|  |  |
| --- | --- |
| Pruebas y Depuración en Entornos de Desarrollo | Ignacio Castillón Salguero  Desarrollo de Aplicaciones Multiplataforma |

# Índice

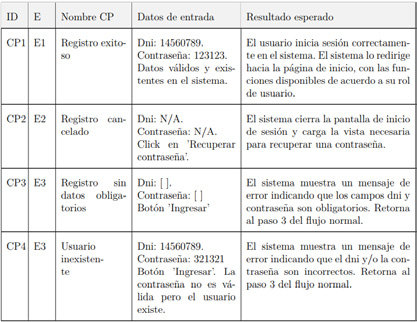
1. [Tipos de pruebas](#_1._Tipos_de)
2. [Casos de prueba](#_1.1_Casos_de)
3. [Herramientas de depuración](#_1.2_Herramientas_de)
4. [Herramientas de prueba](#_1.3_Herramientas_de)
5. [Utilizar herramientas de depuración para definir puntos de ruptura y seguimiento](#_2._Utilizar_herramientas)
6. [Utilizar las herramientas de depuración para examinar y modificar el comportamiento de un programa en tiempo de ejecución](#_2.1_Utilizar_las)
7. [Bibliografía](#_Bibliografía)

# 1. Tipos de pruebas

1. **Pruebas unitarias**: Se enfocan en verificar el correcto funcionamiento de módulos de manera individual.
2. **Pruebas de integración**: Evalúan la comunicación y compatibilidad entre diferentes módulos o componentes.
3. **Pruebas de regresión**: Confirman que nuevas modificaciones en el código no afecten funcionalidades existentes.
4. **Pruebas de validación:** Se prueba el software como conjunto para comprobar si cumple los requisitos funcionales, de rendimiento, seguridad, etc.
5. **Pruebas de sistema y rendimiento**: Se verifica si todos los elementos del sistema se integran adecuadamente y que se realizan las funciones apropiadas.
6. **Pruebas de aceptación**: Se prueba el software en el entorno de explotación y el cliente comprueba si sus requisitos se cumplen.

# 1.1 Casos de prueba

Un caso de prueba es una secuencia de ejecución detalla que valida paso a paso una funcionalidad o un requerimiento. Podremos comprobar si el resultado obtenido coincide con el esperado, en el caso de que no coincidiese, esta secuencia de pasos nos permitiría encontrar y corregir el error.

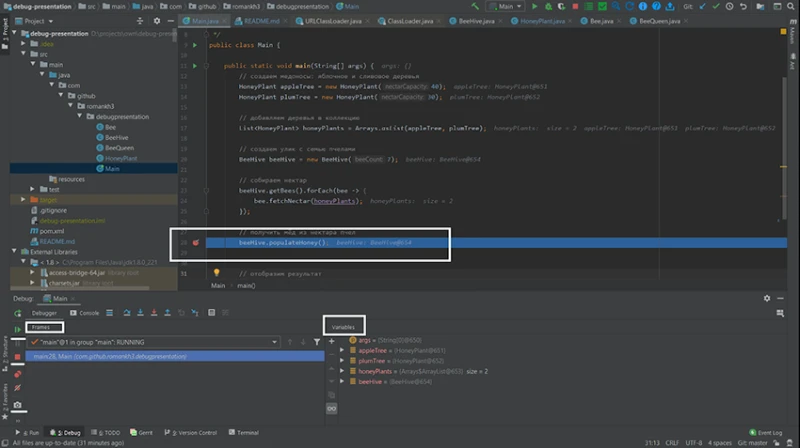


# 1.2 Herramientas de depuración

Se conoce como depuración al proceso por el cual se busca encontrar, aislar y resolver los errores de codificación o errores de software. Esta nos ayuda a descubrir las causas de los errores, a evitar problemas de funcionamiento y a mejorar el rendimiento general.

**Encontramos varias herramientas de depuración**:

**Los compiladores que ejecutan el programa paso a paso, permitiendo comprobar valores intermedios de las variables, las posibles entradas, salidas, etc.  
Los depuradores se deben usar para realizar un seguimiento sobre la forma de actuar de los programas.**



# 1.3 Herramientas de prueba

Son el proceso que permite evaluar y verificar que la aplicación hace lo que se supone que debe hacer. Estas pruebas tienen como beneficio la prevención de errores y la mejora de rendimiento.

