

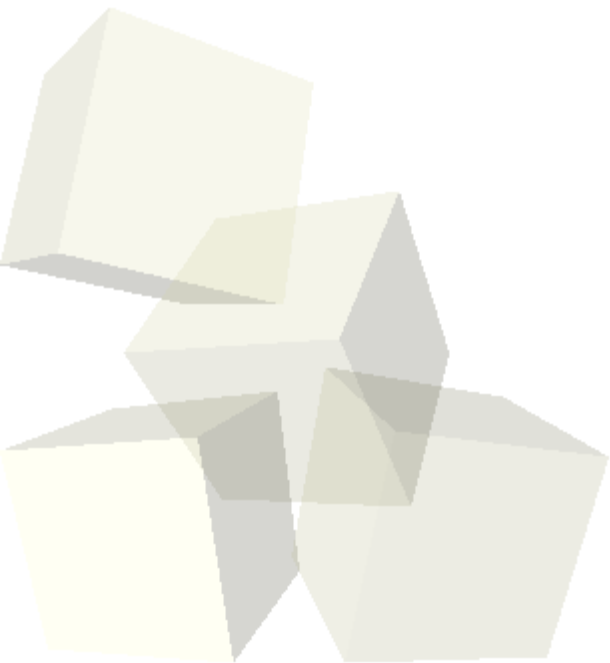


Tema 2

Algoritmos y programas Ejemplos

Informática
Grado en Física
Universitat de València

Francisco.Grimaldo@uv.es
Ariadna.Fuertes@uv.es





Programa: Nota final (1/4)

- Realizar un programa que solicite las notas obtenidas por un estudiante en cada una de las partes de una asignatura y calcule su **nota final** según los siguientes porcentajes:
 - a) Nota del examen final: 40 %
 - b) Nota de los exámenes parciales: 40 %
 - c) Nota de prácticas: 20 %
- La nota final del estudiante se calcula realizando la **media ponderada** si ha obtenido una nota mínima de 4 en cada una de las partes, o tomando **mínimo entre 4 y la media ponderada** en caso contrario.



Programa: Nota final (2/4)

■ 1º.- Análisis del problema:

- ♦ Datos de entrada: 3 notas (necesito 3 variables reales para almacenarlas)
- ♦ Datos de salida: nota final (otra variable real)

■ 2º.-Diseño/búsqueda del algoritmo

- ♦ Procesado de la información: ver si la nota es la media ponderada o si es el mínimo entre ese valor y 4.

■ 3º.-Programación del algoritmo

- ♦ Pseudocódigo u Organigrama

■ 4º.-Traducción/ejecución/comprobación (en lab.)



Programa: Nota final (3/4)

Pseudocódigo:

Variables:

a, b, c, m, n : reales

Inicio

Leer(a)

Leer(b)

Leer(c)

$m \leftarrow (0.4*a + 0.4*b + 0.2*c)$

Si₁ a>4 AND b>4 AND c>4 entonces

n ← m

sino₁

Si₂ m<4 entonces

n ← m

sino₂

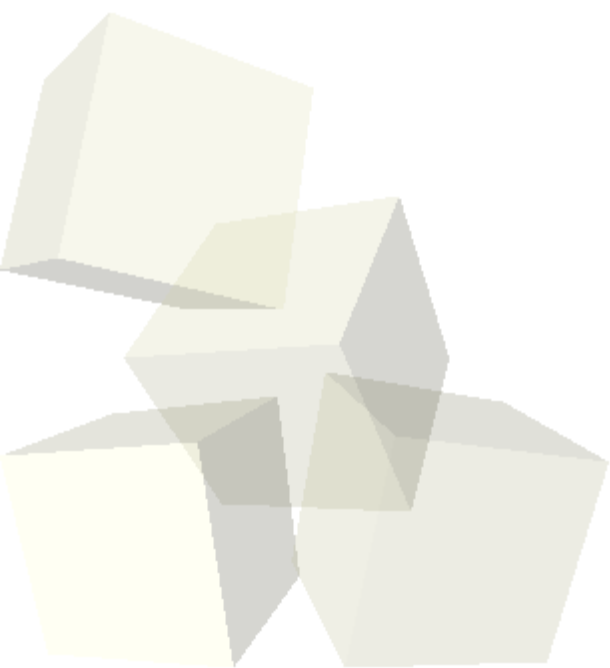
n ← 4

Fin_si₂

Fin_si₁

Escribir(n)

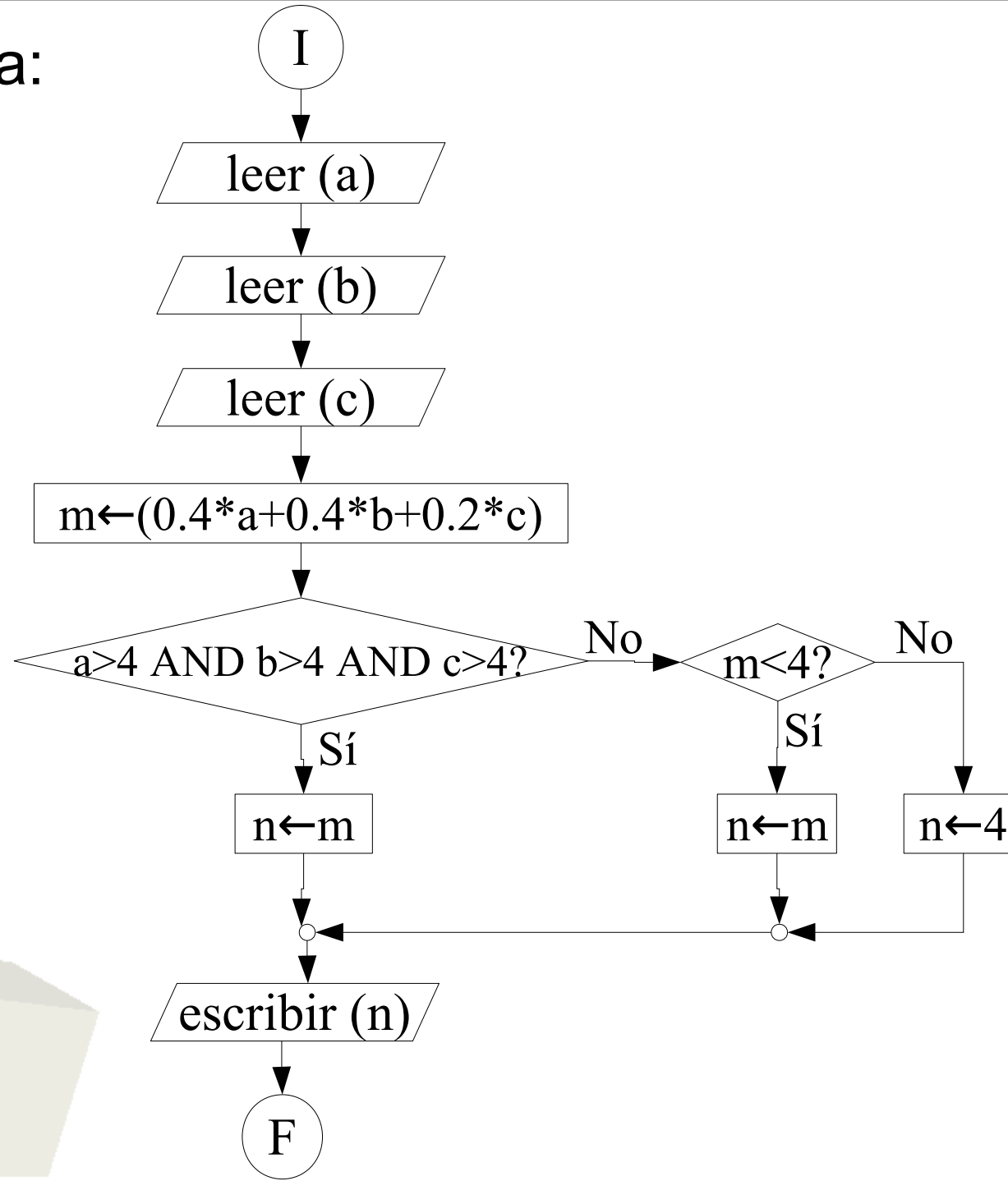
FIN





Programa: Nota final (4/4)

Organigrama:





Programa: Calificación (1/3)

- Escribid un programa que pida al usuario la nota de teoría y de prácticas de un alumno y muestre por pantalla la **nota final y la calificación**, teniendo en cuenta que:
 - ♦ La nota final corresponde al 60% de la nota de teoría y el 40% de la nota de prácticas.
 - ♦ La calificación podrá ser:
 - “Sobresaliente”, que es 9 o más,
 - “Notable”, que va desde 7 hasta 9,
 - “Aprobado”, que es mayor a 5 y menor a 7,
 - “Suspenso”, cuando sea menor a 5.



