EXPLICACIÓN Y RESOLUCIÓN 2º PARCIAL CURSO 2020-2021

EJERCICIO 1 (3 puntos)

El famoso mago Mr. Dados dice que cuando se lanzan p dados m veces, es capaz de adivinar cuántas veces sale un número par y cuantas un número impar y además separar los valores pares y los impares. Sin embargo, por si falla el truco, te pide que realices un algoritmo que siga el siguiente proceso:

Los datos: una matriz DADO de m filas y p columnas que contiene los números que han salido al lanzar los dados (es decir, todos los valores serán números enteros entre 1 y 6).

Obtener una variable llamada SPar que contenga la cantidad de números pares que hay en DADO, otra variable SumaPar que contenga la suma de los valores pares y un vector llamado Par que contenga dichos números.

Obtener una variable llamada SImp que contenga la cantidad de números impares que hay en DADO, otra variable llamada SumaImpar que contenga la suma de los números impares y un vector llamado Impar que contenga dichos números.

Obtener una matriz B de s filas y 2 columnas cuya primera columna sean los números pares y la segunda los impares. El valor de s será el mayor entre SPar y SImp. La matriz B se inicializará a 0 para que las posiciones no ocupadas queden como 0.

NOTA: Para saber si un número D es divisible entre otro número d se puede usar en el algoritmo la instrucción de R: D%%d==0, que indica que el resto de dicha división es 0.

$$\mathrm{DADO}=\begin{pmatrix}4&3&2\\5&4&2\\2&3&4\\6&1&5\\5&6&6\end{pmatrix},\mathrm{m}=5(\mathrm{lanzamientos});\mathrm{p}=3(\mathrm{dados})$$

