**White and Black Box Testing**

**Prework**

* Analizar qué son las pruebas de tipo white box testing
* Analizar qué son las pruebas de tipo black box testing
* Comparar y reconocer las diferencias entre black box y white box testing
* **Black Box Testing:** es un método de prueba de software en el que el probador no conoce la estructura/diseño/implementación interna del elemento que se está probando. Solo se prueba el diseño externo y la estructura.
* **White Box Testing:** es un método de prueba de software en el que el evaluador conoce la estructura/diseño/implementación interna del elemento que se está probando. Se prueba la implementación y el impacto del código.

| **Pruebas de caja negra** | **Pruebas de caja blanca** |
| --- | --- |
| Es una forma de prueba de software en la que se oculta la estructura interna o el programa o el código y no se sabe nada al respecto. | Es una forma de probar el software en la que el probador tiene conocimiento sobre la estructura interna o el código o el programa del software. |
| La implementación del código no es necesaria para las pruebas de caja negra. | La implementación del código es necesaria para las pruebas de caja blanca. |
| En su mayoría, lo realizan probadores de software. | Lo hacen principalmente los desarrolladores de software. |
| No se necesita conocimiento de implementación. | Se requiere conocimiento de implementación. |
| Puede denominarse prueba de software externa o externa. | Es la prueba de software interna o interna. |
| Es una prueba funcional del software. | Es una prueba estructural del software. |
| Esta prueba puede iniciarse sobre la base del documento de especificaciones de requisitos. | Este tipo de prueba de software se inicia después del documento de diseño detallado. |
| No se requieren conocimientos de programación. | Imprescindible tener conocimientos de programación. |
| Es la prueba de comportamiento del software. | Es la prueba lógica del software. |
| Es aplicable a los niveles más altos de pruebas de software. | Es generalmente aplicable a los niveles más bajos de pruebas de software. |
| También se le llama prueba cerrada. | También se denomina prueba de caja transparente. |
| Es el que menos tiempo consume. | Es el que más tiempo consume. |
| No es adecuado o preferido para la prueba de algoritmos. | Es adecuado para la prueba de algoritmos. |
| Se puede hacer por métodos y métodos de prueba y error. | Los dominios de datos junto con los límites internos o internos se pueden probar mejor. |
| **Ejemplo:** buscar algo en google usando palabras clave | **Ejemplo:** por entrada para comprobar y verificar bucles |
| **Tipos de pruebas de caja negra:**    * A. Pruebas funcionales * B. Pruebas no funcionales * C. Pruebas de regresión | **Tipos de pruebas de caja blanca:**    * A. Prueba de ruta * B. Prueba de bucle * C. Pruebas de condición |

**¿Qué es el mock testing?**

Mocking significa crear una versión falsa de un servicio externo o interno que pueda reemplazar al real, esto ayuda a que tus pruebas se ejecuten de manera más rápida y confiable. Cuando tu implementación interactúa con las propiedades de un objeto, en lugar de su función o comportamiento, se puede usar un mock.

**¿Qué es la prueba de stub?**

Tanto Stubbing, como Mocking, significan crear un sustituto, pero un stub solo simula el comportamiento de una parte del objeto. Esto se usa cuando su implementación solo interactúa con un cierto comportamiento del objeto.

Se recomienda el uso de mocking o stubbing cuando el código usa dependencias externas, que pueden ser costosas en tiempo o recursos, como llamadas al sistema o acceder a una base de datos. Usar mocks y stubs para simular la funcionalidad externa nos ayuda a crear pruebas independientes.

**¿Qué es un test fixture?**

Un test fixture es un estado fijo de un conjunto de objetos que se utiliza como base para ejecutar pruebas. El propósito principal de un fixture es garantizar que exista un entorno bien conocido y fijo en el que se ejecuten las pruebas para que los resultados sean repetibles.

Crear un fixture es crear un conjunto de objetos inicializados a ciertos estados.

El propósito de usar Test Fixture es eliminar la duplicación del código común para todos los casos de prueba.

**Work**

* Analizar qué son las pruebas de tipo white box testing
* Analizar qué son las pruebas de tipo black box testing
* Comparar y distinguir las diferencias entre black box y white box testing
* Implementar pruebas de caja negra
* Implementar pruebas de caja blanca

**Postwork**

* Implementar pruebas de caja negra.
* Implementar pruebas de caja blanca.