Algoritmos. Elementos básicos

Cálculo Numérico (2514) Lectura 0

Prof. Gilberto Noguera

Universidad Central de Venezuela
Facultad de Ingeniería
Escuela de Ingeniería Eléctrica

Resumen En este documento nos referimos a algunos elementos básicos de programación que se utilizarán en el curso de Cálculo Numérico, como lo son el diagrama N-S, pseudocódigo y en menor medida los diagramas de flujo. El uso del diagrama Nassi-Shneiderman o Chapin nos facilita la comprensión del algoritmo al igual que el diagrama de flujo, pero con la particularidad de evitarnos el uso del estándar simbólico ANSI.
Al escribir los algoritmos se establecen ciertas reglas, entre otras; palabras reservadas, una sintaxis, etc... , que facilitan con su uso visualizar los pasos a seguir en la resolución de un problema o la interpretación de un proceso.
En Cálculo Numérico se hace uso frecuente de la computadora y los algoritmos adquieren importancia, recordemos que para realizar un proceso en la computadora se le debe suministrar al procesador un algoritmo adecuado, expresado en forma de programa escrito en un lenguaje de programación.

1. Algoritmo

Un algoritmo esencialmente se puede definir como una secuencia de instrucciones que al ser ejecutadas en el orden propuesto solucionan o aproximan una solución de un problema bien definido.

Para llegar a la realización de un programa es necesario el diseño previo de un algoritmo, de lo contrario el programa de existir podría no ser fácil de revisar, por esto un algoritmo es una parte primordial de la documentación del programa.

Por otra parte un algoritmo es independiente del lenguaje de programación utilizado para expresar el programa, como dicen Ün lenguaje de programación es sólo un medio para expresar un algoritmo y una computadora es sólo un procesador para ejecutarlo".

1.1. Características de un algoritmo

Para que un algoritmo este bien diseñado debe cumplir con las siguientes características,

- Preciso. Definirse de manera rigurosa, sin dar lugar a ambigüedades.
- Definido. Si se sigue un algoritmo dos veces, se obtendrá el mismo resultado.
- Finito. Debe terminar en algún momento.
- Debe tener cero o más elementos de entrada, es decir, debe tener por lo menos una instrucción que ordene averiguar el dato o los datos.
- Debe producir un resultado. Los datos de salida serán los resultados de efectuar las instrucciones. Los datos de entrada pueden ser ninguno, pero los de la salida deben ser alguno o algunos.
- Se concluye que un algoritmo debe ser suficiente y breve, es decir, no exceder en las instrucciones ni quedarse corto. Entre dos algoritmos que lleven a un mismo objetivo, siempre será mejor el más corto.

1.2. Etapas básicas para solucionar un problema por medios computacionales

Se considera que para solucionar un problema o aproximarse a una solución del mismo utilizando algún método de computación se debe en principio cumplir con las siguientes etapas,

- Análisis del problema, definición y delimitación. Considerar los datos de entrada, el proceso que debe realizar el computador y los datos de salida.
- Diseño y desarrollo del algoritmo. Pseudocódigo o escritura natural del algoritmo, diagramas de flujo, Diagramas rectangulares.
- Prueba de escritorio. Seguimiento manual de los pasos descritos en el algoritmo. Se hace con valores bajos y tiene como fin detectar errores.
- Codificación. Selección de un lenguaje y digitalización del pseudocódigo haciendo uso de la sintaxis y estructura gramatical del lenguaje seleccionado.
- Compilación o interpretación del programa. El software elegido convierte las instrucciones escritas en el lenguaje a las universales comprendidas por el computador.
- Ejecución. El programa es ejecutado por la máquina para llegar a los resultados esperados.
- Depuración (debug). Operación de detectar, localizar y eliminar errores de mal funcionamiento del programa.
- Evaluación de resultados. Obtenidos los resultados se los evalúa para verificar si son correctos. Un programa puede arrojar resultados incorrectos aún cuando la ejecución sea perfecta.