Programa: Calificación (2/3)

Pseudocódigo:

Variables:

a, b, n : reales

Inicio

Leer(a)

Leer(b)

$n \leftarrow (0.6*a + 0.4*b)$

Escribir(n)

Si₁ $n \geq 9$ entonces

Escribir("Sobresaliente")

sino₁

Si₂ $n \geq 7$ entonces

Escribir("Notable")

sino₂

Si₃ $n \geq 5$ entonces

Escribir("Aprobado")

sino₃

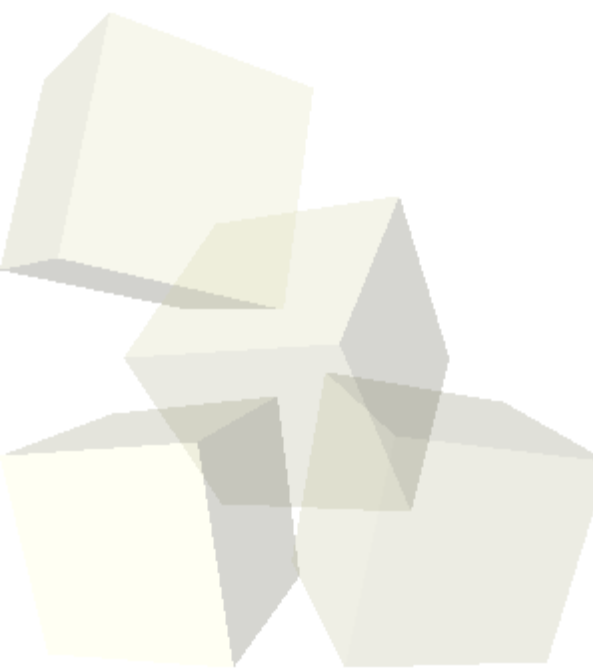
Escribir("suspenso")

Fin_si₃

Fin_si₂

Fin_si₁

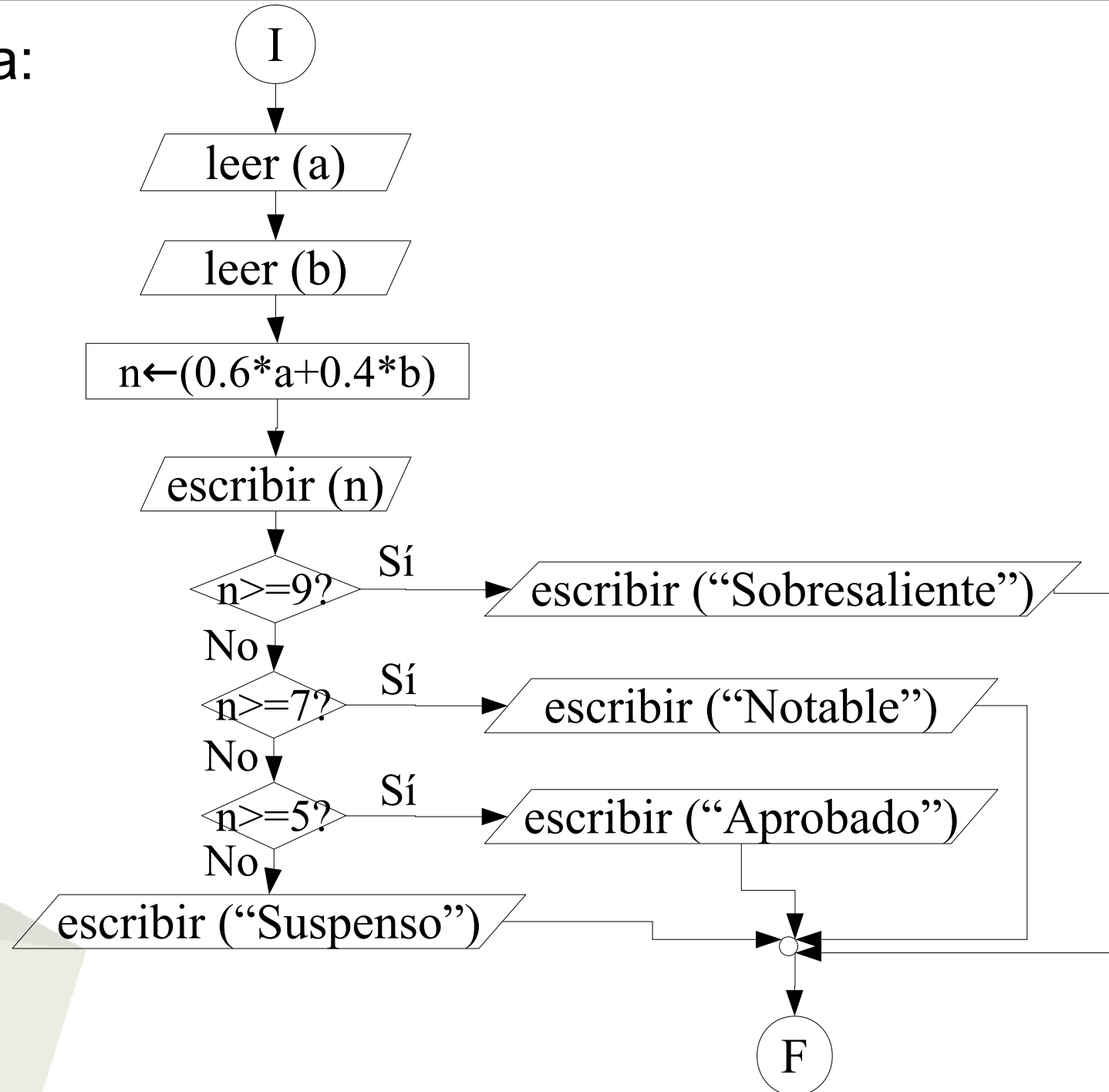
FIN





Programa: Calificación (3/3)

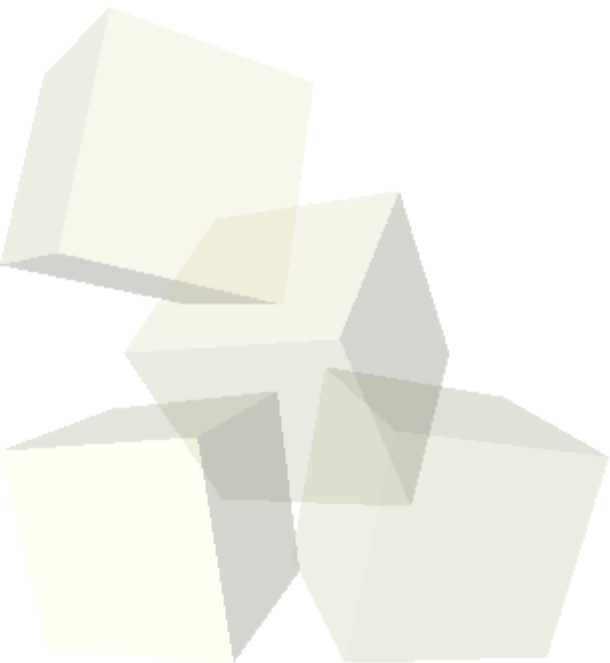
Organigrama:





Programa: Ordenación (1/3)

- Escribid un programa que pida tres números por teclado y diga:
 - ♦ Si están “Ordenados de manera ascendente”,
 - ♦ Si están “Ordenados de manera descendente”,
 - ♦ o si están “Desordenados”.





