



Claase 1



Práctica Integradora

Relacione cada principio de testing con su definición.

1	No puede probar que no hay defectos. Reduce la probabilidad de que queden defectos no descubiertos en el software, pero, incluso si no se encuentran, el proceso de prueba no es una demostración de corrección.	La prueba muestra la presencia de defectos no su ausencia
2	No es posible probar todo —todas las combinaciones de entradas y precondiciones—, excepto en casos triviales. En lugar de intentar realizar pruebas exhaustivas se deberían utilizar el análisis de riesgos, las técnicas de prueba y las prioridades para centrar los esfuerzos de prueba.	La prueba exhaustiva es imposible
3	Para detectar defectos de forma temprana, las actividades de testing, tanto estáticas como dinámicas, deben iniciarse lo antes posible en el ciclo de vida de desarrollo de software para ayudar a reducir o eliminar cambios costosos.	La prueba temprana ahorra tiempo y dinero
4	En general, un pequeño número de módulos contiene la mayoría de los defectos descubiertos durante la prueba previa al lanzamiento, o es responsable de la mayoría de los fallos operativos.	Los defectos se agrupan
5	Si las mismas pruebas se repiten una y otra vez, eventualmente estas pruebas ya no encontrarán ningún defecto nuevo. Para detectarlo, es posible que sea necesario cambiar las pruebas y los datos de prueba existentes.	Cuidado con la prueba del pesticida
6	Por ejemplo, el software de control industrial de seguridad crítica se prueba de forma diferente a una aplicación móvil de comercio electrónico.	La prueba se realiza de manera diferente según el contexto
7	El éxito de un sistema no solo depende de encontrar errores y corregirlos hasta que desaparezcan ya que puede no haber errores, pero sí otros problemas. Existen otras variables a tener en cuenta al momento de medir el éxito.	La ausencia de errores es una falacia

Leer atentamente las siguientes consignas, y realizar los ejercicios de acuerdo a lo solicitado para el siguiente sistema. Enviar la resolución del parcial a este [formulario](#) con la siguiente nomenclatura: [Apellido, Nombre - DD/MM/YYYY]

No se aceptarán links de Drive, sólo documentos adjuntos (máximo 5).
Caso contrario, el examen no será considerado para su corrección.

Nota aclaratoria: al enviar el parcial, esperar la confirmación del profesor **antes de salir de la sala de Zoom** para garantizar que se recibió correctamente para posterior corrección. Caso contrario, no se recibirá la evaluación. **Cada**

pregunta vale 1 (un) punto, de un total de 10 (diez).


Duración: 1 hora 45 minutos.

Parte teórica

1) Mencionar tres (3) principios de Testing y explicarlos

brevemente.

Clase 1

**Certified Tech Developer**
The Ultimate Degree

Práctica Integradora

Relacione cada principio de testing con su definición.		
1	No puede probar que no hay defectos. Reduce la probabilidad de que queden defectos no descubiertos en el software, pero, incluso si no se encuentran, el proceso de prueba no es una demostración de corrección.	La prueba muestra la presencia de defectos no su ausencia
2	No es posible probar todo —todas las combinaciones de entradas y precondiciones—, excepto en casos triviales. En lugar de intentar realizar pruebas exhaustivas se deberían utilizar el análisis de riesgos, las técnicas de prueba y las prioridades para centrar los esfuerzos de prueba.	La prueba exhaustiva es imposible
3	Para detectar defectos de forma temprana, las actividades de testing, tanto estáticas como dinámicas, deben iniciarse lo antes posible en el ciclo de vida de desarrollo de software para ayudar a reducir o eliminar cambios costosos.	La prueba temprana ahorra tiempo y dinero
4	En general, un pequeño número de módulos contiene la mayoría de los defectos descubiertos durante la prueba previa al lanzamiento, o es responsable de la mayoría de los fallos operativos.	Los defectos se agrupan
5	Si las mismas pruebas se repiten una y otra vez, eventualmente estas pruebas ya no encontrarán ningún defecto nuevo. Para detectarlo, es posible que sea necesario cambiar las pruebas y los datos de prueba existentes.	Cuidado con la prueba del pesticida
6	Por ejemplo, el software de control industrial de seguridad crítica se prueba de forma diferente a una aplicación móvil de comercio electrónico.	La prueba se realiza de manera diferente según el contexto
7	El éxito de un sistema no solo depende de encontrar errores y corregirlos hasta que desaparezcan ya que puede no haber errores, pero sí otros problemas. Existen otras variables a tener en cuenta al momento de medir el éxito.	La ausencia de errores es una falacia

1. La prueba muestra la presencia de defectos, no su ausencia:

2) ¿Qué es un caso de prueba? ¿Para qué se utiliza?

- un conjunto de condiciones específicas para probar requerimientos del sistema
- es un documento escrito que proporciona información escrita sobre qué y cómo probar
- Un caso de prueba es un conjunto de acciones que se ejecutan para verificar una característica o funcionalidad particular de una aplicación de software. Es decir, que todas las características de una aplicación de software van a ser representadas por uno o más casos de prueba. Es por ello que este documento es de vital importancia en el mundo de la calidad.

3) Definir brevemente un defecto y quién se responsabiliza por su reporte. ¿Qué diferencia existe con los conceptos de error y falla?

- error: ocasionado por la equivocación de una persona en el código (de tipeo, equivocación de operadores lógicos o matemáticos)
- defecto: se genera por un error del código (problema que tiene) , (manifiesta de ese error en el código) → el responsable de reportar el defecto es el QA
- la falla es la consecuencia que nos da el defecto cuando se ejecuta (se genera cuando ya se ejecuta el programa)

4) ¿A qué gran grupo de pruebas pertenece la prueba de partición de equivalencia? Explique brevemente su uso y aplicación.

