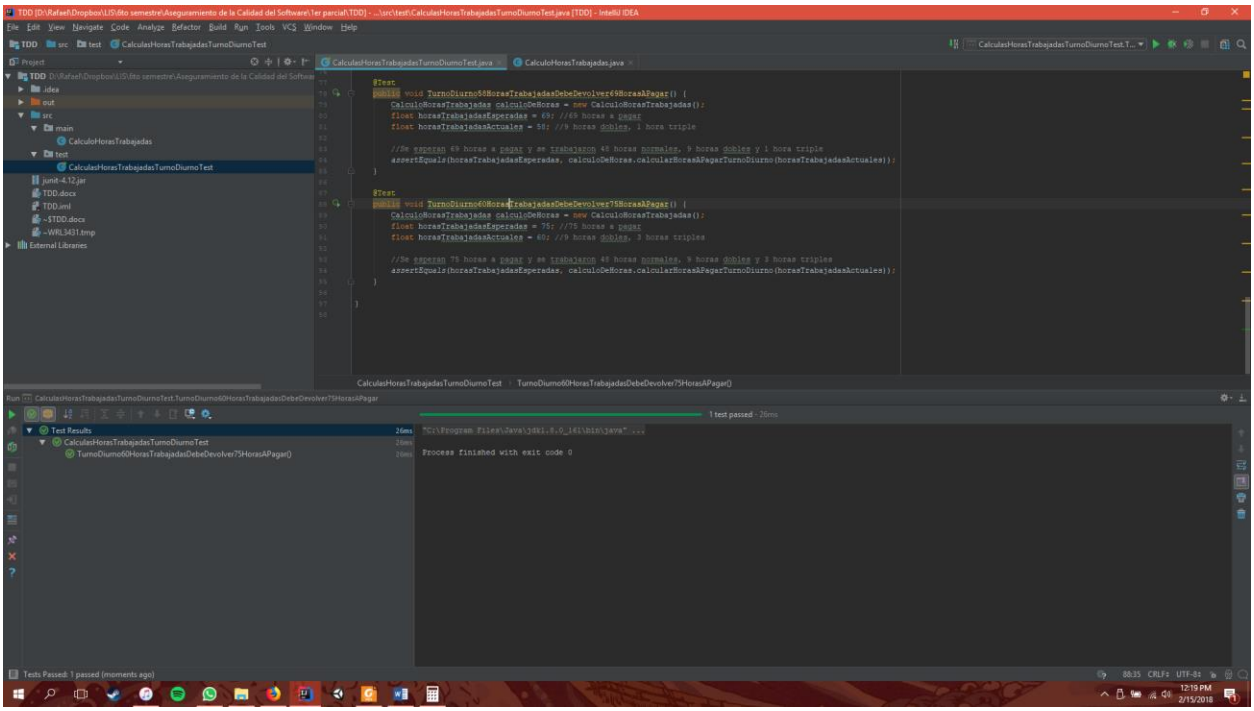

TDD [D:\Rafael\Dropbox\LIS\6to semestre\Aseguramiento de la Calidad del Software\1er parcial\TDD] - ...\src\test\CalculasHorasTrabajadasTurnoDiurnoTest.java [TDD] - IntelliJ IDEA
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
CalculasHorasTrabajadasTurnoDiurnoTest
CalculoHorasTrabajadas.java
//Se esperan 47 horas a pagar y se trabajaron 47 horas
assertEquals(horasTrabajadas, calculoDeHoras.calcularHorasAPagarTurnoDiurno(horasTrabajadas));
@Test
public void TurnoDiurno48HorasTrabajadasDebeDevolver48HorasAPagar() {
CalculoHorasTrabajadas calculoDeHoras = new CalculoHorasTrabajadas();
float horasTrabajadas = 48;
//Se esperan 48 horas a pagar y se trabajaron 48 horas
assertEquals(horasTrabajadas, calculoDeHoras.calcularHorasAPagarTurnoDiurno(horasTrabajadas));
@Test
public void TurnoDiurno49HorasTrabajadasDebeDevolver50HorasAPagar() {
CalculoHorasTrabajadas calculoDeHoras = new CalculoHorasTrabajadas();
float horasTrabajadasEsperadas = 50; //50 horas a pagar
float horasTrabajadasActuales = 49; //1 hora doble
//Se esperan 50 horas a pagar y se trabajaron 48 horas normales y 1 hora doble
assertEquals(horasTrabajadasEsperadas, calculoDeHoras.calcularHorasAPagarTurnoDiurno(horasTrabajadasActuales));
@Test
public void TurnoDiurno64HorasTrabajadasDebeDevolver64HorasAPagar() {
CalculoHorasTrabajadas calculoDeHoras = new CalculoHorasTrabajadas();
float horasTrabajadasEsperadas = 64; //64 horas a pagar
CalculasHorasTrabajadasTurnoDiurnoTest
TurnoDiurno48HorasTrabajadasDebeDevolver48HorasAPagar()
Run CalculasHorasTrabajadasTurnoDiurnoTest.TurnoDiurno48HorasTrabajadasDebeDevolver48HorasAPagar
1 test passed - 28ms
Test Results 28ms
CalculasHorasTrabajadasTurnoDiurnoTest 28ms
