



INGENIERÍA EN DESARROLLO DE SOFTWARE
ASIGNATURA: PRUEBAS Y MANTENIMIENTO DE SISTEMAS DE
SOFTWARE
DOCENTE: RICARDO RODRÍGUEZ NIEVES
ALUMNO: VÍCTOR DAVID VALDEZ GUERRERO
MATRICULA: ES1611317668

CASO DE ESTUDIO:

UNIDAD 2 ACTIVIDAD 2 TÉCNICAS DE PRUEBAS DE SOFTWARE



Elabora y ejecuta los casos de prueba de tres de las técnicas de prueba de caja blanca:

- Cobertura del camino básico
- Cobertura de sentencias
- Cobertura de decisión
- Cobertura de condición
- Cobertura decisión/condición
- Cobertura de condición múltiple (debes elegir tres para desarrollar)

Para el siguiente algoritmo

Inicio

Leer N

Si $N > 60$ entonces

 Imprimir Aprobado

Si no

Si $N < 60$ entonces

 Imprimir No aprobado

Si no Imprimir Número Cero

Fin Si

Fin Si

Primero identificamos el código fuente con base a la importancia del proceso a evaluar:
En este caso lo codifiqué en php.

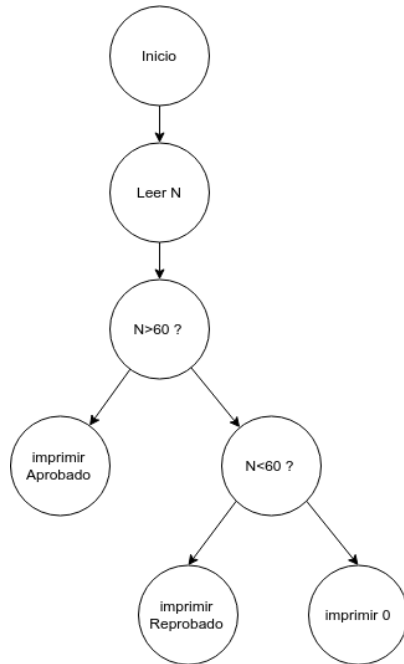
```
<?php
$calificacion=50;

if ($calificacion>60)
{echo "Aprobado";}
else
{
    if ($calificacion<60)
    {echo "Reprobado";}
    else
    {echo "0";}
}

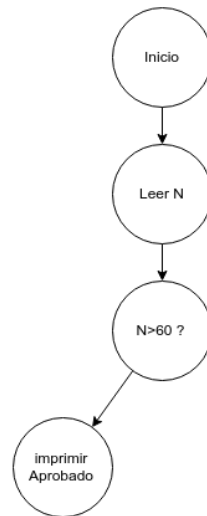
?>
```



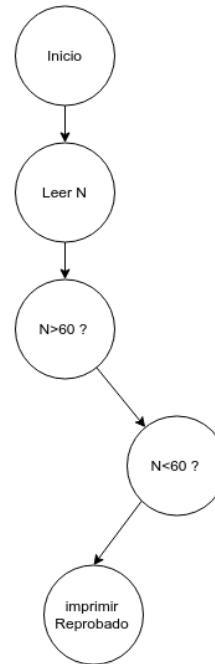
Se dibuja el grafo



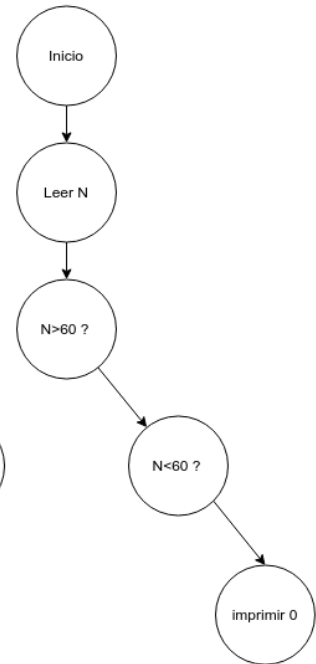
General



Caso 1



Caso 2



Caso 3

Calcular la complejidad ciclomática del grafo.

$$v(G) = a - n + 2$$

Dónde:

a: número de arcos.

n: número de nodos.

v: vértices.

(G): grafo

Soluciones aplicando la fórmula:

Para el primer caso:

$$v(G) = 3 - 4 + 2 = 1$$

Para el segundo caso:

$$v(G) = 6 - 5 + 2 = 1$$

Se determina un conjunto básico de caminos independientes. Los caminos independientes para el caso 1 son: 1,2,3,4 y 1,2,3,5,6 y 1,2,3,5,7



Se diseña el caso de prueba Aprobado para el primer caso, y Reprobado para el segundo caso:

| Numero de caso de prueba | Escenario | Condición | Resultado Esperado |
|--------------------------|-----------|-----------------|--------------------|
| 1 | Ecenario1 | N es mayor a 60 | Aprobado |
| 1 | Ecenario1 | N es menor a 60 | Reprobado |

Cobertura de sentencias

Para estas pruebas se usa la siguiente formula:

$$\text{Cobertura de sentencias} = \frac{\text{Número de sentencias ejercitadas}}{\text{Total de número de sentencias}} * 100\%$$

Para ejecutar la cobertura de sentencia se puede utilizar software (testware), pero para ejecutarla de manera manual se siguen los siguientes lineamientos:

- cada línea se considera una sentencia.
- una sentencia puede ser una sola línea o extenderse en varias.
- una línea puede contener más de una sentencia, solo una, o una parte.
- las sentencias pueden contener otras sentencias dentro de ellas.