1.3. Tipos de algoritmos

Un algoritmo es cualitativo cuando en sus pasos o instrucciones no están involucrados cálculos numéricos.

Los algoritmos cuantitativos involucran cálculos numéricos. Por ejemplo, solución de una ecuación de segundo grado, solución de un factorial.

1.4. Representación de algoritmos

Un algoritmo se representa por algún método que permite independizarlo del lenguaje de programación elegido para la implementación en máquina.

Para la representación de un algoritmo, antes de ser convertido a lenguaje de programación, se utilizan algunos métodos de representación gráfica o numérica.

Algunos de estos métodos son:

- Diagrama libre (Diagramas de flujo).
- Pseudocódigo.
- Diagramas Nassi-Shneiderman o Chapin.

1.5. Componentes elementales de un algoritmo

Para diseñar un algoritmo se debe comenzar por identificar las tareas más importantes para resolver el problema y disponerlas en le orden en el que han de ser ejecutadas.

Mostramos en un diagrama los tres procesos que consideramos,

Inicial: Información dada al algoritmo, obtenida por ingreso manual del usuario o lectura de archivos.

Principal: Operaciones o cálculos necesarios para encontrar la solución del problema.

Final: Respuestas dadas por el algoritmo o resultados finales de los cálculos, que se mostrarán o almacenarán según requerimientos.

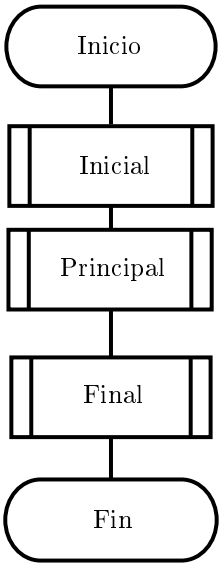

Figura 1: Diagrama de procesos

Los pasos en esta primera descripción de actividades deberán ser refinados añadiendo detalles a los mismo e incluso, algunos de ellos, pueden requerir un refinamiento adicional antes que podamos obtener un algoritmo claro preciso y completo.

En el proceso Inicial debemos considerar como mínimo los siguientes aspectos:

- Inicializar las variables que requerirá el programa.
- ¿Cómo se obtendrán los datos? (1) Manualmente, esto es, se muestra un mensaje en pantalla solicitando que se digitalicen los datos y el formato de ingreso. (2) Se leen o cargan los datos desde un archivo que los contiene.
- ¿Cuántos datos se requerirán?
- ¿Cómo se validarán los datos?
- Asignación de variables que utilizará el proceso de tratamiento o computo de datos.

El proceso Principal, en nuestro caso computo de datos depende exclusivamente del tipo de problema a solucionar, por lo tanto las consideraciones u operaciones están sujetas a las particularidades del problema.

En el proceso Final o de salida se consideran entre otros, aspectos como:

- ¿Cuáles son los datos de salida?
- ¿Cuántos datos de salida se producirá?
- ¿Qué precisión tendrán los resultados?
- ¿Se bebe imprimir una cabecera?
- ¿Se mostrarán los datos por pantalla? ¿Se generará un reporte? ¿Se almacenan los resultados en uno o varios archivos y de que tipo?

1.6. Pseudocódigo

Tal como adelantamos en la introducción otra forma de representar algoritmo que utilizaremos será el pseudocódigos, cuyo objetivo es representar de la forma más detallada posible, y a su vez parecida a un lenguaje de programación.

El pseudocódigo facilita la traducción del algoritmo al lenguaje de programación escogido para la implementación.

A continuación se describen los componentes esenciales del pseuducódigo, también llamado pseudolenguaje.

Alfabeto:

Letras tanto mayúsculas como minúsculas de la A a la Z.

Caracteres y símbolos especiales: $\{ \ \} \ \# \ \% \ \tilde{} \ \$ \ \_ \ \hat{} \ ª \ º \ | \ " @ \ ' \ ` \ ! \ ¿ \ ? \ [\ ] \ : \ ; \ \sim \ - \ + \ =$ espacio en blanco.