TurnoDiurno48HorasTrabajadasDebeDevolver48HorasAPagar() 28ms
"C:\Program Files\Java\jdk1.8.0_161\bin\java" ...
Process finished with exit code 0
Tests Passed: 1 passed (moments ago)
39:31 CRLF: UTF-8:
12:17 PM
2/15/2018


TDD [D:\Rafael\Dropbox\LIS\6to semestre\Aseguramiento de la Calidad del Software\1er parcial\TDD] - ...\src\test\CalculasHorasTrabajadasTurnoDiurnoTest.java [TDD] - IntelliJ IDEA
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
CalculasHorasTrabajadasTurnoDiurnoTest
CalculoHorasTrabajadas.java
TurnoDiurno49HorasTrabajadasDebeDevolver50HorasAPagar() {
CalculoHorasTrabajadas calculoDeHoras = new CalculoHorasTrabajadas();
float horasTrabajadasEsperadas = 50; //50 horas a pagar
float horasTrabajadasActuales = 49; //1 hora doble
//Se esperan 50 horas a pagar y se trabajaron 48 horas normales y 1 hora doble
assertEquals(horasTrabajadasEsperadas, calculoDeHoras.calcularHorasAPagarTurnoDiurno(horasTrabajadasActuales));
TurnoDiurno56HorasTrabajadasDebeDevolver64HorasAPagar() {
float horasTrabajadasEsperadas = 64; //64 horas a pagar
float horasTrabajadasActuales = 56; //8 horas dobles
//Se esperan 64 horas a pagar y se trabajaron 48 horas normales y 8 horas dobles
TurnoDiurno57HorasTrabajadasDebeDevolver66HorasAPagar() {
CalculasHorasTrabajadasTurnoDiurnoTest > TurnoDiurno49HorasTrabajadasDebeDevolver50HorasAPagar()
Run CalculasHorasTrabajadasTurnoDiurnoTest.TurnoDiurno49HorasTrabajadasDebeDevolver50HorasAPagar
1 test passed - 27ms
Test Results 27ms
CalculasHorasTrabajadasTurnoDiurnoTest 27ms
TurnoDiurno49HorasTrabajadasDebeDevolver50HorasAPagar() 27ms
"C:\Program Files\Java\jdk1.8.0_161\bin\java" ...
Process finished with exit code 0
Tests Passed: 1 passed (moments ago)
48:30 CRLF: UTF-8:
12:17 PM
2/15/2018


