Informática para la Ingeniería
Metodología de la programación

Ejercicios de Análisis y diseño estructurado para no Informáticos

(Ed. 2020-1)

Dr. Manuel Pérez Cota
Área de Linguaxes e Sistemas Informáticos
Departamento de Informática
Universidade de Vigo – 2020

Índice:

-Introducción.

En este documento se presenta la evolución real de un ciclo de vida de software, partiendo de la fase de definición del problema y terminando las pruebas de los programas. Es importante señalar que los ejercicios están desarrollados con la idea de explicar conceptos, por lo que no todos tienen características de completitud, así mismo se han diseñado con la idea de ir avanzando en complejidad, con el fin de que los conceptos de programación se vean claramente.

Es importante indicar que los diagramas son genéricos, es decir, pueden valer para cualquier lenguaje por lo que, a la hora de codificar, pueden detectarse pequeñas modificaciones que dependen del lenguaje utilizado.

Este documento incluye la codificación de los programas en lenguaje Phyton (v3.9.0) corriendo bajo Windows 10-Pro, realizados bajo la plataforma Thonny 3.3.0 de la Universidad de Tartu, Estonia.

Se visualiza, la codificación y ejecución de los programas en modo consola para simplificar. Se recuerda, nuevamente, que están escritos a modo de ejemplo de aprendizaje creciente, de forma que el alumno pueda ir aprendiendo las características de la forma de programar más fácilmente.

-Ejercicio 1:

Problema: Diseñar un programa que sea capaz de sumar dos números reales.

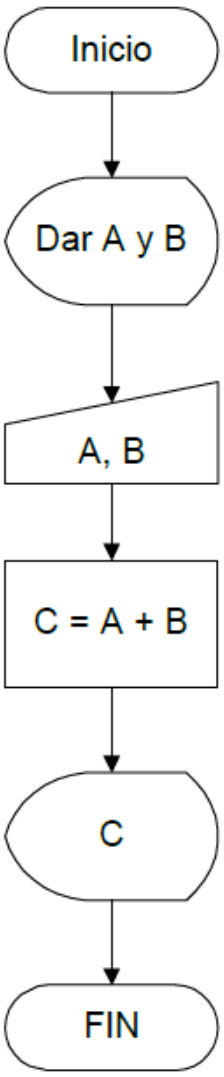
Análisis: Empiezo con un ejemplo que traduzco a una fórmula:

$$\begin{array}{r} 3{,}2 \\ +4{,}3 \\ \hline 7{,}5 \end{array} \quad \Rightarrow \quad \begin{array}{r} A \\ +B \\ \hline C \end{array}$$

Informáticamente podría ser: C = A + B

Restricciones:
No se ven restricciones.

Diseño (diagrama de flujo):



Codificación:

```
# Programa para calcular la suma de 2 números reales
# Autor: Dr. Manuel Pérez Cota
# v20201126
a = float(input ("Dame el valor de a "))
b = float(input ("Dame el valor de b "))
c = a + b
print ("El resultado es ", c)
```

Prueba en ejecución del programa:

```
>>> %Run probasuma20201126.py

Dame el valor de a 5.3
Dame el valor de b 3.5
El resultado es  8.8

>>>
```

-Ejercicio 2:

Problema: Diseñar un programa que sea capaz de restas dos números reales.

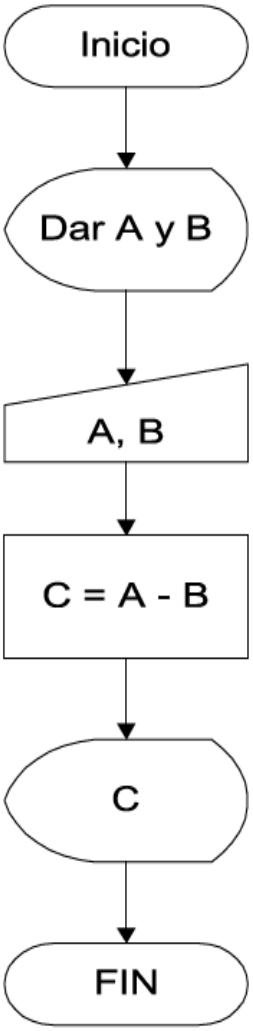
Análisis: Empiezo con un ejemplo que traduzco a una fórmula:

$$\begin{array}{r} 3{,}2 \\ -4{,}3 \\ \hline -1{,}1 \end{array} \Rightarrow \begin{array}{r} A \\ -B \\ \hline C \end{array}$$

Informáticamente podría ser: C = A - B

Restricciones:
No se ven restricciones, aunque se debe tener en consideración lo siguiente:
Obsérvese que, en el caso de la resta, tenemos que preocuparnos de qué variable es el minuendo y qué variable es el sustraendo pues, no da igual restar A de B que B de A, en este caso A es el minuendo y B es el sustraendo.

Diseño (diagrama de flujo):



Codificación:

```
# Programa para calcular la diferencia de 2 números reales
# Autor: Dr. Manuel Pérez Cota
# v201201126
a = float(input ("Dame el valor de a "))
b = float(input ("Dame el valor de b "))
c = a - b
print ("El resultado es ", c)
```

Pruebas del programa (corresponden a ejecuciones seguidas):

```
>>> %Run probaresta20201126.py

Dame el valor de a 5.3
Dame el valor de b 3.5
El resultado es  1.7999999999999998

>>> %Run probaresta20201126.py

Dame el valor de a 3.5
Dame el valor de b 5.3
El resultado es  -1.7999999999999998

>>>
```

Nota: ¡Ojo al error de exactitud de la utilización de los números reales en Phyton!

-Ejercicio 3:

Problema: Diseñar un programa que sea capaz de multiplicar dos números reales.

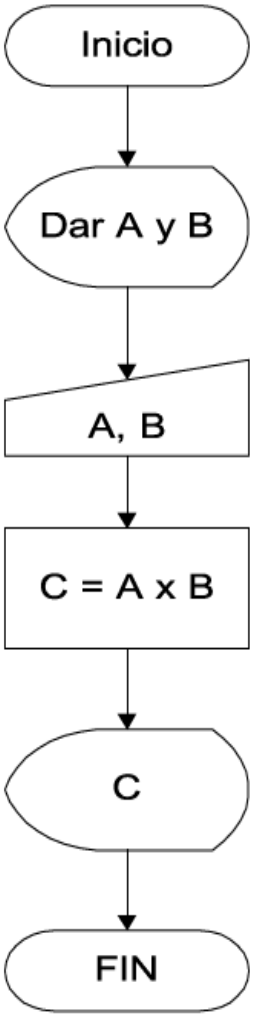
Análisis: Se empieza con un ejemplo que se traduce en una fórmula:

$$\begin{array}{r} 3{,}2 \\ x\ 4{,}3 \\ \hline 96 \\ 128 \\ \hline 16{,}96 \end{array} \quad \Rightarrow \quad \begin{array}{r} A \\ x\ B \\ \hline C \end{array}$$

Informáticamente podría ser: C = A * B

Restricciones:
No se ven restricciones.
A diferencia del anterior, de la resta, aquí si da igual quién sea el multiplicando y quién el multiplicador, pues el orden de los factores no altera el producto.

Diseño (diagrama de flujo):



Codificación:

```
# Programa para calcular el producto de 2 números reales
# Autor: Dr. Manuel Pérez Cota
# v20201126
a = float(input ("Dame el valor de a "))
b = float(input ("Dame el valor de b "))
c = a * b
print ("El resultado es ", c)
```

Prueba del programa:

```
>>> %Run probamultiplica20201126.py

Dame el valor de a 5.3
Dame el valor de b 3.5
El resultado es  18.55

>>>
```

-Ejercicio 4:

Problema: Diseñar un programa que sea capaz de dividir dos números reales.

Análisis: Se empieza con un ejemplo que se traduce en una fórmula:

```
35,5 | 10,0        A | B
05 5 |  3,5        R | C
 0 5 |
```

Informáticamente podría ser: C = A / B

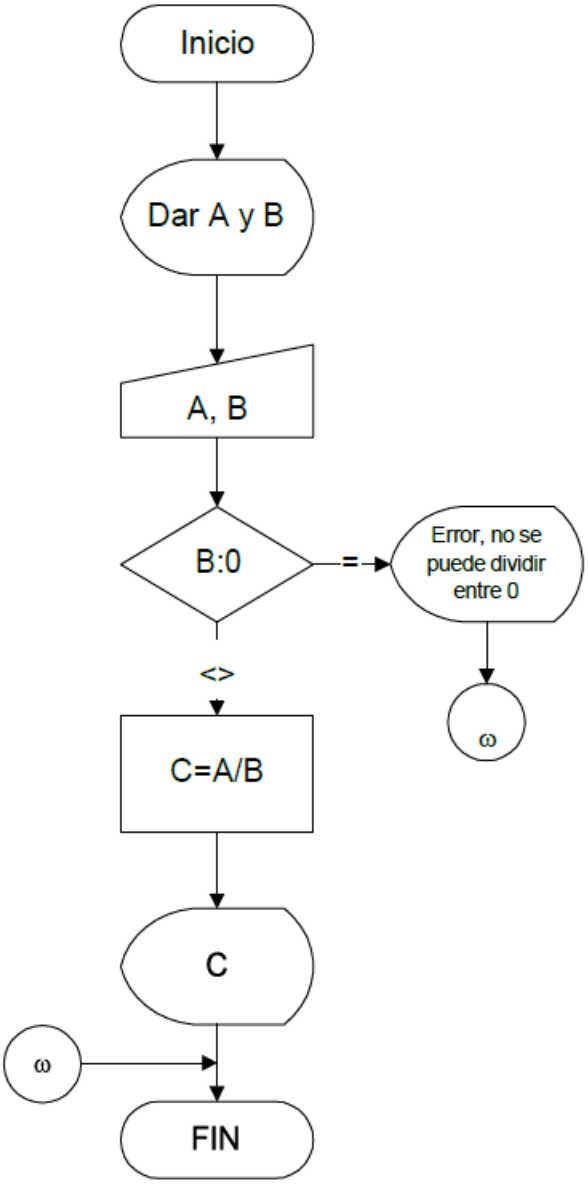
Restricciones:

Al igual que en la resta, es importante que quede claro qué variable es el dividendo y qué variable es el divisor. En este caso, A es el dividendo y B es el divisor.