Programa: Ordenación (2/3)

Pseudocódigo:

Variables:

a, b, c : reales

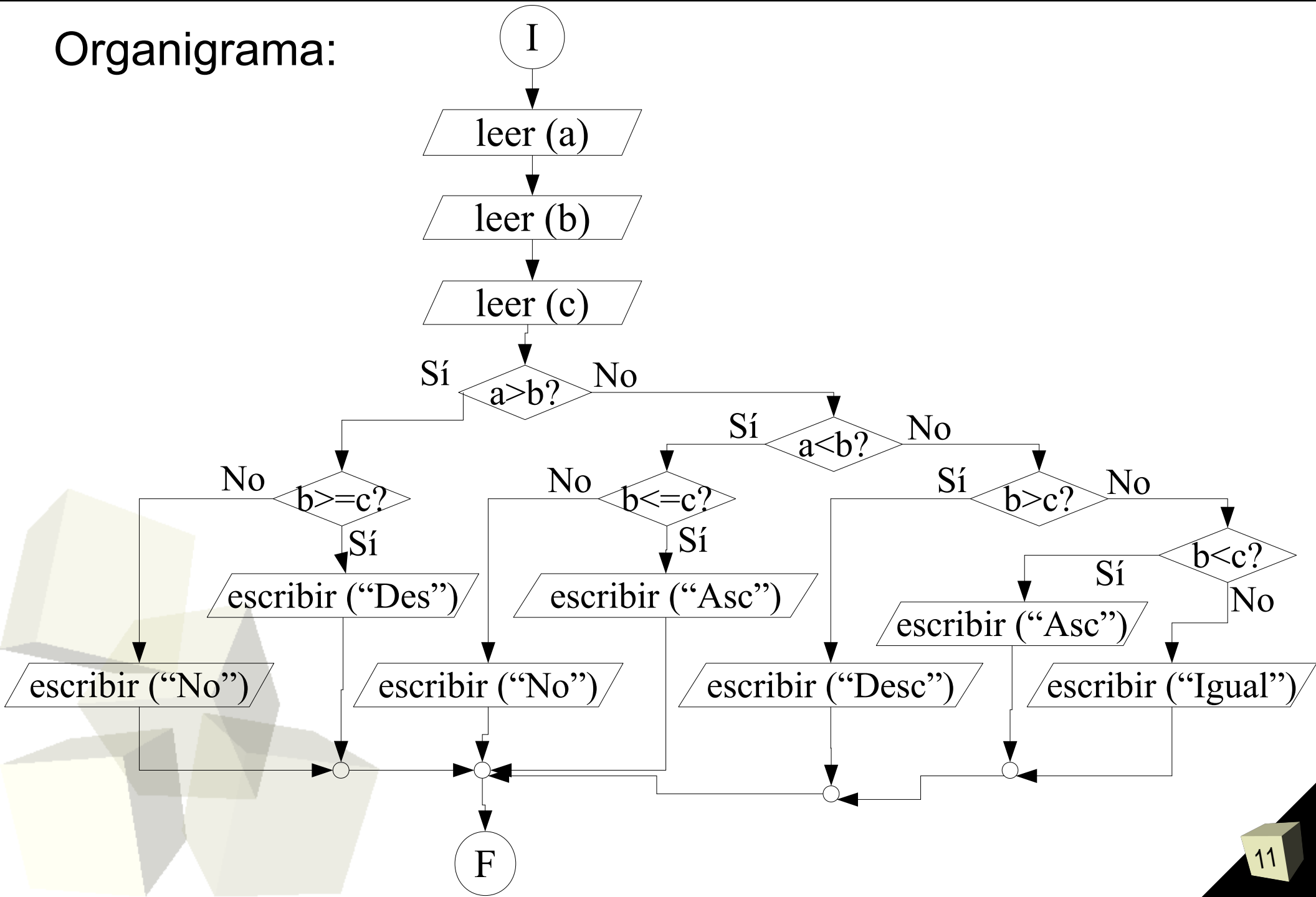
Nota: Al ser reales, no podemos comparar con el operador =, es decir, no son "exactamente iguales"

```
Inicio
Leer(a)
Leer(b)
Leer(c)
Si1 a>b entonces
    Si2 b>=c entonces
        Escribir("Descendente")
    sino2
        Escribir("No ordenados")
    Fin_si2
sino1
    Si3 a<b entonces
        Si4 b<=c entonces
            Escribir("Ascendente")
        sino4
            Escribir("No ordenados")
        Fin_si4
    sino3
        Si5 b>c entonces
            Escribir("Descendente")
        sino5
            Si6 b<c entonces
                Escribir("Ascendente")
            sino6
                Escribir("Son todos iguales")
            Fin_si6
        Fin_si5
    Fin_si3
Fin_si1
FIN
```



Programa: Ordenación (3/3)

Organigrama:





Programa: Triángulo (1/3)

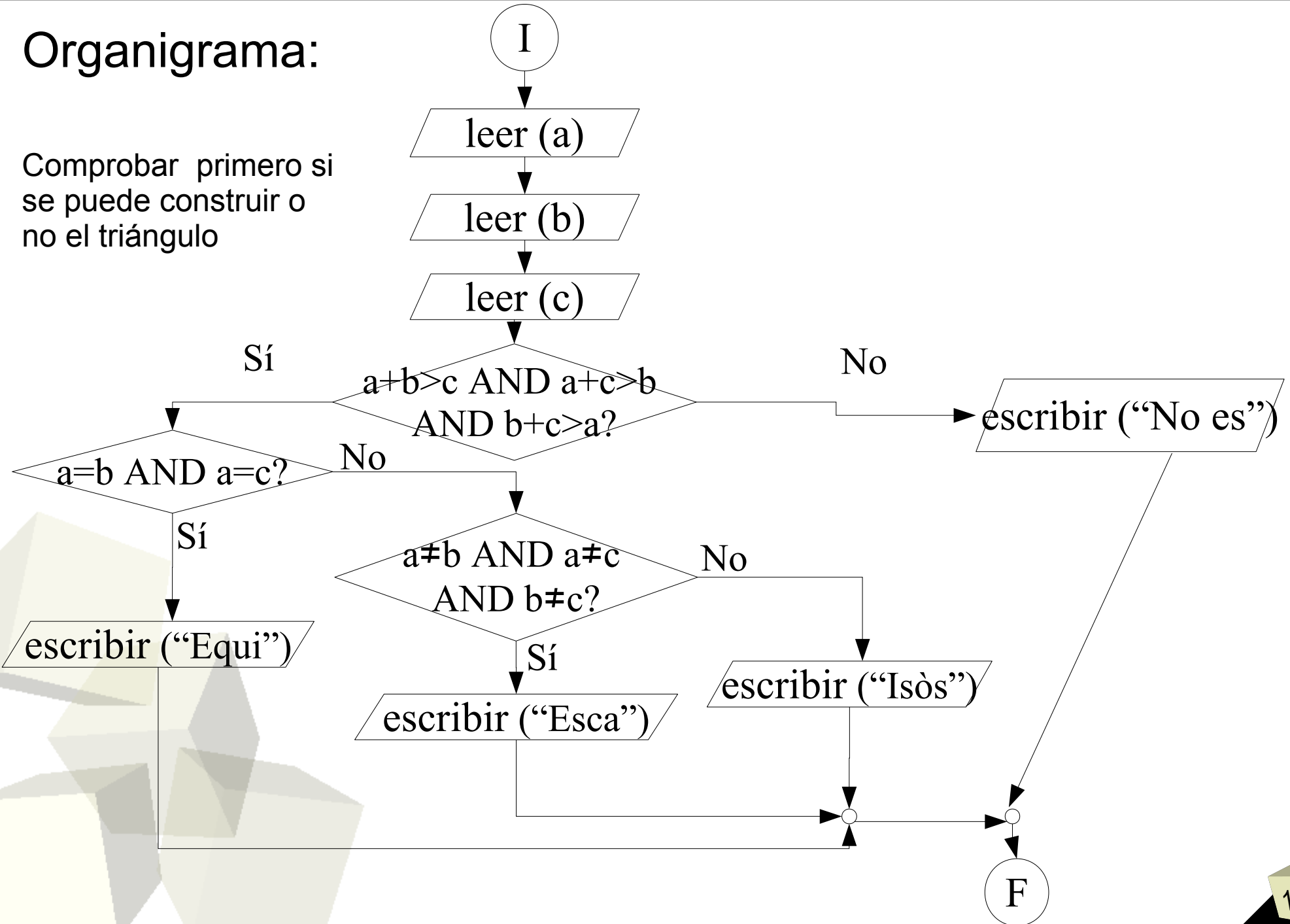
- Un triángulo cumple siempre la siguiente regla:
“La suma de las longitudes de dos lados cualesquiera es estrictamente mayor que la longitud del tercer lado”.
- Cuando los tres lados son diferentes decimos que el triángulo es escaleno, cuando los tres son iguales lo llamamos equilátero y cuando al menos dos lados son iguales el triángulo es isósceles.
- Cread un programa que pida al usuario la longitud de tres lados y compruebe si a partir de ellos se puede **formar un triángulo y de qué tipo sería.**