TDD [D:\Rafael\Dropbox\LIS\6to semestre\Aseguramiento de la Calidad del Software\1er parcial\TDD] - ...\src\test\CalculasHorasTrabajadasTurnoDiurnoTest.java [TDD] - IntelliJ IDEA
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
CalculasHorasTrabajadasTurnoDiurnoTest
CalculoHorasTrabajadas.java
CalculoHorasTrabajadas calculoDeHoras = new CalculoHorasTrabajadas();
float horasTrabajadasEsperadas = 50; //50 horas a pagar
float horasTrabajadasActuales = 49; //1 hora doble
//Se esperan 50 horas a pagar y se trabajaron 48 horas normales y 1 hora doble
assertEquals(horasTrabajadasEsperadas, calculoDeHoras.calcularHorasAPagarTurnoDiurno(horasTrabajadasActuales));
TurnoDiurno56HorasTrabajadasDebeDevolver64HorasAPagar() {
float horasTrabajadasEsperadas = 64; //64 horas a pagar
float horasTrabajadasActuales = 56; //8 horas dobles
//Se esperan 64 horas a pagar y se trabajaron 48 horas normales y 8 horas dobles
TurnoDiurno57HorasTrabajadasDebeDevolver66HorasAPagar() {
float horasTrabajadasEsperadas = 66; //66 horas a pagar
float horasTrabajadasActuales = 57; //9 horas dobles
//Se esperan 66 horas a pagar y se trabajaron 48 horas normales y 9 horas dobles
CalculasHorasTrabajadasTurnoDiurnoTest > TurnoDiurno56HorasTrabajadasDebeDevolver64HorasAPagar()
Run CalculasHorasTrabajadasTurnoDiurnoTest.TurnoDiurno56HorasTrabajadasDebeDevolver64HorasAPagar
1 test passed - 26ms
Test Results 26ms
CalculasHorasTrabajadasTurnoDiurnoTest 26ms
TurnoDiurno56HorasTrabajadasDebeDevolver64HorasAPagar() 26ms
"C:\Program Files\Java\jdk1.8.0_161\bin\java" ...
Process finished with exit code 0
Tests Passed: 1 passed (moments ago)
58:31 CRLF: UTF-8:
12:18 PM
2/15/2018


float horasTrabajadasActuales = 56; //8 horas dobles
//Se esperan 64 horas a pagar y se trabajaron 48 horas normales y 8 horas dobles
assertEquals(horasTrabajadasEsperadas, calculoDeHoras.calcularHorasAPagarTurnoDiurno(horasTrabajadasActuales));
@Test
public void TurnoDiurno57HorasTrabajadasDebeDevolver66HorasAPagar() {
CalculoHorasTrabajadas calculoDeHoras = new CalculoHorasTrabajadas();
float horasTrabajadasEsperadas = 66; //66 horas a pagar
float horasTrabajadasActuales = 57; //9 horas dobles
//Se esperan 66 horas a pagar y se trabajaron 48 horas normales y 9 horas dobles
assertEquals(horasTrabajadasEsperadas, calculoDeHoras.calcularHorasAPagarTurnoDiurno(horasTrabajadasActuales));
@Test
public void TurnoDiurno58HorasTrabajadasDebeDevolver69HorasAPagar() {
CalculoHorasTrabajadas calculoDeHoras = new CalculoHorasTrabajadas();
float horasTrabajadasEsperadas = 69; //69 horas a pagar
float horasTrabajadasActuales = 58; //9 horas dobles, 1 hora triple
//Se esperan 69 horas a pagar y se trabajaron 48 horas normales, 9 horas dobles y 1 hora triple
assertEquals(horasTrabajadasEsperadas, calculoDeHoras.calcularHorasAPagarTurnoDiurno(horasTrabajadasActuales));
CalculasHorasTrabajadasTurnoDiurnoTest
TurnoDiurno57HorasTrabajadasDebeDevolver66HorasAPagar()
1 test passed - 29ms
"C:\Program Files\Java\jdk1.8.0_161\bin\java"...
Process finished with exit code 0


