

## IP2022. Tests de ejecuciones por consola en Jenkins

Estos tests lanzan los programas Java mediante la ejecución del método main de la clase implementada: para ello, automatizan la introducción por teclado de uno o varios valores para cada caso de prueba y comprueban que el valor esperado aparece correctamente como parte de la salida por consola. Los valores utilizados para los casos de prueba son los que aparecen en los guiones de las sesiones, y algunos casos más.

Veamos un ejemplo: para la clase `Euclides.java` del ejercicio 2 de la sesión 3 se ejecuta un caso de prueba en el que se introducen los valores -3, 5 y 25. El resultado debe ser el máximo común divisor de 5 y 25, descartando el -3. En el siguiente ejemplo, el caso de prueba de ha fallado. El mensaje de error tiene la siguiente estructura:

```
Para los valores de entrada -3; 5; 25; el texto esperado no aparece en la salida:
[ESPERADO]
5 y 25 es 5
[SALIDA]
Introduce el primer valor
Introduce el primer valor
Introduce el segundo valor
El maximo comun divisor de 25 y 5 es: 5
```

Así, para cada caso de prueba fallido se muestra:

- los **valores de entrada** del caso de prueba, separados por punto y coma:

-3; 5; 25;

- El **resultado esperado**, tras el texto [ESPERADO]

5 y 25 es 5

- La **salida por consola** de la ejecución del programa, tras el texto [SALIDA]

```
Introduce el primer valor
Introduce el primer valor
Introduce el segundo valor
El maximo comun divisor de 25 y 5 es: 5
```

El motivo del fallo es que el texto ESPERADO debe aparecer como parte del texto de salida, y no es así. En este ejemplo, en la salida se ha añadido el carácter ':' (dos puntos) que no está en el texto esperado, por lo que la prueba ha fallado. La salida debe ser exactamente igual a la que se muestra en los enunciados de las sesiones.

*En estos test de consola (consoleIO), para tener acceso a la entrada por teclado y la salida por consola en JUnit utilizamos el framework System Rules (<https://stefanbirkner.github.io/system-rules/>).*

*Además, para ejecutar el mismo test con distintos casos de prueba (valores de entrada y resultado esperado) utilizamos tests parametrizados de JUnit ([https://www.tutorialspoint.com/junit/junit\\_parameterized\\_test.htm](https://www.tutorialspoint.com/junit/junit_parameterized_test.htm)).*

Estos tests por consola tienen un nombre y un número como sufijo para cada caso de prueba:

**org.ip2022:ip2022\_sesion04**

Test Name
+ org.ip.console.sesion04.CilindrolOTest.salidaConsola[Cilindro_0]
+ org.ip.console.sesion04.CilindrolOTest.salidaConsola[Cilindro_1]
+ org.ip.console.sesion04.CilindrolOTest.salidaConsola[Cilindro_2]
+ org.ip.console.sesion04.CilindrolOTest.salidaConsola[Cilindro_3]

Se han incorporado tests de consola a partir de la **sesión 03** en adelante. A continuación, os exponemos algunos ejemplos de fallos que se producen en este tipo de tests. Veréis que algunos fallos se pueden considerar *falsos positivos*, es decir, el test falla cuando no debería fallar. Aunque hemos tratado de minimizarlos al máximo, **en algunas ocasiones el mensaje de salida del ejercicio implementado por el estudiante no coincide con la salida esperada según el enunciado**, así que el **test falla**, como en el ejemplo anterior.

Insistimos en que la salida debe ser **exactamente igual** a la que se muestra en los enunciados de las sesiones.

A continuación, se muestran ejemplos de fallos habituales detectados en las sesiones 03 y 04.

### Sesion 03: TablaMultiplicar.java

Falla cuando no se controla que el valor de entrada esté entre 1 y 10.

- **Ejemplo 1:** El caso de prueba 1 introduce el valor 0, que debería ser controlado.

```
org.ip.console.sesion03.TablaMultiplicarOTest.salidaConsola[TablaMultiplicar_1]

= Detalles del error

Para los valores de entrada 0; 2; el texto esperado no aparece en la salida:
[ESPERADO]
2 x 8 = 16
[SALIDA]
Introduzca un numero (de 1 a 10):
Tabla del 0
0 x 1 = 0
0 x 2 = 0
0 x 3 = 0
0 x 4 = 0
0 x 5 = 0
0 x 6 = 0
0 x 7 = 0
0 x 8 = 0
0 x 9 = 0
0 x 10 = 0
```

- **Ejemplo 2:** El caso de prueba 2 introduce el valor 11, que debería ser controlado.

