

# Prueba de la unidad

## **1. PRUEBA DE LA UNIDAD.**

Las pruebas de la unidad centran su proceso de verificación en la menor unidad de software, el módulo. Una unidad de programa es lo suficientemente pequeña como para que el programador pueda probarla minuciosamente tomando como guía el diseño detallado.

A este nivel las pruebas más importantes son del tipo de la caja blanca. Se denominan de la caja blanca a aquellas pruebas que examinan el código al más bajo nivel, es decir, aseguran que las operaciones internas se ajustan a las especificaciones, aunque también se llevan a cabo algunas del tipo de la caja negra que van orientadas a probar el interfaz, es decir, demuestran que cada función es completamente operativa.

Se realizarán las siguientes pruebas y en el siguiente orden:

### **. Prueba de Interfaz.**

Tratan de asegurar que la información fluye de forma adecuada hacia y desde la unidad de programa.

Para ello haremos las siguientes comprobaciones:

El número, tipo, modo y orden de los parámetros formales de la declaración debe ser igual a los reales de la llamada.

Comprobar la consistencia de las variables globales entre los módulos.

Comprobar que las restricciones entre los módulos se pasan como parámetros, no como variables globales.

Además, si en el módulo se trabaja con ficheros, habrá de comprobarse si los atributos del fichero son correctos, si se manejan errores de E/S y si son correctas las aperturas y cierres.

### **. Pruebas de estructuras de datos locales.**

Examina las estructuras de datos locales para asegurar que los datos que se mantienen temporalmente en el sistema mantienen su integridad, para ello nos fijaremos en lo siguiente:

Si los tipos de datos son consistentes.

Si los nombres de las variables son consistentes.

Vigilar la inicialización y desbordamiento de las estructuras de datos.

Ver como afectan los datos globales al módulo.

Dentro de este tipo de pruebas hay que destacar las de las cadenas de definición de uso que determinan para cada variable el conjunto de instrucciones en las que se define y usa y esas instrucciones son las que se prueban.

### **. Prueba de los caminos independientes de ejecución.**

Se trata de probar todos los caminos básicos de la estructura de control con el fin de asegurar que todas las sentencias del módulo se ejecutan por lo menos una vez.

Dentro de estos tipos de pruebas destacamos las del grafo de flujo o camino básico que hayan la complejidad del algoritmo y basándose en ella, implementa pruebas para cada uno de los caminos del grafo.

Para hacer estas pruebas nos fijaremos sobre todo en lo siguiente:

Comparaciones incorrectas o flujos de control inapropiados. Dentro de estas destacaremos a las del camino básico antes mencionadas, las de las condiciones y las de los bucles.

Detección de cálculos incorrectos por ejemplo por inicializaciones incorrectas, falta de precisión, mala precedencia entre operadores, mal los símbolos de la expresión, igualdad esperada cuando hay errores de precisión, variables o comparadores incorrectos, terminación de bucles inexistentes o inapropiada, variables de bucles modificadas incorrectamente, .

**. Pruebas para comprobar la manipulación de errores.** Se trata de asegurar que el programa siempre da una respuesta, aunque esta sea un mensaje de error. Habrá que comprobar que se manipulan correctamente cada una de las condiciones de error posibles, para lo cual debemos probar que la descripción del error se entienda, sea lo suficientemente significativa para averiguar

la causa, se corresponda con lo que es y se de cuando se produzca.

**. Prueba de los límites.**

Para asegurar que el módulo funciona correctamente en los límites establecidos como restricciones del procesamiento. Se trata de probar los casos que ejerciten las estructuras de datos en los límites de sus dimensiones, el flujo de control y los valores de los datos por debajo de los máximos y mínimos permitidos.

Dentro de este tipo de pruebas están algunas pruebas de la caja negra como las de partición equivalente y análisis de valores límite.