Programa: Triángulo (2/3)

Organigrama:

Comprobar primero si se puede construir o no el triángulo





Programa: Triángulo (3/3)

Pseudocódigo:

Variables:

a, b, c : enteros

Inicio

Leer(a)

Leer(b)

Leer(c)

Si₁ $a+b>c$ AND $a+c>b$ AND $b+c>a$ entonces

Si₂ $a=b$ AND $a=c$ entonces

Escribir("Equilatero")

sino₂

Si₃ $a\neq b$ AND $a\neq c$ AND $b\neq c$ entonces

Escribir("Escaleno")

sino₃

Escribir("Isosceles")

Fin_si₃

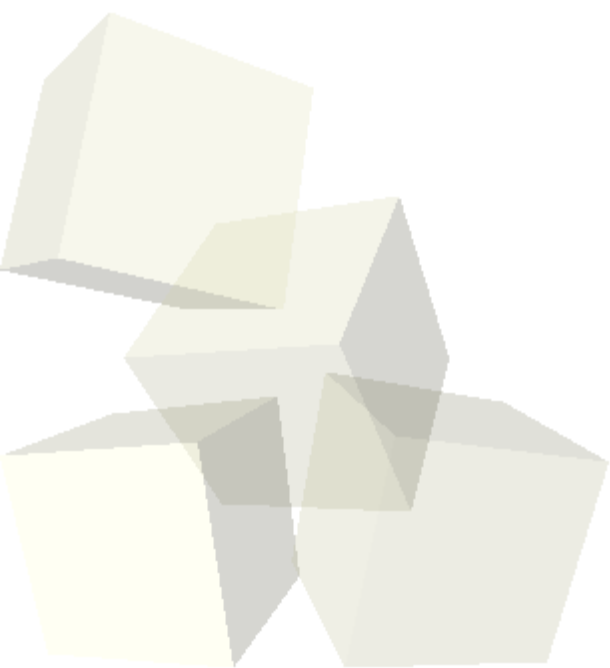
Fin_si₂

sino₁

Escribir("No se puede construir un triangulo")

Fin_si₁

FIN





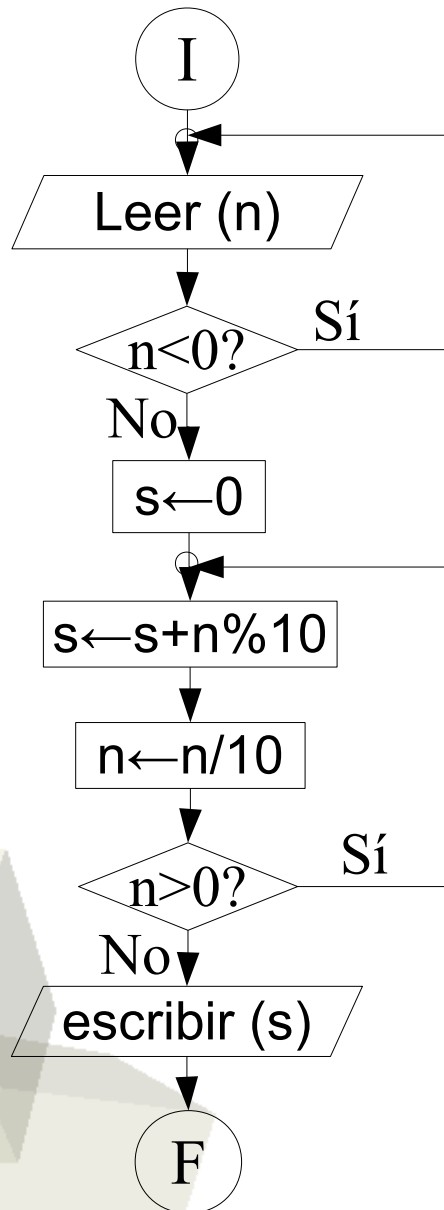
Programa: Sumar cifras (1/2)

- Realizar un programa que devuelva la **suma de las cifras de un número entero positivo**.
- **1º.- Análisis del problema:**
 - ♦ Datos de entrada: 1 número entero positivo (necesito 1 variable entera para almacenarlo)
 - ♦ Datos de salida: suma de las cifras (otra variable entera)
- **2º.-Diseño/búsqueda del algoritmo**
 - ♦ No permitir que se introduzca un número negativo. Cuando sea positivo, extraer cada una de las cifras dividiendo sucesivamente por 10 y quedándonos con el resto (que es la cifra que buscamos) hasta que no se pueda dividir más.
- **3º.-Programación del algoritmo**



Programa: Sumar cifras (2/2)

Organigrama:



Pseudocódigo:

Variables:

n, s : enteras

Inicio

Hacer

Leer(n)

Mientras₁ (n < 0)

s ← 0

Hacer

s ← s + n % 10

n ← n / 10

Mientras₂ (n > 0)

Escribir(s)

FIN

Programa: Secuencia numérica (1/3)

- Escribir un programa que solicite al usuario un **número entero positivo n** (si se introduce un valor negativo se volverá a pedir otro número hasta que este sea positivo) y muestre una **secuencia de salida** similar a (para $n=6$):

