***PRE-EXAMEN PARCIAL***

**Historia de usuario:**

<https://drive.google.com/file/d/1FUdRUvRYm0pa_Qluycf_oM1yi_2l8yMf/view?usp=sharing>

1. Se pide:
2. Seleccionar un requerimiento y aplicar la técnica ***Tabla de decisiones***. (es posible modificar la tabla)

Debe ser 36 posibles opciones, por temas visuales solo se tomará 8 caso, de los cuales solo 6 son positivos.

| **Condición** | **R1** | **R2** | **R3** | **R4** | **R5** | **R6** | **R7** | **R8** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Seccionar 1 Producto | V | V | V | v | f | f | F |  |
| Seleccionar +1 producto | F | V | V | v | v | v | V |  |
| No Seleccionar Productos | F | F | F | f | f | f | F |  |
| Eliminar 1 producto | V | V | V | f | v | v | F |  |
| Eliminar +1 Producto | F | V | F | v | f | v | V |  |
| No Eliminar productos | F | F | F | f | f | f | F |  |
| A**cción** |  |  |  |  |  |  |  |  |
| Habilitar Botón | V | V | V | v | v | v | V |  |

1. De acuerdo a la técnica aplicada anteriormente, Diseñar un caso de prueba **NEGATIVO** y **POSITIVO**. (utilizar template)

<https://docs.google.com/spreadsheets/d/18J7ZITos0NzN_MvMRaHyjd3A-58Byp4YpxLKwzAWYoM/edit#gid=57923827>

| Relacione cada principio de Testing con su definición. | | |
| --- | --- | --- |
| **1** | 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 paradoja del pesticida |
| **2** | 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 |
| **3** | 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 |
| **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** | 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 |
| **6** | 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 |
| **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. Ejecutando los casos de prueba, se detectó que el resultado real/obtenido no corresponde a lo que indica nuestro producto de trabajo. ¿Qué debe hacer?

* Continuar ejecutando los casos de prueba hasta finalizar el primer ciclo de casos de prueba.
* Reportar un defecto
* Ejecutar el proceso de revisión formal para identificar lo que está sucediendo con el caso.

1. El equipo de desarrollo entregó un defecto que fue reportado por el equipo de QA. ¿Cuál es la primera prueba que se debería ejecutar? Justifique la respuesta.

Prueba de Humo debido a primero se verifica que las funciones básicas del sistema estén funcionando. Esta prueba es rápida y ayuda a identificar problemas graves antes de realizar pruebas más detalladas. Si el sistema no pasa la Prueba de Humo, no tiene sentido ejecutar pruebas más exhaustivas.

Luego se procede a realizar la Prueba de Regresión, con el fin de asegurar que las nuevas modificaciones o correcciones no afecten negativamente a otras partes del sistema.Después de corregir un defecto, es crucial asegurarse de que la corrección no haya introducido nuevos problemas en otras áreas del sistema. La Prueba de Regresión aborda este aspecto.

1. Reportar el siguiente defecto utilizando el témplate.

**Defecto**: Se detectó en el formulario de “datos de contacto” en el campo apellido es posible ingresar ***caracteres especiales.***Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

[***https://docs.google.com/spreadsheets/d/18J7ZITos0NzN\_MvMRaHyjd3A-58Byp4YpxLKwzAWYoM/edit?usp=sharing***](https://docs.google.com/spreadsheets/d/18J7ZITos0NzN_MvMRaHyjd3A-58Byp4YpxLKwzAWYoM/edit?usp=sharing)

1. Las pruebas de regresión: ¿Se debe aplicar en un entorno de STG/Preproducción?

* Verdadero
* Falso.

1. Cuando hablamos de las técnicas de caja blanca, decimos que no es posible a través de esta técnica entender cómo es el funcionamiento interno del flujo.

* Verdadero
* Falso.

1. Ejemplificar una prueba funcional y una prueba no funcional (***SOLO ES NECESARIO EL TITUTLO DE LA PRUEBA)***

***Prueba Funcional:*** Verificación de inicio de sesión con credenciales válidas.

***Prueba No Funcional:*** Prueba de rendimiento de carga de página web bajo alta concurrencia.

1. En el proceso de ejecución de casos de prueba, el líder nos avisa que hubo un cambio de prioridades. Por lo tanto, es necesario entregar al final del día los 10 casos de prueba diseñados. ¿Qué acción usted como tester debe tomar?

* Se debe quedar horas extras para terminar de ejecutar todos los casos de prueba.
* Se debe terminar de ejecutar todos los casos de prueba con prioridad media.
* Se debe ejecutar solo los casos de prueba positivos.
* Se debe ejecutar los casos de prueba definidos con prioridad alta.

1. Nivel de prueba unitario o componentes: En este nivel de pruebas, se ejecutan estos casos de pruebas diseñados con el objetivo de encontrar defectos y que esos defectos no se propaguen a los niveles superiores. Justificar su elección.

* Verdadero: La intención es asegurarse de que los módulos individuales funcionen correctamente y detectar posibles problemas antes de que se propaguen a niveles superiores del sistema.
* Falso.

1. Decimos que es necesario contar con el sistema en funcionamiento o bien desplegado en algún entorno, para poder aplicar lo que denominamos “Pruebas Estáticas”. Justificar su elección.

* Verdadero
* Falso: Las pruebas estáticas no requieren que el sistema esté en funcionamiento o desplegado en algún entorno. Las pruebas estáticas se centran en la revisión y análisis estático del código, la documentación, y otros artefactos sin necesidad de ejecutar el software.