$\mathrm{SPar}=9$

$\mathrm{Par}=(4,2,4,2,2,4,6,6,6)$

$\mathrm{SumaPar}=4+2+4+2+2+4+6+6+6$

$\mathrm{SImp}=5$

$\mathrm{Impar}=(3,5,3,1,5,5)$

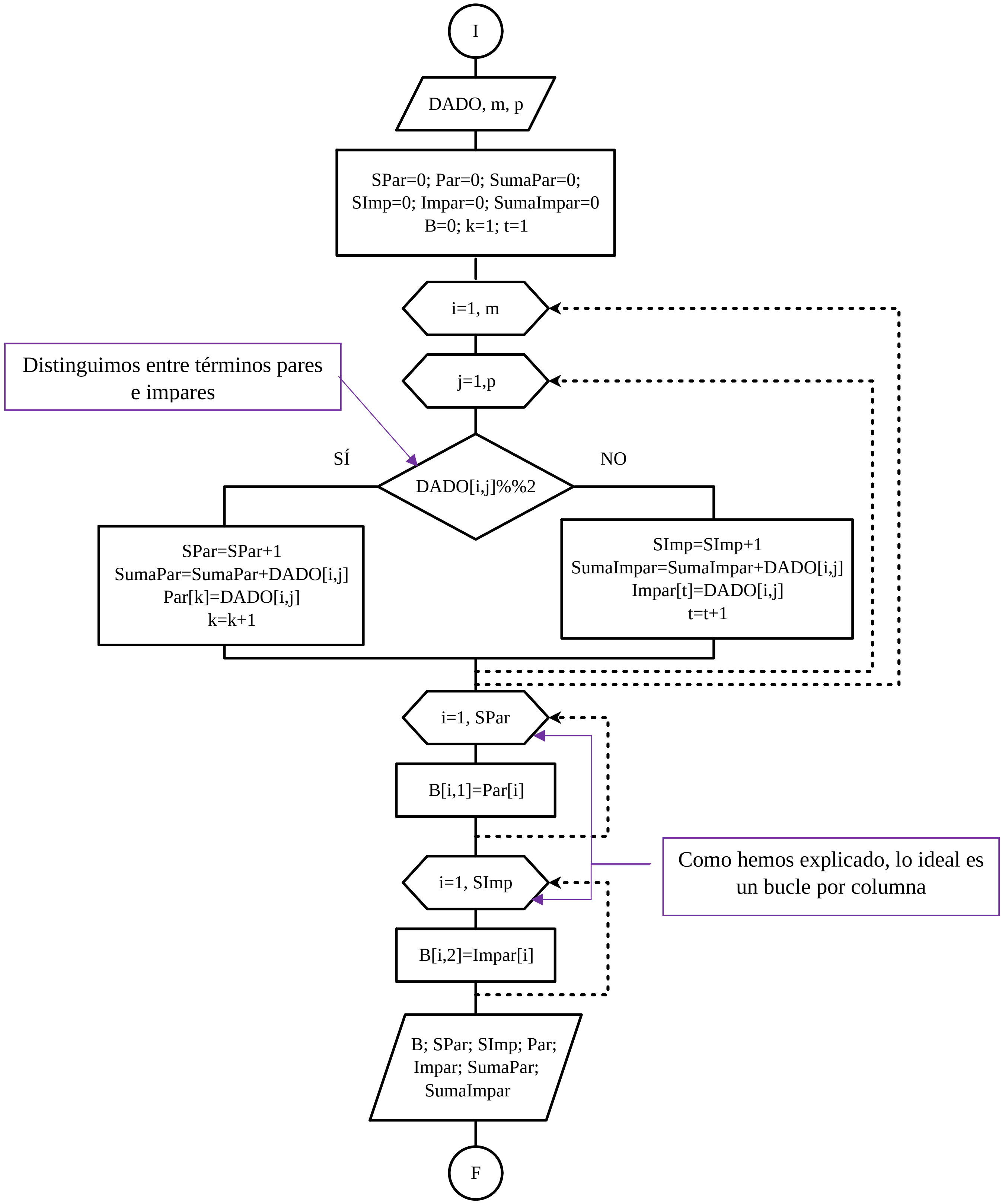
$\mathrm{SumaImpar}=3+5+3+1+5+5$

Empezaremos dividiendo nuestro algoritmo en dos partes: la primera corresponde a la obtención de SPar, SumaPar, Par, SImp, SumaImpar e Impar. Todos estos vectores y variables se obtendrán dentro de la misma estructura IF; los que correspondan a los valores impares irán en una rama y los que correspondan a los pares irán en la otra. La condición nos la da el propio enunciado, si el valor de DADO es divisible entre 2, será par; si no, será impar. Dado que hay que comprobar la divisibilidad de todos los valores de la matriz DADO, la estructura if deberá estar dentro de dos bucles anidados, uno con i variando desde 1 hasta m para las filas, y otro con j variando desde 1 hasta p para las columnas.

En la rama SÍ del if tendremos que calcular SPar, SumaPar y Par. SumaPar se obtendrá simplemente como un sumatorio $SumaPar = \sum DADO[i,j]$, y por tanto se tiene que inicializar a 0 fuera de los bucles. En cuanto a SPar, ya que contabiliza cuántos números pares hay debe inicializarse a 0 fuera del bucle y, si se cumple la condición, incrementará en 1 su valor (es decir, SPar=SPar+1). Finalmente, para el vector Par, es importante que no queden términos nulos (ceros), y por tanto debe tener un subíndice diferente a los de la matriz. A este le llamaremos k y empezará siendo igual a 1; posteriormente igualaremos Par[k] a DADO[i,j], y una vez igualado, k incrementará en 1 su valor para que el siguiente valor par que encuentre el algoritmo ocupe la siguiente posición del vector.

En la rama NO del if tendremos que hacer exactamente lo mismo, pero para los valores impares. SImp, SumaImpar e Impar se obtendrán de la misma manera, con la diferencia de que, para el vector Impar, tendremos que crear un subíndice diferente a k (por ejemplo, t), que también incrementará en 1 su valor.

Para la siguiente parte, consideramos que hay un fallo en el enunciado, pues al programar es posible que diese errores si el valor de s no coincide con la longitud de uno de los vectores. Por tanto lo óptimo sería inicializar B a 0 y hacer dos bucles diferentes para cada columna, uno desde 1 hasta SPar para el vector Par y otro desde 1 hasta SImp para el vector Impar. De esta manera nos aseguraríamos de evitar que el programa nos diese errores debidos a la dimensión del vector. Resolveremos el algoritmo de esta manera, que es la que consideramos correcta.



EJERCICIO 2 (3 puntos).

Se considera la función $g(x) = a + be^{1-x}$ y el conjunto formado por los puntos (-1, 20), (0, 15), (0.5, 9), (1, 10). Se pide:

A) Obtener la expresión matricial del sistema de ecuaciones al que se llega mediante un ajuste por mínimos cuadrados de los puntos dados mediante la función g(x) (2 puntos).