Se ve, además, que existe algo a tener en consideración:

1) Si B = 0 existe un error, no se puede dividir entre 0.

Diseño (diagrama de flujo):



Codificación:

```
# Programa para calcular la división de 2 números reales
# Autor: Dr. Manuel Pérez Cota
# v20201126x
a = float(input ("Dame el valor de a "))
b = float(input ("Dame el valor de b "))
if b != 0.0:
    c = a / b
    print ("El resultado es ", c)
else:
    print ("No se puede dividir entre 0")
```

Pruebas del programa (corresponden a ejecuciones seguidas):

```
>>> %Run probadivide20201126.py

Dame el valor de a 3
Dame el valor de b 4
El resultado es  0.75

>>> %Run probadivide20201126.py
Dame el valor de a 3
Dame el valor de b 0
No se puede dividir entre 0

>>>
```

-Ejercicio 5:

Problema: Diseñar un programa que resuelva una ecuación lineal de primer grado.

Análisis: La fórmula es conocida:

ax + b = c

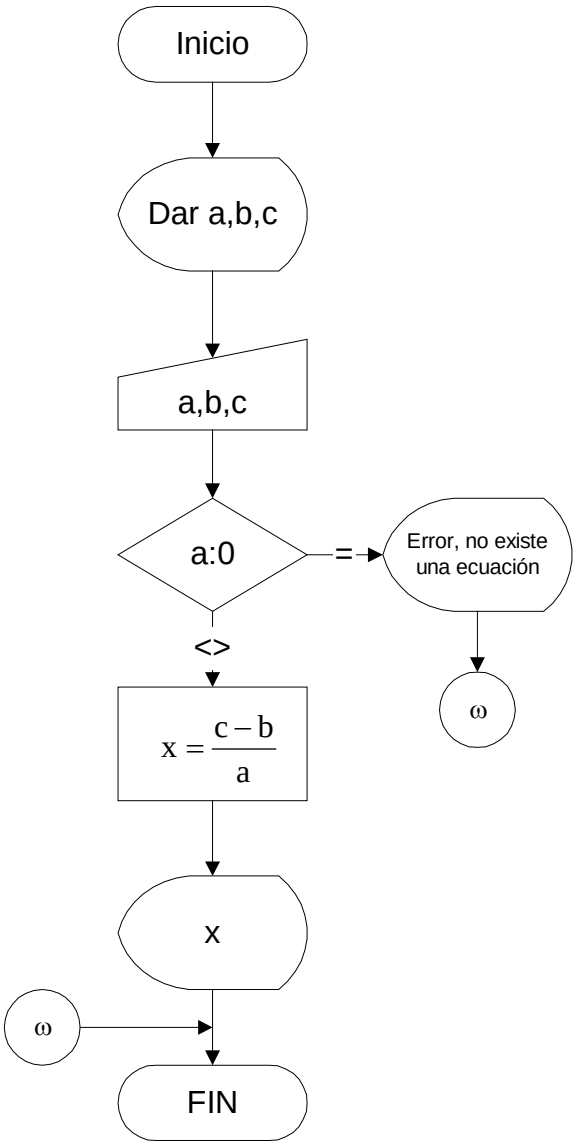
siendo ***a, b*** y ***c*** constantes y ***x*** la incógnita despejando x se obtiene:

$$x = \frac{c - b}{a}$$

Restricciones:

1)Si $a = 0$ entonces no existe ninguna ecuación, es un error.

Diagrama de flujo:



Codificación:

```
# Programa para calcular x en una ecuación de 1er. grado
# Autor: Dr. Manuel Pérez Cota
# v20201126x
a = float(input ("Dame el valor de a "))
b = float(input ("Dame el valor de b "))
c = float(input ("Dame el valor de c "))
if a != 0.0:
    x = (c-b)/a
    print ("El resultado de x es: ", x)
else:
    print ("Error: no existe una ecuación")
```

Pruebas del programa (corresponden a ejecuciones

seguidas): >>> %Run probaec1grau20201126x.py

```
Dame el valor de a 2
Dame el valor de b 3
Dame el valor de c 4
El resultado de x es:  0.5

>>> %Run probaec1grau20201126x.py

Dame el valor de a 0
Dame el valor de b 3
Dame el valor de c 4
Error: no existe una ecuación

>>>
```

-Ejercicio 6.

Problema: Diseñar un programa que resuelva un sistema de ecuaciones lineales de primer grado con dos incógnitas.

Análisis: Se empieza observando cómo es un sistema de ese tipo, con un ejemplo:

$$\left.\begin{array}{l} 3x + 2y = 5 \\ 2x - 4y = -2 \end{array}\right\}$$

Se convierte en un sistema con letras algebraico:

$$\left.\begin{array}{l} ax + by = c \\ dx + ey = f \end{array}\right\}$$

Se observa que existen muchas formas de resolverlo: sustitución, igualación, determinantes, matrices, gráfico …..

Se debe optar, en este caso, por la forma más sencilla y directa; se opta por tanto por el método de sustitución.

De la primera ecuación: ax + by = c se despeja x:

$$x = \frac{c - by}{a}$$

Que es la solución de x en función de y.

Sustituyendo en la segunda ecuación dx + ey = f

$$d\left(\frac{c-by}{a}\right) + ey = f$$

$$dc - dby + aey = af$$

$$aey - dby = af - dc$$

$$y = \frac{af - dc}{ae - db}$$

Que es la solución para y; se tiene ahora el valor de "y" que se sustituirá en la primera ecuación y obtendremos "x".

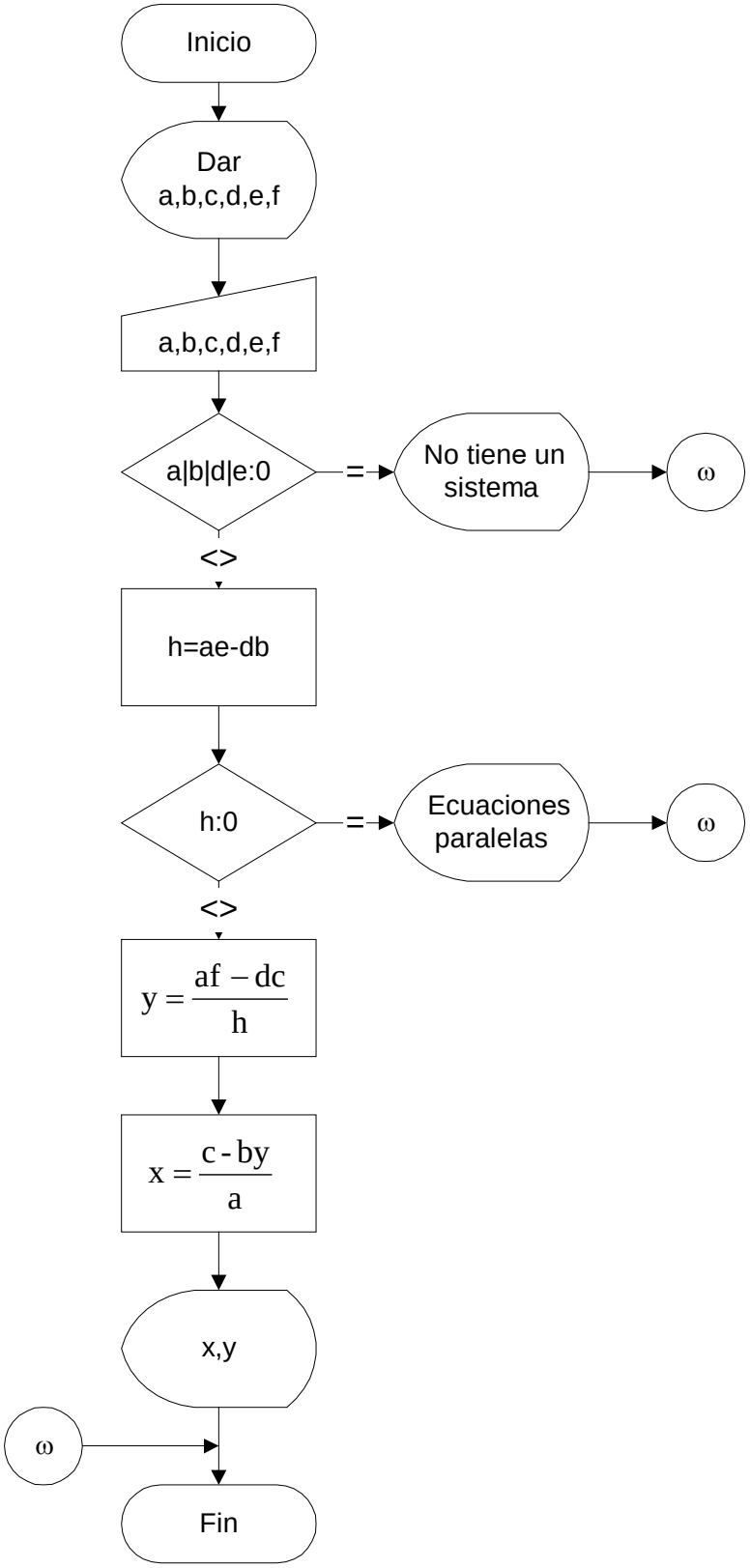
Restricciones:

1)Sí a ó b ó d ó e = 0 no se tiene un sistema.

