Universidad Andrés Bello®
Conectar · Innovar · Liderar

# ANÁLISIS DINÁMICO

## TÉCNICAS DE CAJA NEGRA Y CAJA BLANCA

Sara González Catalán
sarit.gonzalez@uandresbello.edu.

# CALIDAD DENTRO DEL DESARROLLO DE SOFWARE

Universidad Andrés Bello

- El principal objetivo del desarrollo de software es conseguir productos de **alta calidad**.
- La calidad involucra **confiabilidad, mantenibilidad, interoperabilidad, usabilidad** entre otros atributos.
- La confiabilidad es uno de los atributos de calidad más importantes. *¿Porqué crees que digo eso?*
- Una de las métricas de calidad más utilizada es la cantidad de defectos por líneas de código. *¿Esta frase tiene relación con confiabilidad?*

*Mayor cantidad de defectos hace al software menos confiable*

# Familias de Técnicas de pruebas

| | | |
|---|---|---|
| Criterios de adecuación | Conocimiento de la implementación | - Caja Blanca<br>- Caja Negra |
| | Enfoque | - Estructurales<br>- Basados en faltas<br>- Basados en errores |

# Clasificación de Técnicas de pruebas

| Implementación / Enfoque | Caja Blanca | Caja Negra |
|---|---|---|
| **Estructurales** | Flujo de control<br>Flujo de datos | |
| **Basadas en faltas** | Mutación | |
| **Basadas en errores** | | Funcionales<br>Aleatorias |

# Análisis Dinámico

- Dado que, la ejecución  del programa que exponen posibles fallas este análisis observa comportamiento del programa y concluye acerca de la calidad del sistema.
- Para que el análisis dinámico resulte efectivo el programa a ser analizado se debe ejecutar con los suficientes casos de prueba como para producir un comportamiento interesante.
- Considerando que las pruebas exhaustivas no son viables, es necesario seleccionar sobre el universo de entradas al sistema aquellas que ayuden a predecir la calidad del software con mayor exactitud por ello nace la necesidad de técnicas que ayuden a la selección de casos de prueba

# Análisis Dinámico: Principios Básicos

Universidad Andrés Bello

- Su objetivo es descubrir bugs.
- Caso de prueba "Adecuado", es aquel que tiene altas probabilidades de detectar un bug.
- Un caso de prueba debe tener la descripción del resultado esperado.
- Los casos de prueba se generan para condiciones de entradas válidas e inválidas.
- Un programador debe evitar testear su propio programa.
- El test completo es casi imposible.

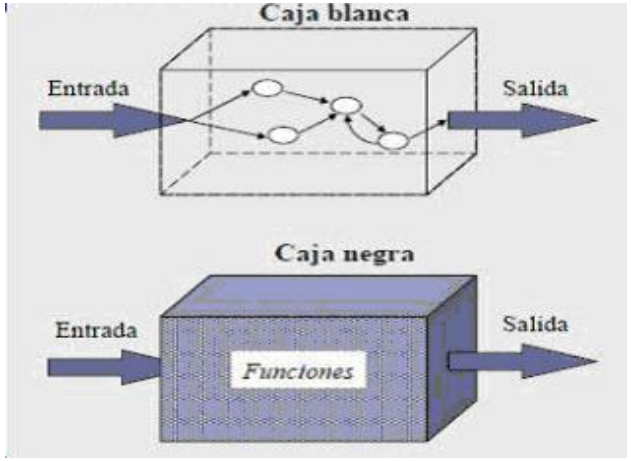
# Técnicas Pruebas Dinámicas

- ✓ **Caja Negra**

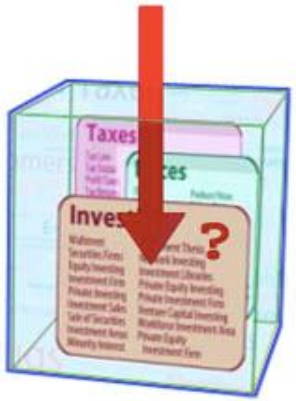
  Funcionalidad de los objetos a prueba

- ✓ **Caja Blanca**

  Estructura interna de los objetos a prueba





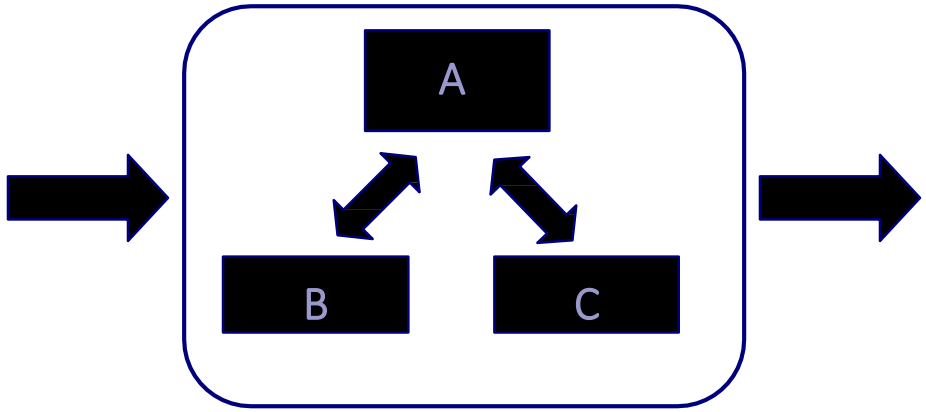


# Técnicas Pruebas Dinámicas: Caja Blanca

Para la creación de los casos de prueba se examina la estructura y la lógica interna del programa.

**Comprende las técnicas de:**

1. Prueba de caminos posibles. Flujos de control
2. Prueba de estructuras de datos. Flujo de datos
3. Prueba de decisiones y estructuras lógicas. Flujos de control



# Técnicas Pruebas Dinámicas: Caja Blanca

*Grado en que los casos de prueba cubren la lógica del programa (código fuente)*

Existen diferentes caminos:

- IF – ELSE
- DO WHILE
- DO UNTIL.

} *Cubrirlos todos es impráctico*

***Ejemplo:*** Número de caminos posibles para un módulo que contiene 10 sentencias condicionales simples implica aproximadamente

$2^{10}$ es 1024 caminos ➔ al menos 1024 casos.

# Técnicas Pruebas Dinámicas: Caja Blanca

Universidad Andrés Bello

***Idea:***

No es posible probar todos los caminos de ejecución distintos, pero sí *confeccionar un conjunto de casos de prueba* que garanticen que se *verifican todos los caminos de ejecución* llamados ***independientes.***

# Técnicas Pruebas Dinámicas: Caja Blanca

Universidad Andrés Bello

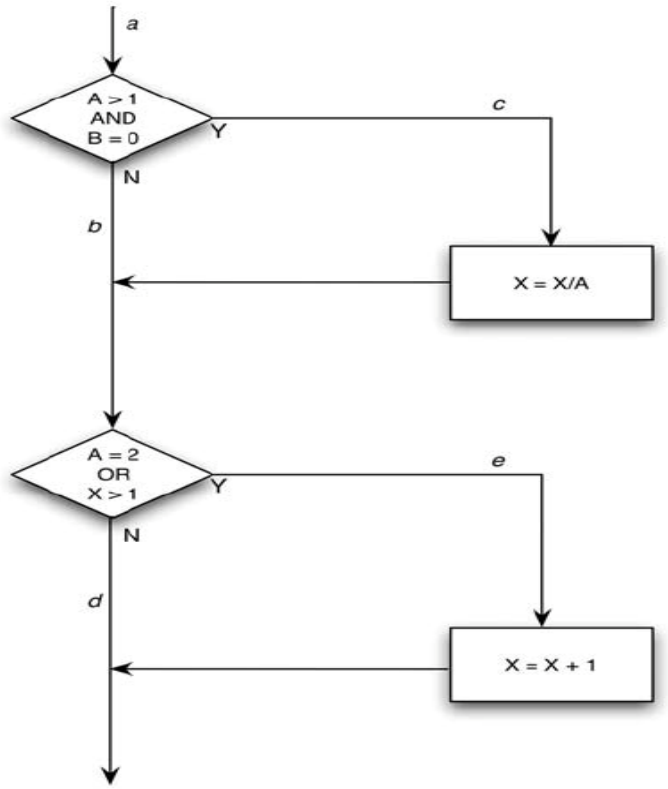
## CRITERIOS DE COBERTURA

1. **Cobertura de Sentencias:** Cada sentencia o instrucción del programa se ejecute al menos una vez.
2. **Cobertura de Decisiones:** Cada decisión tenga, por lo menos una vez, un resultado verdadero y, al menos una vez, uno falso.
3. **Cobertura de Condiciones:** Cada condición de cada decisión adopte el valor verdadero al menos una vez y el falso al menos una vez
4. **Criterio de Decisión/Condición:** Consiste en exigir el criterio de cobertura de condiciones obligando a que se cumpla también el criterio de decisiones.
5. **Criterio de Condición múltiple:** Debe garantizar todas las combinaciones posibles de las condiciones dentro de cada decisión.

# Técnicas Pruebas Dinámicas: Caja Blanca

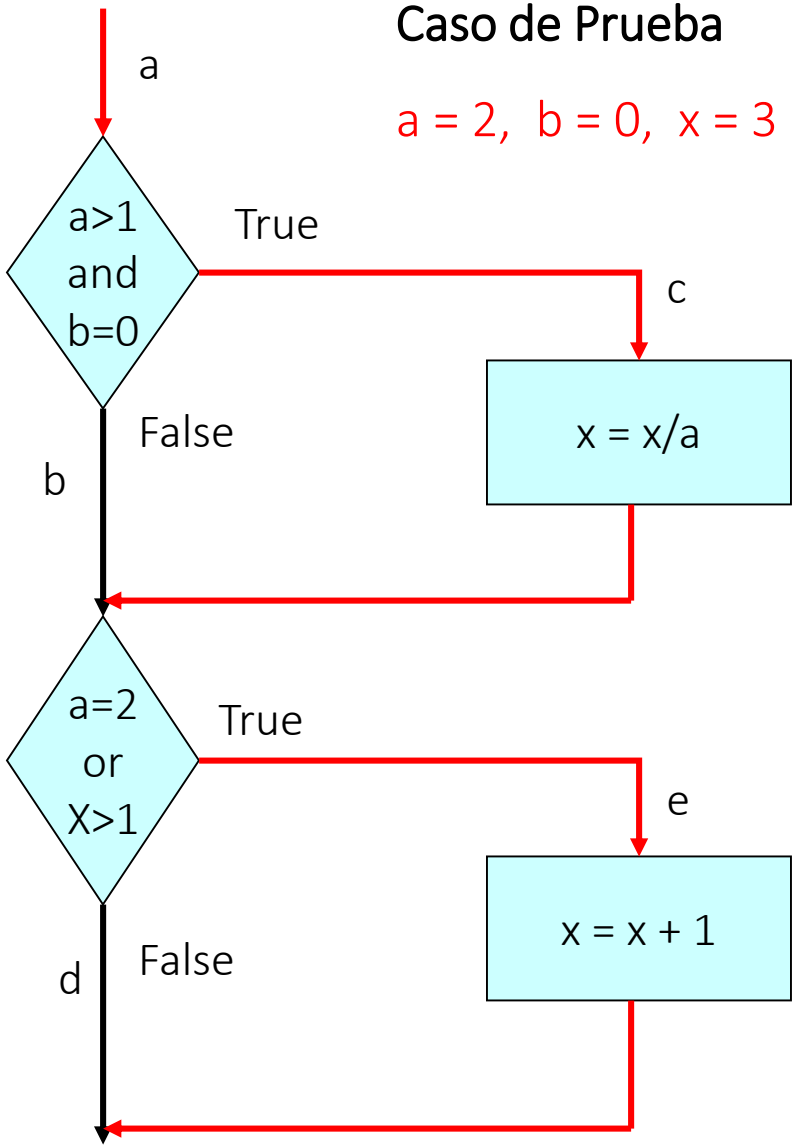
## 1.- TÉCNICA COBERTURA DE SENTENCIAS (CS)

- ✓Se requiere que toda línea sea ejecutada al menos una vez.
- ✓Se utiliza un diagrama de flujo para realizar la acción.
- ✓Cuantos casos de prueba se requieren para cubrir todas las lineas de código



# Técnicas Pruebas Dinámicas: Caja Blanca

## 1.- TÉCNICA COBERTURA DE SENTENCIAS (CS)



# Técnicas Pruebas Dinámicas: Caja Blanca

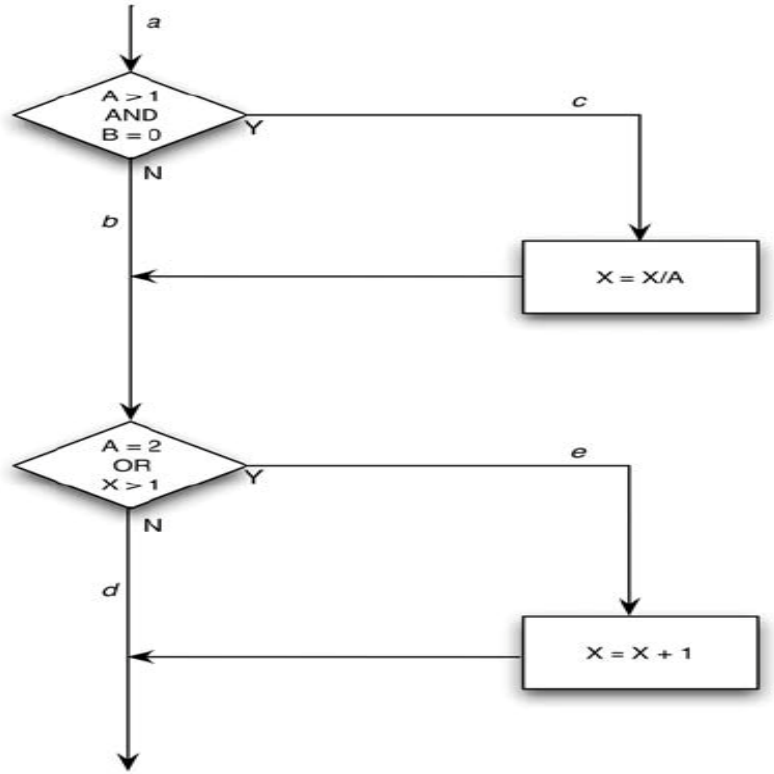
## 2.- CRITERIO DE CUBRIMIENTO DE DECISIÓN

Cada decisión dentro del código toma al menos una vez el valor true y otra vez el valor false para el CCP

```
public void foo(int a, int b, int x) {
    if (a>1 && b==0)  {
        x=x/a;
    }
    if (a==2 || x>1)  {
        x=x+1;
    }

}
```

¿Cuántos casos de prueba se requieren para cubrir todas las líneas de código?