123456

12345

1234

123

12

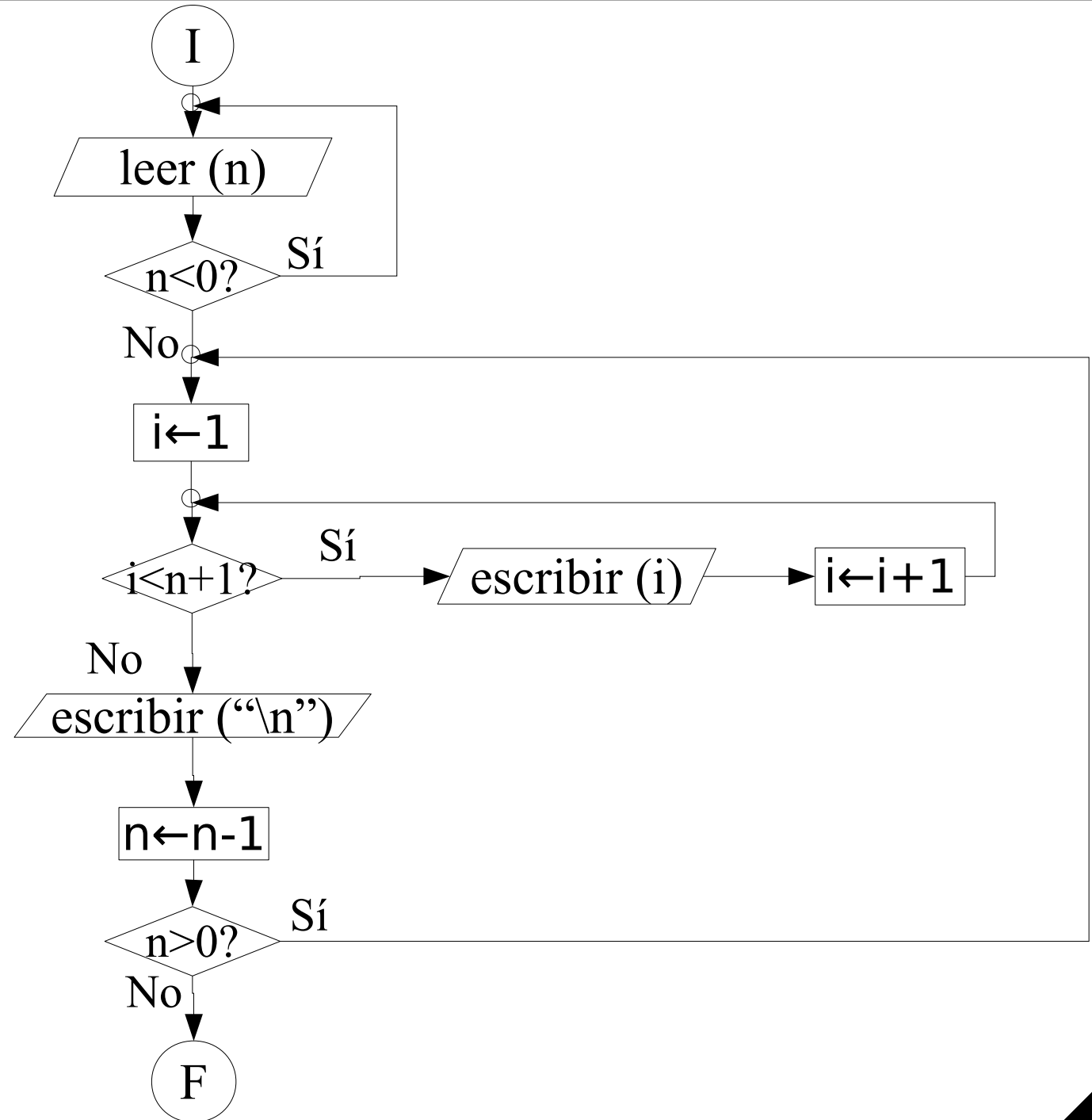
1



Programa: Secuencia numérica (2/3)

Organigrama:

Hacer...Mientras





Programa: Secuencia numérica (3/5)

Pseudocódigo:

Hacer...Mientras

Variables:

n, i : enteras

Inicio

Hacer

Leer(n)

Mientras₁ (n<0)

Hacer

Desde i←1 hasta n hacer

Escribir(i)

i ← i+1

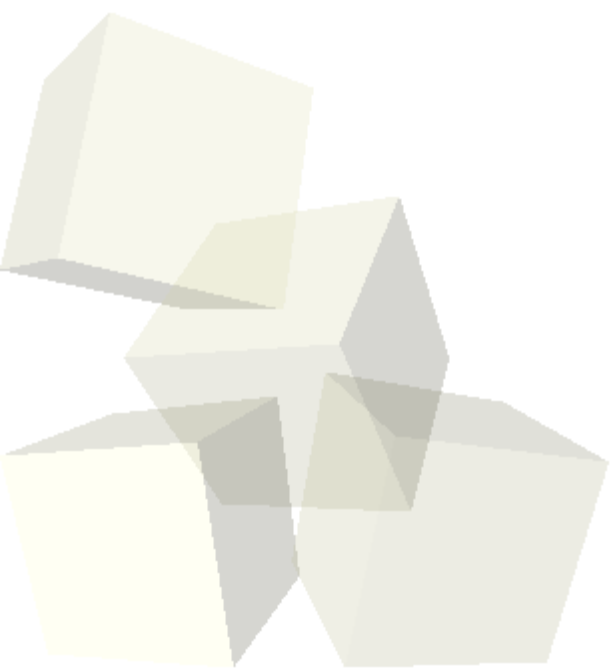
fin_desde

Escribir("\n")

n ← n - 1

Mientras₂ (n>0)

FIN

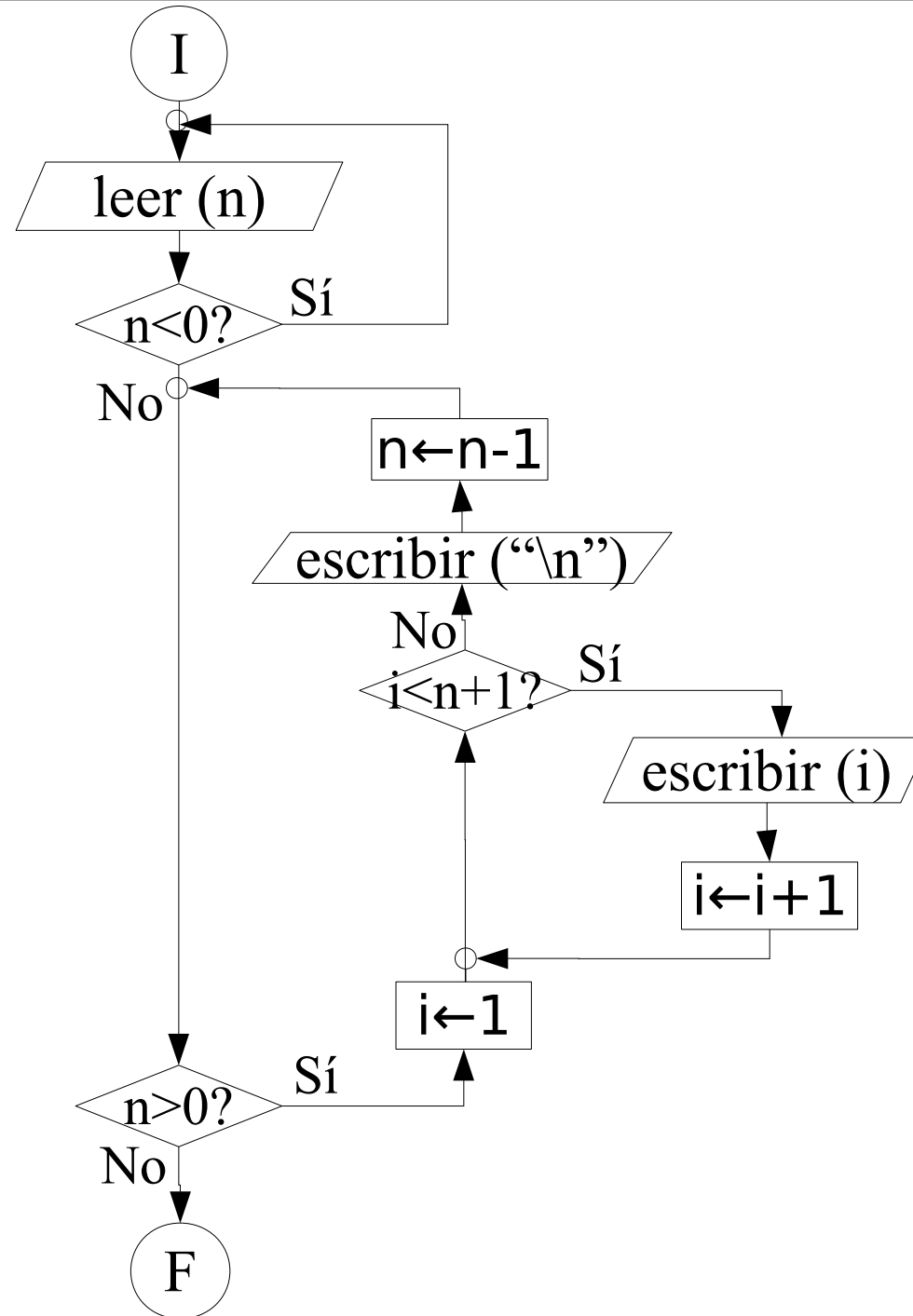




Programa: Secuencia numérica (4/5)