(Cabe indicar aquí que se podría decidir la solución con la falta de cualquiera de ellos, más por evidenciar lo necesario de este ejercicio se deja con esta restricción abierta).

2)Sí ae – db = 0 se tienen ecuaciones paralelas.

Diagrama de flujo:



Codificación:

```
# Programa para calcular un sistema de 2 ecuaciones de 1er. grado
# Autor: Dr. Manuel Pérez Cota
# v20201126x
# Carga de valores
a = float(input ("Dame el valor de a "))
b = float(input ("Dame el valor de b "))
c = float(input ("Dame el valor de c "))
d = float(input ("Dame el valor de d "))
e = float(input ("Dame el valor de e "))
f = float(input ("Dame el valor de f  "))
# Revisión de los valores de a, b, d, e
if a == 0.0 or b == 0.0 or d == 0.0 or e == 0.0:
   print ("No tiene un sistema")
else:
   # Creación de una variable temporal y comprobación de su valor
   h = a*e - d*b
   if h == 0:
      print ("Ecuaciones paralelas")
   else:
      # Calculo del los valores de x e y
      y = (a*f - d*c)/h
      x = (c - b*y)/a
      print("x = ", x, "\n", "y = ", y)
```

Pruebas del programa (corresponden a ejecuciones

seguidas): >>> %Run probasistec1grau20201126x.py

```
Dame el valor de a 1
Dame el valor de b 2
Dame el valor de c 3
Dame el valor de d 4
Dame el valor de e 5
Dame el valor de f 6
x =  -1.0
 y =  2.0
```

>>> %Run probasistec1grau20201126x.py

```
Dame el valor de a 2
Dame el valor de b 3
Dame el valor de c 4
Dame el valor de d 4
Dame el valor de e 6
Dame el valor de f 8
Ecuaciones paralelas
```

>>> %Run probasistec1grau20180802x.py

Dame el valor de a 2
Dame el valor de b 0
Dame el valor de c 3
Dame el valor de d 4
Dame el valor de e 3
Dame el valor de f 5
No tiene un sistema

>>>

-Ejercicio 7.

Problema: Diseñar un programa que resuelva la ecuación cuadrática mixta completa.

Análisis: Después de preguntar, normalmente, se debe entender sí el proponente se referirá a la siguiente ecuación:

$$ax^2+bx+c=0$$

De ser así, en esta ecuación es bien conocida la fórmula para resolverla:

$$x=\frac{-b\pm\sqrt{b^2-4ac}}{2a}$$

Restricciones:

1)Si a = 0 no se tiene una ecuación de segundo grado, no obstante, sí que se puede tener una ecuación de 1er. grado de la forma:

$bx + c = 0$

Pero en este caso también se puede encontrar alguna restricción, es decir:

1.1) Si b=0 se tiene c = 0, es decir una constante igual a 0, que:

1.1.1.) Si el valor de c es cero se tiene 0 = 0 y se puede considerar el mensaje de este tipo al usuario "No tiene ecuación sino 0 = 0".

1.1.2.) En el caso de que c sea diferente de cero se considerará directamente un error.

1.2) Si b es diferente de 0 la solución es:

$$x=\frac{-c}{b}$$

2)Sí el discriminante de la raíz (b^2-$4ac$)es menor que cero, la solución se debe obtener por complejos:

$$r=\frac{-b}{2a}$$

$$i=\left|\frac{\sqrt{b^2-4ac}}{2a}\right|$$

Por lo tanto las soluciones serán:

$x_1 = r + ij$

$x_2 = r - ij$

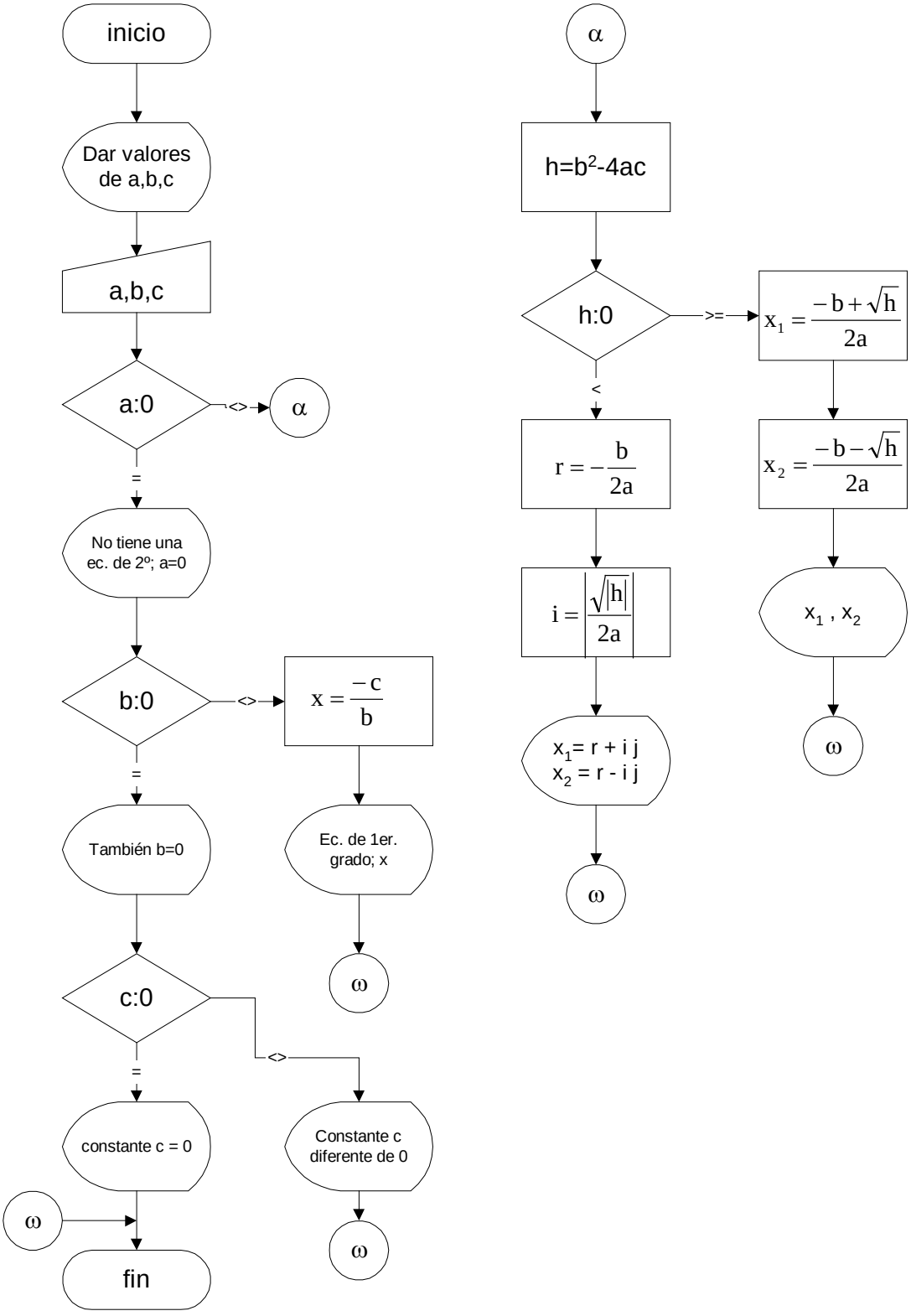
Se debe recordar, aquí, que las j únicamente indican que i es un valor complejo.