⇒ [org.ip.console.sesion03.TablaMultiplicarIOTest.salidaConsola\[TablaMultiplicar\\_2\]](#)

#### ⇒ Detalles del error

Para los valores de entrada 11; 9; el texto esperado no aparece en la salida:

[ESPERADO]

9 x 10 = 90



[SALIDA]

Escribe un numero (de 1 a 10)

Tabla del 11

11 x 1 = 11

11 x 2 = 22

11 x 3 = 33

11 x 4 = 44

11 x 5 = 55

11 x 6 = 66

11 x 7 = 77

11 x 8 = 88

11 x 9 = 99

11 x 10 = 110

+ Traza de la pila

- **Ejemplo 3:** El caso de prueba 3 introduce los valores 20 y -3 que deberían ser controlados:

⇒ [org.ip.console.sesion03.TablaMultiplicarIOTest.salidaConsola\[TablaMultiplicar\\_3\]](#)

#### ⇒ Detalles del error

Para los valores de entrada 20; -3; 8; el texto esperado no aparece en la salida:

[ESPERADO]

8 x 2 = 16

[SALIDA]

Introduzca un numero (de 1 a 10):

Tabla del 20

20 x 1 = 20

20 x 2 = 40

20 x 3 = 60

20 x 4 = 80

20 x 5 = 100

20 x 6 = 120

20 x 7 = 140

20 x 8 = 160

20 x 9 = 180

20 x 10 = 200

+ Traza de la pila

**Ejemplo 4:** Este test también falla cuando la salida por pantalla no es correcta:

⇒ [org.ip.console.sesion03.TablaMultiplicarIOTest.salidaConsola\[TablaMultiplicar\\_0\]](#)

#### – Detalles del error

Para los valores de entrada 5; el texto esperado no aparece en la salida:

[ESPERADO]

5 x 3 = 15

[SALIDA]

Introduzca un numero (de 1 a 10): Tabla del 5

5 x 1 = 5

5 x 1 = 10

5 x 1 = 15

5 x 1 = 20

5 x 1 = 25

5 x 1 = 30

5 x 1 = 35

5 x 1 = 40

5 x 1 = 45

5 x 1 = 50

#### + Traza de la pila

**Ejemplo 5:** El test también falla si no se dejan los espacios en blanco adecuados. Insistimos, los mensajes de salida deben ser los mismos (**exactamente iguales**) que se proporcionan en los guiones de prácticas.

⇒ [org.ip.console.sesion03.TablaMultiplicarIOTest.salidaConsola\[TablaMultiplicar\\_1\]](#)

#### – Detalles del error

Para los valores de entrada 0; 2; el texto esperado no aparece en la salida:

[ESPERADO]

2 x 8 = 16

[SALIDA]

Introduce un numero (del 1 al 10)

Introduce un numero (del 1 al 10)

TABLA DEL 2

2x1= 2

2x2= 4

2x3= 6

2x4= 8

2x5= 10

2x6= 12

2x7= 14

2x8= 16

2x9= 18

2x10= 20

#### + Traza de la pila

## Sesion 04: Cilindro.java

**Ejemplo 1:** El test falla si el estudiante ha olvidado poner dos puntos (:) en la salida.

[org.ip2022:ip2022\\_sesion04](#)

**Test Name**

`org.ip.console.sesion04.CilindrolOTest.salidaConsola[Cilindro_0]`

**Detalles del error**

Para los valores de entrada 5,5; 2,7; 1; el texto esperado no aparece en la salida:

[ESPERADO]

El area del cilindro es de: 283.37165735379926

[SALIDA]

Introduzca radio: Introduzca altura: Que desea calcular (1 (area) / 2 (volumen):

El area del cilindro es de 283.37165735379926

+ Traza de la pila

**Ejemplo 2:** El test falla si el estudiante añade espacio en blanco antes de los dos puntos.

[org.ip2022:ip2022\\_sesion04](#)

**Test Name**

`org.ip.console.sesion04.CilindrolOTest.salidaConsola[Cilindro_0]`

**Detalles del error**

Para los valores de entrada 5,5; 2,7; 1; el texto esperado no aparece en la salida:

[ESPERADO]

El area del cilindro es de: 283.37165735379926

[SALIDA]

Introduzca radio : Introduzca altura : Que desea calcular (1 (area) / 2 (volumen) :

El area del cilindro es de : 283.37165735379926

+ Traza de la pila

**Ejemplo 3:** El test falla si el estudiante no añade espacio en blanco después de los dos puntos.

`org.ip.console.sesion04.CilindrolOTest.salidaConsola[Cilindro_0]`

**Detalles del error**

Para los valores de entrada 5,5; 2,7; 1; el texto esperado no aparece en la salida:

