Resumen extraído de "El arte de probar el Software", **por Glenford J. Myers - Ed. El Ateneo**

El arte de probar el Software

Se parte de la idea de que se usa con frecuencia una definición totalmente incorrecta de la palabra probar, y ésta es la causa principal de que las pruebas resulten deficientes.

Malas definiciones

"Probar es demostrar que no hay errores presentes en un programa"

"El propósito de probar es mostrar que el programa realiza correctamente las funciones esperadas"

"Probar es el proceso que lleva a confiar en que un programa hará lo que se supone que debe hacer"

Cuando se prueba un programa se desea agregarle algún valor (es decir, puesto que la prueba es una actividad costosa, se desea recuperar parte de este costo por medio de un incremento del valor del programa). Agregar valor significa aumentar su calidad o confiabilidad, lo que, a su vez, significa encontrar y eliminar errores.

"Prueba es el proceso de ejecutar un programa con la intención de encontrar errores."

"Un buen caso de prueba es el que tiene una probabilidad elevada de encontrar un error aún no descubierto."

"Un caso de prueba exitoso es el que descubre un error aún encubierto."

PRINCIPIOS DE PRUEBA

La definición del resultado esperado a la salida del programa es una parte integrante y necesaria de un caso de prueba

Es uno de los errores más frecuentes en prueba de programas. Este es un hecho basado en la psicología humana. Existe la posibilidad de que un resultado plausible pero erróneo se interprete como correcto por efecto del fenómeno que se describe por "el ojo ve lo que quiere ver".

Por ello un caso de prueba debe constar de dos componentes: una descripción de los datos de entrada al programa y una descripción precisa de la salida esperada para esos datos de prueba. ("Si no hay expectativas, no puede haber sorpresas" Copi).

Un programador debe evitar probar su propio programa

La prueba de programas es un proceso destructivo, es extremadamente difícil para un programador que ha sido constructivo durante todo el tiempo de escritura de un programa cambiar súbitamente y adoptar una actitud totalmente destructiva con respecto a su mismo programa. Además, el programa puede contener errores debidos a falsa interpretación por parte del programador del enunciado del problema o de la especificación del programa.

Una empresa de programación no debería probar sus propios programas

Una empresa programadora de un proyecto es, en muchos sentidos, un organismo vivo con problemas psicológicos parecidos a lo visto en el concepto anterior.

Inspeccionar concienzudamente el resultado de cada prueba

Probablemente sea el principio más obvio, pero parece ocurrir que un importante número de errores que son puestos de manifiesto casualmente son errores expuestos previamente por casos de prueba pero que escaparon a la detección debido a fallas en la inspección cuidadosa de los resultados de esos casos previos de prueba.

Los casos de prueba deben ser escritos tanto para condiciones de entrada inválidas e inesperadas como para condiciones válidas y esperadas

Los casos de prueba que representan condiciones de entrada inválidas e inesperadas parecen tener una mayor facilidad para detectar errores que los casos preparados para condiciones de entrada válidas. Además, muchos errores descubiertos repentinamente en programas de producción aparecen cuando se utilizan esos programas en una forma nueva o "inesperada".

Examinar un programa para comprobar que no hace lo que se supone que debe hacer es solo la mitad del problema. La otra mitad consiste en ver si el programa hace lo que no se supone que debe hacer

Los programas deben ser examinados con respecto a efectos laterales indeseados.

Evite los casos de prueba desechables a menos que el programa sea verdaderamente un programa desechable

Es muy común sentarse ante un terminal e inventar casos de prueba "al vuelo" y pasar estos casos a través del programa. El mayor problema es que los casos de prueba representan una inversión valiosa que, en este caso, se pierde después que la prueba se ha dado por terminada. Si este caso de prueba ha producido un error, una vez corregido este, habría que inventar un nuevo caso de prueba (tiempo valioso) que rara vez es tan riguroso como la prueba original, lo que significa que si la modificación causa un error en una parte del programa que anteriormente era correcta, este error a menudo pasa desapercibido.

No planear un esfuerzo de prueba con la suposición tácita de que no se encontrarán errores

Es cometido a menudo por jefes de proyecto y es una indicación del uso de una definición incorrecta de prueba.

La probabilidad de encontrar errores adicionales en una sección del programa es proporcional al número de errores ya encontrados en esa misma sección

Es un fenómeno que se ha observado en muchos programas; los errores parecen presentarse en grupos, y en un programa típico, algunas secciones son mucho más propensas a tener errores que otras. Este fenómeno sugiere que los esfuerzos adicionales para las pruebas deberán dirigirse con preferencia a esa sección del programa.

Las pruebas constituyen una tarea altamente creativa y son un desafío intelectual

Existen métodos que permiten elaborar un conjunto razonable de casos de prueba para un programa, pero tales métodos requieren un grado considerable de creatividad. Probablemente sea cierto que la creatividad requerida para probar programas extensos supera la necesaria para concebirlos.