3)Por último; si el discriminante de la raíz es mayor o igual a cero, es necesario separar la ecuación con el "+" y la ecuación con el "-"; en este caso si el discriminante es igual a 0 es un caso particular del caso general y se deja la misma ecuación, x_1 y x_2 serán iguales:

$$x_1=\frac{-b+\sqrt{b^2-4ac}}{2a}$$

$$x_2=\frac{-b-\sqrt{b^2-4ac}}{2a}$$

Diagrama de flujo:



Codificación:

```
# Programa para calcular la ecuación cuadrática mixta completa
# Autor: Dr. Manuel Pérez Cota
# v20201126x
# importo librería matemática
import math
a = float(input ("Dame el valor de a "))
b = float(input ("Dame el valor de b "))
c = float(input ("Dame el valor de c "))
# revisiones
if a == 0.0:
    print ("No tiene una ecuación de 2º; a = 0")
    if b == 0.0:
        print ("También b = 0")
        if c == 0.0:
            print ("Constante c = 0")
        else:
            print ("Constante c <> 0")
    else:
        x = -c/b
        print ("Ecuación de 1º grau; x = ",x)
else:
    h = b**2 - 4*a*c
    if h < 0.0:
        # Cálculo parte imaginaria
        r = -b/(2.0*a)
        i = math.sqrt(abs(h))/(2.0*a)
        print ("x1 = ",r, "+",i,"j\n","x2 = ",r,"-",i,"j")
    else:
        # Cálculo parte real
        x1=(-b+math.sqrt(h))/(2.0*a)
        x2=(-b-math.sqrt(h))/(2.0*a)
        print("x1 = ",x1,"\nx2 = ",x2)
```

Pruebas del programa (corresponden a ejecuciones seguidas):

```
>>> %Run probaec2grau20201126x.py

Dame el valor de a 2
Dame el valor de b -4
Dame el valor de c 3
x1 =  1.0 + 0.7071067811865476 j
 x2 =  1.0 - 0.7071067811865476 j

>>> %Run probaec2grau20201126x.py

Dame el valor de a 2
Dame el valor de b 7
```

```
Dame el valor de c 2
x1 =  -0.31385933836549285
x2 =  -3.186140661634507

>>> %Run probaec2grau20201126x.py
Dame el valor de a 0
Dame el valor de b 2
Dame el valor de c 4
No tiene una ecuación de 2º; a = 0
Ecuación de 1º grau; x =  -2.0

>>> %Run probaec2grau20201126x.py

Dame el valor de a 0
Dame el valor de b 0
Dame el valor de c 3
No tiene una ecuación de 2º; a = 0
También b = 0
Constante c <> 0

>>> %Run probaec2grau20201126x.py

Dame el valor de a 0
Dame el valor de b 0
Dame el valor de c 0
No tiene una ecuación de 2º; a = 0
También b = 0
Constante c = 0

>>>
```

-Ejercicio 8.

Problema: Diseñar un programa capaz de calcular el factorial de un número entero positivo.

NOTA 1: Este ejemplo se resolverá utilizando dos tipos de sentencias de ciclos diferentes, a saber: for y while.

Análisis:
Se tiene que, por ejemplo, el factorial de 5 = 5x4x3x2x1 = 120,
Por lo tanto: n! = n(n-1)(n-2)...1

Restricciones:

1) $\forall n \in Z$

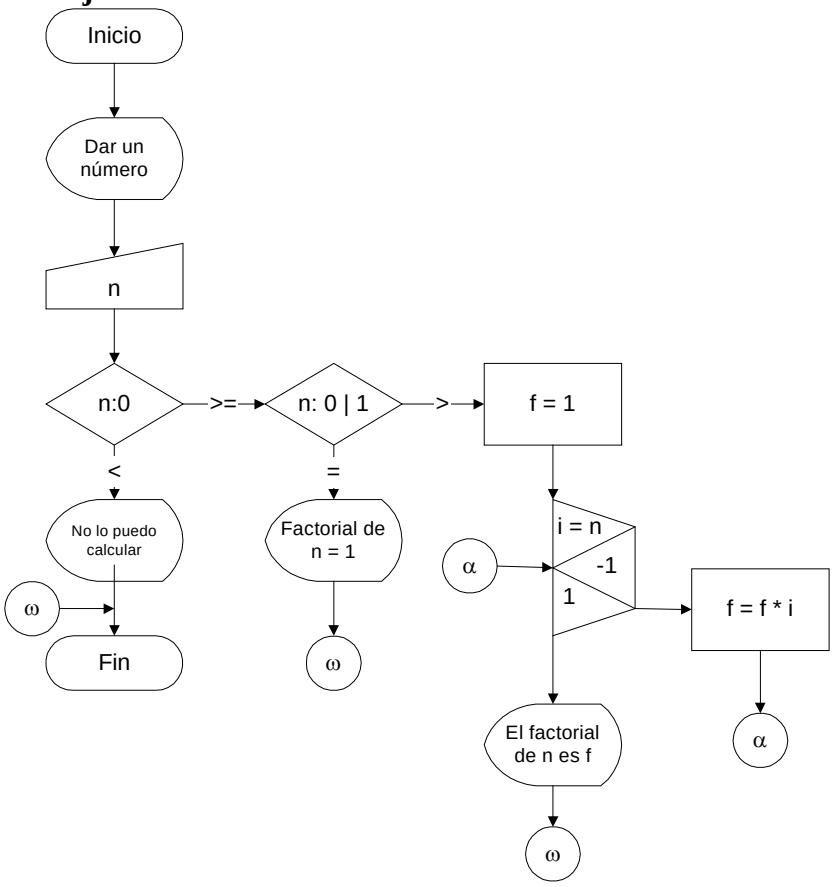
NOTA 2: Aquí se presentan 2 conceptos nuevos, el concepto de acumulador y el de contador.

Acumulador: Aquella variable que almacenara valores de operaciones en los que la propia variable resultante está involucrada, v. gr.: A = A + W.

Contador: Un tipo especial de acumulador en el que el incremento o decremento de la variable depende de una constante entera positiva o negativa, v.gr.: C = C + 1.

El computador realiza estas operaciones atendiendo a la siguiente lógica: primero resuelve las operaciones que se encuentran en segundo término (en los ejemplos A + W y C + 1 y guarda el resultado en la variable que se encuentra en el primer término, es decir: A y C, perdiendo éstas el valor anterior que poseyeran.

Diagrama de flujo: Utilizando el ciclo for.



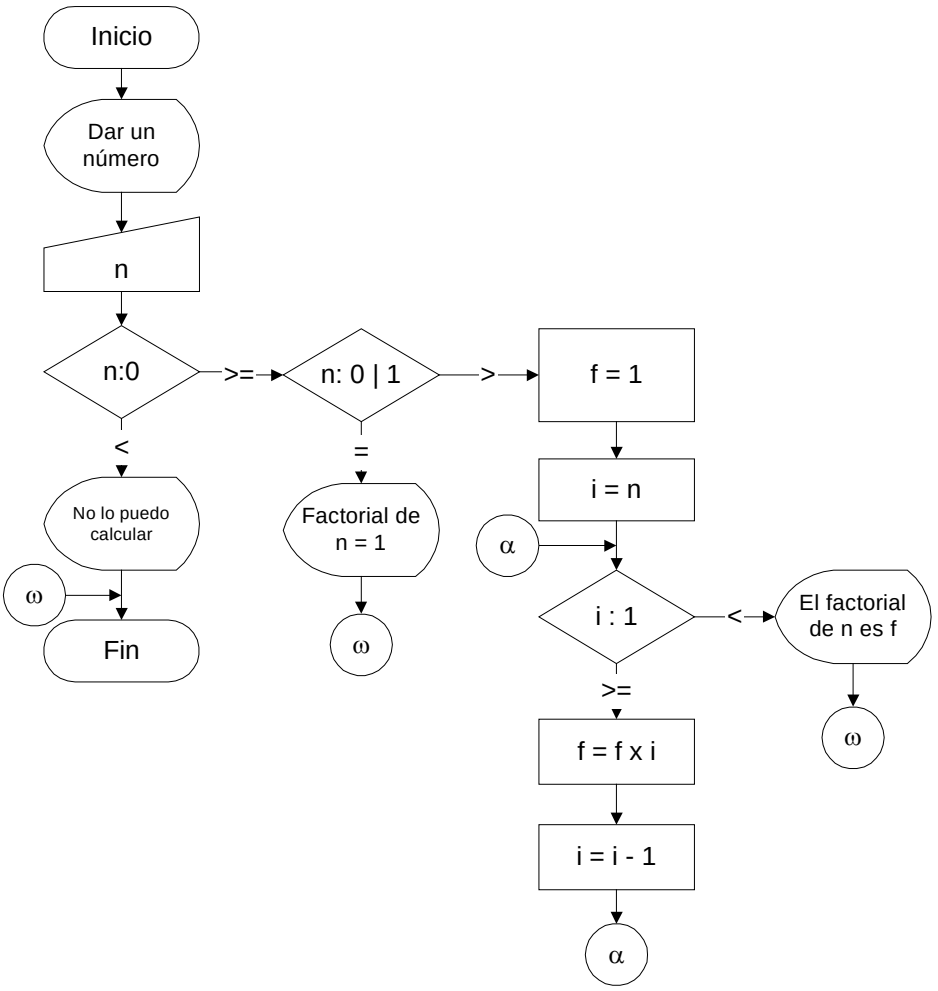
Codificación:

```
# Programa para calcular el factorial de un número (ciclos)
# Autor: Dr. Manuel Pérez Cota
# v20201126x
n = int(input ("Dame el valor de n "))
if n < 0:
   print ("No se puede calcular el factorial de un número negativo")
elif n == 0:
   print ("El factorial de 0 = 1")
else:
   f = 1
   for i in range(n,1,-1):
       f *= i
   print("El factorial de ", n, "es = ", f)
```