Organigrama:

Mientras...Hacer





Programa: Secuencia numérica (5/5)

Pseudocódigo:

Mientras...Hacer

Variables:

n, i : enteras

Inicio

Hacer

Leer(n)

Mientras₁ ($n < 0$)

Mientras₂ ($n > 0$) hacer

Desde $i \leftarrow 1$ hasta n hacer

Escribir (i)

$i \leftarrow i + 1$

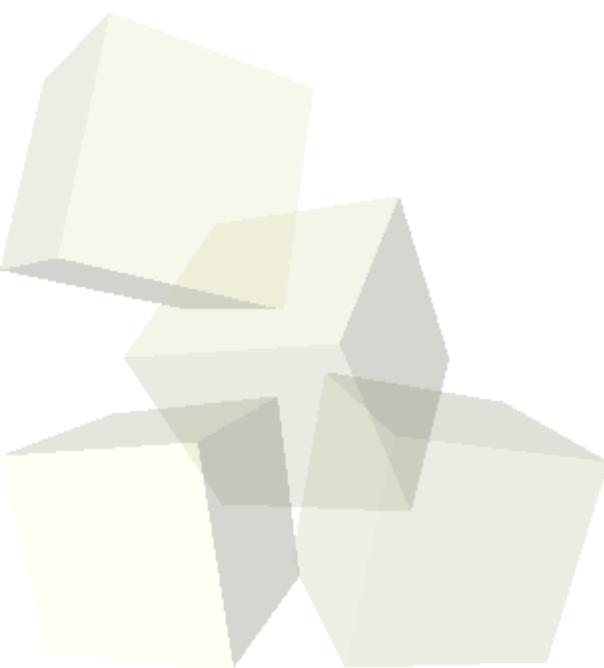
fin_desde

Escribir("\n")

$n \leftarrow n - 1$

Fin_mientras₂

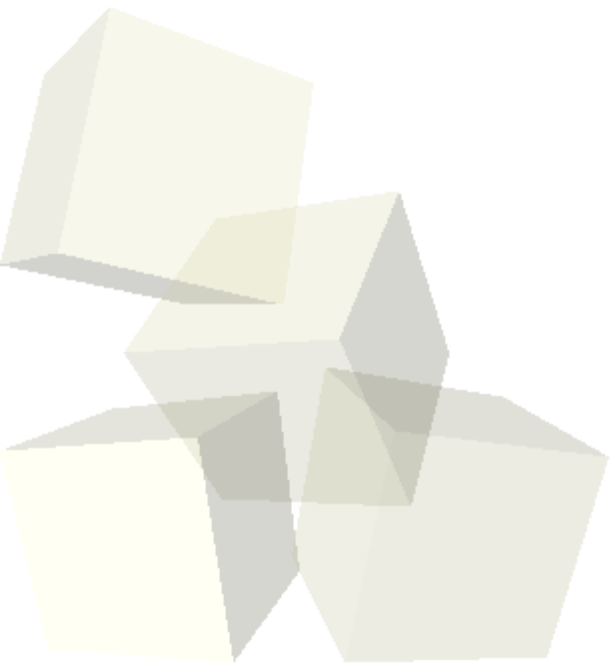
FIN





Programa: Cuenta números (1/5)

- Escribir un programa que lea números enteros positivos hasta que el usuario introduzca un 0 (“valor centinela”).
- Además, el programa mostrará por pantalla la cantidad de números leídos, el mayor, el menor y el valor medio de los números leídos.

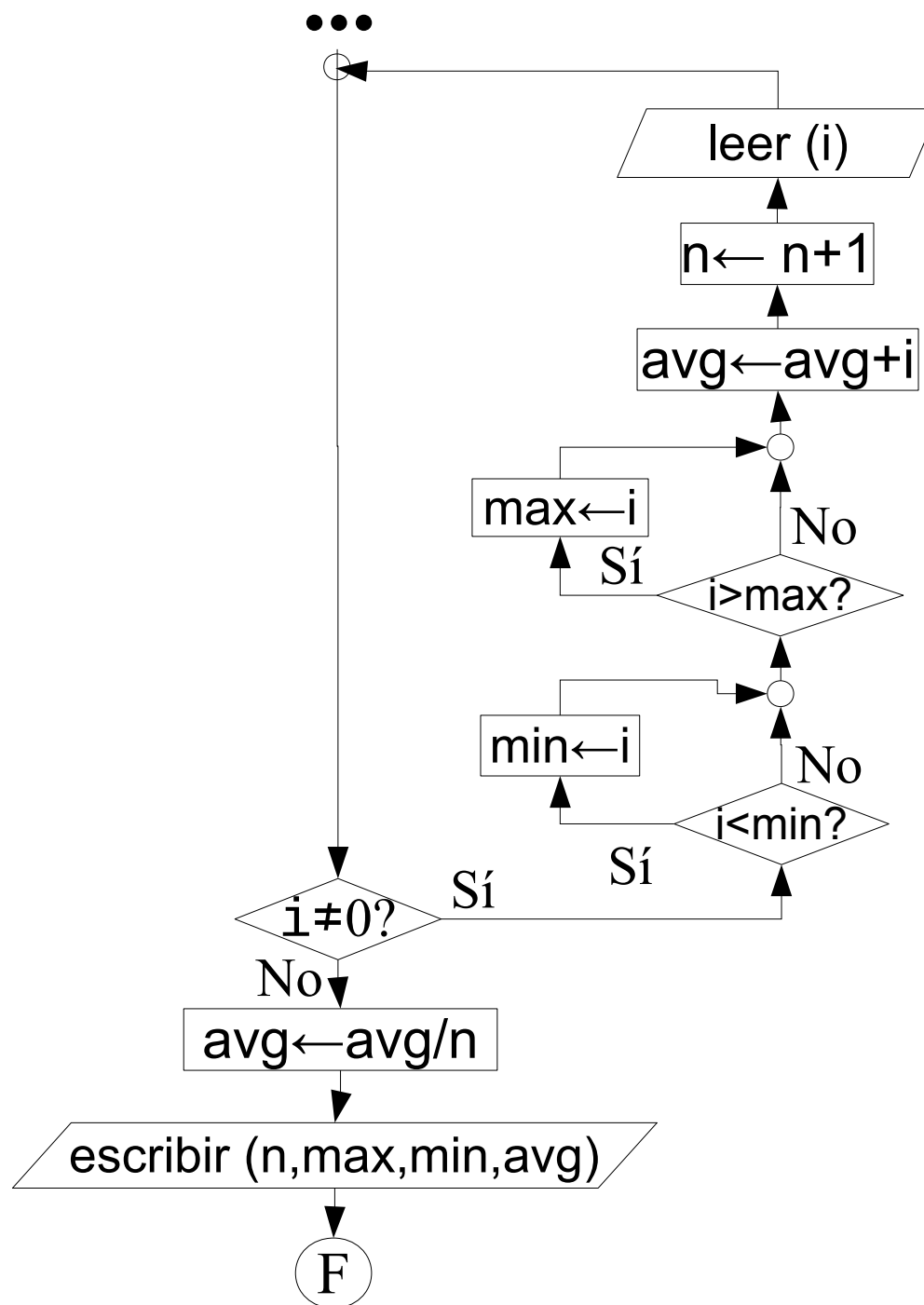
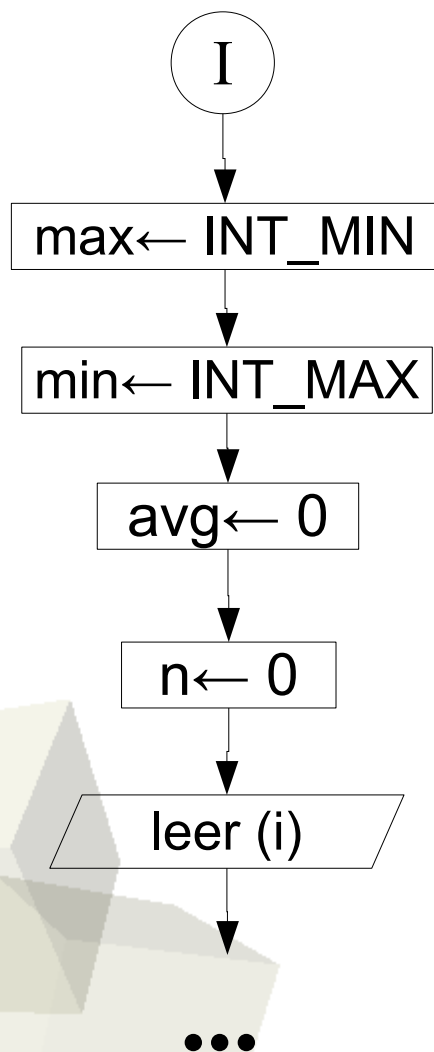