B) Realizar un algoritmo para obtener la matriz y el vector de términos independientes del sistema de ecuaciones generado en el apartado A). (1 punto)

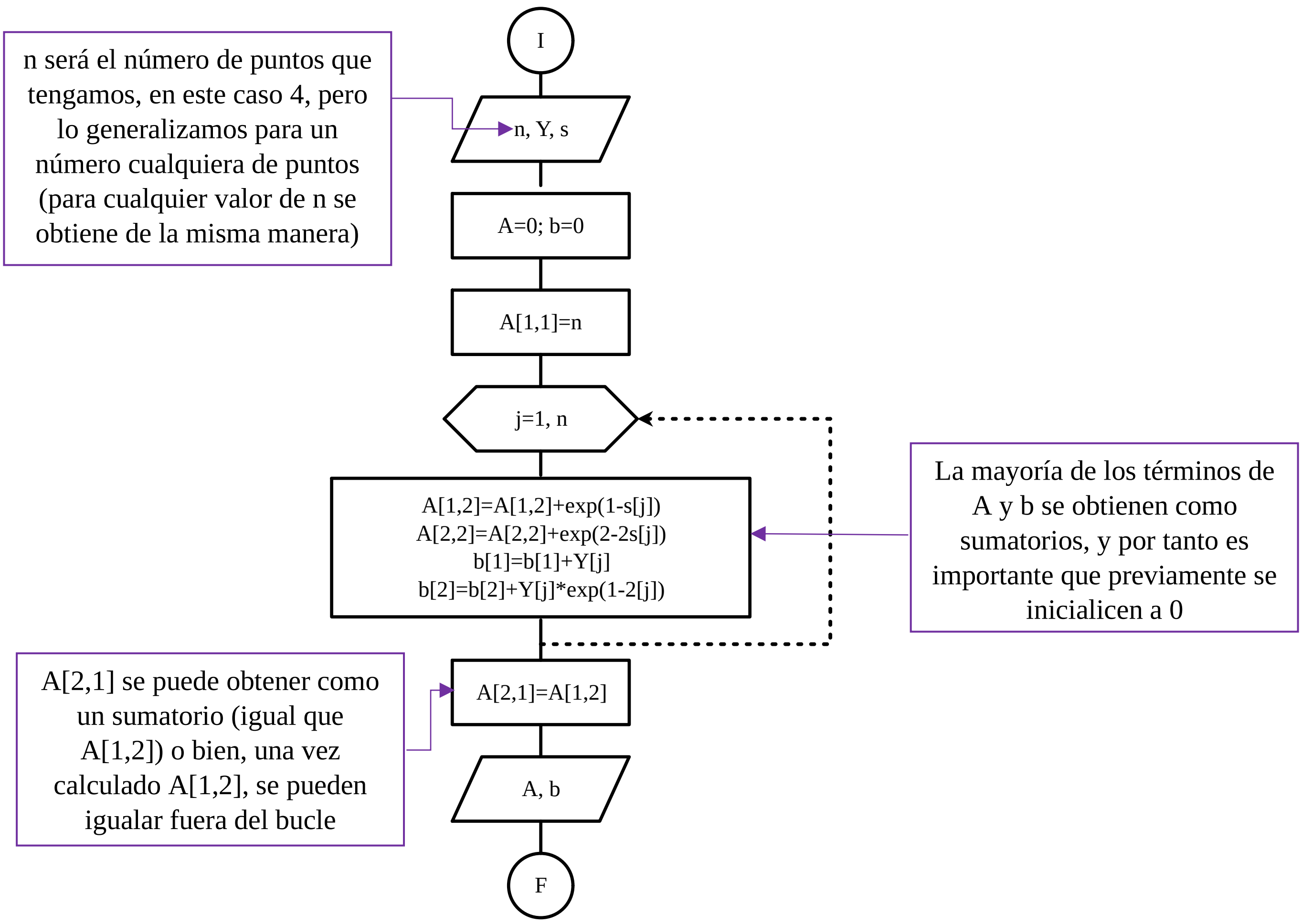
Nos centraremos solamente en la resolución del algoritmo y no se explicará el proceso operativo del apartado A, partiremos del sistema de ecuaciones ya planteado

$$4a + b\sum_{j=1}^{4} e^{1-s_j} = \sum_{j=1}^{4} Y_j$$

$$a\sum_{j=1}^{4} e^{1-s_j} + b\sum_{j=1}^{4} e^{2(1-s_j)} = \sum_{j=1}^{4} Y_j \cdot e^{1-s_j}$$

La expresión matricial será: $\begin{pmatrix} 4 & \sum_{j=1}^{4} e^{1-s_j} \\ \sum_{j=1}^{4} e^{1-s_j} & \sum_{j=1}^{4} e^{2(1-s_j)} \end{pmatrix} \begin{pmatrix} a \\ b \end{pmatrix} = \begin{pmatrix} \sum_{j=1}^{4} Y_j \\ \sum_{j=1}^{4} Y_j \cdot e^{1-s_j} \end{pmatrix}$

Desde aquí se puede obtener muy fácilmente el algoritmo correspondiente a la matriz A, de coeficientes y el vector b, de términos independientes.