Pruebas del programa (corresponden a ejecuciones seguidas):

```
>>> %Run probafactor20201126x.py

Dame el valor de n 0
El factorial de 0 = 1

>>> %Run probafactor20201126x.py

Dame el valor de n 1
El factorial de  1 es =  1

>>> %Run probafactor20201126x.py

Dame el valor de n 5
El factorial de  5 es =  120

>>> %Run probafactor20201126x.py

Dame el valor de n -5
No se puede calcular el factorial de un número negativo

>>>
```

Diagrama de flujo: Utilizando el ciclo tipo while



Codificación:

```
# Programa para calcular el factorial de un número (ciclos)
# Autor: Dr. Manuel Pérez Cota
# v20201126x
n = int(input ("Dame el valor de n "))
if n < 0:
   print ("No se puede calcular el factorial de un número negativo")
elif n == 0:
   print ("El factorial de 0 = 1")
else:
   f = 1
   i = n
   while i > 1:
      f *= i
      i -= 1
   print("El factorial de ", n, "es = ", f)
```

Pruebas del programa (corresponden a ejecuciones seguidas):

```
>>> %Run probafactorwhile20201126x.py

Dame el valor de n -3
No se puede calcular el factorial de un número negativo

>>> %Run probafactorwhile20201126x.py

Dame el valor de n 6
El factorial de  6 es =  720

>>> %Run probafactorwhile20201126x.py

Dame el valor de n 0
El factorial de 0 = 1

>>> %Run probafactorwhile20201126x.py

Dame el valor de n 1
El factorial de  1 es =  1

>>>
```

-Ejercicio 9.

Problema: Diseñar un programa capaz de resolver la serie de Fibonacci genérica, para n valores a partir de los 2 iniciales dados por el usuario.

Análisis: se empieza observando cómo es la serie de Fibonacci:

Serie que aparecería dados 0 y 1 como valores de semilla (serie original)

0 1 1 2 3 5 8 13 ...

Creando una secuencia con variables se obtiene:

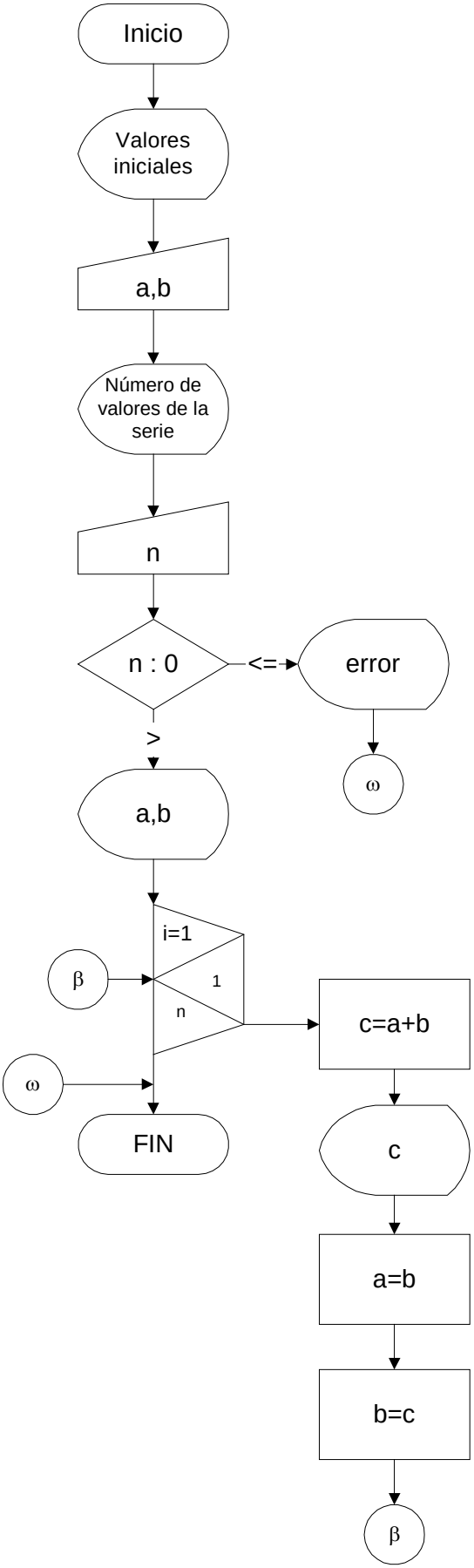
a	b	c
0	1	1
1	1	2
1	2	3
2	3	5
3	5	8

c = a + b
a = b
b = c

Obsérvese como se van moviendo los valores, justamente eso es lo que nos permite encontrar las fórmulas c = a + b y después ver como se pasa el valor de *b* a *a* y el valor de *c* en *b*.

Es importante que se siga esta secuencia, porque, si no es así el programa no funcionará correctamente.

Diagrama de flujo:



Codificación:

```
#Serie de Fibonacci genérica
#Autor: Dr. Manuel Pérez Cota
#v20201126x
a = int(input("Dame el primer valor de semilla "))
b = int(input("Dame segundo valor se semilla "))
n = int(input("Dame el número de valores de la serie "))
if n <= 0:
    print("No se piden valores de la serie")
else:
    print(a, b, end=" ")
    for i in range(1,n):
        c = a + b
        print(c, end=" ")
        a = b
        b = c
```

Pruebas del programa (corresponden a ejecuciones seguidas):

```
>>> %Run probafibona20201126x.py

Dame el primer valor de semilla 0
Dame segundo valor se semilla 1
Dame el número de valores de la serie 10
0 1 1 2 3 5 8 13 21 34 55

>>> %Run probafibona20201126x.py

Dame el primer valor de semilla 3
Dame segundo valor se semilla 4
Dame el número de valores de la serie 0
No se piden valores de la serie

>>> %Run probafibona20201126x.py

Dame el primer valor de semilla 2
Dame segundo valor se semilla 5
Dame el número de valores de la serie 8
2 5 7 12 19 31 50 81 131

>>>
```

-Ejercicio 10.

Problema: Diseñar un programa que sea capaz de sumar dos matrices.

Análisis:

Como en ejercicios anteriores, se realiza con un ejemplo que sume la matriz A con la matriz B generando una tercera matriz C de la forma:

$$\begin{pmatrix} 2 & 3 \\ 4 & 5{,}2 \\ -2{,}3 & 7 \end{pmatrix} + \begin{pmatrix} 2 & 3 \\ 4 & -4{,}1 \\ 2.3 & 7 \end{pmatrix} = \begin{pmatrix} 4 & 6 \\ 8 & 1{,}1 \\ 0 & 14 \end{pmatrix}$$

A + B = C

$$\begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \\ a_{31} & a_{32} \end{pmatrix} + \begin{pmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \\ b_{31} & b_{32} \end{pmatrix} = \begin{pmatrix} c_{11} & c_{12} \\ c_{21} & c_{22} \\ c_{31} & c_{32} \end{pmatrix}$$

Por lo tanto: $a_{11} + b_{11} = c_{11}$, o si se prefiere: $c_{11} = a_{11} + b_{11}$