//Se esperan 66 horas a pagar y se trabajaron 48 horas normales y 9 horas dobles
assertEquals(horasTrabajadasEsperadas, calculoDeHoras.calcularHorasAPagarTurnoDiurno(horasTrabajadasActuales));
@Test
public void TurnoDiurno58HorasTrabajadasDebeDevolver69HorasAPagar() {
CalculoHorasTrabajadas calculoDeHoras = new CalculoHorasTrabajadas();
float horasTrabajadasEsperadas = 69; //69 horas a pagar
float horasTrabajadasActuales = 58; //9 horas dobles, 1 hora triple
//Se esperan 69 horas a pagar y se trabajaron 48 horas normales, 9 horas dobles y 1 hora triple
assertEquals(horasTrabajadasEsperadas, calculoDeHoras.calcularHorasAPagarTurnoDiurno(horasTrabajadasActuales));
@Test
public void TurnoDiurno60HorasTrabajadasDebeDevolver75HorasAPagar() {
CalculoHorasTrabajadas calculoDeHoras = new CalculoHorasTrabajadas();
float horasTrabajadasEsperadas = 75; //75 horas a pagar
float horasTrabajadasActuales = 60; //9 horas dobles, 3 horas triples
//Se esperan 75 horas a pagar y se trabajaron 48 horas normales, 9 horas dobles y 3 horas triples
assertEquals(horasTrabajadasEsperadas, calculoDeHoras.calcularHorasAPagarTurnoDiurno(horasTrabajadasActuales));
CalculasHorasTrabajadasTurnoDiurnoTest
TurnoDiurno58HorasTrabajadasDebeDevolver69HorasAPagar()
1 test passed - 27ms
"C:\Program Files\Java\jdk1.8.0_161\bin\java"...
Process finished with exit code 0



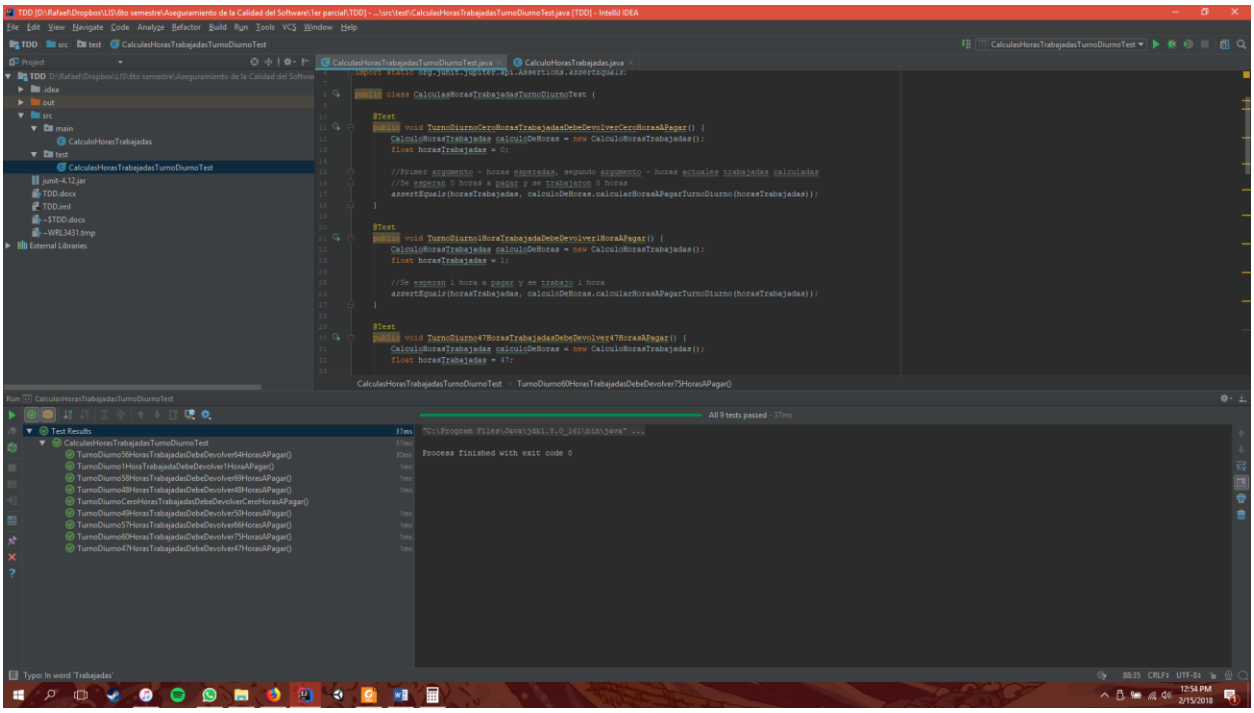
Podemos apreciar que las pruebas fallaron como se esperaba, por lo tanto, es correcto decir que las pruebas funcionan de la manera adecuada.