Las pruebas de software se dividen en varias categorías según su objetivo y alcance:

* **Pruebas funcionales**

Verifican si el software cumple con los requisitos definidos. Incluyen:

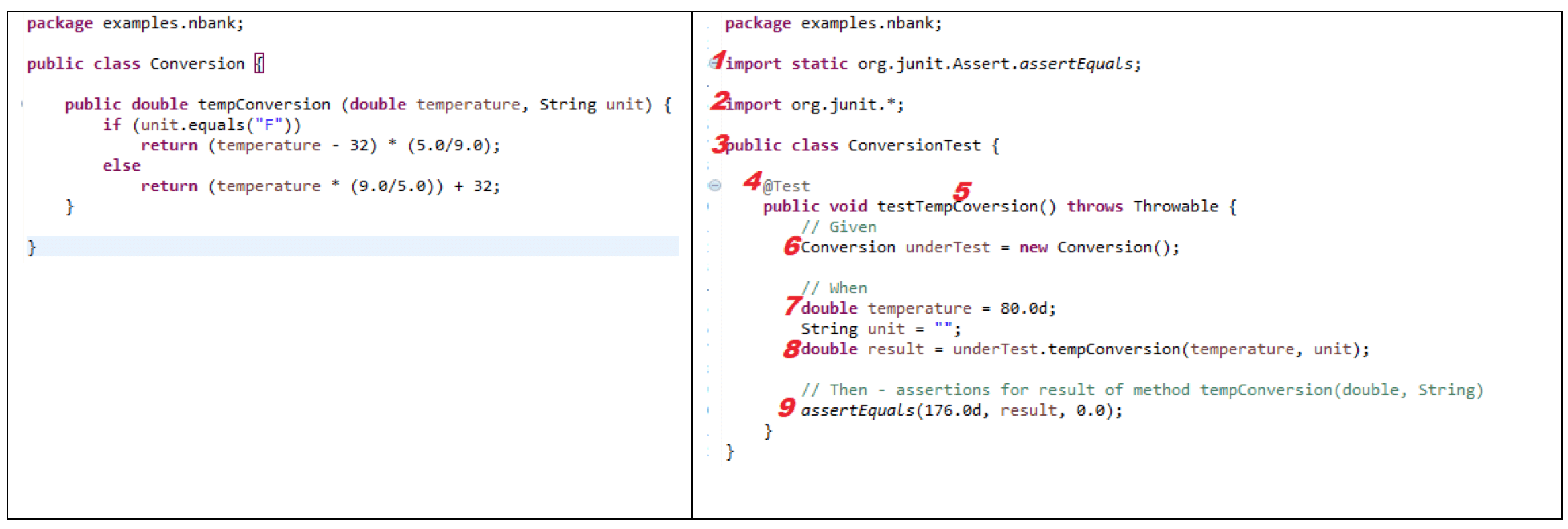
* **Pruebas unitarias:** Evaluación de módulos individuales.
* **Pruebas de integración:** Verificación de la comunicación entre módulos.
* **Pruebas de sistema:** Validación del software en su totalidad.
* **Pruebas de aceptación:** Comprobación final con el usuario o cliente.
* **Pruebas no funcionales**

Evalúan características no relacionadas con la funcionalidad del sistema, como:

* **Pruebas de rendimiento:** Analizan velocidad, estabilidad y consumo de recursos.
* **Pruebas de carga:** Evalúan el comportamiento del sistema con múltiples usuarios.
* **Pruebas de seguridad:** Identifican posibles vulnerabilidades.
* **Pruebas de usabilidad:** Analizan la experiencia del usuario fina