[ESPERADO]

El area del cilindro es de: 283.37165735379926

[SALIDA]

Introduzca radio:

Introduzca altura:

Que desea calcular (1/(area) / 2 (volumen):

El area del cilindro es de:283.37165735379926

+ Traza de la pila

#### Ejemplo 4: El test falla si el resultado de los cálculos a realizar no es el correcto:

⇒ `org.ip.console.sesion04.CilindrolOTest.salidaConsola[Cilindro_0]`

##### — Detalles del error

```
Para los valores de entrada 5,5; 2,7; 1; el texto esperado no aparece en la salida:  
[ESPERADO]  
El area del cilindro es de: 283.37165735379926  
[SALIDA]  
Introduzca el radio  
Introduzca la altura  
Que desea calcular 1 = Area. 2 = Volumen.  
El area del cilindro es de: 51.5221195188726  
Opcion no valida
```

##### + Traza de la pila

---

#### Ejemplo 5: El test falla si se comente errores ortográficos.

⇒ `org.ip.console.sesion04.CilindrolOTest.salidaConsola[Cilindro_0]`

##### — Detalles del error

```
Para los valores de entrada 5,5; 2,7; 1; el texto esperado no aparece en la salida:  
[ESPERADO]  
El area del cilindro es de: 283.37165735379926  
[SALIDA]  
Introduzca el radio:  
Introduzca la altura:  
Que desea calcular (1 (area) / 2 (volumen):  
El area del cicilindro es de: 283.37165735379926
```

##### + Traza de la pila

## Sesion 04: DivisoresPrimos.java

**Ejemplo 1:** El test falla si el resultado no es correcto.

= [org.ip.console.sesion04.DivisoresPrimosIOTest.salidaConsola\[DivisoresPrimos\\_1\]](#)

### = Detalles del error

Para los valores de entrada 23; el texto esperado no aparece en la salida:

[ESPERADO]

Los divisores primos de 23 son: 23

[SALIDA]

Introduzca un numero :

Los divisores primos de 23 son :

### + Traza de la pila

**Ejemplo 2:** El test falla si el estudiante añade espacio en blanco entre la palabra “son” y los dos puntos.

= [org.ip.console.sesion04.DivisoresPrimosIOTest.salidaConsola\[DivisoresPrimos\\_3\]](#)

### = Detalles del error

Para los valores de entrada 1690; el texto esperado no aparece en la salida:

[ESPERADO]

Los divisores primos de 1690 son: 2 5 13

[SALIDA]

Introduzca un numero :

Los divisores primos de 1690 son : 2 5 13

### + Traza de la pila

**Ejemplo 3:** El test falla si se faltan palabras en la salida. Insistimos, los mensajes de salida deben ser los mismos (**exactamente iguales**) que se proporcionan en los guiones de prácticas.

= [org.ip.console.sesion04.DivisoresPrimosIOTest.salidaConsola\[DivisoresPrimos\\_0\]](#)

### = Detalles del error

Para los valores de entrada 510510; el texto esperado no aparece en la salida:

[ESPERADO]

Los divisores primos de 510510 son: 2 3 5 7 11 13 17

[SALIDA]

Introduzca un numero:

Divisores Primos: 2 3 5 7 11 13 17

### + Traza de la pila

**Ejemplo 4:** El test falla si en la salida se muestra un contenido que no es el esperado, aunque signifique lo mismo. Insistimos, los mensajes de salida deben ser los mismos (**exactamente iguales**) que se proporcionan en los guiones de prácticas.

⇒ org.ip.console.sesion04.DivisoresPrimosIOTest.salidaConsola[DivisoresPrimos\_1]

#### – Detalles del error

Para los valores de entrada 23; el texto esperado no aparece en la salida:

[ESPERADO]

Los divisores primos de 23 son: 23

[SALIDA]

Introduzca un numero:

El numero introducido es primo, no tiene divisores mas que si mismo y el 1

#### + Traza de la pila

**Ejemplo 5:** El test falla si se cometen errores ortográficos. Insistimos, los mensajes de salida deben ser los mismos (**exactamente iguales**) que se proporcionan en los guiones de prácticas.

⇒ org.ip.console.sesion04.DivisoresPrimosIOTest.salidaConsola[DivisoresPrimos\_0]

#### – Detalles del error

Para los valores de entrada 510510; el texto esperado no aparece en la salida:

[ESPERADO]

Los divisores primos de 510510 son: 2 3 5 7 11 13 17

[SALIDA]

Introduzca un numero Los divisores primos de 510510 son: 2 3 5 7 11 13 17

#### + Traza de la pila