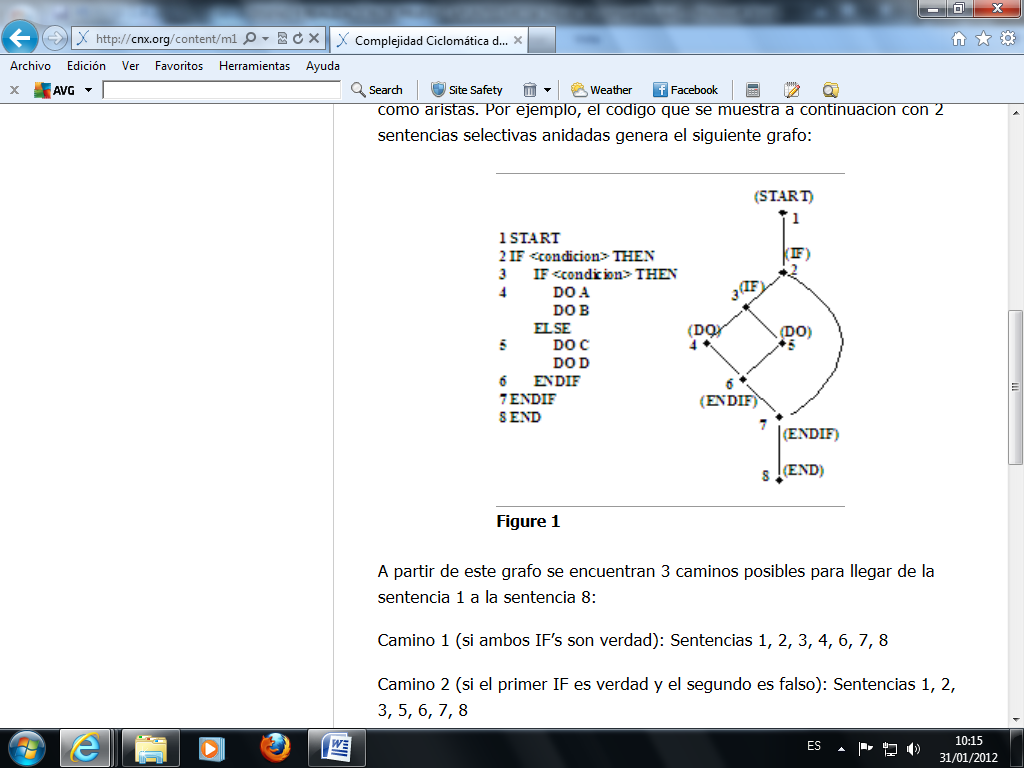
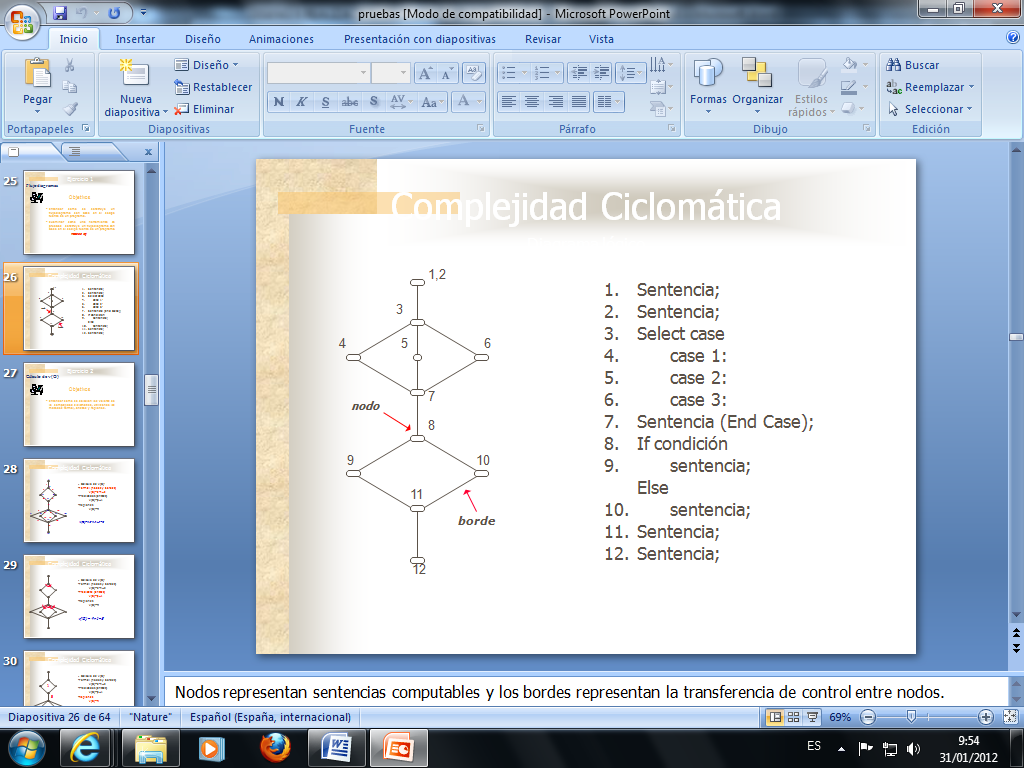
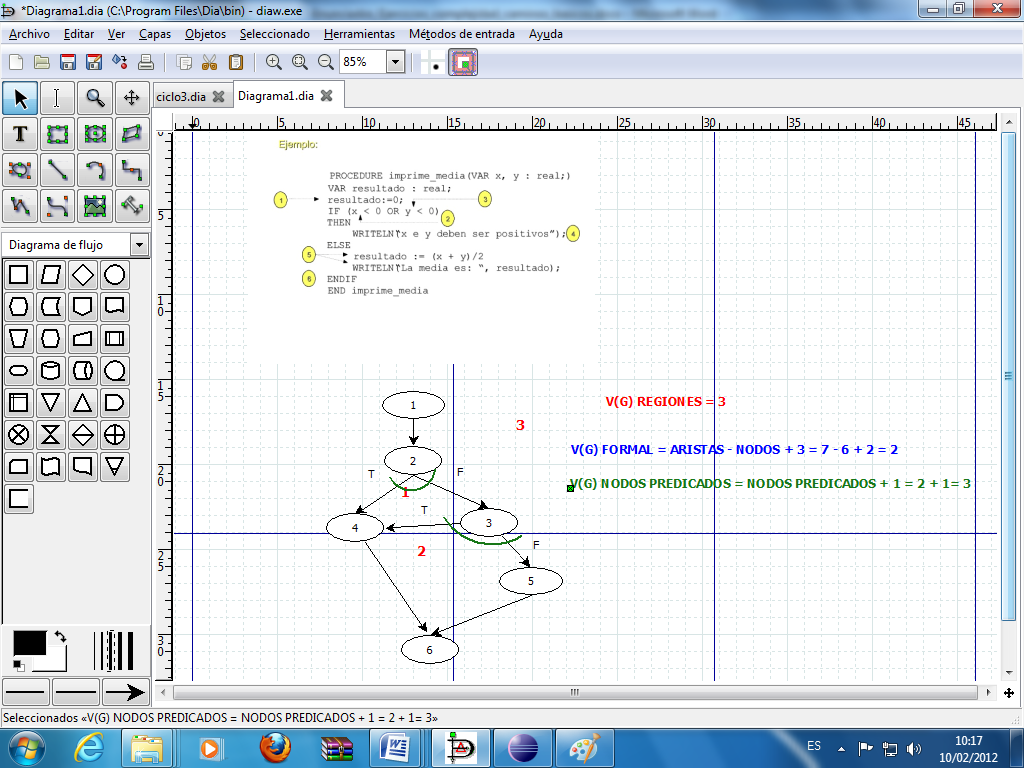
***Ejercicio 1*** :**Hallar el grafo del algoritmo y la complejidad ciclomática.**