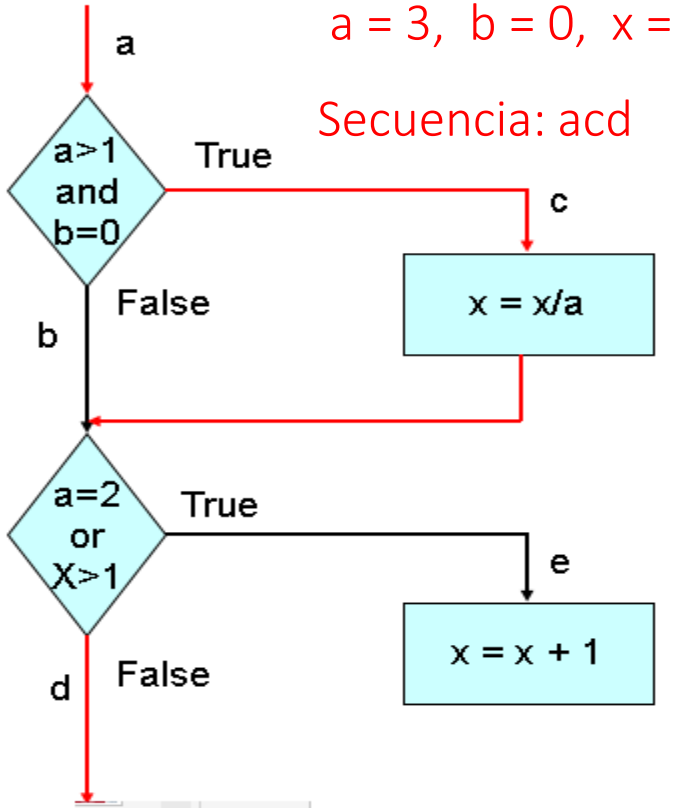
# Técnicas Pruebas Dinámicas: Caja Blanca

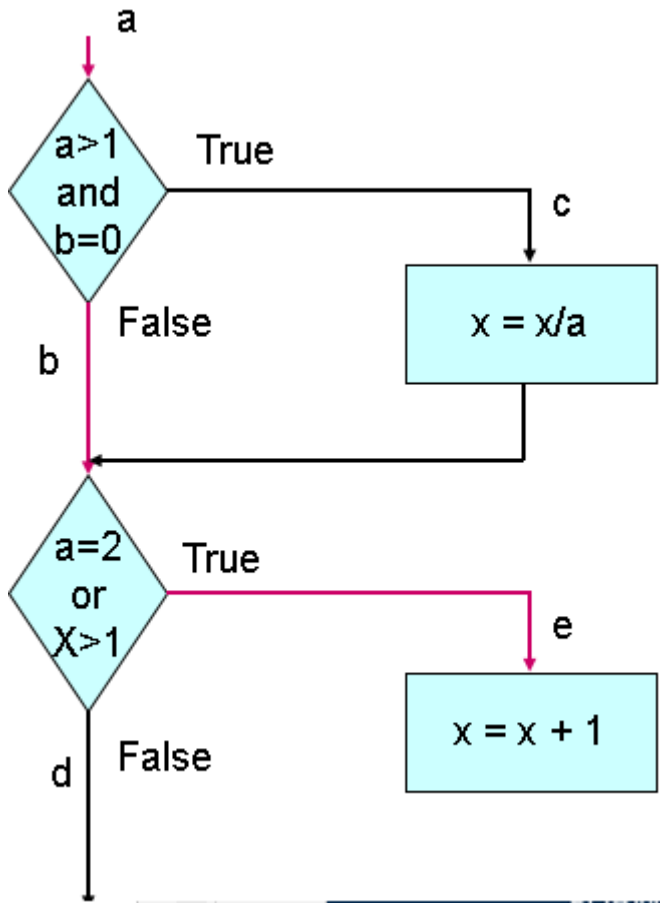
## 2.- CRITERIO DE CUBRIMIENTO DE DECISIÓN

Caso de Prueba 1

a = 3, b = 0, x = 3

Secuencia: acd




Caso de Prueba 2

a = 2, b = 1, x = 1

Secuencia: abe

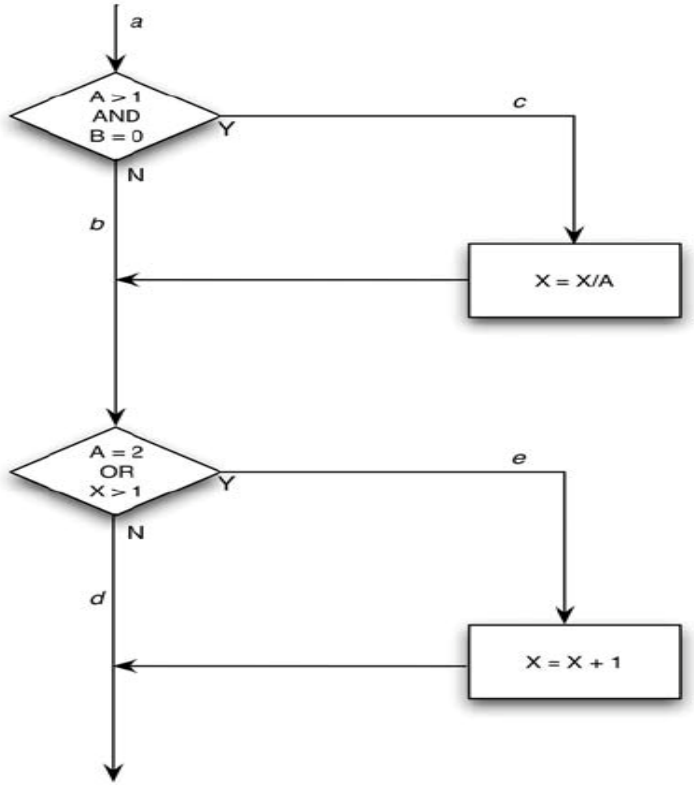
# Técnicas Pruebas Dinámicas: Caja Blanca

## 3.- CRITERIO DE CUBRIMIENTO DE CONDICIÓN

Cada condición dentro de una decisión debe tomar al menos una vez el valor true y otra el false para el CCP

```
public void foo(int a, int b, int x) {
    if (a>1 && b==0)  {
        x=x/a;
    }
    if (a==2 || x>1)  {
        x=x+1;
    }

}
```




¿Cuántos casos de prueba se requieren para cubrir todas las líneas de código?

# Técnicas Pruebas Dinámicas: Caja Blanca

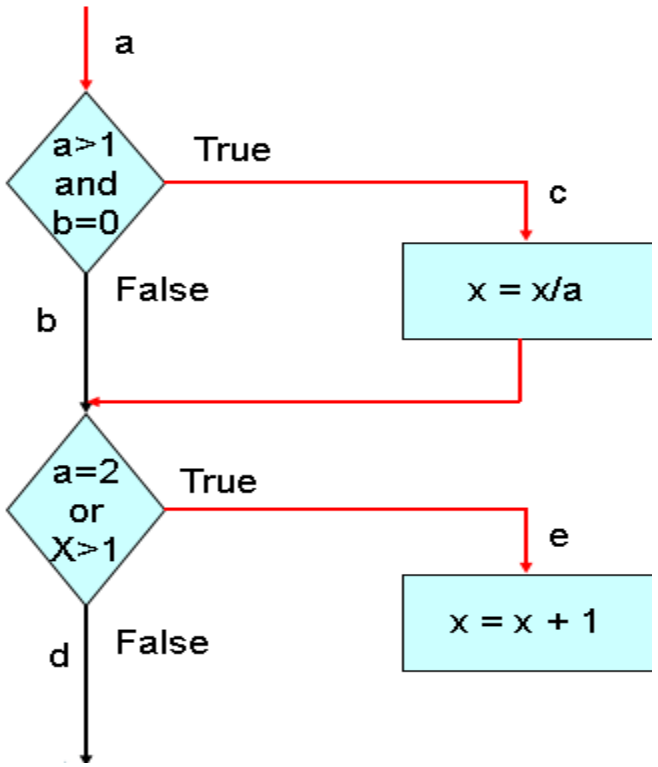
## 3.- CRITERIO DE CUBRIMIENTO DE CONDICIÓN

Caso de Prueba 1

CP1 a = 2, b = 0, x = 4

Secuencia: ace

C1 V
C2 V
C3 V
C4 V



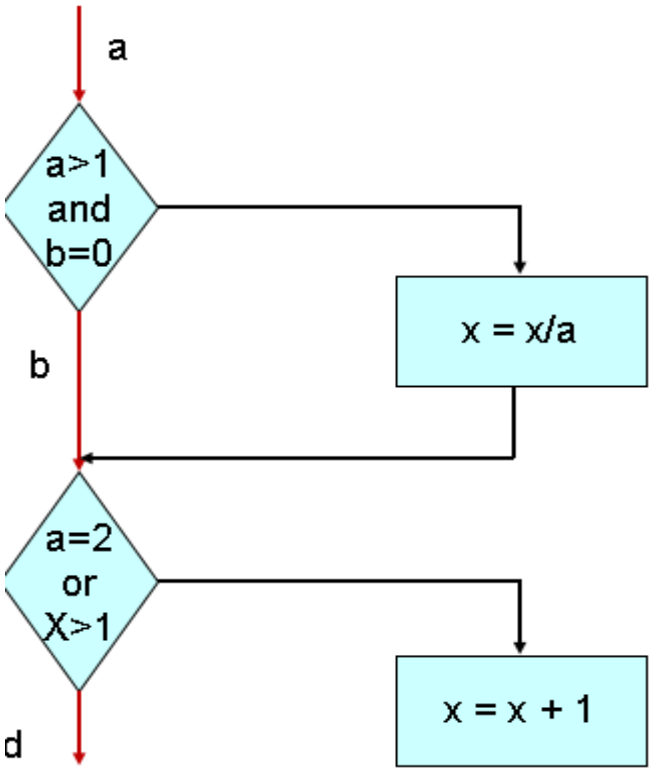

Este criterio es <u>generalmente</u> más fino que el de decisión

Caso de Prueba 2

CP1 a = 1, b = 1, x = 1

Secuencia: abd

C1 F
C2 F
C3 F
C4 F



# Técnicas Pruebas Dinámicas: Caja Blanca

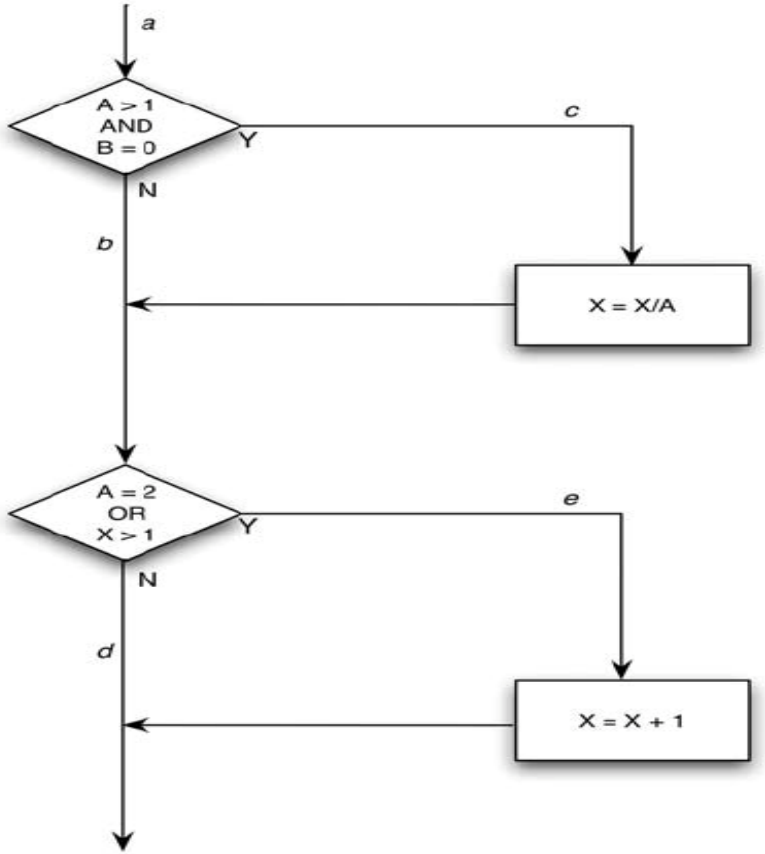
## 4.- CRITERIO DE CUBRIMIENTO DE DECISIÓN/CONDICIÓN

Es la combinación de los dos criterios anteriores

```
public void foo(int a, int b, int x) {
    if (a>1 && b==0)  {
        x=x/a;
    }
    if (a==2 || x>1)  {
        x=x+1;
    }

}
```

¿Cuántos casos de prueba se requieren para cubrir todas las líneas de código?



# Técnicas Pruebas Dinámicas: Caja Blanca

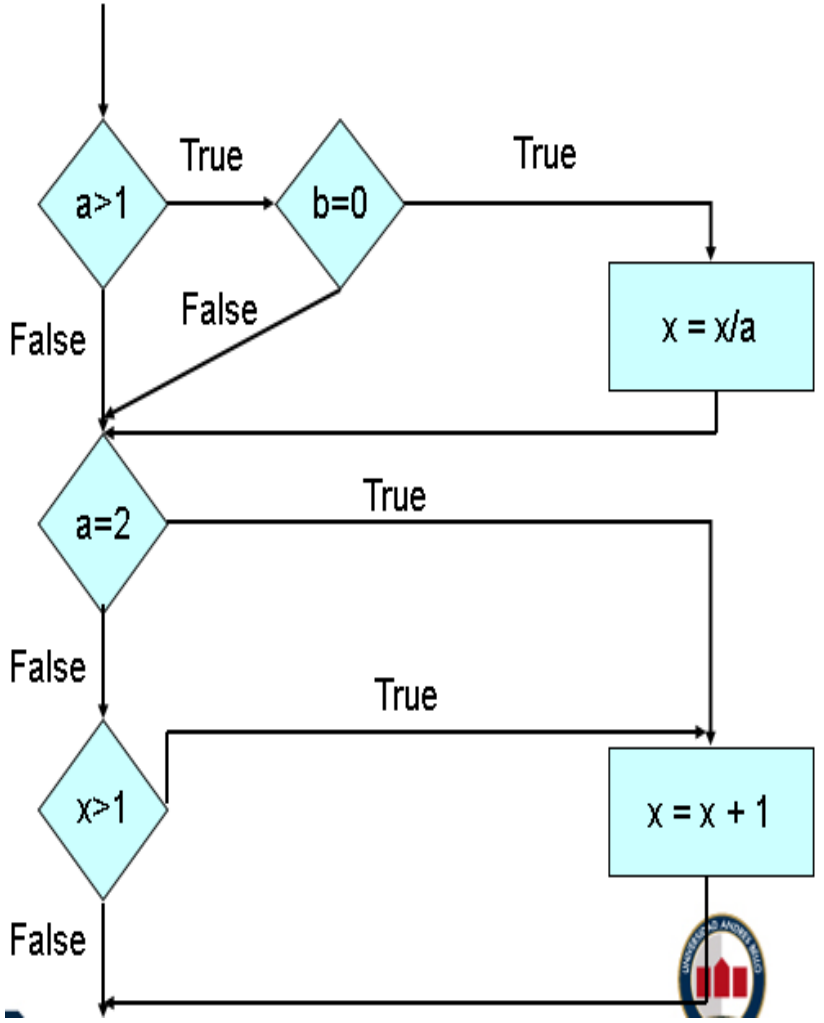
## 4.- CRITERIO DE CUBRIMIENTO DE DECISIÓN/CONDICIÓN

Es la combinación de los dos criterios anteriores

```
public void foo(int a, int b, int x) {
    if (a>1 && b==0)  {
        x=x/a;
    }
    if (a==2 || x>1)  {
        x=x+1;
    }

}
```

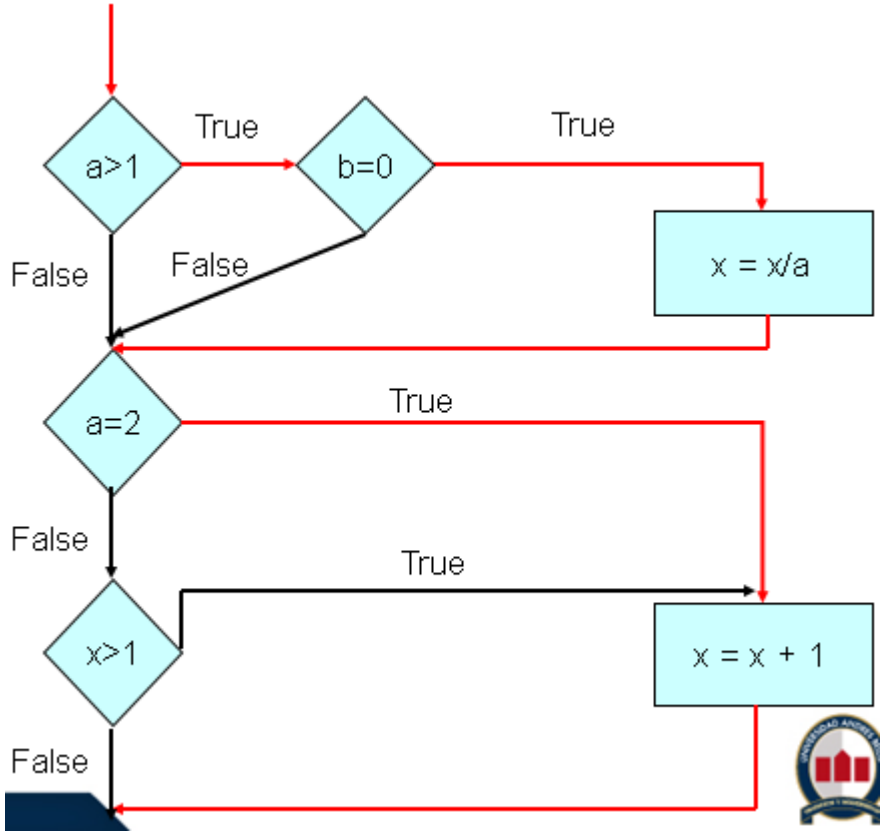
¿Cuántos casos de prueba se requieren para cubrir todas las líneas de código?