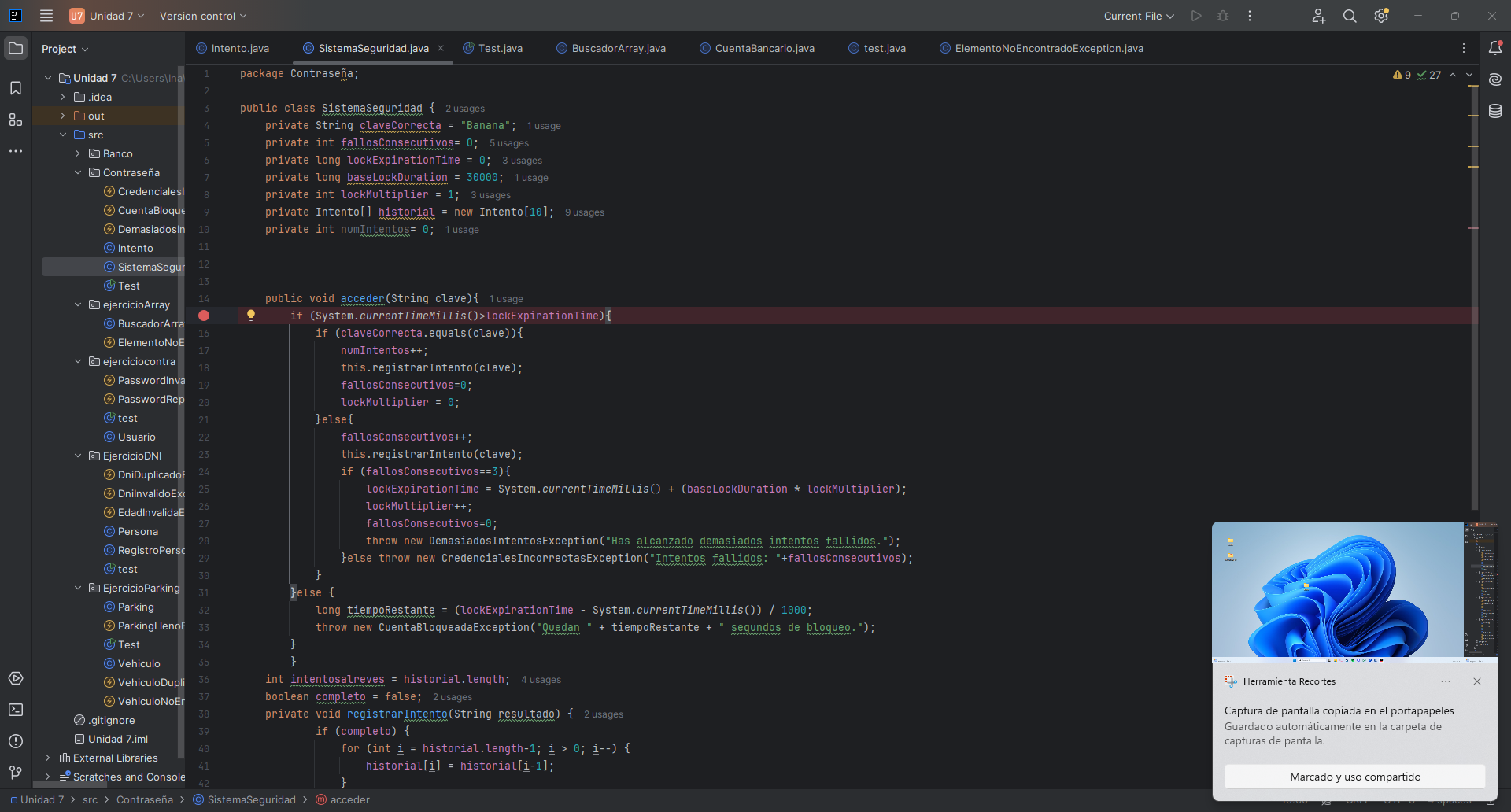
Herramientas de prueba populares

1. **JUnit (Java):** Para pruebas unitarias automatizadas.
2. **Selenium:** Para pruebas de interfaz de usuario en aplicaciones web.



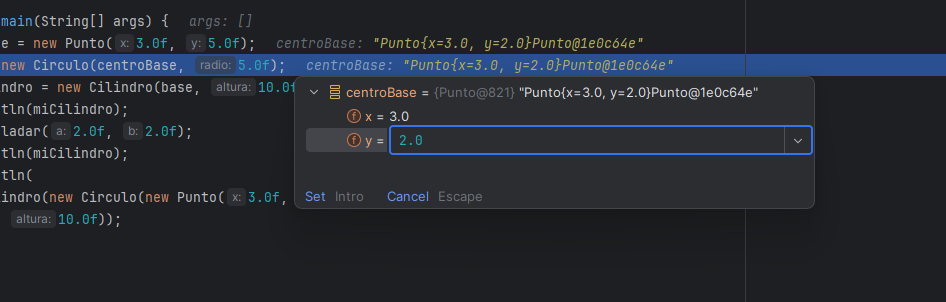
# 2. Utilizar herramientas de depuración para definir puntos de ruptura y seguimiento

Los puntos de rupturas son indicadores que se colocan en el código para que cuando se depure este se detenga donde se encuentra el punto.



# 2.1 Utilizar las herramientas de depuración para examinar y modificar el comportamiento de un programa en tiempo de ejecución

Las herramientas de depuración además de permitir verificar y evaluar los códigos para así solucionar errores, nos permite modificar las variables durante la depuración.



# Bibliografía

Casos de prueba: <https://es.linkedin.com/pulse/qué-es-un-caso-de-prueba-servicios-de-informatica-profesion-gyyzf>

Herramientas de depuración: <https://www.ibm.com/es-es/topics/debugging>

Herramientas de prueba: <https://www.ibm.com/es-es/topics/software-testing>