***Ejercicio 2*** :**Hallar el grafo del algoritmo y la complejidad ciclomática.**



***Ejercicio 3*** :**Hallar el grafo del algoritmo y la complejidad ciclomática (ciclo3.dia).**



***Ejercicio 4*** :**Hallar el grafo del algoritmo y la complejidad ciclomática**

If (a>1) and (b>5) and (c<2) then

x=x+1;

else

x= x-1;

**SOLUCION**

If (a>1) and (b>5) and (c<2) then

4

3

2

1

x=x+1;

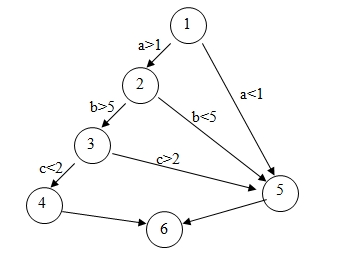
else

5

x= x-1;

6

....................



V(G) Regiones = 4

V(G) Formal = Aristas - Nodos + 2 = 8 – 6 + 2 = 4

V(G) Predicado = 3 + 1 = 4

Camino 1 = 1 – 2 – 3 – 4 - 6

Camino 2 = 1 – 2 – 5 – 6

Camino 3 = 1 – 2 – 3 – 5 – 6

Camino 4 = 1 – 5 – 6

|  |  |  |  |
| --- | --- | --- | --- |
| **Prueba** | **a** | **b** | **c** |
| Camino 1 | a>1 | b>5 | C<2 |
| Camino 2 | a>1 | b<5 |  |
| Camino 3 | a>1 | b>5 | c>2 |
| Camino 4 | a<1 |  |  |

***Ejercicio 5*** :***Dado el mismo fragmento de código del ejercicio anterior con dos cláusulas OR, calcular el número mínimo de casos de prueba que es necesario para que se cumpla la cobertura de condiciones.***

1. Señalamos en el código los nodos :

If (a>1) OR (b>5) OR (c<2) then

4

3

2

1

x=x+1;

else

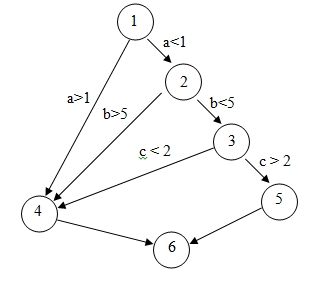
5

x= x-1;

6

....................

1. Dibujamos el grafo de flujo. En este caso al trarase de claúsulas OR en vez de AND el dibujo cambia notablemente:



1. Calculamos las complejidad ciclomática:

V(G)= 8 – 6 + 2 = 4 caminos independientes

Camino 1 🡺 1 – 4

Camino 2 🡺 1 – 2 – 4

Camino 3 🡺 1 – 2 – 3 – 4

Camino 4 🡺 1 – 2 – 3 – 5 – 6

Para que se cumpla el criterio de cobertura de condiciones, es decir, que cada condición de cada decisión tome el valor falso y verdadero al menos una vez, deberán ejecutarse los cuatro caminos críticos. Es dicir, el número mínimo de pruebas que deberemos realizar será cuatro.

***Ejercicio 6*** :***Dado el siguiente fragmento de programa en java:***

1. ***Calcular la complejidad ciclomática de McCabe V(G)***

Para calcular la complejidad ciclomática de McCabe, lo primero que tenemos que hacer es dibujar el grafo de flujo. Para esto seguiremos los siguientes pasos:

* 1. Señalamos en el código los pasos para dibujar el grafo de flujo

Import java.io.\*;

Public class Maximo

{

public static void main (String args[]) throws IOException

{

1

BufferedReader entrada = new BufferedReader (new InputStreamReader(System.in));

Int x,y,z,max;

System.out.println(“Introduce x,y,z: ”);

x = Integer.parseInt (entrada.readLine());

y = Integer.parseInt (entrada.readLine());

z = Integer.parseInt (entrada.readLine());

3

2

if (x>y && x>z)

4

max = x;

else

5

if (z>y)

6

max = z;

7

else

max = y;