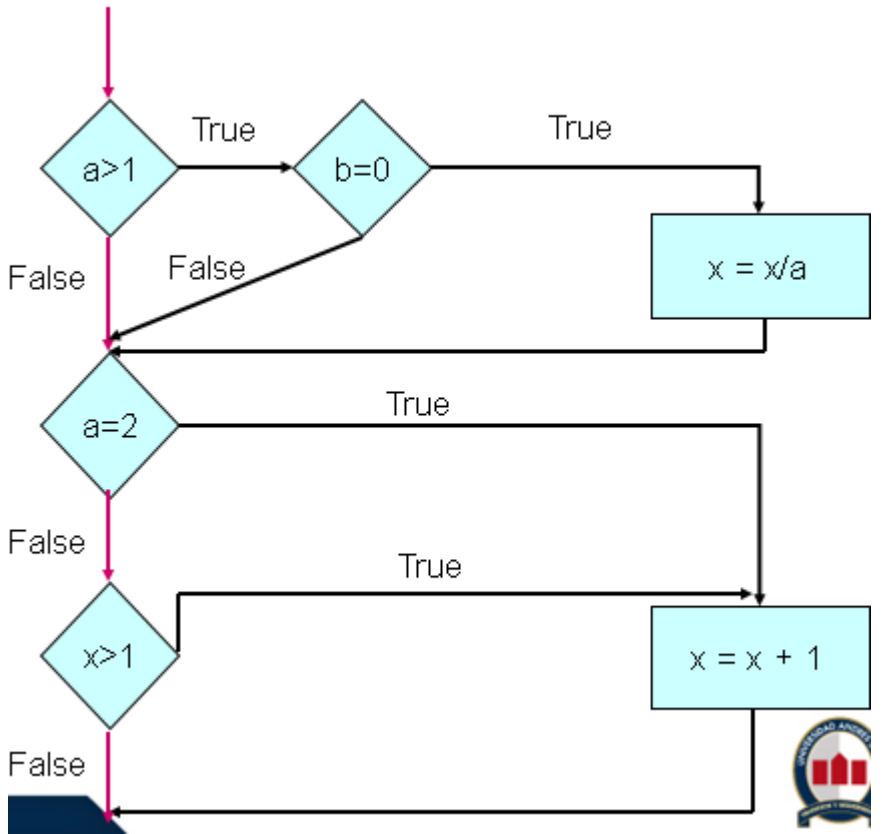
# Técnicas Pruebas Dinámicas: Caja Blanca

## 4.- CRITERIO DE CUBRIMIENTO DE DECISIÓN/CONDICIÓN

CP1 V (C1 V C2 V) F (C3 F C4 F)
a = 2, b = 0, x = 4




CP2 F (C1 F C2 F) V (C3 V C4 V)
CP1 a = 1, b = 1, x = 1

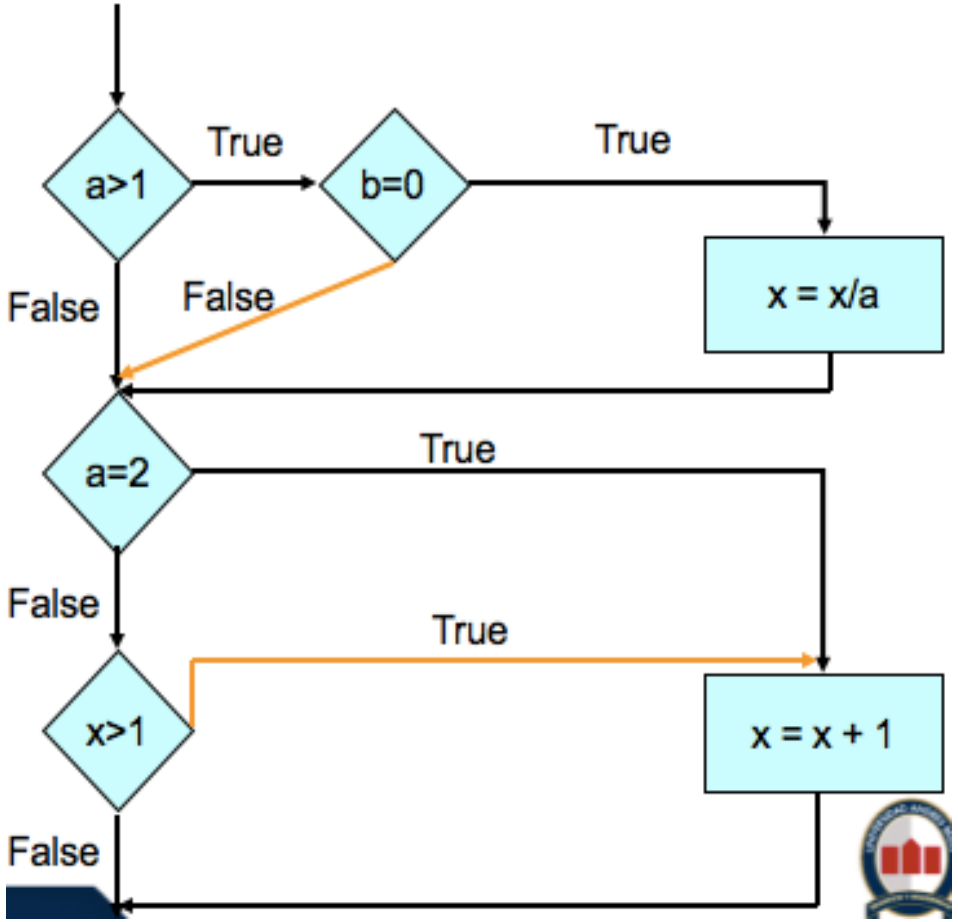


# Técnicas Pruebas Dinámicas: Caja Blanca

## 4.- CRITERIO DE CUBRIMIENTO DE DECISIÓN/CONDICIÓN

PROBLEMA

Hay caminos que no se evalúan



# Técnicas Pruebas Dinámicas: Caja Blanca

## 5.- CRITERIO DE CUBRIMIENTO DE CONDICIÓN MÚLTIPLE

Todas las combinaciones posibles de resultados de condición dentro de una decisión se ejecuten al menos una vez

Se deben satisfacer 8 combinaciones:

1) a>1 (V), b=0 (V) 2) a>1 (V), b<>0 (F)

3) a<=1(F), b=0 (V) 4) a<=1 (F), b<>0 (F)

5) a=2 (V), x>1 (V) 6) a=2 (V), x<=1 (F)

7) a<>2 (F), x>1 (V) 8) a<>2 (F), x<=1 (F)

Los casos 5 a 8 aplican en el punto b

CP1 a=2, b=0, x=4 cubre 1 y 5

CP2 a=2, b=1, x=1 cubre 2 y 6

CP3 a=1, b=0, x=2 cubre 3 y 7

CP4 a=1, b=1, x=1 cubre 4 y 8

- La secuencia acd queda sin ejecutar → este criterio no asegura testear todos los caminos posibles

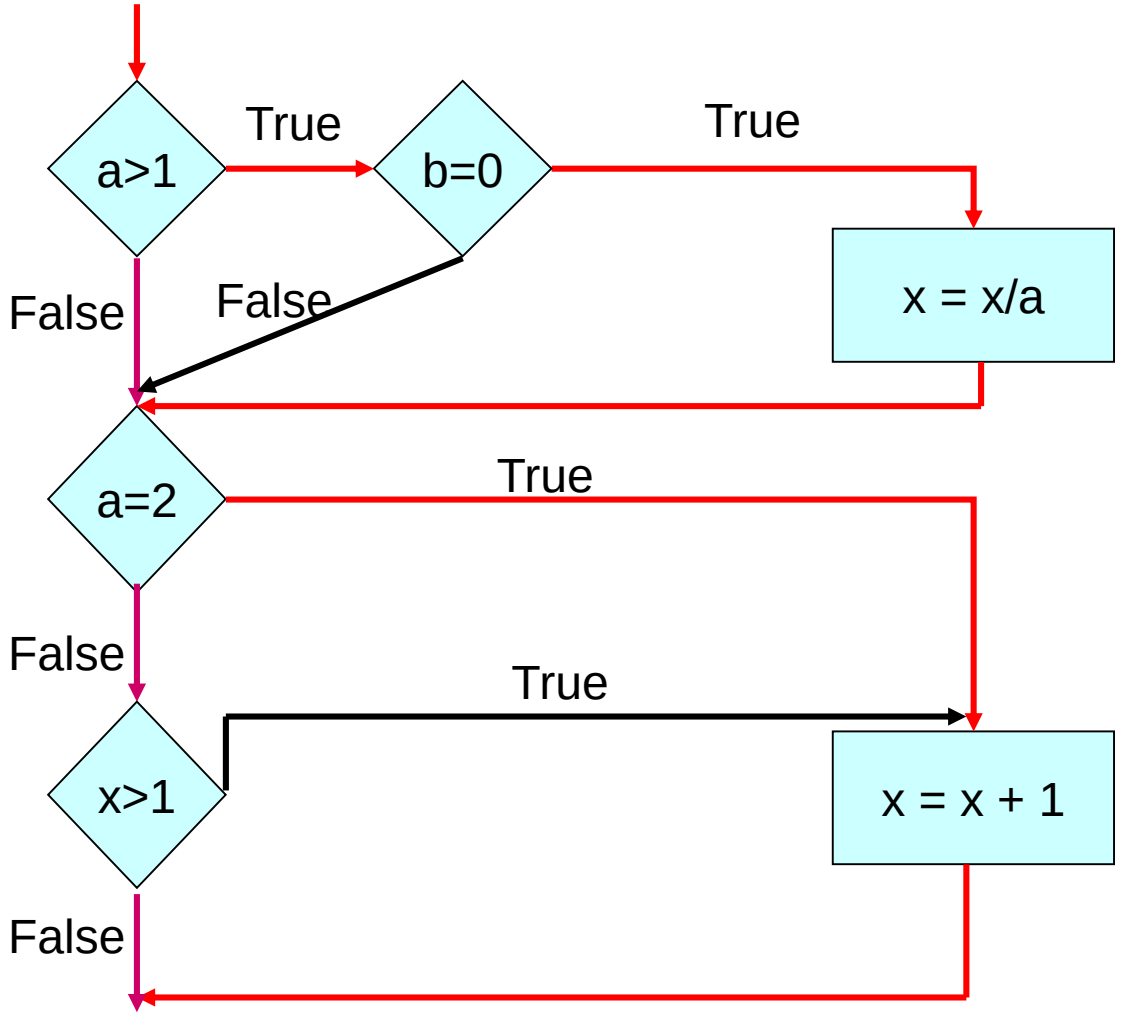
Incluye al criterio de condición/decisión.