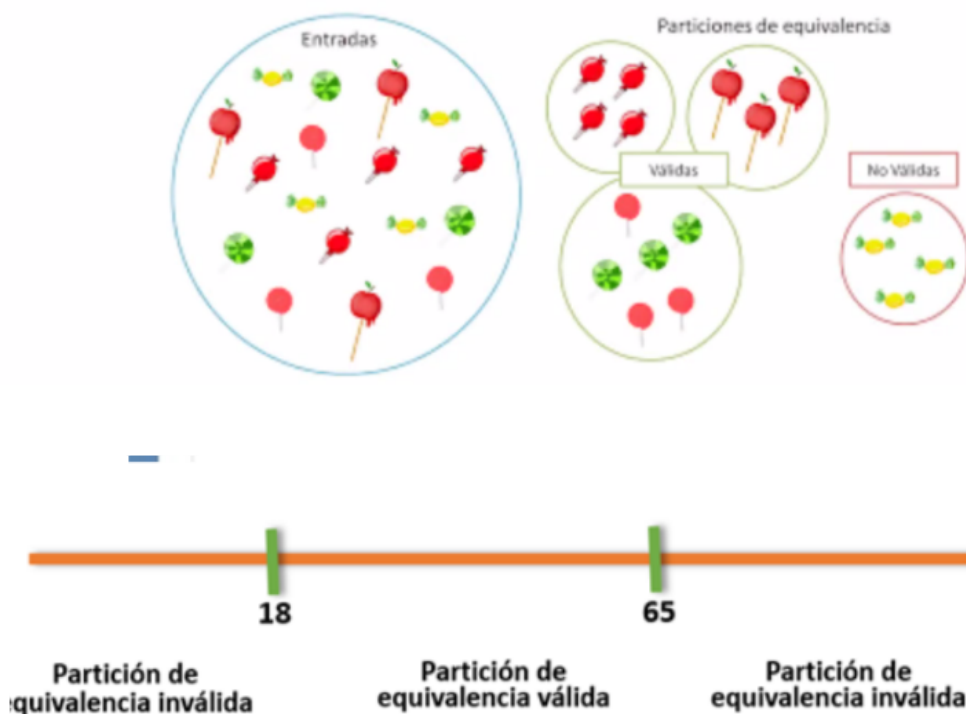


Técnicas de caja negra:

se basan en el comportamiento extraído del análisis de los documentos que son base de prueba (documentos de requisitos formales, casos de uso, historias de usuario, etc). Son aplicables tanto para pruebas funcionales como no funcionales. Se concentran en las entradas y salidas sin tener en cuenta la estructura interna.

Partición de Equivalencia

En esta técnica se dividen los datos en particiones conocidas como **clases de equivalencia** donde cada miembro de estas clases o particiones es procesado de la misma manera.



NOTA : análisis de valores límites es una subsección de partición de equivalencia, donde se analiza el comportamiento

- para la partición de equivalencia se necesita desarrollar un caso mínimo en cada partición

5) ¿Cuál es la diferencia principal entre pruebas estáticas y dinámicas?

pruebas estáticas: leer el código (revisar, analizar, entender, etc)

pruebas dinámicas: ver el sistema (código) en funcionamiento (**ejecutar** el software)

6) Explica brevemente qué pruebas puede realizar en cada ambiente de desarrollo de software.

R/ producción: las pruebas nunca se hacen en producción (hipótesis)

pre-producción: se hacen pruebas (aceptación)

desarrollo: se hacen pruebas

Parte práctica

1

Enunciado (Relevamiento)

Se tiene un software denominado **Comida Ya!** El cual nos permite realizar la compra y venta de productos. El mismo cuenta con las siguientes funcionalidades: un registro (tanto para un usuario administrador y un usuario cliente), un Login, un ABM de productos y pedidos (Alta-Baja-Modificación) y Listar productos y pedidos.

<https://ctd-app-resto.herokuapp.com/admin/dashboard/>

Usted ha sido seleccionado para probar esta app, desde la visión del usuario **CLIENTE** y que cuenta con los siguientes requerimientos mínimos: • El sistema debe permitir registrar un tipo de usuario cliente. Los datos necesarios son: nombre, apellido, email y contraseña. Se debe validar que todos los campos estén completos y tengan el formato correcto. • El sistema debe permitir loguear a los usuarios. En caso de que el email o la contraseña sean incorrectos, se debe mostrar un mensaje de error. • El usuario cliente podrá ver todos los productos disponibles y agregarlos al carrito de compras. Al hacer clic en un producto se debe mostrar un mensaje indicando que el producto fue añadido al carrito.

- El usuario cliente podrá ver su carrito de compras, elegir su forma de pago y confirmar la compra. Si el carrito no posee productos el botón

“confirmar compra” debe aparecer deshabilitado.

- La aplicación debe ser responsive, es decir debe adaptarse a las diferentes resoluciones del navegador hasta llegar a la versión móvil.
- La aplicación debe ser capaz de operar adecuadamente con hasta 300.000 usuarios con sesiones concurrentes.

Consignas

7) Redactar un (1) caso de prueba aplicando partición de equivalencia y otro caso de prueba aplicando la técnica de tabla de decisión. **Para redactar el caso de prueba, debes utilizar el template desarrollado en clase.**

8) Reportar dos (2) defectos del sistema en cualquiera de los menús disponibles. **Para dicho reporte, debes utilizar el template de defectos visto en clases.**

9) Escribir un (1) caso de prueba positivo y un (1) caso de prueba negativo. **No se requiere escribirlos en formato de template.**

10) Redactar brevemente una (1) prueba funcional y una (1) prueba no funcional. **No se requiere escribirlos en formato de template.**