EJERCICIO 3 (4 puntos)

PARTE A) (3 puntos).

Se considera que la acumulación de cierto fármaco en el organismo está dada por la integral: $\int_a^b f(x)\,dx$ siendo x el tiempo transcurrido entre dos instantes: a, b. Se desea obtener el valor de la integral mediante una fórmula de Gauss compuesta, con un número creciente de sub-intervalos.

El proceso comenzará considerando todo el intervalo [a,b] y se irá duplicando el número de sub-intervalos mientras la diferencia entre el valor exacto de la integral y el aproximado sea mayor que cierta tolerancia prefijada.

Para ello, se realizará un ALGORITMO tal que, dados: una variable f que contiene la expresión de la función que representa la acumulación del fármaco, las variables a, b que son los extremos del intervalo de integración, una variable valex, que contiene el valor exacto de la integral (supuesto conocido), una variable tol que es la tolerancia admisible en el proceso iterativo desarrollado, se aplique una fórmula de Gauss compuesta, que emplee un soporte de dos puntos en cada sub-intervalo, de manera que el número de sub-intervalos se vaya duplicando mientras la diferencia en valor absoluto entre el valor exacto y el aproximado (variable dif) sea mayor que una tolerancia prefijada (tol).

La fórmula de Gauss compuesta con dos puntos de soporte en cada sub-intervalo viene dada por $\int_a^b f(x)dx \approx \frac{h}{2}\sum_{i=1}^{N}(f\left(\frac{s_i+s_{i+1}}{2} - \frac{h\sqrt{3}}{3}\right) + f\left(\frac{s_i+s_{i+1}}{2} + \frac{h\sqrt{3}}{3}\right))$ siendo h= $s_{i+1} - s_i$

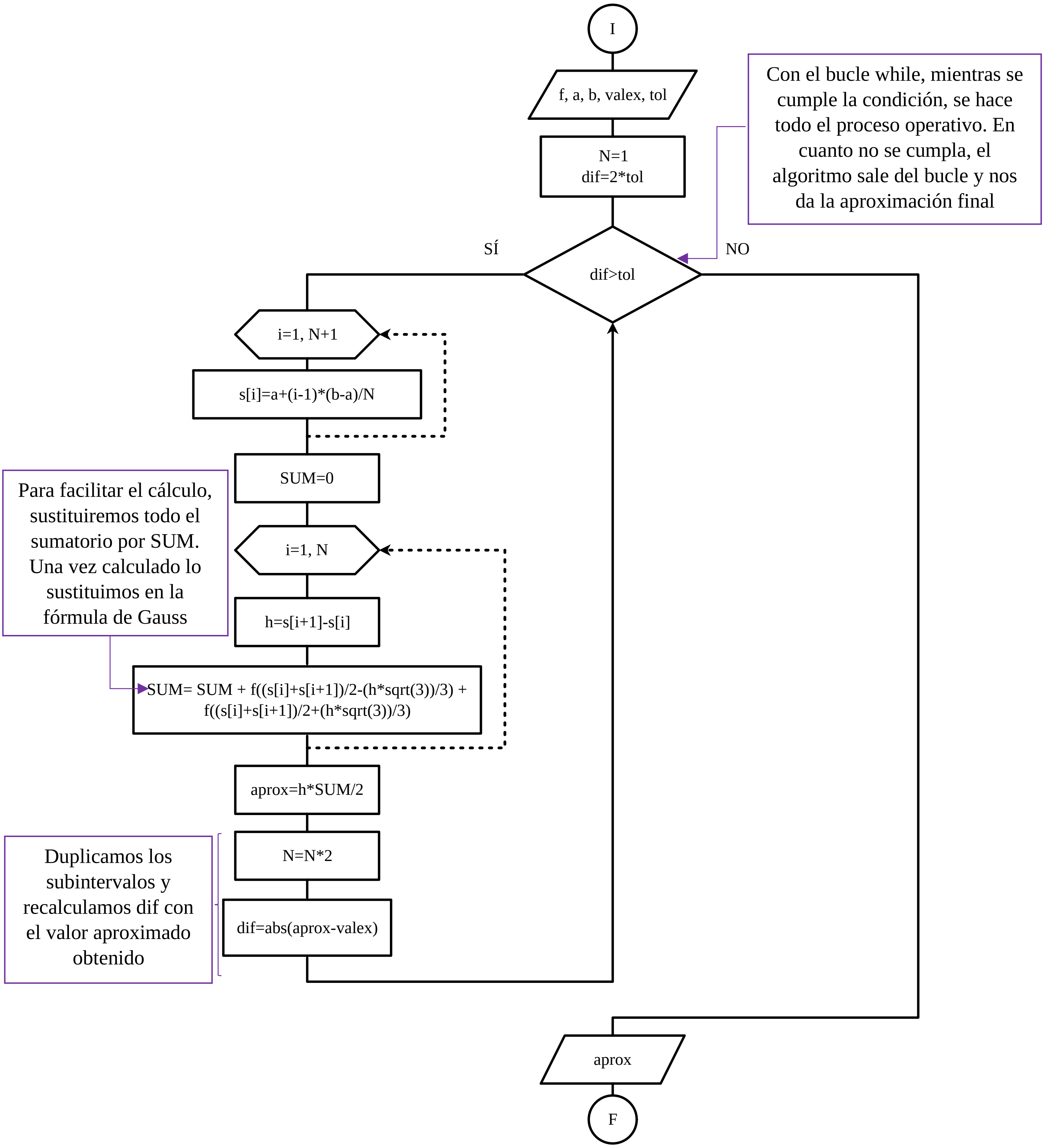
donde N es el número de sub-intervalos que se van considerando durante las iteraciones. El vector s contiene las coordenadas de los puntos que constituyen los extremos de cada sub-intervalo y que habrá que calcular dentro del proceso iterativo.