Myers lo considera un criterio aceptable

# TÉCNICA CAJA BLANCA

## 5.- CRITERIO DE CUBRIMIENTO DE CONDICIÓN MÚLTIPLE

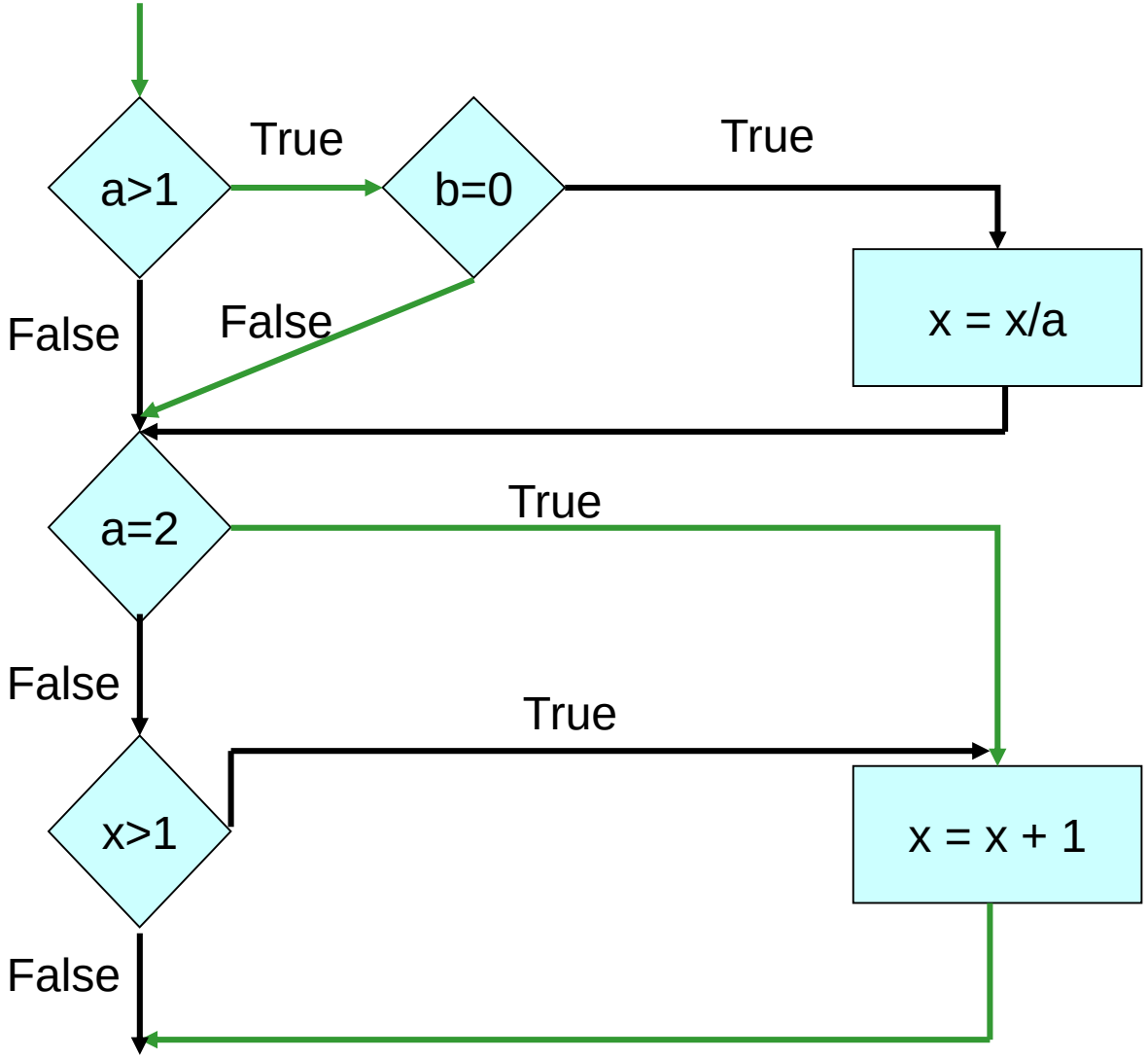
CP1 y CP4 ya los vimos



# TÉCNICA CAJA BLANCA

## 5.- CRITERIO DE CUBRIMIENTO DE CONDICIÓN MÚLTIPLE

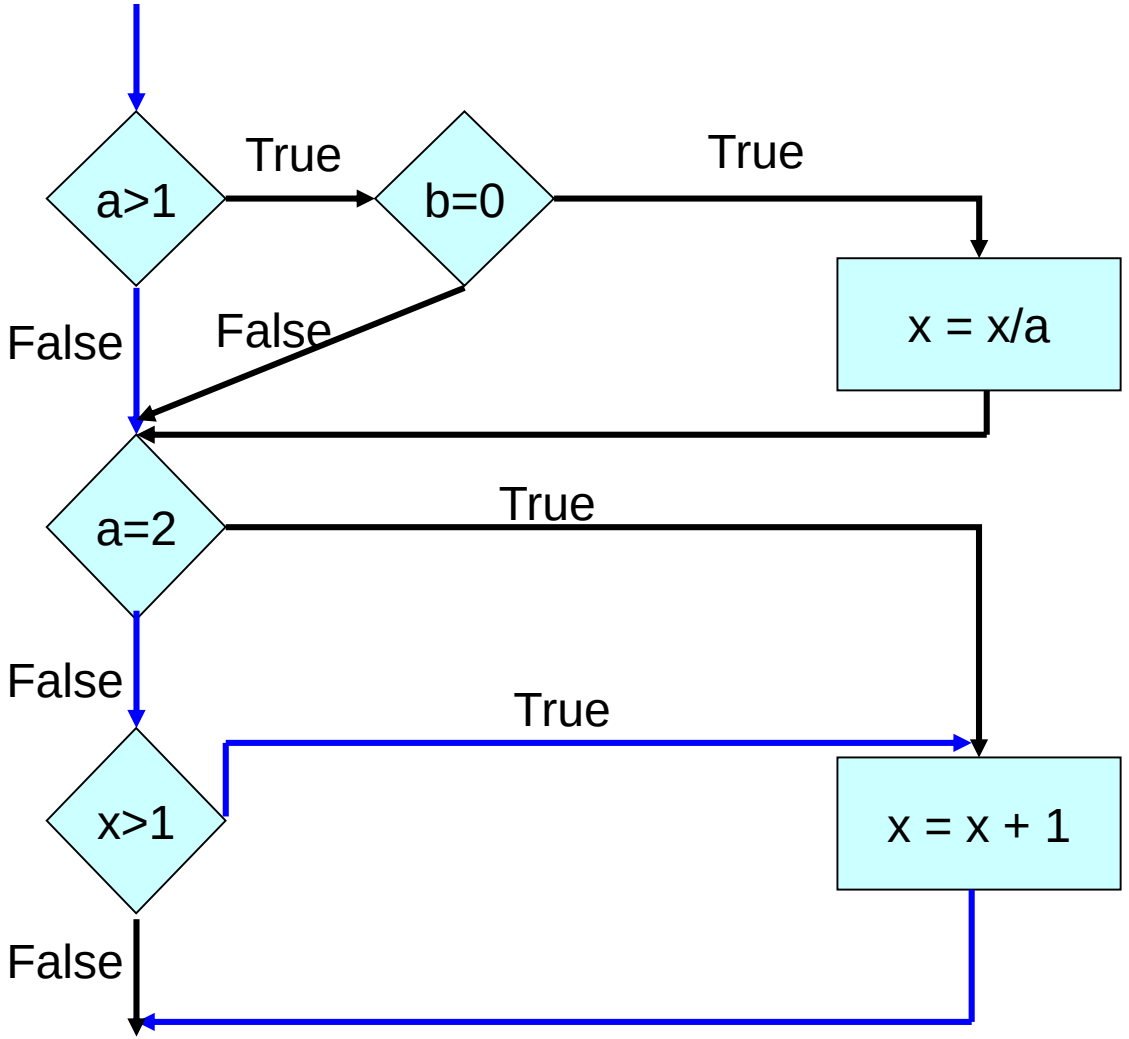
CP2 a = 2, b = 1, x = 1



# Técnicas Pruebas Dinámicas: Caja Blanca

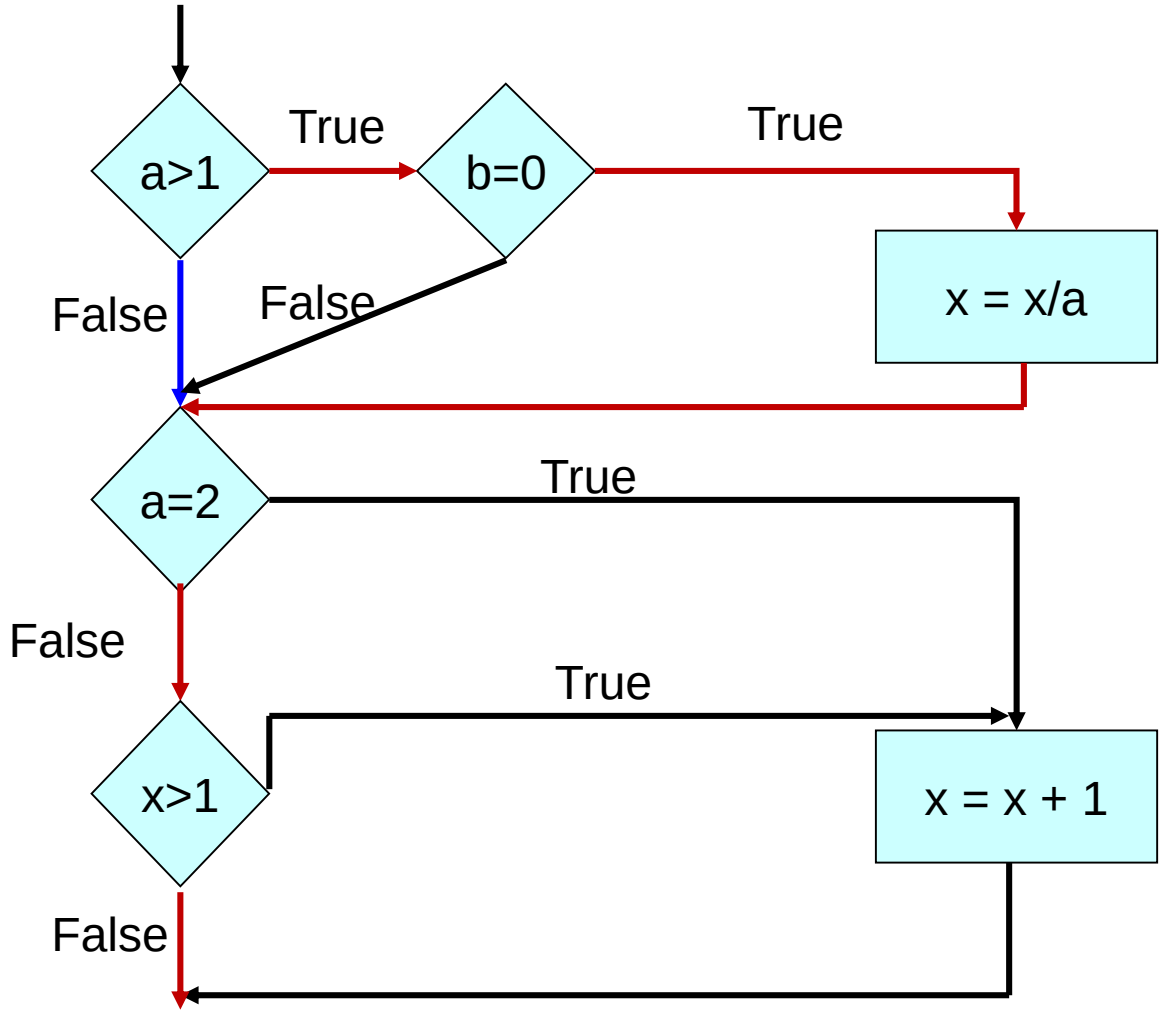
## 5.- CRITERIO DE CUBRIMIENTO DE CONDICIÓN MÚLTIPLE

CP3 a = 1, b = 0, x = 2



# Técnicas Pruebas Dinámicas: Caja Blanca

## 5.- CRITERIO DE CUBRIMIENTO DE CONDICIÓN MÚLTIPLE



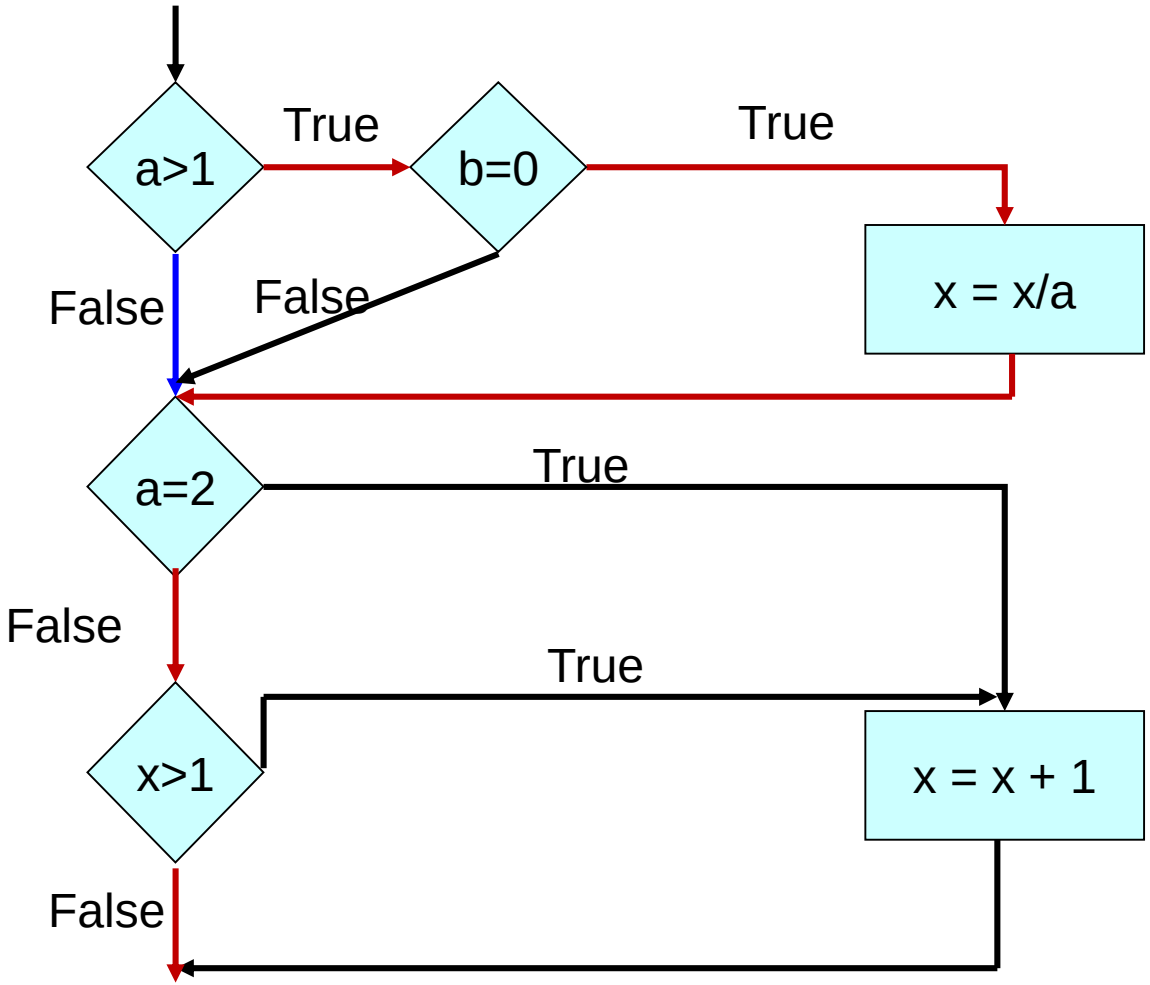

La secuencia acd queda sin ejecutar → este criterio no asegura testear todos los caminos posibles

Incluye al criterio de condición/decisión.

Myers lo considera un criterio aceptable

# Técnicas Pruebas Dinámicas: Caja Blanca

## 5.- CRITERIO DE CUBRIMIENTO DE CONDICIÓN MÚLTIPLE




La secuencia acd queda sin ejecutar → este criterio no asegura testear todos los caminos posibles

Incluye al criterio de condición/decisión.

