Bien, comenzamos entonces la unidad 3 y vamos a mostrarles, a ver, bien, en esta unidad vamos a aprender cómo se diseñan los casos de prueba. Seguimos viendo casos de prueba, porque como les decía anteriormente, los casos de pruebas son lo más importante que hacemos en nuestro nuestro trabajo diario como testers.

Entonces, ¿qué vamos a repasar en esta unidad? Vamos a ver los principios del testing, un poquito de teoría que tiene que ver con la certificación ISTQB, es una certificación que se puede hacer para certificar como testers y bueno, esa tiene un poco de teoría que volcamos también en este curso. Vamos a ver cuáles son las características de un "test case", vamos a ver, vamos a hablar, digamos sobre el diseño de "test case" y vamos a ver algunos ejemplos. Bien, entonces en este video "Principios del testing".

Son siete principios, el principio uno dice que la prueba muestra la presencia de defectos. La prueba puede mostrar que los defectos están presentes, pero no puede probar que no hay defectos, nosotros no podemos hacer una prueba y después de probar decir: "bueno, no hay errores". No podemos asegurar que hay errores. Podemos asegurar que hay errores si los encontramos y podemos decir también, que no encontramos nosotros más errores, por lo menos, en estos, con estas pruebas que hicimos.

Pero la prueba reduce la probabilidad de los defectos no descubiertos restantes en el software, sí, o sea que reduce la posibilidad de que haya más errores, pero incluso si no se encuentran defectos, no es una prueba de corrección, ¿está bien?, entonces, la prueba no demuestra que no hay errores, demuestra los errores que hay y que no se encontraron más pero no significa que no haya más.

Principio número dos, que la prueba exhaustiva es imposible, o sea, probar todo, todas las combinaciones de entradas y de precondiciones no es posible excepción de casos triviales, o sea, ofrece un sistema muy sencillo únicamente, si no, es imposible probar todas las combinaciones de casos que pueda haber, todos los escenarios posibles, es muy difícil que entonces en vez de la prueba exhaustiva, se usa el riesgo, yo les comentaba en un video anterior, se usa el riesgo y prioridades para enfocar los esfuerzos de prueba. Entonces, uno prueba lo que es más peligroso y prueba lo que es más probable que pase. Está bien, pero se prueba lo más importante, sí, pero no se prueba exhaustivamente porque no se puede.

Bien, principio tres, la prueba temprana. Las actividades de prueba deben comenzar tan pronto como sea posible en el ciclo de vida de desarrollo del software o del sistema, y deben enfocarse en los objetivos definidos, ya que cuanto antes comencemos a probar, esto también lo hemos dicho, cuanto antes comencemos a hacer las pruebas, muchos menos costos va a tener, después el error si yo encuentro un error en forma temprana es más barato, si lo encuentro cuando ya está en producción es un error caro.

Bien, después el tema del agrupamiento de defectos. Se dice que un número pequeño de módulos, contiene la mayoría de los defectos y ¿esto es porque es así?, bueno, porque hay un número pequeño de módulos que son los que tienen las reglas de negocio más