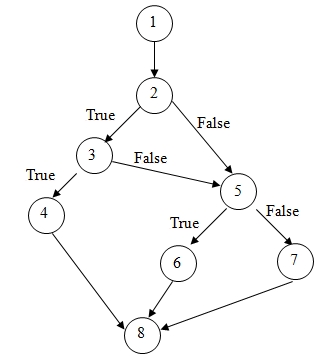
8

System.out.println (“El máximo es ”+ max);

}

}

* 1. Dibujamos el grafo de flujo



1. Calculamos la complejidad ciclomática, que da el número de caminos independientes :

V(G) = a – n + 2 = 10 – 8 + 2 = 4

V(G) = r = 4

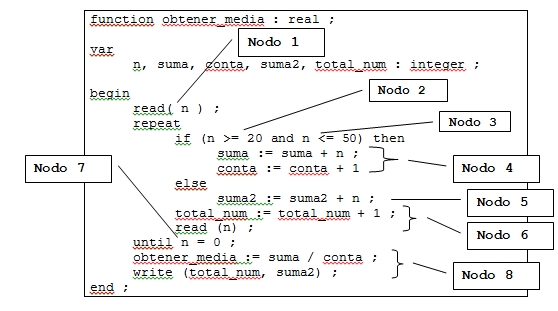
V(G) = c + 1 = 3 + 1 = 4

1. Mirando el grafo de flujo deducimoslos siguientes caminos independientes:

* Camino 1 🡺 1 - 2 – 3 – 4 – 8
* Camino 2 🡺 1 – 2 – 3 – 5 – 6 –8
* Camino 3 🡺 1 – 2 – 3 - 5 – 7 – 8
* Camino 4 🡺 1 – 2 – 5 – 7 - 8

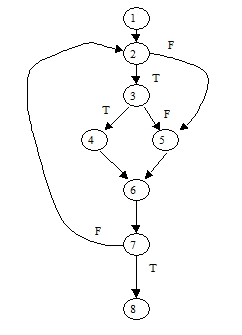
|  |  |  |  |
| --- | --- | --- | --- |
| **Prueba** | **X** | **Y** | **Z** |
| **Camino 1** | 5 | 2 | 1 |
| **Camino 2** | 5 | 2 | 8 |
| **Camino 3** | 5 | 10 | 1 |
| **Camino 4** | 5 | 10 | 7 |

***Ejercicio 7*** : **Hallar el grafo del algoritmo, la complejidad ciclomática y los caminos independientes**



La complejidad ciclomática del grafo anterior nos proporciona el número máximo de caminos linealmente independientes:

* V(G) = a – n + 2 = 10 – 8 + 2 = 4
* V(G) = c + 1 = 3 + 1 = 4
* V(G) = r = 4.

El conjunto de caminos linealmente independientes sería:

1-2-3-4-6-7-8 (1)