Myers lo considera un criterio aceptable

# Técnicas Pruebas Dinámicas: Caja Blanca

Universidad Andrés Bello

## 6.- PRUEBA DEL CAMINO BÁSICO

- La prueba del camino básico es una técnica de prueba de la caja blanca propuesta por Tom McCabe.
- La idea es derivar casos de prueba a partir de un conjunto dado de caminos independientes por los cuales puede circular el flujo de control.
- Camino independiente es aquel que introduce por lo menos una sentencia de procesamiento (o condición) que no estaba considerada en el conjunto de caminos independientes calculados hasta ese momento.
- Para obtener el conjunto de caminos independientes se construirá el Grafo de **Flujo asociado** y se calculará su **Complejidad Ciclomática.**

# Técnicas Pruebas Dinámicas: Caja Blanca

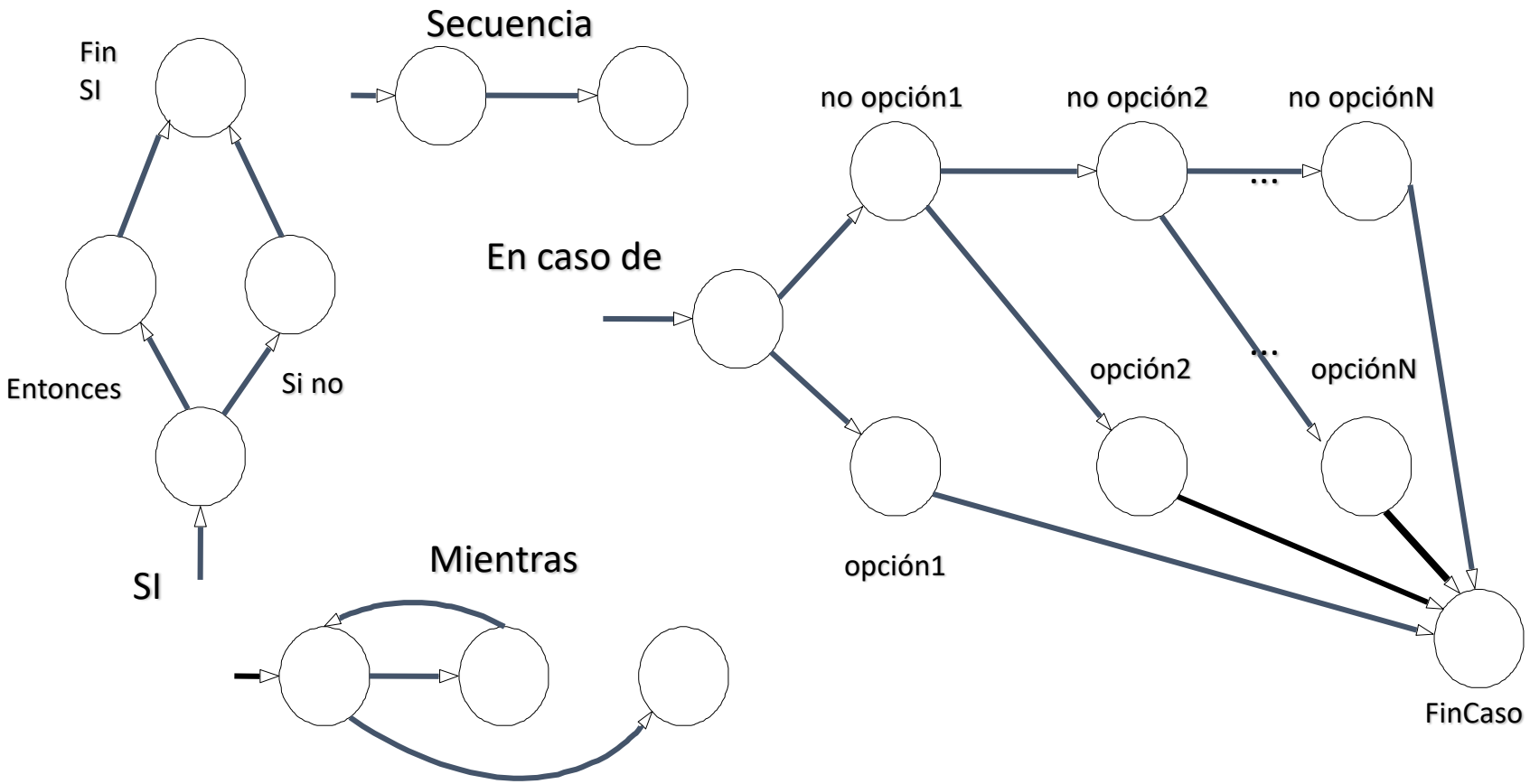
Universidad Andrés Bello

## PRUEBA DEL CAMINO BÁSICO: GRAFOS DE FLUJO

- El flujo de control de un programa puede representarse por un grafo de flujo.
- Cada  nodo del grafo corresponde a una o más sentencias de código fuente
- Cada nodo que representa una condición se denomina nodo predicado.
- Cualquier representación del diseño procedural se puede traducir a un grafo de flujo.
- Un camino independiente en el grafo es el que incluye alguna arista nueva, es decir, que no estaba presente en los caminos definidos previamente.

# Técnicas Pruebas Dinámicas: Caja Blanca

## PRUEBA DEL CAMINO BÁSICO: GRAFOS DE FLUJO



# Técnicas Pruebas Dinámicas: Caja Blanca

Universidad Andrés Bello

## PRUEBA DEL CAMINO BÁSICO: GRAFOS DE FLUJO

- Si se diseñan casos de prueba que cubran los caminos básicos, se garantiza la ejecución de cada sentencia al menos una vez y la prueba de cada condición en sus dos posibilidades lógicas (verdadera y falsa).

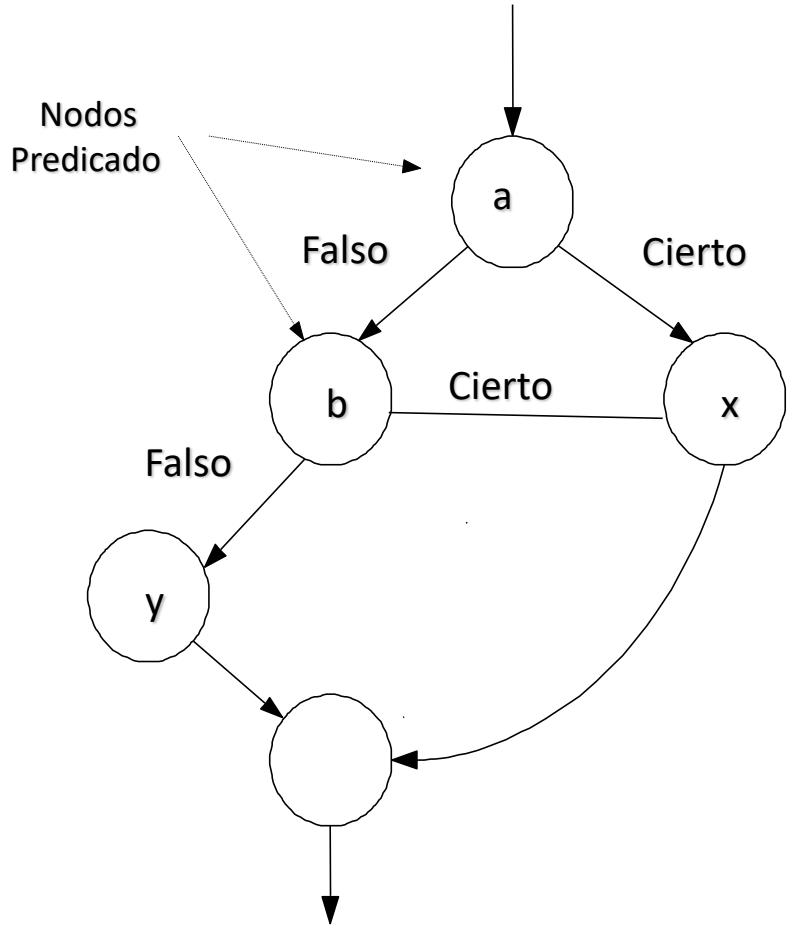
- El conjunto básico para un grafo dado puede no ser único, depende del orden en que se van definiendo los caminos.

- Cuando aparecen condiciones lógicas compuestas la confección del grafo es más compleja.

# PRUEBA DEL CAMINO BÁSICO: GRAFOS DE FLUJO

Ejemplo:

SI a O b
**Entonces**
hacer x
**Si No**
hacer y
FinSI



# PRUEBA DEL CAMINO BÁSICO: COMPLEJIDAD CICLOMATICA

- Complejidad ciclomática de un grafo de flujo, V(G), indica el *número máximo de caminos independientes en el grafo*.
- Puede calcularse de tres formas alternativas:

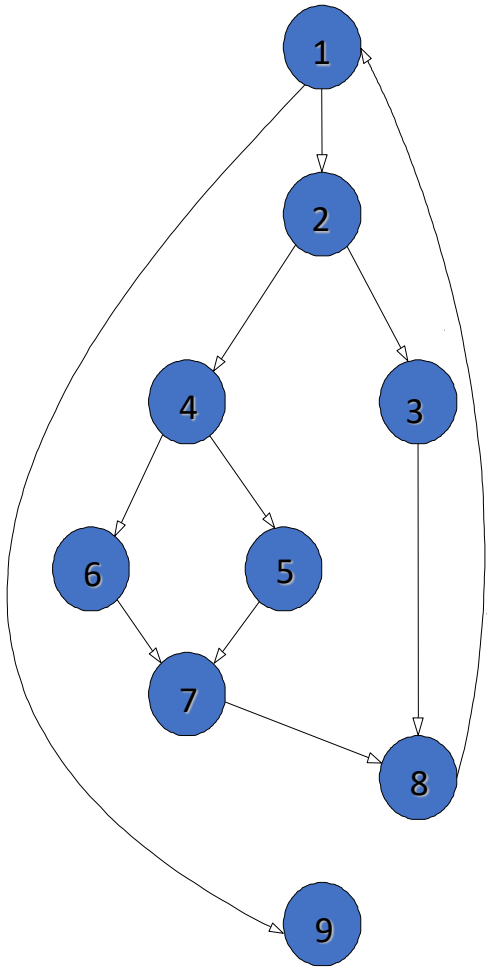
  - **El número de regiones** en que el grafo de flujo divide el plano.
  - **V(G) = A - N + 2**, donde A es el número de aristas y N es el número de nodos.
  - **V(G) = P + 1,** donde P es el número de nodos predicado.

# Técnicas Pruebas Dinámicas: Caja Blanca

## PRUEBA DEL CAMINO BÁSICO: COMPLEJIDAD CICLOMATICA

V(G) = 4

- El grafo de la figura delimita cuatro regiones.
- 11 aristas - 9 nodos + 2 = 4
- 3 nodos predicado + 1 = 4



# Técnicas Pruebas Dinámicas: Caja Blanca

## PRUEBA DEL CAMINO BÁSICO: COMPLEJIDAD CICLOMATICA

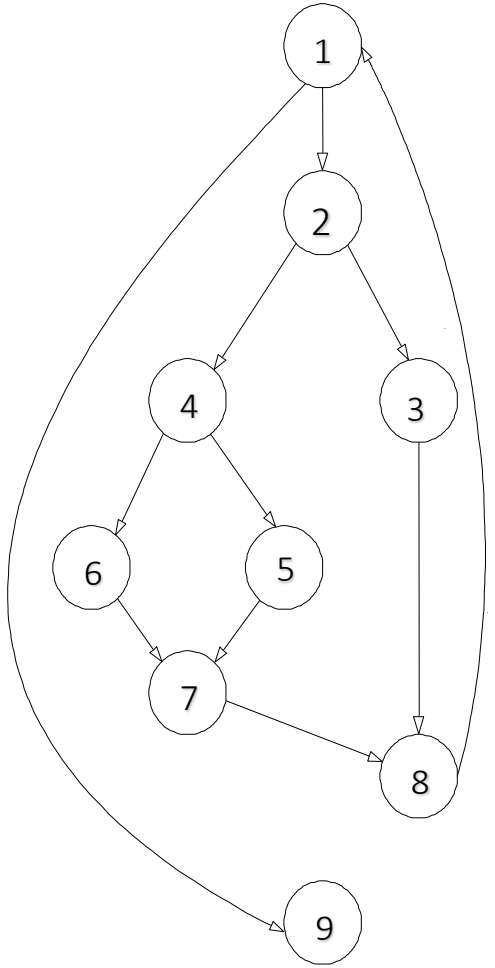
- El conjunto de caminos independientes del grafo será 4.
  - Camino 1: 1-9
  - Camino 2: 1-2-4-5-7-8-1-9
  - Camino 3: 1-2-4-6-7-8-1-9
  - Camino 4: 1-2-3-8-1-9

- Cualquier otro camino no será un camino independiente, ejemplo:

  1-2-4-5-7-8-1-2-3-8-1-2-4-6-7-8-1-9

  ya que es simplemente una combinación de caminos ya especificados

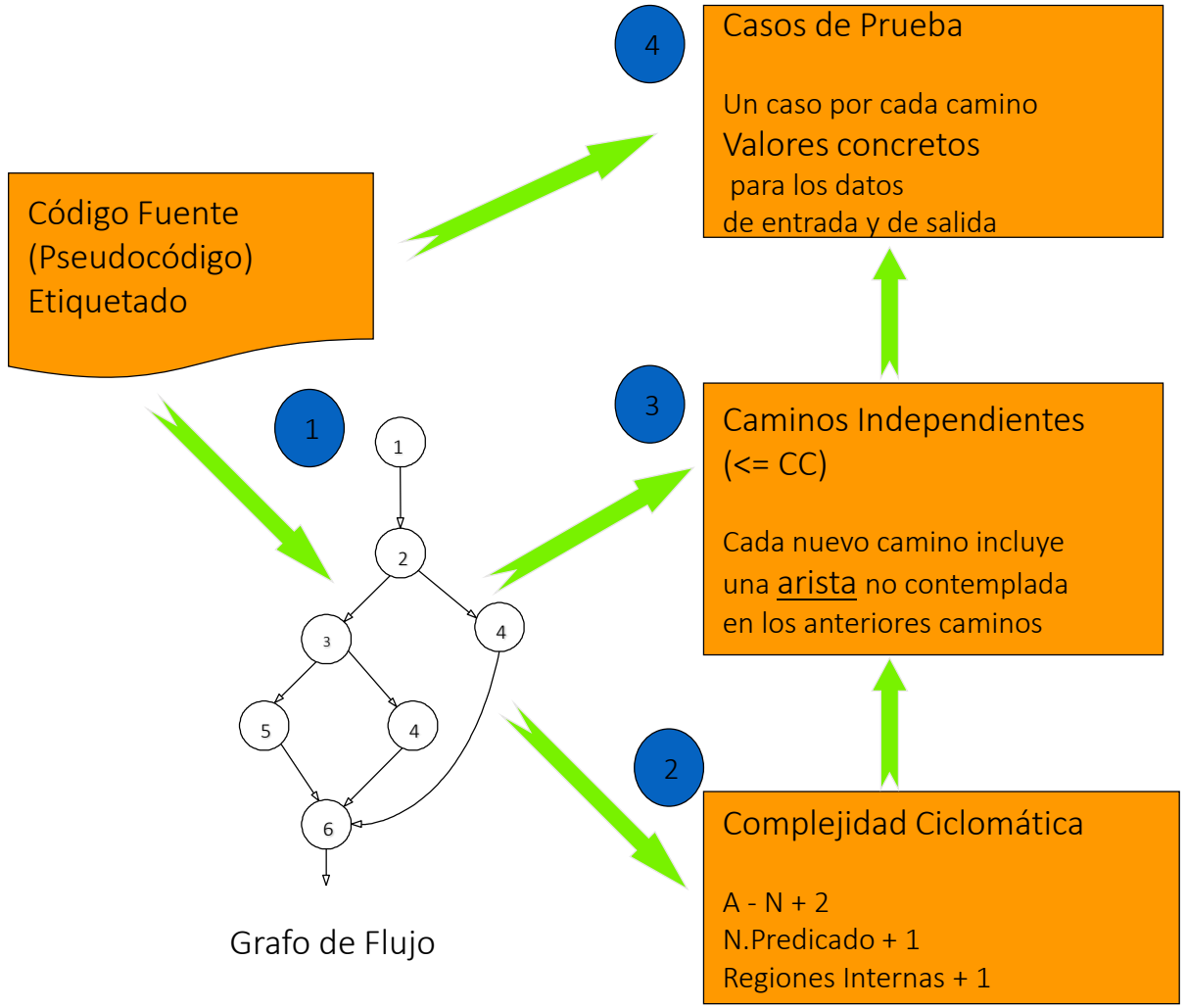
*Los cuatro caminos anteriores constituyen un conjunto básico para el grafo*