Para comenzar primero enumeramos cada línea del código para considerar a cada una como sentencia:

```
1  <?php $calificacion=100;  
2  if ($calificacion>60) {echo "Aprobado";}   
3  else   
4  {   
5      if ($calificacion<60) {echo "Reprobado";}   
6      else {echo "0"; } } ?>
```

Como el editor ya me enumera las líneas de código lo único que hago quitar los espacios para que cada línea sea una sentencia.

Después se le asignan valores arbitrarios a la variable de entrada “Calificacion”

Conjunto de prueba 1

Prueba 1_4: entero = 50, Imprime Reprobado

Prueba 1_5: entero = 100, Imprime Aprobado

Se cubren 3 de las 6 sentencias por lo que la cobertura es del 50% con tres pruebas $3/6=5.3*100=50\%$

Cobertura de decisión

La cobertura de decisión requiere que en las ramas se contemple verdadera y falsa y se requiere lograr una cobertura al 100%

La fórmula es:

$$\text{Cobertura de decisión} = \frac{\text{Número de salidas de decisiones ejercitadas}}{\text{Número total de de decisiones de salida}} * 100\%$$



Enumeramos el código y consideramos cada línea como sentencia.

```
1 <?php $calificacion=100;  
2 if ($calificacion>60) {echo "Aprobado";}   
3 else   
4 {   
5     if ($calificacion<60) {echo "Reprobado";}   
6     else {echo "0";} } ?>
```

Se plantean el conjunto de pruebas 1, asignando valores arbitrarios a la variable “calificacion”

1_1: entero = 50.

1_2: entero = 100.

Se identifica si la condición es verdadera o falsa:

Resultado Falso

En la prueba 1_1 el valor de “calificacion” será de 50. La condición de $calificacion > 60$ es falsa y entra el segundo if, con la condición de $calificacion < 60$, la cual es verdadera y se imprime “Reprobado”

Resultado Verdadero

En la prueba 1_2 el valor de “calificacion” será de 100. La condición de $calificacion > 60$ es verdadero y se imprime Aprobado

La prueba cubre el 100% de cobertura de sentencia, y el resultado de la decisión verdadero al 50%

Cobertura de decisión = $\frac{1}{2} \times 100 = 50\%$

Cobertura de condición.

En esta prueba se requiere que cada estado individual de cada sentencia de decisión sea probado como verdadero o falso.

Para garantizar los criterios de cobertura de condición entero, i, y suma deben ser evaluados al menos una vez

La condición dice que si i es verdadera suma lo es también.

Se asignan valores de verdadero, falso y no evaluado a las variables, entero, i, y suma, una vez cada expresión booleana.

| Número de caso de prueba | entero | i | Mayor que 60 | condición |
|--------------------------|-----------|-------------|--------------|-----------|
| 1_1 | VERDADERO | NO EVALUADO | FALSO | FALSO |
| 1_2 | FALSO | VERDADERO | VERDADERO | VERDADERO |



Desarrolla una de las técnicas de prueba de caja negra

Abrimos el navegador (haciendo previamente una pequeña interfaz web para estos fines)

Las clases de equivalencia (que esperamos cubrir)son las siguientes:



- 1.Calificación numérica(válido)
- 2.Calificación alfabética(no válido)
- 3.Calificación mayor a 100 (no válido)
- 4.Calificación menor a 0 (no válido)

Una vez listadas las clases de equivalencia, se agrupan en una tabla como la siguiente para mostrar los casos de prueba y lo que se espera

Se validarán los siguientes casos:

| Numero de caso de prueba | Datos de prueba | Salida Esperada | Clases cubiertas |
|--------------------------|-----------------|-----------------|------------------|
| 1 | 101 | F | 1,3 |
| 2 | -5 | F | 1,4 |
| 3 | 55 | V | 1 |
| 4 | 99 | V | 1 |

No permitir valores fuera del rango

| | |
|---|--|
| <p>Capture calificación: <input type="text" value="101"/> <input type="button" value="Enviar"/></p> <p> El valor debe ser inferior o igual a 100</p> | <p>Capture calificación: <input type="text" value="-5"/> <input type="button" value="Enviar"/></p> <p> El valor debe ser superior o igual a 0</p> |
|---|--|



No permitir valores nulos

Capture calificación: Enviar

! Completa este campo

Caso Falso

Capture calificación: 55 Enviar

Reprobado

Caso Verdadero

Capture calificación: 99 Enviar

Aprobado

Conclusión

Las diferentes técnicas de detección de errores que hemos visto nos ayudan a detectar errores en nuestros módulos de código para así poder corregirlos, pero esto no garantiza que todos los errores sean detectados por lo que es importantes someter nuestros módulos a más de una prueba para asegurarnos que el código se encuentra en óptimas condiciones y cuenta con la calidad específica para la que fue desarrollado.

Referencias

- UNADM Unidad 2. Pruebas de sistema de Software (2019)
[https://unadmexico.blackboard.com/webapps/blackboard/execute/modulepage/view?course_id= 65090_1&cmp_tab_id= 124047_1&mode=view](https://unadmexico.blackboard.com/webapps/blackboard/execute/modulepage/view?course_id=65090_1&cmp_tab_id=124047_1&mode=view)