Programa: Cuenta números (2/5)

Organigrama:

1ª aproximación: Inicializar el mínimo a algo muy pequeño y el máximo a algo muy grande





Programa: Cuenta números (3/5)

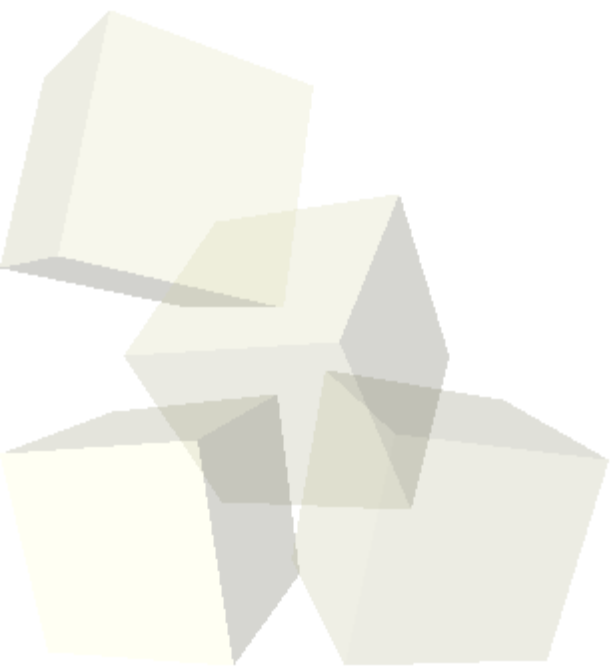
Pseudocódigo:

Variables:

n, max, min, i : enteros
avg : real

Inicio

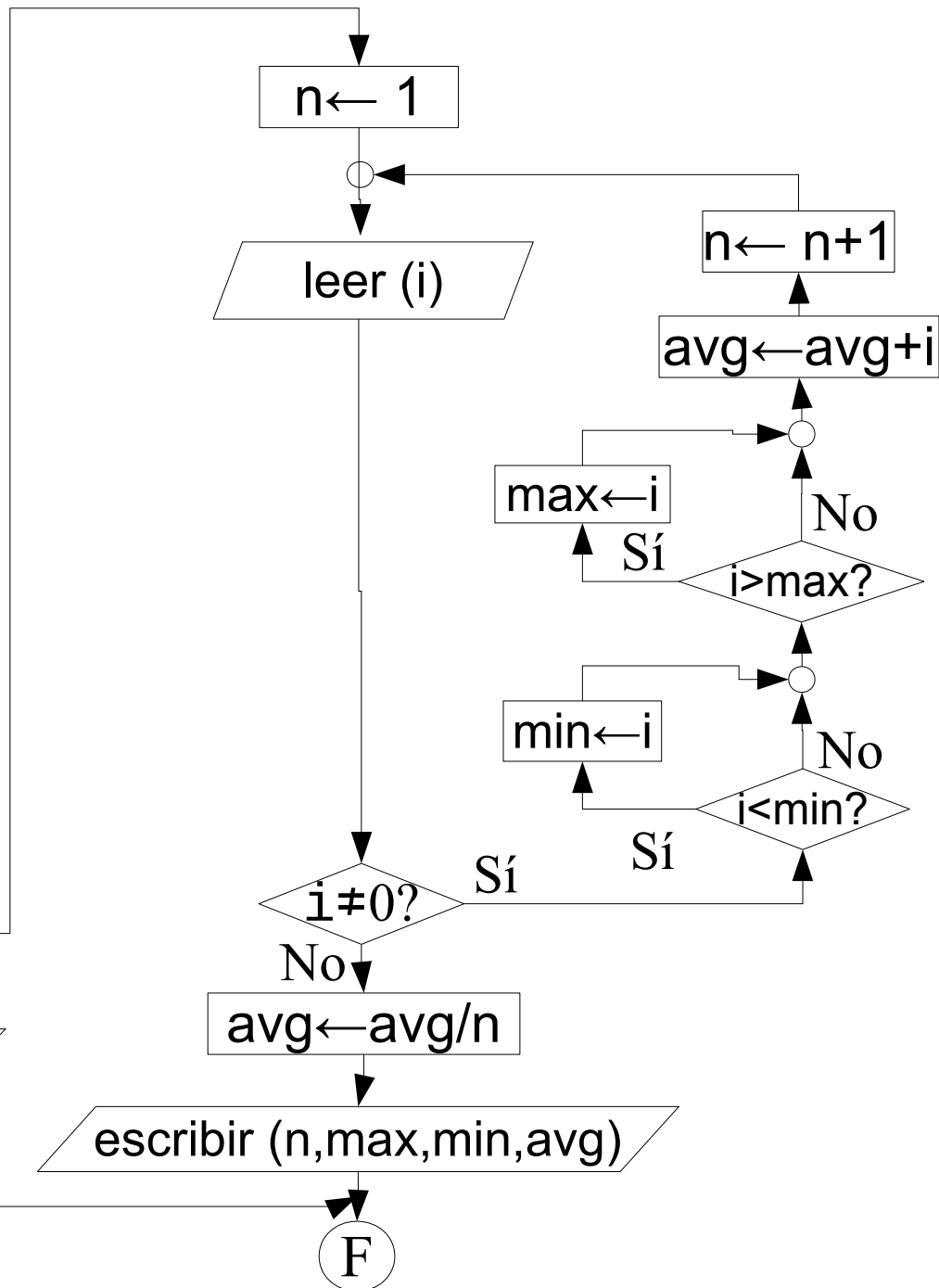
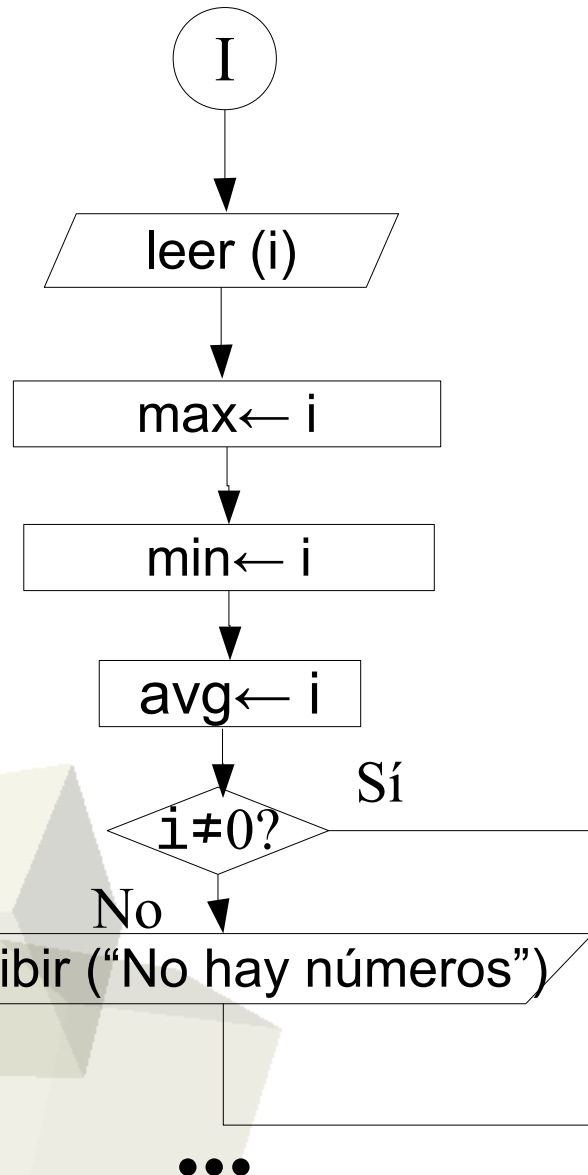
```
max ← INT_MIN
min ← INT_MAX
avg ← 0
n ← 0
Leer(i)
Mientras (i ≠ 0) hacer
    Si  $i < \text{min}$  entonces
        min ← i
    fin_si1
    Si  $i > \text{max}$  entonces
        max ← i
    fin_si2
    avg ← avg + i
    n ← n + 1
    Leer(i)
Fin_mientras
avg ← avg / n
Escribir(n, max, min, avg)
FIN
```