$c_{12} = a_{12} + b_{12}$

$c_{21} = a_{21} + b_{21}$

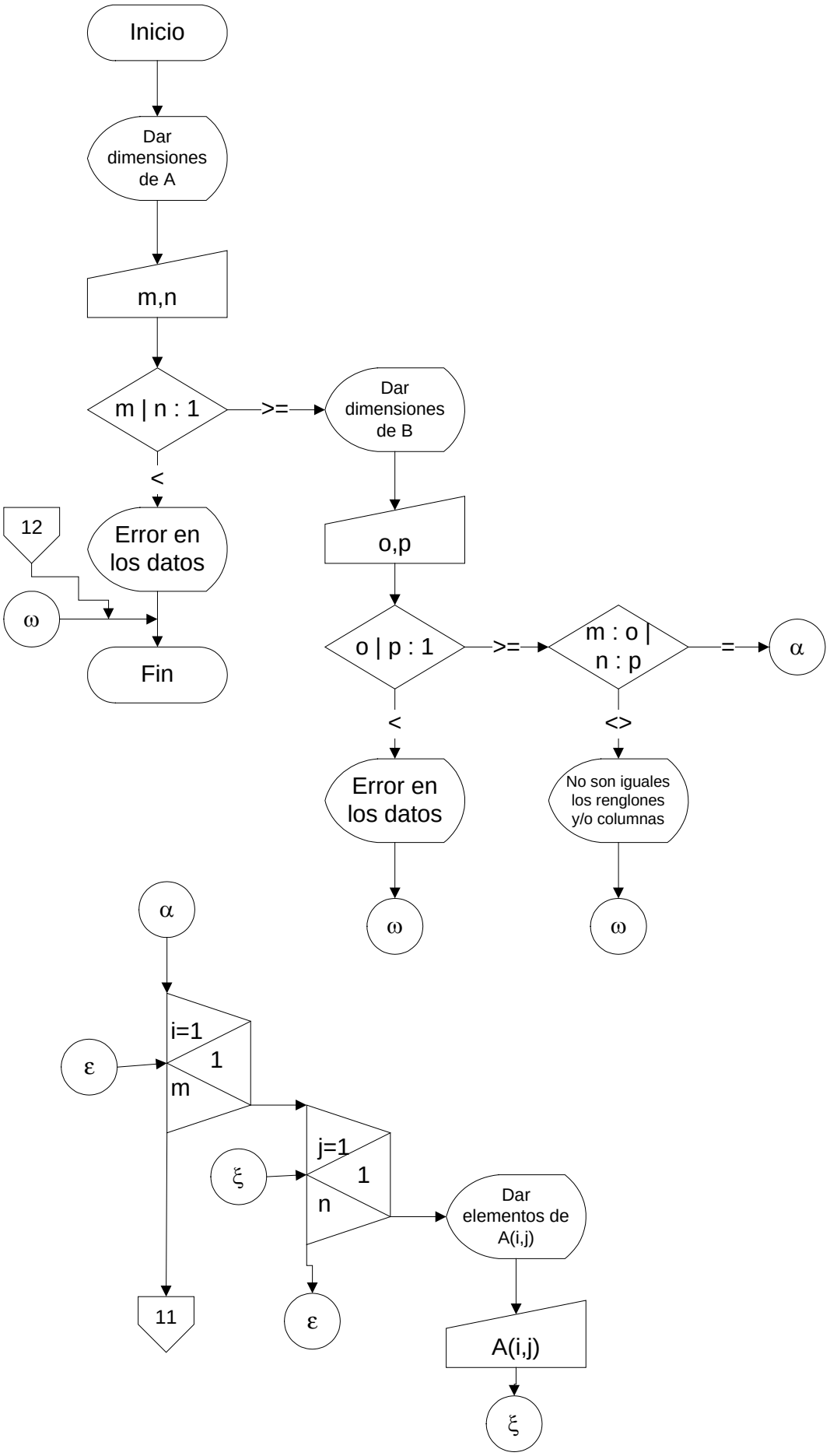
: :

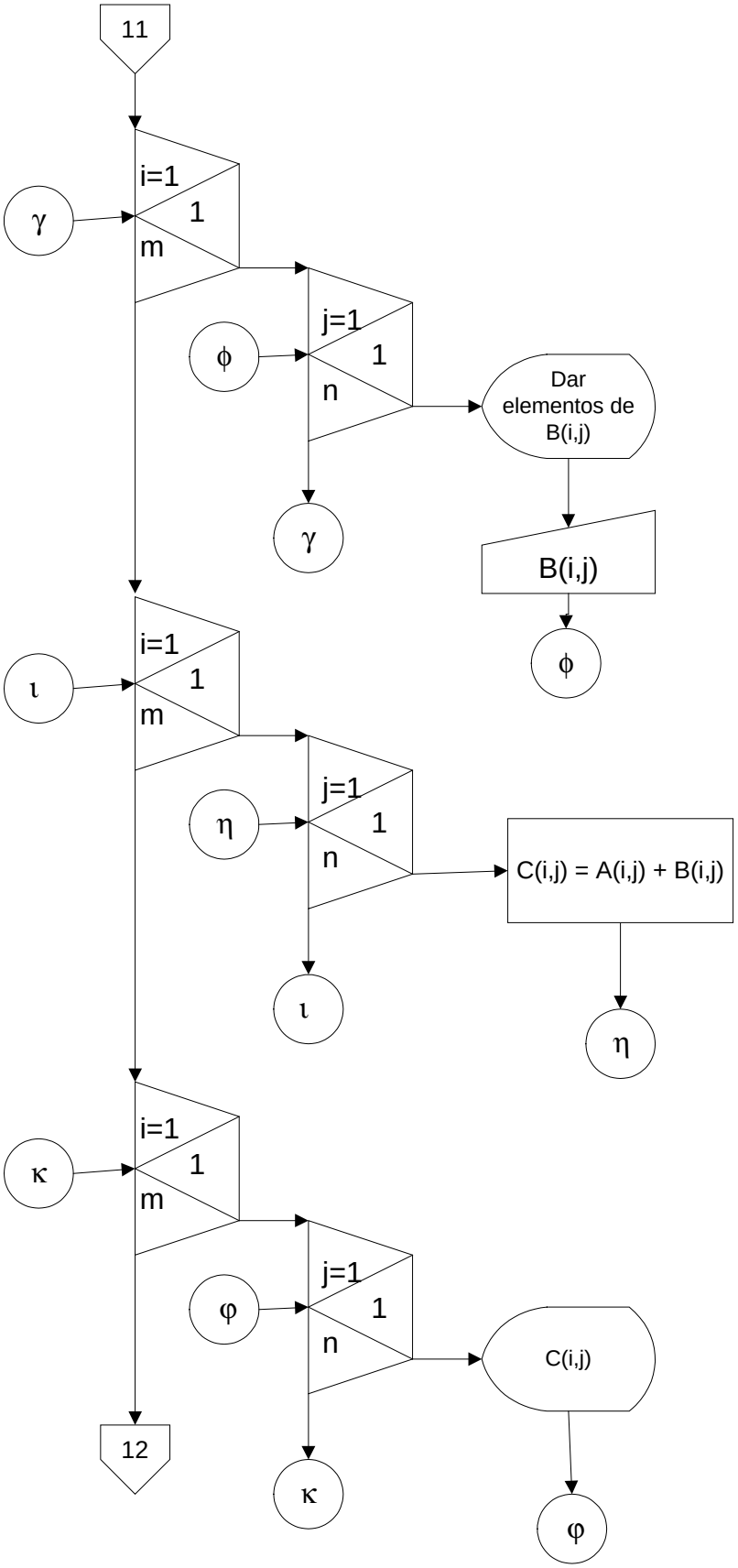
Genéricamente $c_{ij} = a_{ij} + b_{ij}$ $\forall (i, j) >= 1 \, y \in Z$

Restricciones:

1) El número de reglones y columnas de las matrices debe ser iguales, es decir, número de renglones de A = número de renglones de B y número de columnas de A = número de columnas de B.
2) El número de renglones y columnas debe ser siempre un entero positivo.

Diagrama de flujo:




11
i=1
1
m
γ
j=1
1
n
ϕ
Dar elementos de B(i,j)
γ
B(i,j)
ϕ
i=1
1
m
ι
j=1
1
n
η
C(i,j) = A(i,j) + B(i,j)
ι
η
i=1
1
m
κ
j=1
1
n
φ
C(i,j)
12
κ
φ

Codificación:

```
#Suma de matrices
#Autor: Dr. Manuel Pérez Cota
#v20201126x
m = int(input("Dame el número de renglones de la matriz A "))
n = int(input("Dame el número de columnas de la matriz A "))
if m < 1 or n < 1:
   print("Error en los datos, no pueden ser menores que 1")
else:
   o = int(input("Dame el número de renglones de la matriz B "))
   p = int(input("Dame el número de columnas de la matriz B "))
   if o < 1 or p < 1:
      print("Error en los datos, no pueden ser menores que 1")
   else:
      if m != o or n != p:
         print("No es igual en número de renglones y/o columnas")
      else:
         print ("Dame los valores de la matriz A")
         A=[[float(input("A ")) for i in range(n)] for j in range(m)]
         print ("Dame los valores de la matriz A")
         B=[[float(input("B ")) for i in range(n)] for j in range(m)]
         #Inicialización de C para Python
         C=[[0 for i in range(n)] for j in range(m)]
         for i in range(m):
            for j in range(n):
               C[i][j] = A[i][j] + B[i][j]
         for i in range(m):
            for j in range(n):
               print("C(",i+1,",",j+1,")=", C[i][j])
```

Pruebas del programa (corresponden a ejecuciones seguidas):

```
>>> %Run probasumamat20201126x.py

Dame el número de renglones de la matriz A 2
Dame el número de columnas de la matriz A 2
Dame el número de renglones de la matriz B 2
Dame el número de columnas de la matriz B 2
Dame los valores de la matriz A
A 1
A 2
A 3
A 4
Dame los valores de la matriz B
B 1
B 2
B 3
B 4
```

```
C( 0 , 0 )= 2.0
C( 0 , 1 )= 4.0
C( 1 , 0 )= 6.0
C( 1 , 1 )= 8.0

>>> %Run probasumamat20201126x.py

Dame el número de renglones de la matriz A 2
Dame el número de columnas de la matriz A 2
Dame el número de renglones de la matriz B 2
Dame el número de columnas de la matriz B 2
Dame los valores de la matriz A
A 1
A 2
A 3
A 4
Dame los valores de la matriz B
B 1
B 2
B 3
B 4
C( 1 , 1 )= 2.0
C( 1 , 2 )= 4.0
C( 2 , 1 )= 6.0
C( 2 , 2 )= 8.0

>>> %Run probasumamat20201126x.py

Dame el número de renglones de la matriz A 1
Dame el número de columnas de la matriz A 1
Dame el número de renglones de la matriz B 1
Dame el número de columnas de la matriz B 1
Dame los valores de la matriz A
A 1
Dame los valores de la matriz B
B 2
C( 1 , 1 )= 3.0

>>> %Run probasumamat20201126x.py

Dame el número de renglones de la matriz A 2
Dame el número de columnas de la matriz A 4
Dame el número de renglones de la matriz B 2
Dame el número de columnas de la matriz B 4
Dame los valores de la matriz A
A 1
A 2
A 3
A 4
A 5
A 6
A 7
```

A 8
Dame los valores de la matriz B
B 1
B 2
B 3
B 4
B 5
B 6
B 7
B 8
C(1 , 1)= 2.0
C(1 , 2)= 4.0
C(1 , 3)= 6.0
C(1 , 4)= 8.0
C(2 , 1)= 10.0
C(2 , 2)= 12.0
C(2 , 3)= 14.0
C(2 , 4)= 16.0

>>> %Run probasumamat20201126x.py

Dame el número de renglones de la matriz A 3
Dame el número de columnas de la matriz A 4
Dame el número de renglones de la matriz B 3
Dame el número de columnas de la matriz B 5
No es igual en número de renglones y/o columnas

>>> %Run probasumamat20201126x.py

Dame el número de renglones de la matriz A 0
Dame el número de columnas de la matriz A 1
Error en los datos, no pueden ser menores que 1

>>>

-Ejercicio 11.