El siguiente paso es modificar el método "*calcularHorasAPagarTurnoDiurno()*" para que realice el cálculo adecuado.

```
package main;

public class CalculoHorasTrabajadas {

    /**
     * Método para realizar el cálculo de horas a pagar. Para la primera prueba sólo
se desea verificar que la prueba
     * aprueba o falla, comprobando así que la prueba es correcta.
     * @param turno El turno del empleado bajo el cual se hace el cálculo.
     * @param horas Número de otras trabajadas en la semana laboral.
     * @return Número cualquiera. Sólo se usará para verificar el funcionamiento de la
prueba.
     */
    public float calcularHorasAPagar(String turno, float horas) {
        return (float) 50.0;
    }

    /**
     * Recibe el número de horas trabajadas durante la semana laboral en el turno
diurno y calcula el número total de horas
     * a pagar, mutiplicando por 2 las horas dobles trabajadas y por 3 las horas
triples trabajadas. Así queda una suma
     * del total de horas.
     * @param horasTrabajadas Número de horas trabajadas en el turno.
     * @return La suma del total de horas a pagar.
     */
    public float calcularHorasAPagarTurnoDiurno(float horasTrabajadas) {
        float horasActualesTrabajadas = horasTrabajadas;
```

```
        float horasAPagar = 0;

        //Cantidade horas obligatorias a cumplir en el turno
        final float HORAS_NORMALES_OBLIGATORIAS = 48;

        //Cantidad de horas máxima que se pueden laborar al cubrir el máximo de horas
dobles
        final float MAXIMO_HORAS_DOBLES = 57;


        if ( (horasActualesTrabajadas >= 0) && (horasActualesTrabajadas <=
HORAS_NORMALES_OBLIGATORIAS) ) {
            horasAPagar = horasActualesTrabajadas;
        }
        else if( (horasActualesTrabajadas > HORAS_NORMALES_OBLIGATORIAS) &&
(horasActualesTrabajadas <= MAXIMO_HORAS_DOBLES) ) {
            float horasDoblesTrabajadas = horasActualesTrabajadas -
HORAS_NORMALES_OBLIGATORIAS;

            horasAPagar = HORAS_NORMALES_OBLIGATORIAS + (2 * horasDoblesTrabajadas);
        }
        else if ( horasActualesTrabajadas > 57) {
            float horasTriplesTrabajadas = horasActualesTrabajadas -
MAXIMO_HORAS_DOBLES;
            float horasDoblesTrabajadas = MAXIMO_HORAS_DOBLES -
HORAS_NORMALES_OBLIGATORIAS;

            horasAPagar = HORAS_NORMALES_OBLIGATORIAS + (2 * horasDoblesTrabajadas) +
(3 * horasTriplesTrabajadas);
        }

        return horasAPagar;
    }

    public float calcularHorasAPagarTurnoNocturno(float horas) {
        float horasAPagar = 100;

        return horasAPagar;
    }

    public float calcularHorasAPagarTurnoMixto(float horas) {
        float horasAPagar = 100;

        return horasAPagar;
    }

}
```

Y al ejecutar las pruebas podemos con los mismos valores de entrada y mismos valores esperados que definimos anteriormente, podeos ver que todas pasan.



Ahora hace falta implementar las pruebas para el turno nocturno y el turno mixto, después probar que el código de prueba es corriendo forzando a que este falle y pase, y después implementar el código de la aplicación. Los pasos para este proceso son análogos a los realizados para el turno diurno, descritos en este documento.