Principios del Testing

 Un programador debería evitar probar su propio código.

- Una unidad de programación no debería probar sus propios desarrollos.

- Examinar el software para probar que no hace lo que se supone que debería hacer es la mitad de la batalla, la otra mitad es ver que hace lo que no se supone que debería hacer.

- No planificar el esfuerzo de testing sobre la suposición de que no se van a encontrar defectos.

- El testing es una tarea extremadamente creativa e intelectualmente desafiante.

- Testing temprano. Cuando se hacen los requerimientos es el momento de comenzar con los casos de prueba. El objetivo es verificar que el requerimiento es testeable, legible y evito que llegue a desarrollo con errores.

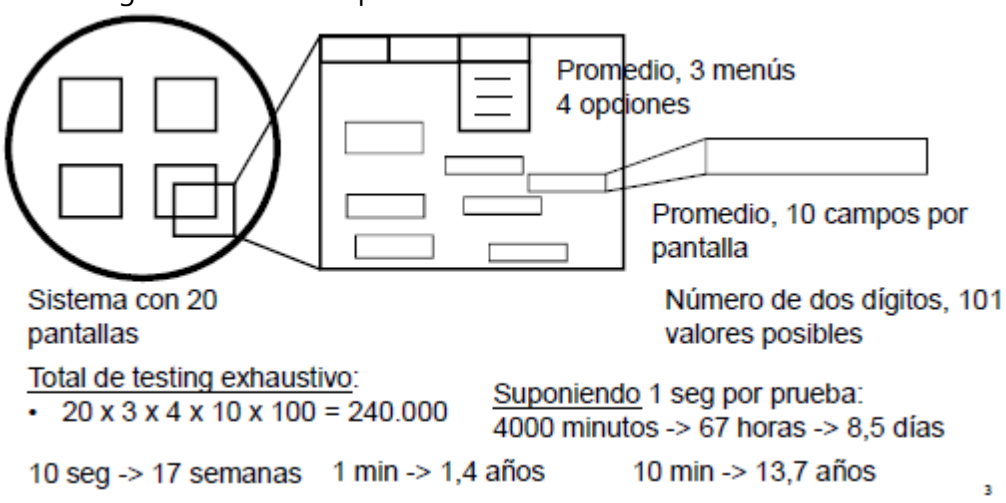
- Los test cases deben ser actualizados y rediseñados, para evitar la paradoja del pesticida (Cada método que utilices para prevenir o encontrar errores deja un residuo de errores sutiles contra los que esos métodos resultan ineficaces).

- El testing es dependiente del contexto, para lo cual hay q evaluar los métodos, costos riesgos y recursos. Los riesgos se utilizan para priorizar casos de prueba.

- Falacia sobre la ausencia de errores. El testing es necesario porque la existencia de errores en el software es inevitable.

¿Cuánto testing es suficiente?

 El testing exhaustivo es imposible.



- Decidir cuánto testing es suficiente depende de:
  - Evaluación del nivel de riesgo
  - Costos asociados al proyecto

- Usamos los riesgos para determinar:
  - Que testear primero
  - A qué dedicarle más esfuerzo de testing
  - Que no testear (por ahora)

Criterios de Aceptación

El criterio de aceptación es lo que comúnmente se usa para resolver el problema de determinar cuándo una determinada fase de testing ha sido completada. Estos pueden estar definidos en término de:

- Costos

- % de tests corridos sin fallas
- Fallas predichas que aún permanecen en el software
- No hay defectos de una determinada severidad en el software.
- Pasa exitosamente el conjunto de pruebas diseñado y la cobertura estructural.
- Good Enough: Cierta cantidad de fallas no críticas es aceptable.
- Defectos detectados es similar a la cantidad de defectos estimados.