Problema: Diseñar un programa que sea capaz de multiplicar dos matrices.

Análisis:

Multiplicar la matriz A por la matriz B genera una tercera matriz C de la forma:

$$\begin{pmatrix} 2 & 2 \\ 4 & 3 \\ -2{,}3 & 1 \end{pmatrix} x \begin{pmatrix} 1 & -2{,}1 \\ 3 & 2 \end{pmatrix} = \begin{pmatrix} 8 & -0{,}2 \\ 13 & -2{,}6 \\ 1{,}3 & 6{,}83 \end{pmatrix}$$

A(3,2) x B(2.,2) = C(3,2)

$$\begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \\ a_{31} & a_{32} \end{pmatrix} x \begin{pmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{pmatrix} = \begin{pmatrix} a_{11}b_{11} + a_{12}b_{21} & a_{11}b_{12} + a_{12}b_{22} \\ a_{21}b_{11} + a_{22}b_{21} & a_{21}b_{12} + a_{22}b_{22} \\ a_{31}b_{11} + a_{32}b_{21} & a_{31}b_{12} + a_{32}b_{22} \end{pmatrix} = \begin{pmatrix} c_{11} & c_{12} \\ c_{21} & c_{22} \\ c_{31} & c_{32} \end{pmatrix}$$

Por lo tanto: $c_{11} = a_{11}b_{11} + a_{12}b_{21}$

$c_{12} = a_{11}b_{12} + a_{12}b_{22}$

$c_{21} = a_{21}b_{11} + a_{22}b_{21}$

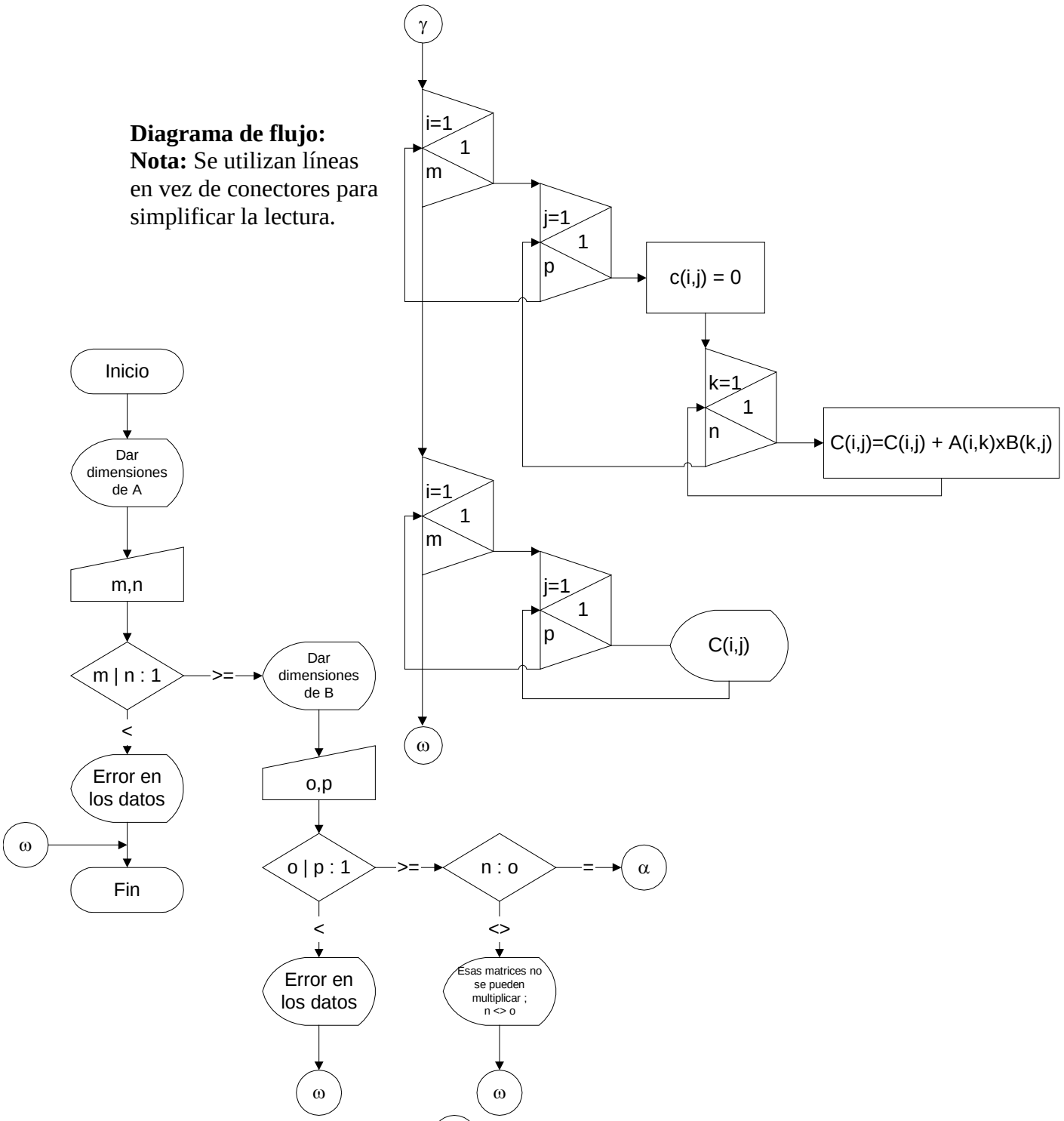
: :

Genéricamente $c_{ij} = a_{ik}b_{kj} + a_{ik}b_{kj} \quad \forall(i, j, k) >= 1 \, y \in Z$

Restricciones:

1) El número de columnas de la primera matriz (A) debe ser igual al número de renglones de la segunda (B).
2) La matriz resultante (C) tendrá el número de renglones de la primera matriz (A) y el número de columnas de la segunda matriz (B).
3) El número de renglones y columnas debe ser siempre un entero positivo.

Diagrama de flujo:
Nota: Se utilizan líneas en vez de conectores para simplificar la lectura.



Codificación:

```
#Multiplicación de matrices
#Autor: Dr. Manuel Pérez Cota
#v20201126x
m = int(input("Dame el número de renglones de la matriz A "))
n = int(input("Dame el número de columnas de la matriz A "))
if m < 1 or n < 1:
   print("Error en los datos, no pueden ser menores que 1")
else:
   o = int(input("Dame el número de renglones de la matriz B "))
   p = int(input("Dame el número de columnas de la matriz B "))
   if o < 1 or p < 1:
      print("Error en los datos, no pueden ser menores que 1")
   else:
      if n != o:
         print("No se pueden multiplicar las matrices cA <> rB")
      else:
         print ("Dame los valores de la matriz A")
         A=[[float(input("A ")) for i in range(n)] for j in range(m)]
         print ("Dame los valores de la matriz B")
         B=[[float(input("B ")) for i in range(p)] for j in range(o)]
         #Inicialización de C para Python
         #antes del proceso para que sea posible utilizarla
         C=[[0 for i in range(p)] for j in range(m)]
         for i in range(m):
            for j in range(p):
               for k in range(n):
                  C[i][j] = C[i][j] + A[i][k] * B[k][j]
         for i in range(m):
            for j in range(p):
               print("C(",i+1,",",j+1,")=", C[i][j])
```

Pruebas del programa (corresponden a ejecuciones seguidas):

```
>>> %Run probamultiplicamat20201126x.py

Dame el número de renglones de la matriz A 2
Dame el número de columnas de la matriz A 2
Dame el número de renglones de la matriz B 2
Dame el número de columnas de la matriz B 2
Dame los valores de la matriz A
A 1
A 2
A 3
A 4
Dame los valores de la matriz B
B 1
B 2
```

B 3
B 4
C(1 , 1)= 7.0
C(1 , 2)= 10.0
C(2 , 1)= 15.0
C(2 , 2)= 22.0

>>> %Run probamultiplicamat20201126x.py

Dame el número de renglones de la matriz A 2
Dame el número de columnas de la matriz A 3
Dame el número de renglones de la matriz B 2
Dame el número de columnas de la matriz B 3
No se pueden multiplicar las matrices cA <> rB

>>> %Run probamultiplicamat20201126x.py

Dame el número de renglones de la matriz A 2
Dame el número de columnas de la matriz A 3
Dame el número de renglones de la matriz B 3
Dame el número de columnas de la matriz B 2
Dame los valores de la matriz A
A 1
A 2
A 3
A 4
A 5
A 6
Dame los valores de la matriz B
B 1
B 2
B 3
B 4
B 5
B 6
C(1 , 1)= 22.0
C(1 , 2)= 28.0
C(2 , 1)= 49.0
C(2 , 2)= 64.0

>>> %Run probamultiplicamat20201126x.py

Dame el número de renglones de la matriz A 1
Dame el número de columnas de la matriz A 2
Dame el número de renglones de la matriz B 2
Dame el número de columnas de la matriz B 3
Dame los valores de la matriz A
A 1
A 2
Dame los valores de la matriz B
B 3

B 4
B 5
B 6
B 7
B 8
C(1 , 1)= 15.0
C(1 , 2)= 18.0
C(1 , 3)= 21.0

>>>

-Ejercicio 12.