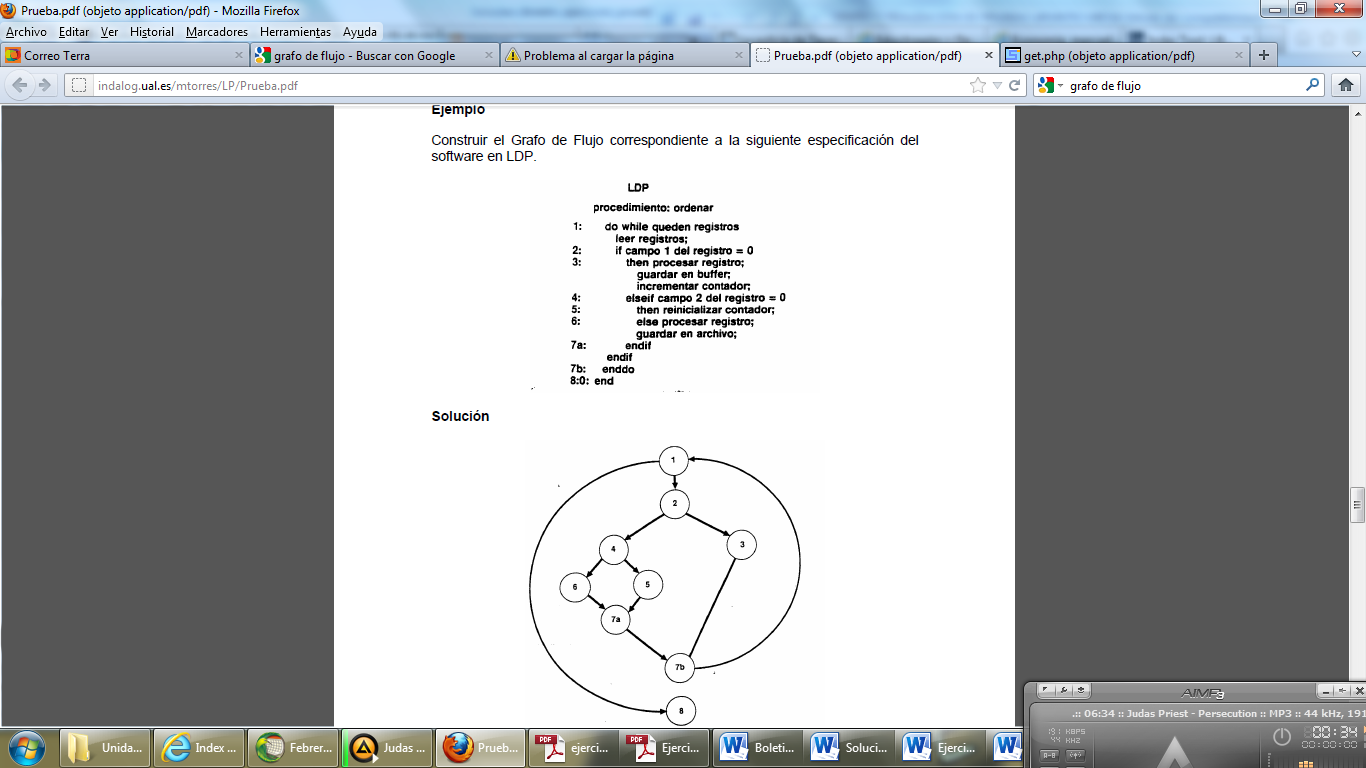
1-2-5-6-7-8 (2)

1-2-3-5-6-7-8 (3)

1-2-3-4-6-7-2- ... (4)

|  |  |  |
| --- | --- | --- |
| PRUEBA | Valor n nodo1 | Valor n nodo 6 |
| Camino1 | 20 | 0 |
| Camino2 | 19 | 0 |
| Camino3 | 51 | 0 |
| Camino4 | 50 | 1 |

***Ejercicio 8*** :



V(G )= 4

Camino 1: 1 - 2 – 3 – 7b – 1 – 8

Camino 2: 1 – 2 – 4 – 6 – 7a – 7b – 1 – 8

Camino 3: 1 – 2 – 4 – 5 – 7a – 7b – 1 – 8

Camino 4: 1 – 8

**DE OTRA MANERA MÁS CLARA**

|  |  |
| --- | --- |
| ej8.png | diagrama_ej8.png |

Camino 1: 1 - 2 – 3 – 8 - 9 - 1... -10

Camino 2: 1 – 2 – 4 – 6 – 7 – 8 - 9 – 1 - ….-10

Camino 3: 1 – 2 – 4 – 5 – 7 –8 –9 – 1 - ….-10

Camino 4: 1 – 10

***EJERCICIO 9***

TYPE ……

**1**

…

**3**

**2**

Suma = 0

Do while valor[i] <>-999 AND total.entrada <100 …..

**4**

Increment

**5**

**6**

If valor[i] >= minimo AND valor[i] <= maximo THEN

Increment total.valido by 1;

**7**

Suma=suma + valor[i];

Else

**8**

Skip

End if

**9**

Increment i by 1;

**10**

End do

**11**

If total.valido >0 THEN

**12**

Media = suma / total.valido;

Else

**13**

Media= -999;

**14**

End if

***EJERCICIO 10***