# Técnicas Pruebas Dinámicas: Caja Blanca

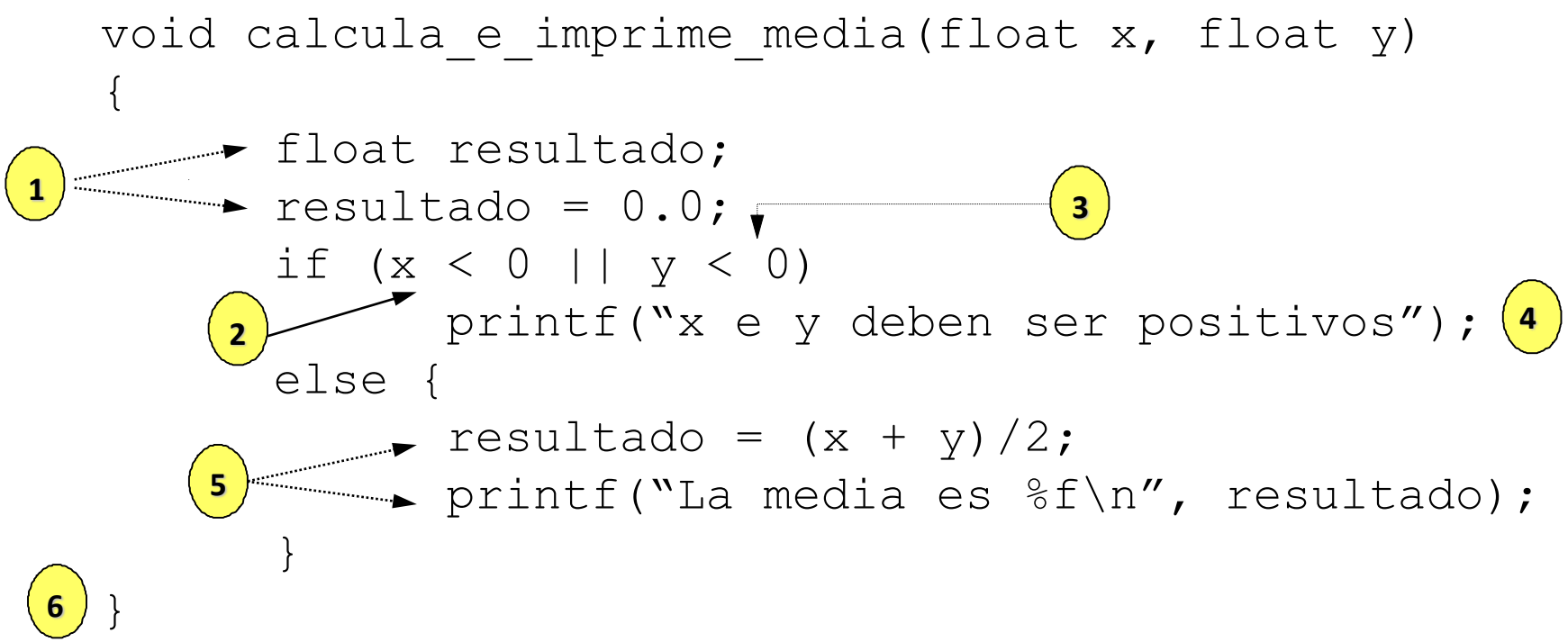
## PRUEBA DEL CAMINO BÁSICO: COMPLEJIDAD CICLOMATICA

Como realizar los Casos de prueba



# Técnicas Pruebas Dinámicas: Caja Blanca

## PRUEBA DEL CAMINO BÁSICO: COMPLEJIDAD CICLOMATICA



# Técnicas Pruebas Dinámicas: Caja Blanca

Universidad Andrés Bello

## Pruebas de Estructura de Control

- La técnica de prueba del camino básico del punto anterior es una de las muchas existentes para la pruebas de la estructura de control.
- Aunque sea alta la efectividad de la prueba del camino básico, no nos asegura completamente la corrección del software.
- Existen variantes las que amplían el abanico de pruebas mejorando la fiabilidad de las pruebas de caja blanca.

# Técnicas Pruebas Dinámicas: Caja Blanca

Universidad Andrés Bello

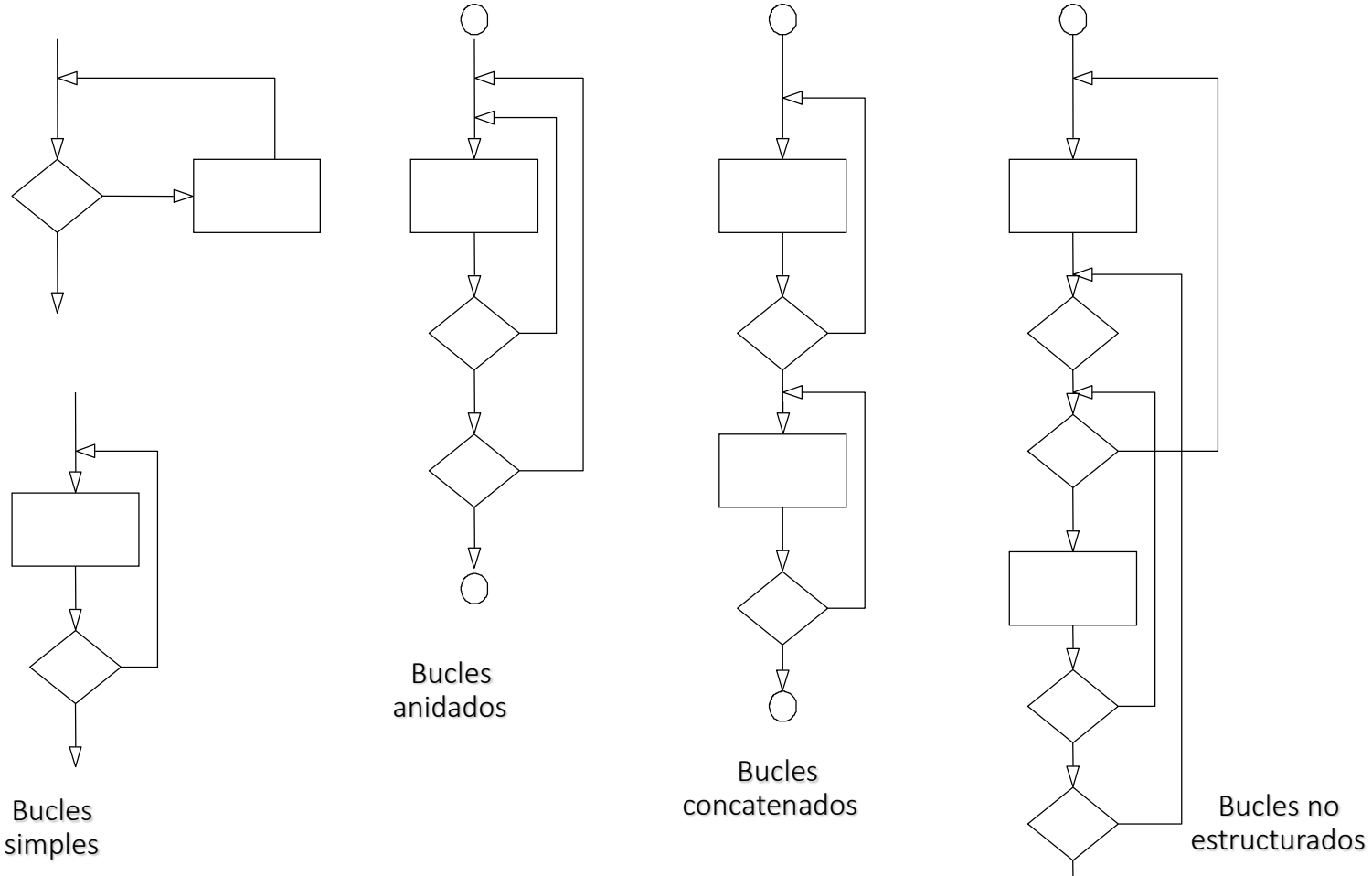
## Pruebas de Estructura de Control: Prueba de Condiciones

La prueba de condiciones es un método de diseño de casos de prueba que ejercita las condiciones lógicas contenidas en el módulo de un programa. Los tipos de errores que pueden aparecer en una condición son los siguientes:

- Existe un error en un operador lógico (que sobra, falta o no es el que corresponde).
- Existe un error en un paréntesis lógico (cambiando el significado de los operadores involucrados).
- Existe un error en un operador relacional (operadores de comparación de igualdad, menor o igual, etc.).
- Existe un error en una expresión aritmética.

# Técnicas Pruebas Dinámicas: Caja Blanca

## Pruebas de Estructura de Control: Prueba de Condiciones



# Técnicas Pruebas Dinámicas: Caja Blanca

Universidad Andrés Bello

## Pruebas de Estructura de Control: Prueba de Condiciones

Pruebas para Bucles Simples. (n es el número máximo de iteraciones permitidos por el bucle)

- Pasar por alto totalmente el bucle
- Pasar una sola vez por el bucle
- Pasar dos veces por el bucle
- Hacer m pasos por el bucle con m < n
- Hacer n-1, n y n + 1 pasos por el bucle

# Técnicas Pruebas Dinámicas: Caja Blanca

Universidad Andrés Bello

## Pruebas de Estructura de Control: Prueba de Condiciones

- Pruebas para Bucles anidados
  - Comenzar en el bucle más interior estableciendo los demás bucles en sus valores mínimos.
  - Realizar las pruebas de bucle simple para el más interior manteniendo los demás en sus valores mínimos.
  - Avanzar hacia fuera confeccionando pruebas para el siguiente bucle manteniendo todos los externos en los valores mínimos y los demás bucles anidados en sus valores típicos.
  - Continuar el proceso hasta haber comprobado todos los bucles.
- Pruebas para Bucles concatenados
  - Siempre que los bucles concatenados sean independientes se puede aplicar lo relativo a bucles simples/anidados. En caso de ser dependientes se evaluarán como bucles anidados.

# Caja Negra

*Evalua la satisfacción de un sistema o componente con los requerimientos funcionales especificados.*

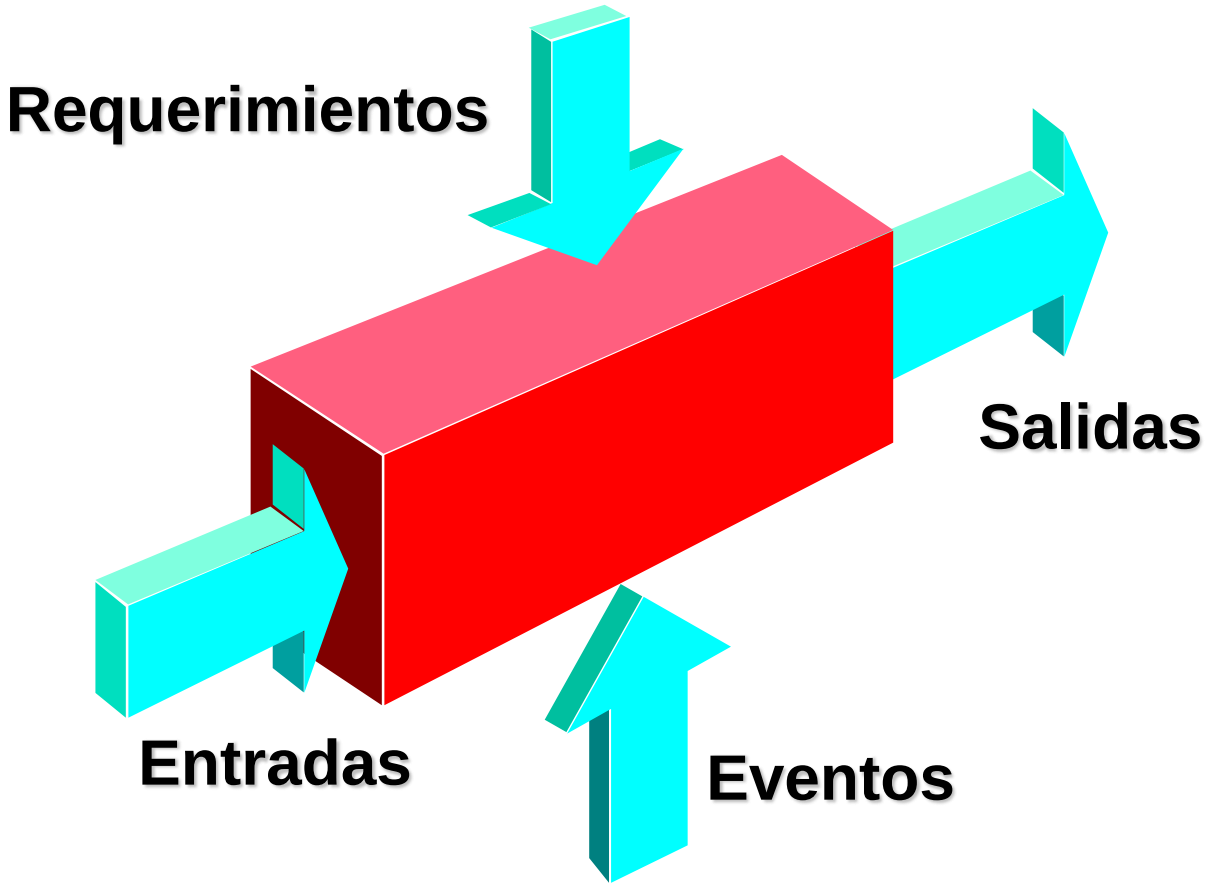
Partición de equivalencia

Análisis de valor frontera

Grafo causa efecto

Adivinanza de errores

# Técnicas Pruebas Dinámicas: Caja Negra



# Técnicas Pruebas Dinámicas: Caja Negra

Universidad Andrés Bello

No son una alternativa, complementan las de Caja Blanca

Intenta encontrar errores tipo:

- Funciones incorrectas o faltantes
- Errores de interfaz
- Errores en las estructuras de datos o en el acceso a BD externas
- Errores de comportamiento o rendimiento
- Errores de inicialización y terminación

Las pruebas se diseñan para responder a preguntas como:

- Cómo se prueba la validez funcional?
- Cómo se prueba el comportamiento y rendimiento?
- Qué clases de entrada harán buenos casos de prueba?
- El sistema es particularmente sensible a ciertos valores de entrada?
- Cómo se aíslan las fronteras de una clase de datos?
- Qué tasas y volumen de datos puede tolerar el sistema?
- Qué efecto tendrá sobre la operación algunas combinaciones de datos?

# Técnicas Pruebas Dinámicas: Caja Negra

Universidad Andrés Bello

Las pruebas de caja negra se llevan a cabo **sobre la interfaz del software**, obviando el comportamiento interno y la estructura del programa.

Los casos de prueba de la caja negra pretenden demostrar que:

- Las funciones del software son operativas
- La entrada se acepta de forma correcta
- Se produce una salida correcta
- La integridad de la información externa se mantiene

# Técnicas Pruebas Dinámicas: Caja Negra

Universidad Andrés Bello

También son llamadas pruebas de comportamiento, ya que se centran en los **requisitos funcionales** del software.

Consideran la función específica para la cuál fue creado el producto (lo que hace).

Las pruebas se llevan a cabo sobre la interfaz del sistema. Reduce el número de casos de prueba mediante la elección de condiciones de entrada y salida válidas y no válidas que ejercitan toda la funcionalidad del sistema.

# Técnicas Pruebas Dinámicas: Caja Negra

Universidad Andrés Bello

Tiende a aplicarse durante las fases posteriores de la prueba, ya que la prueba de caja negra ignora intencionalmente la estructura de control, centra su atención en el campo de la información.

## *Casos de Prueba*

Reducen, en un coeficiente que es mayor que uno, el número de casos de prueba adicionales que se deben diseñar para alcanzar una prueba razonable.