Problema: Diseñar un programa que ordene de menor a mayor los elementos de un vector.

Análisis:
El proceso se estudiará con un ejemplo.
Para empezar se asume que se tiene el siguiente vector que se desea ordenar de menor a mayor:

(1, 7, 4, 2, 9)

El proceso de ordenación se hará comparando el número que ocupa la primera posición del vector a_1 con los números de las siguientes posiciones (es decir de la 2 a la 5, en el caso del ejemplo), si el valor de la posición uno es menor o igual a cada uno del resto de los valores comparados, se deja en su sitio, si es mayor se cambiará el número de esa posición a la otra posición y se traerá el número de la otra posición a ésta (proceso de intercambio también conocido como swap), esto se repetirá avanzando la posición original a la siguiente y así sucesivamente hasta la anterior a la última, donde se realizará la última comparación.

A esta forma de ordenar se le denomina de burbuja.

Representando el proceso paso a paso se tiene (contenido del vector): 1,7,4,2,9

El número en negrita e itálica es el pivote, el número subrayado es contra el que se hace la comparación)

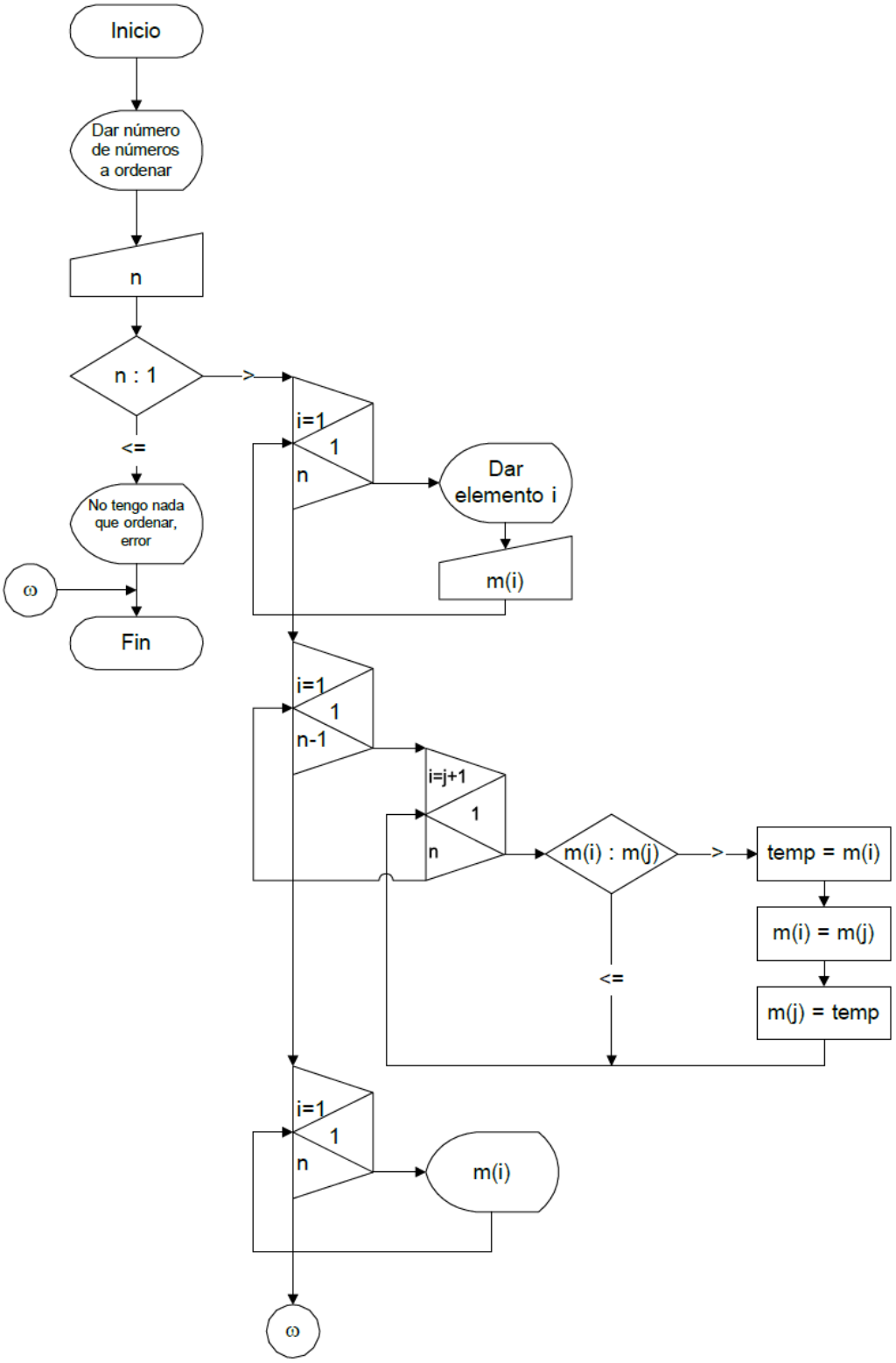
(***1***, 7, 4, 2, 9)
(***1***, 7, 4, 2, 9)
(***1***, 7, 4, 2, 9)
(***1***, 7, 4, 2, 9)
(1, ***7***, 4, 2, 9) Intercambiar 7 y 4
(1, ***4***, 7, 2, 9) Intercambiar 4 y 2
(1, ***2***, 7, 4, 9)
(1, 2, ***7***, 4, 9) Intercambiar 7 y 4
(1, 2, ***4***, 7, 9)
(1, 2, 4, ***7***, 9)

Restricciones:

1)El número de valores debe ser mayor que uno, pues si solo existe un valor no se tiene nada para ordenar.

Diagrama de flujo:

Nota: En este diagrama se utilizan practicamente todas las flechas en vez de conectores para que la persona lectora pueda ver la diferencia, a favor de los conectores, pues las flechas rompen el esquema siempre en sentido arriba-abajo y izquierda-derecha.



Codificación:

```
#Ordenación de valores
#Autor: Dr. Manuel Pérez Cota
#v20201126x
n = int(input("Dame el número de valores a ordenar "))
if n <= 1:
   print("No tengo nada para ordenar")
else:
   M = [float(input("Dame valor ")) for i in range(n)]
   for i in range(n-1):
      for j in range (i,n):
         if M[i] > M[j]:
            temp = M[i]
            M[i] = M[j]
            M[j] = temp
   for i in range(n):
      print(M[i])
```

Pruebas del programa (corresponden a ejecuciones

seguidas): >>> %Run probaordena20201126x.py

```
Dame el número de valores a ordenar 5
Dame valor 1
Dame valor 7
Dame valor 4
Dame valor 2
Dame valor 9
1.0
2.0
4.0
7.0
9.0
```

>>> %Run probaordena20201126x.py

```
Dame el número de valores a ordenar
Dame valor 1
Dame valor 4
Dame valor 5
Dame valor 3
Dame valor 4
1.0
3.0
4.0
4.0
5.0
```

>>> %Run probaordena20201126x.py

```
Dame el número de valores a ordenar 1
No tengo nada para ordenar

>>> %Run probaordena20201126x.py

Dame el número de valores a ordenar 8
Dame valor 7
Dame valor 3
Dame valor 5.2
Dame valor 6.2
Dame valor 3.1
Dame valor -4.0
Dame valor -3
Dame valor 8
-4.0
-3.0
3.0
3.1
5.2
6.2
7.0
8.0

>>>
```

Bibliografía

Microsoft Corporation (MSN). (2018). *Microsoft*. Obtenido de www.microsoft.com/es-es/: https://www.microsoft.com/es-es/

Pérez-Cota, M. (2008). *Fundamentos de Informática; Metodología de la programación.* Vigo.

Python Software Foundation (PSF). (2020). *Python*. Obtenido de www.python.org: https://www.python.org

University of Tartu (Estonia). (2020). *Thonny*. Obtenido de thonny.org: https://thonny.org