Al iniciar el proceso iterativo se tomará N=1 (se irá duplicando durante el proceso iterativo) y dif=2*tol.

Este ejercicio es un poco más complicado que los anteriores, pero yendo paso a paso se puede resolver con relativa facilidad. Empezaremos inicializando N=1 y dif=2*tol, y posteriormente tendremos que abrir un bucle WHILE, pues como dice el enunciado, se tiene que ir repitiendo todo el proceso de cálculo MIENTRAS la diferencia entre el valor exacto y el aproximado sea mayor que la tolerancia. Por tanto, al final del proceso debemos recalcular el valor de dif y duplicar el número de subintervalos para el siguiente cálculo de la integral.

Dado que en la fórmula de Gauss aparece el vector s tenemos que empezar por su cálculo. s contiene los extremos de los subintervalos, que irán aumentando conforme se repite el proceso iterativo. S se puede calcular de varias maneras: una de ellas consiste en asignar s[1]=a, y los siguientes términos los calculamos a partir de ese primer punto. Una fórmula general para la distancia entre dos puntos consecutivos es $\frac{b-a}{N}$, y por tanto cada término se puede obtener como

$s_i = s_{i-1} + \frac{b-a}{N}$, con i variando desde 2 (pues el primero está calculado previamente) hasta N+1 (ya que si hay N subintervalos, el vector tiene N+1 elementos). Otra manera sería obtener cada término como $s_i = a + \frac{b-a}{N}(i-1)$, con i variando desde 1 hasta N+1.


I
f, a, b, valex, tol
N=1
dif=2*tol
Con el bucle while, mientras se cumple la condición, se hace todo el proceso operativo. En cuanto no se cumpla, el algoritmo sale del bucle y nos da la aproximación final
SÍ
dif>tol
NO
i=1, N+1
s[i]=a+(i-1)*(b-a)/N
SUM=0
Para facilitar el cálculo, sustituiremos todo el sumatorio por SUM. Una vez calculado lo sustituimos en la fórmula de Gauss
i=1, N
h=s[i+1]-s[i]
SUM= SUM + f((s[i]+s[i+1])/2-(h*sqrt(3))/3) + f((s[i]+s[i+1])/2+(h*sqrt(3))/3)
aprox=h*SUM/2
Duplicamos los subintervalos y recalculamos dif con el valor aproximado obtenido
N=N*2
dif=abs(aprox-valex)
aprox
F