Casos de prueba que nos dicen algo sobre la presencia o ausencia de clases de errores en lugar de errores asociados solamente con la prueba que estamos realizando.

# Técnicas Pruebas Dinámicas: Caja Negra

Universidad Andrés Bello

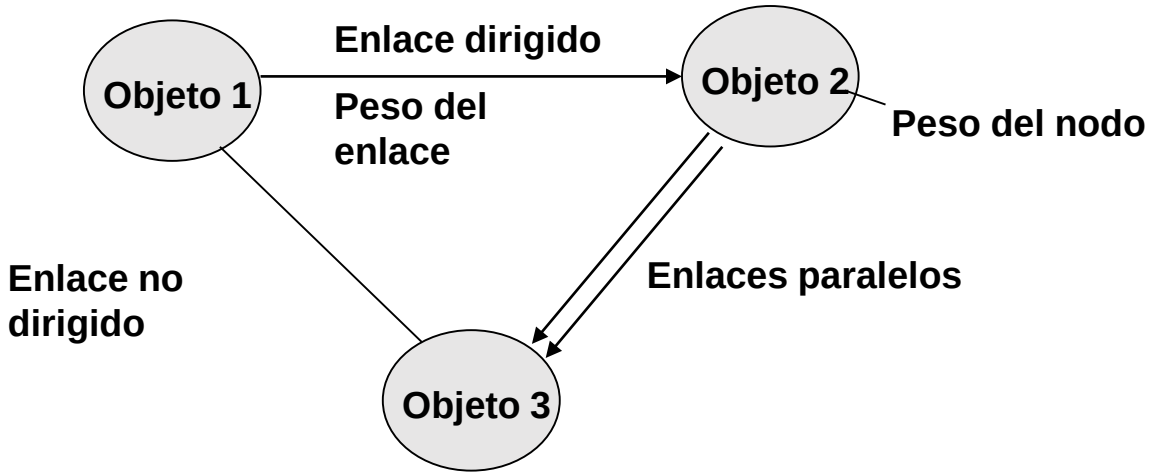
Técnicas de pruebas de caja negra.

a) Métodos de prueba basados en grafos.
b) Partición Equivalente.
c) Análisis de valores límite.
d) Prueba de Comparación.
e) Conjetura de Errores.

# Técnicas Pruebas Dinámicas: Caja Negra

## MÉTODOS DE PRUEBA BASADOS EN GRAFOS

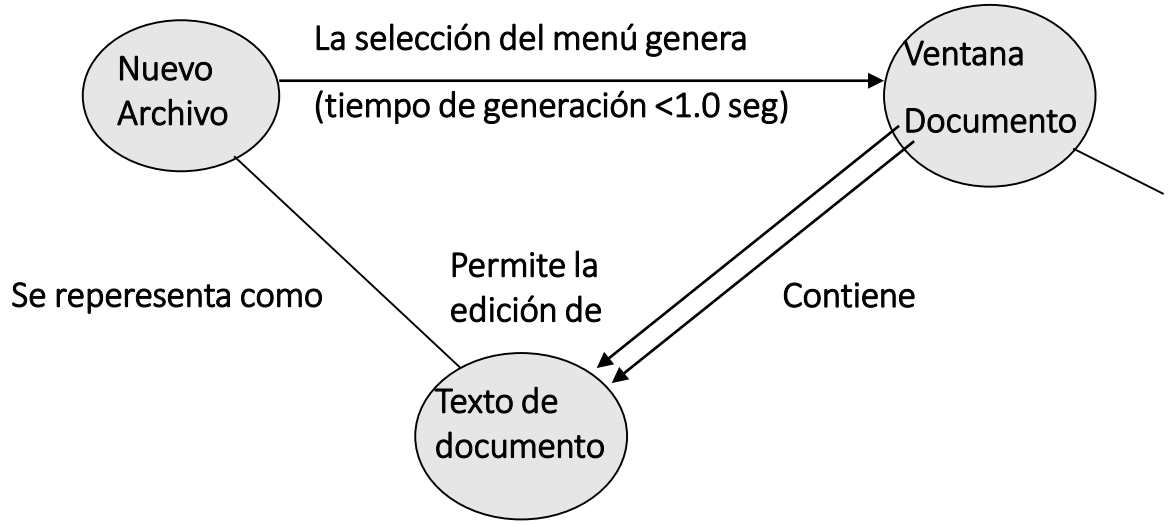
1. Se modelan los objetos del sistema y las relaciones entre ellos mediante un grafo.
2. El siguiente paso es definir una serie de pruebas que verifiquen que todos los objetos tienen las relaciones esperadas entre ellos
3. Se ejercitan todos los objetos y sus relaciones para descubrir errores



# Técnicas Pruebas Dinámicas: Caja Negra

## MÉTODOS DE PRUEBA BASADOS EN GRAFOS

Ejemplo: Creación de un nuevo archivo en un procesador de texto.




Atributos:

Dimensiones de inicio:

Configuración o preferencias predeterminadas.

Color de fondo: blanco

Color de texto: color o preferencias predeterminadas

Una selección del menú en Nuevo Archivo genera una ventana del documento.
El peso del nodo de Ventana Documento proporciona una lista de atributos de la Ventana, que se esperan cuando se genera una ventana.
El peso del enlace indica que la ventana se tiene que generar en menos de 1.0 segundos.

# Técnicas Pruebas Dinámicas: Caja Negra

Universidad Andrés Bello

## MÉTODOS DE PRUEBA BASADOS EN GRAFOS

En este método se debe entender los objetos que se modelan en el software y las relaciones que conectan a estos objetos. Una vez que se ha llevado a cabo esto, el siguiente paso es definir una serie de pruebas que verifiquen que todos los objetos tienen entre ellos las relaciones esperadas.

*En este método:*

- ✓Se crea un grafo de objetos importantes y sus relaciones.
- ✓Se diseña una serie de pruebas que cubran el grafo de manera que se ejerciten todos los objetos y sus relaciones para descubrir errores.

# Técnicas Pruebas Dinámicas: Caja Negra

Universidad Andrés Bello

## MÉTODOS DE PRUEBA BASADOS EN GRAFOS

Beizer describe un número de modelos para pruebas de comportamiento que pueden hacer uso de los grafos:

*Modelado del Flujo de Transacción:* Los nodos representan los pasos de alguna *transacción* y los enlaces representan las conexiones lógicas entre los pasos

*Modelado de Estado Finito:* Los nodos representan *diferentes estados* del software observables por el usuario y los enlaces representan las transiciones que ocurren para moverse de estado a estado.

# Técnicas Pruebas Dinámicas: Caja Negra

Universidad Andrés Bello

## MÉTODOS DE PRUEBA BASADOS EN GRAFOS

*Modelado de Flujo de Datos:* Los nodos objetos de datos y los enlaces son las transformaciones que ocurren para convertir un objeto de datos en otro.

*Modelado de Planificación:* Los nodos son objetos de programa y los enlaces son las conexiones secuenciales entre esos objetos. Los pesos de enlace se usan para especificar los tiempos de ejecución requeridos al ejecutarse el programa.

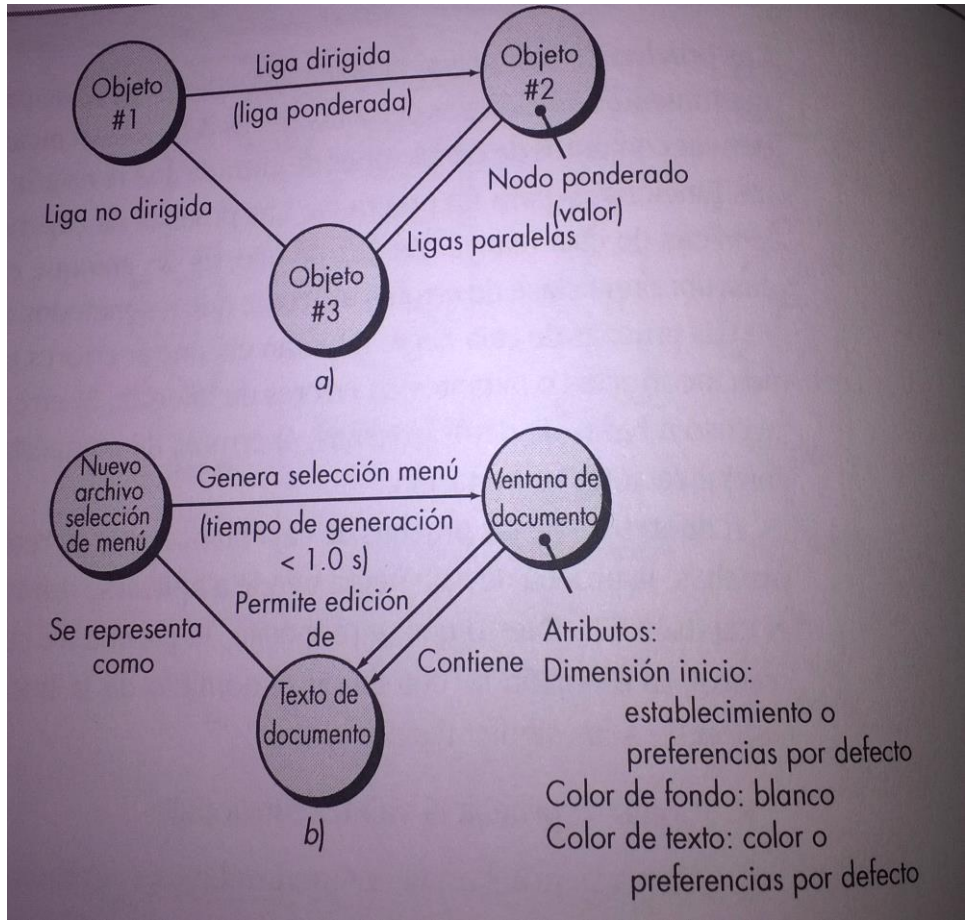
*Gráfica Causa-efecto:* La gráfica Causa-efecto representa una ayuda gráfica en seleccionar, de una manera sistemática, un gran conjunto de casos de prueba. Tiene un efecto secundario beneficioso en precisar estados incompletos y ambigüedades en la especificación.

# Técnicas Pruebas Dinámicas: Caja Negra

## MÉTODOS DE PRUEBA BASADOS EN GRAFOS

Pasos a seguir para una prueba de caja Negra:

- Las pruebas basadas en grafos empiezan con la definición de todos los nodos y peso de nodos.
- Una vez que se han identificado los nodos, se deberían establecer los enlaces y los pesos de enlaces.
- Cada relación es estudiada separadamente, de manera que se puedan obtener casos de prueba.
- Definir una serie de pruebas que verifique que *"todos los objetos tienen entre ellos la relaciones esperadas"*



# MÉTODOS GRAFOS CAUSA/EFECTO Y TABLA DE DECISIÓN

Universidad Andrés Bello

El uso de grafos de causa - efecto es una técnica de casos de prueba que proporciona una concisa representación de las condiciones lógicas y sus correspondientes acciones.

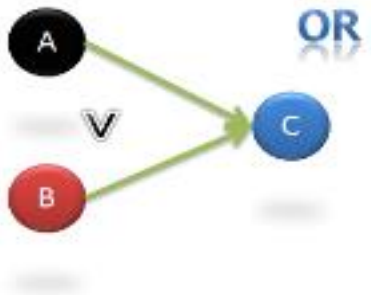
## La técnica sigue cuatro pasos:

1. Se listan para un módulo las causas (condiciones de entrada) y los efectos (acciones), asignando un identificador a cada uno de ellos.
2. Se desarrolla un grafo causa - efecto
3. Se convierte el grafo en una tabla de decisión
4. Se convierten las reglas de la tabla de decisión a casos de prueba

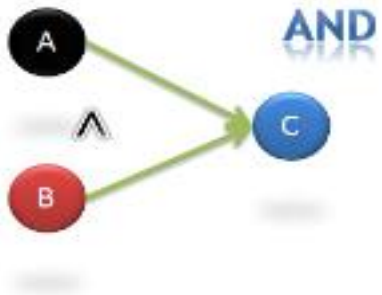
# MÉTODOS GRAFOS CAUSA/EFECTO Y TABLA DE DECISIÓN

Universidad Andrés Bello

Para la construcción del grafo entonces se parte de los operadores booleanos, partimos de los siguientes conceptos:




IF A = 1 OR B=1
THEN C=1




IF A = 1 AND B=1
THEN C=1


IF A = 1 THEN B=1


IF A = 1
THEN B=0
ELSE B=1

# MÉTODOS GRAFOS CAUSA/EFECTO Y TABLA DE DECISIÓN

Universidad Andrés Bello

Ejemplo

Los requerimientos para calcular el valor de los seguros de un auto son:

- Para mujeres de menos de 65 años, la prima es de $500
- Para hombres de menos de 25 años, la prima es de $3000
- Para hombres entre 25 y 64 años, la prima es de $1000
- Para cualquiera de mas de 65 años, la prima es de $1500

# MÉTODOS GRAFOS CAUSA/EFECTO Y TABLA DE DECISIÓN

Universidad Andrés Bello

Ejemplo

**1 Paso:** Identificar Causas (Entradas) y los efectos (Salidas)

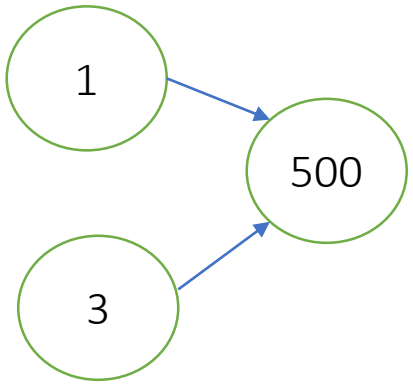
| CAUSA | EFECTO |
|---|---|
| 1.- Sexo: Mujer | Prima= 500 |
| 2.- Sexo: Hombre | Prima= 3.000 |
| 3.- Edad <25 | Prima=1.000 |
| 4.- Edad >=25 and <65 | Prima=1.500 |
| 5.- Edad >=65 | |
| | |

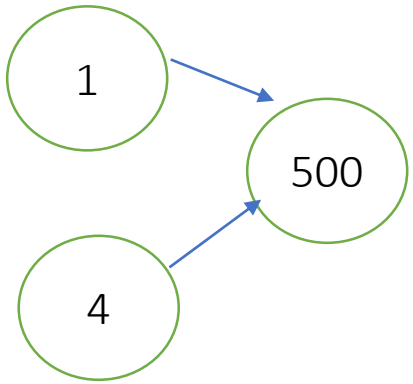
# MÉTODOS GRAFOS CAUSA/EFECTO Y TABLA DE DECISIÓN