Programa: Cuenta números (4/5)

Organigrama:

2ª aproximación: Inicializar los valores máximo y mínimo al primer dato





Programa: Cuenta números (5/5)

Pseudocódigo:

Variables:

n, max, min, i : enteros
avg : real

Inicio

```
Leer(i)
max ← i
min ← i
avg ← i
Si i ≠ 0 entonces
    n ← 1
    Leer(i)
    Mientras (i ≠ 0) hacer
        Si  $i < \min$  entonces
            min ← i
        fin_si1
        Si  $i > \max$  entonces
            max ← i
        fin_si2
        avg ← avg + i
        n ← n + 1
        Leer(i)
    Fin_mientras
    avg ← avg / n
    Escribir(n, max, min, avg)
Si_no
    Escribir("No hay números")
Fin_si
```

FIN



Programa: Fibonacci (1/3)

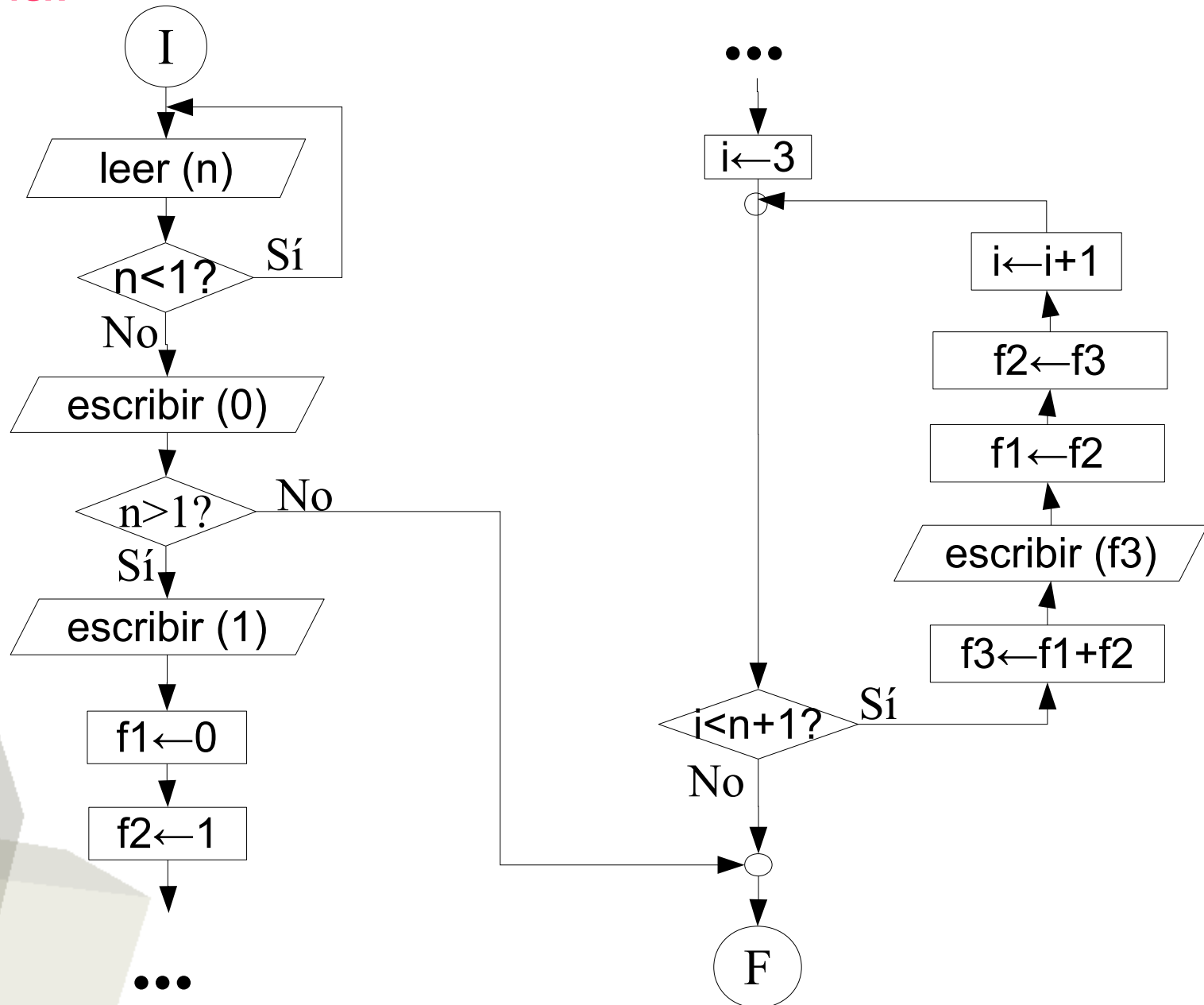
- La sucesión de Fibonacci es una secuencia infinita en la cual el primer elemento es un 0, el segundo es un 1 y el resto de elementos se pueden expresar como la suma de los dos elementos anteriores.
- Escribir un programa que muestre por pantalla los **primeros n términos de la sucesión de Fibonacci**, donde n es un número obtenido desde teclado.





Programa: Fibonacci (2/3)

Organigrama:





Programa: Fibonacci (3/3)

Pseudocódigo:

Variables:

n, i, f1, f2, f3 : enteras

Inicio

Hacer

Leer(n)

Mientras₁ (n<1) hacer

Escribir(0)

Si n > 1 entonces

Escribir(1)

f1 ← 0

f2 ← 1

Desde i←3 hasta n hacer

f3 ← f2+f1

Escribir(f3)

f1←f2

f2←f3

i ← i+1

Fin_desde

Fin_si

FIN