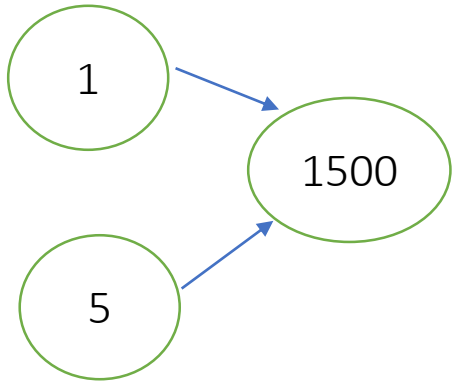
Universidad Andrés Bello

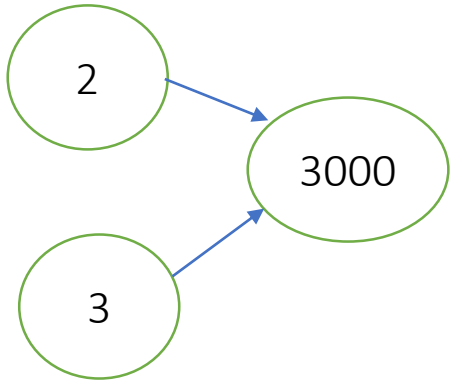
Ejemplo

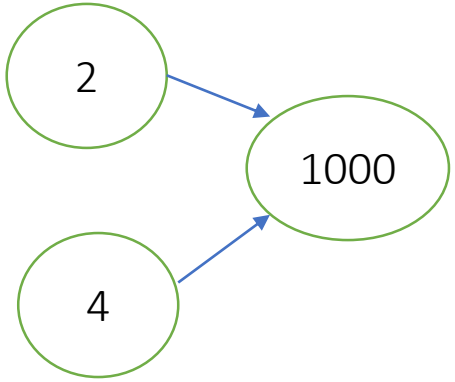
**2 Paso:** Elaborar Grafos

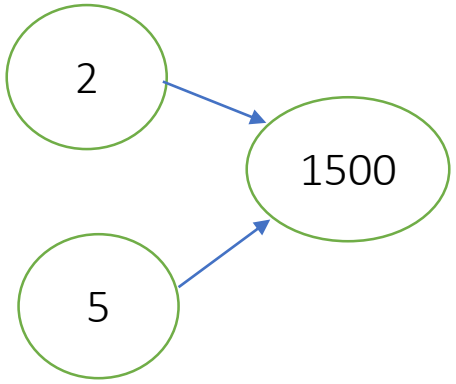


# MÉTODOS GRAFOS CAUSA/EFECTO Y TABLA DE DECISIÓN

Universidad Andrés Bello

Ejemplo

**3 Paso:** Elaborar Tabla de decisión

| Casos de Prueba | G1 | G2 | G3 | G4 | G5 | G6 |
|---|---|---|---|---|---|---|
| **Causas** | | | | | | |
| 1.- Sexo: Mujer | 1 | 1 | 1 | 0 | 0 | 0 |
| 2.- Sexo: Hombre | 0 | 0 | 0 | 1 | 1 | 1 |
| 3.- Edad <25 | 1 | 0 | 0 | 1 | 0 | 0 |
| 4.- Edad >=25 and <65 | 0 | 1 | 0 | 0 | 1 | 0 |
| 5.- Edad >=65 | 0 | 0 | 1 | 0 | 0 | 1 |
| **Efectos** | | | | | | |
| Prima= 500 | 1 | 1 | 0 | 0 | 0 | 0 |
| Prima= 3.000 | 0 | 0 | 0 | 1 | 0 | 0 |
| Prima=1.000 | 0 | 0 | 0 | 0 | 1 | 0 |
| Prima=1.500 | 0 | 0 | 1 | 0 | 0 | 1 |

# MÉTODOS GRAFOS CAUSA/EFECTO Y TABLA DE DECISIÓN

Universidad Andrés Bello

Ejemplo

**4 Paso:** Generar Casos de Prueba

| Casos de Prueba | Entradas (Causas) | | Salidas (Efectos) |
|---|---|---|---|
| | **SEXO** | **EDAD** | **PRIMA** |
| **1** | Mujer | Edad <25 | 500 |
| **2** | Mujer | Edad >=25 and <65 | 500 |
| **3** | Mujer | Edad >=65 | 1500 |
| **4** | Hombre | Edad <25 | 3000 |
| **5** | Hombre | Edad >=25 and <65 | 1000 |
| **6** | Hombre | Edad >=65 | 1500 |

# Técnicas Pruebas Dinámicas: Caja Negra

Universidad Andrés Bello

## MÉTODOS DE PRUEBA PARTICIÓN DE EQUIVALENCIA

Se basa en la *división del campo de entrada* en un conjunto de clases de datos denominadas *clases de equivalencia*.

**Clase de Equivalencia:** Un conjunto de datos de entrada que definen estados válidos y no válidos del sistema.

1. Clase válida: Genera un valor esperado
2. Clase no válida: Genera un valor inesperado

Se obtienen a partir de las condiciones de entrada descritas en las especificaciones.

# Técnicas Pruebas Dinámicas: Caja Negra

Universidad Andrés Bello

## MÉTODOS DE PRUEBA PARTICIÓN DE EQUIVALENCIA

La partición equivalente se dirige a la definición de casos de prueba que descubran clases de errores, reduciendo así el número total de casos de prueba que hay que desarrollar.

El objetivo de partición equivalente es reducir el posible conjunto de casos de prueba en uno más pequeño.

Se toma un riesgo porque se escoge no probar todo.

Se necesita tener mucho cuidado al escoger las clases.

# Técnicas Pruebas Dinámicas: Caja Negra

Universidad Andrés Bello

## MÉTODOS DE PRUEBA PARTICIÓN DE EQUIVALENCIA

| Condiciones de Entrada | Número de clases de equivalencia válida | Número de clases de equivalencia no válida |
|---|---|---|
| 1. Rango de valores | **1** clase que contemple los valores del rango | **2** clases fuera del rango, una por encima y otra por debajo de éste |
| 2. Valor específico | **1** clase que contemple dicho valor | **2** clases que representen un valor por encima y otro por debajo |
| 3. Elementos de un conjunto tratados de forma diferente por el programa | **1** clase de equivalencia por cada elemento | **1** clase que represente un elemento fuera del conjunto |
| 4. Condición lógica | **1** clase que cumpla la condición | **1** clase que no cumpla la condición |

Las clases de equivalencia se pueden definir según:

- ✓ **Condición de entrada especifica un *rango*:** Se define una clase de equivalencia válida y dos no válidas.
- ✓ **Condición de entrada requiere un *valor* específico**: Se define una clase de equivalencia válida y dos no válidas.
- ✓ **Condición de entrada un miembro de un *conjunto*:** Se define una clase de equivalencia válida y una no válida.
- ✓ **Condición de entrada es *lógica*:** Se define una clase de equivalencia válida y una no válida.

# Técnicas Pruebas Dinámicas: Caja Negra

Universidad Andrés Bello

## MÉTODOS DE PRUEBA PARTICIÓN DE EQUIVALENCIA

Pasos para identificar clases de equivalencia.

1. Identificación de las condiciones de entrada del programa.

2.- Identificar las Clases de Equivalencia:
   a) Datos válidos.
   b) Datos no válidos.

# Técnicas Pruebas Dinámicas: Caja Negra

Universidad Andrés Bello

## MÉTODOS DE PRUEBA PARTICIÓN DE EQUIVALENCIA

Pasos para identificar casos de prueba

1. Asignar un número único para cada Clase de Equivalencia.
2. Escribir Casos de pruebas para todas las Clases Válidas.
3. Escribir casos de pruebas para todas las Clases no Válidas.

# Técnicas Pruebas Dinámicas: Caja Negra

Universidad Andrés Bello

## MÉTODOS DE PRUEBA PARTICIÓN DE EQUIVALENCIA

*Aplicación Bancaria en la que el operador debe proporcionar un código, un nombre y una operación para realizar operación.*

*Clases de Equivalencia*

| Condición de Entrada | Clases Válidas | Clases Inválidas |
|---|---|---|
| **Código de Área**<br># de 3 dígitos que no empieza con 0 ni 1: | 1) 200≤ código ≤ 999 | 2) Código < 200.<br>3) Código > 999.. |
| **Nombre**<br>Para identificar la operación máximo 6 dígitos | 4) Menor a seis caracteres. | 5) Mayor 6 caracteres.<br>6) Entrada nula |
| **Orden**<br>Una de las Siguientes | **7)** “Cheque”<br>**8)** “Depósito”<br>**9)** “Pago factura”<br>**10)“**Retiro de fondos**”** | 11) Ninguna orden válida |

# Técnicas Pruebas Dinámicas: Caja Negra

Universidad Andrés Bello

## Ejemplo Métodos de prueba Partición de Equivalencia

*Casos de PRUEBA*

| Caso | Clase de Equivalencia | Codigo Area | Nombre | Orden | Resultado |
|---|---|---|---|---|---|
| 1 | 1,4,7 | 300 | Alejan | Cheque | Operación reaizada |
| 2 | 2,4,8 | 100 | Alejan | Deposito | Opreación no realizada |
| 3 | 3,4,11 | | | | Operación no realizada |
| 4 | | | | | |
| 5 | | | | | |
| 6 | | | | | |
| 7 | | | | | |
| 8 | | | | | |
| 9 | | | | | |
| 10 | | | | | |

# Técnicas Pruebas Dinámicas: Caja Negra

Universidad Andrés Bello

## MÉTODOS DE PRUEBA ANÁLISIS DE VALORES LIMITES

Se basa en la evidencia experimental de que los errores suelen aparecer con mayor probabilidad en los *extremos de los campos de entrada*.

Un análisis de las condiciones límites de las clases de equivalencia aumenta la eficiencia de la prueba.

**Condiciones límites :** Valores justo por encima y por debajo de los márgenes de la clase de equivalencia.

# Técnicas Pruebas Dinámicas: Caja Negra

Universidad Andrés Bello

## MÉTODOS DE PRUEBA ANÁLISIS DE VALORES LIMITES

El mayor número de errores ocurre en las fronteras del dominio de entrada y no en el centro.

Complementa a la partición de equivalencia

Las reglas para identificar las clases son:

1. Si una condición de entrada especifica un rango que deben generar casos para los extremos.
2. Si la condición de entrada especifica un número finito y consecutivo de valores, escribir casos para los números máximo, mínimo, uno más del máximo y uno menos del mínimo de valores
3. Usar la regla 1 para la condición de salida.
4. Usar la regla 2 para cada condición de salida.

# Técnicas Pruebas Dinámicas: Caja Negra

Universidad Andrés Bello

## MÉTODOS DE PRUEBA ANÁLISIS DE VALORES LIMITES

Generar tantos casos de prueba como sean necesarios para ejercitar las condiciones límites de las clases de equivalencia.

| Condiciones de la especificación | Obtención de los casos de prueba |
|---|---|
| 1. Rango de valores como condición de entrada | **1** caso que ejercite el valor máximo del rango |
| | **1** caso que ejercite el valor mínimo del rango |
| | **1** caso que ejercite el valor justo por encima del máximo del rango |
| | **1** caso que ejercite el valor justo por debajo del mínimo del rango |
| 2. Valor numérico específico como condición de entrada | **1** caso que ejercite el valor numérico específico |
| | **1** caso que ejercite el valor justo por encima del valor numérico específico |
| | **1** caso que ejercite el valor justo por debajo de valor numérico específico |
| 3. Rango de valores como condición de salida | Generar casos de prueba según el criterio 1 que ejerciten dichas condiciones de salida |
| 4. Valor numérico específico como condición de salida | Generar casos de prueba según el criterio 2 que ejerciten dichas condiciones de salida |
| 5. Estructura de datos como condición de salida o de entrada | **1** caso que ejercite el primer elemento de la estructura |
| | **1** caso que ejercite el último elemento de la estructura |

# Técnicas Pruebas Dinámicas: Caja Negra

Universidad Andrés Bello

## MÉTODOS DE PRUEBA ANÁLISIS DE VALORES LIMITES

### Ejemplo:

Programa que establece a partir de tres valores de entrada de que tipo de triángulo que es.

Condición: A+B>C and A+C>B and B+C>A

Según el método de partición equivalente deberíamos considerar una clase equivalente válida y otra no válida

{A=4, B=5, C=3}, {A=1, B=2, C= 5}

- No detectaría un error en A+B ≥ C
- AVL incluiría el caso {A=1, B=2, C=3}

# Técnicas Pruebas Dinámicas: Caja Negra

Universidad Andrés Bello

## MÉTODOS DE PRUEBA ANÁLISIS DE VALORES LIMITES

| Condición | Descripción de los casos de prueba |
|---|---|
| Entre 5 y 15 caracteres | **1** caso con nº de caracteres identificados=15 |
| | **1** caso con nº de caracteres identificados=5 |
| | **1** caso con nº de caracteres identificados=16 |
| | **1** caso con nº de caracteres identificados=4 |

| Condición | Casos de prueba |
|---|---|
| Entre 5 y 15 caracteres | Num-1-let-3---d3 (15) |
| | Numd3 (5) |
| | Num-1-letr-3---d3 (16) |
| | Nud3 (4) |

# Técnicas Pruebas Dinámicas: Caja Negra

Universidad Andrés Bello

## MÉTODOS DE PRUEBA DE COMPARACIÓN

- Se desarrollan versiones independientes de una aplicación con las mismas especificaciones.
- Probar todas las versiones con los mismos datos de prueba.
- Luego se ejecutan las versiones en paralelo y se hace una comparación en tiempo real de los resultados.

# Técnicas Pruebas Dinámicas: Caja Negra

Universidad Andrés Bello

## MÉTODOS DE PRUEBA PARTICIÓN DE EQUIVALENCIA

- ✓Enumerar una lista de equivocaciones que pueden cometer los desarrolladores.
- ✓Generar casos de prueba en base a dicha lista.
- ✓La generación de casos se obtiene en base a la intuición o la experiencia.

# Técnicas Pruebas Dinámicas: Caja Negra

Universidad Andrés Bello

## MÉTODOS DE PRUEBA PARTICIÓN DE EQUIVALENCIA

Ejemplo: Subrutina que ejecuta la búsqueda binaria

- Un caso que compruebe
  - La presencia de un solo dato en la lista.
  - Que las posiciones de la lista sea una potencia de dos
  - Que las posiciones de la lista sea uno más de una potencia de dos
  - Que las posiciones de la lista sea uno menos de una potencia de dos

# Técnicas Pruebas Dinámicas: Caja Negra

Universidad Andrés Bello

## MÉTODOS DE PRUEBAS ALEATORIAS

- ✓Se simula los posibles datos de entrada en la secuencia y frecuencia que pueden aparecer en la practica.
- ✓Si el proceso de generación se ha realizado correctamente, se crearán eventualmente todas las posibles entradas del programa en todas las posibles combinaciones y permutaciones.
- ✓Baja probabilidad de encontrar errores.

# Técnicas Pruebas Dinámicas: Caja Negra

Universidad Andrés Bello

## ESTRATEGIAS PARA EL DISEÑO DE CASO DE PRUEBAS

Cada Técnica:

- Evalúa una fuente diferente de errores.
- El diseño debe involucrar una combinación de estas técnicas.

Procedimiento:

- En todos los casos utilizar análisis de valores límites
- Complementar con casos de prueba derivados del método de partición equivalente
- Añadir nuevos casos mediante la conjetura de errores

# CONCLUSIONES

El proceso de prueba consiste en ejecutar el programa con el fin de localizar defectos.

La prueba es una actividad incompleta.

**Propósito de las Técnicas**: Reducir el número de casos de prueba sin mermar la efectividad de la prueba.

Existen dos enfoques a la hora de abordar el diseño de los casos de prueba:

- **Caja Negra**: evalúa la funcionalidad del sistema (lo que hace)
- **Caja Blanca:** evalúa la lógica interna (como lo hace)

# REFERENCIAS


- ✓The Art Of Software Testing G. Myers Capítulo 4
- ✓Software Testing and Quality Assurance Theory and Practice. Naik,Tripathy. Capítulo 4.
- ✓Manage Software Testing. Farrell y Vinay. Capítulo 11 y 12
- ✓Software Quality Assurance From Theory to Implementation (Galin). Capítulo 9
- ✓Software Engineering Theory And Practice S. L. Pfleeger. Capítulo 9.