Constantes:

enteras: 1, 5678, 0, -3, -1278, etc...

reales: 0.23, -12.567, 12345.78, 10^{-5}, 3×10^{9}, etc...

lógicas: Verdadero (V) y Falso (F)

Operadores:

exponenciación: a^b

suma y resta: $+$ -

relacionales: $==$, $<$, $>$, $\neq$, $\leq$, $\geq$

lógicas: NO , O , Y

Algunas funciones:

$\cos$, $\cosh$, $\tan$, $\arctan$, $\log_e$, $\log_2$, $\log_{10}$, etc...

Álgebra:

vector: $a = [2, 3, \ldots]$

matriz: $A = \begin{bmatrix} a\ b\ c \\ d\ e\ f \\ g\ h\ i \end{bmatrix}$

elemento de vector: $a(i)$ o $a(j)$, etc..

elemento de matriz: $A(i, j)$ etc...

Las sentencias de pseudocódigo asociadas a los bloques estructurales de un diagrama N-S, son

A) Asignación

$y \leftarrow 1$

B) Decisión simple

si <condición> **entonces**
  Acciónes
fin si

C) Decisión binaria

si <condición> **entonces**
  Acciones /* acciones caso verdadero */
si no
  Acciones /* acciones caso falso */
fin si

D) Selección

si <Valor=?> **entonces**
  Acción 1 /* acciones si valor es 1 */
si no
  Acción 2 /* acciones si valor es 2 */
si no
  Acción 3 /* acciones si valor es 3 */
si no

Acción 4 /* acciones si valor es 4 */
si no
Acción 5 /* acciones si valor es 5 */
fin si

D) Para

para $i \leftarrow 1$, n, p **hacer**
Acciones
fin para

E) Mientras

mientras <condición> **hacer**
Acciones
fin mientras

F) Repetir

repetir
Acciones
hasta que <condición>

Nota: Las palabras en negrillas están reservadas para diagramas y pseudocódigos.

Ejemplo 1

Algoritmo 1 Cálculo de raíces de una ecuación de segundo grado

Entrada: $a, b,$
Salida: R, I, R_1, R_2
$d \leftarrow b^2 - 4ac$
si $d < 0$ **entonces**
$R \leftarrow (-b)/(2a)$
$I \leftarrow \sqrt{-d}/(2a)$
si no
si $d = 0$ **entonces**
$R \leftarrow (-b)/(2a)$
si no
$R_1 \leftarrow (-b + \sqrt{d})/(2a)$
$R_2 \leftarrow (-b + \sqrt{d})/(2a)$
fin si
fin si

1.7. Diagrama Nassi-Schneidermane

El diagrama N-S de Nassi-Schneiderman, también conocido como Chapin, es como un diagrama de flujo en el que se omiten las flechas de unión y las cajas son contiguas. Las acciones sucesivas se escriben en cajas sucesivas y como en los diagramas de flujo se pueden escribir diferentes acciones en una caja.

A continuación presentamos los elementos de un algoritmo acorde con la filosofía del diagrama Chapin. Se construye un diagrama N-S colocando los bloque estructurales de arriba hacia abajo y de derecha a izquierda según los requerimientos asociado al algoritmo.

Los elementos estructurales de un diagrama N-S son;

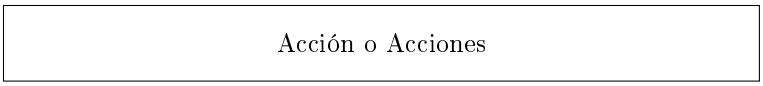

Figura 2: Acción o Acciones

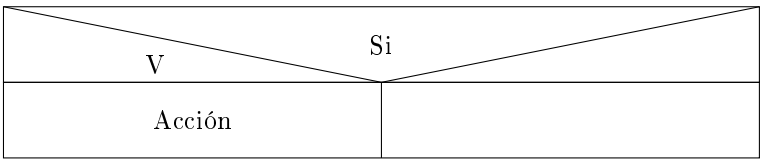

Figura 3: Decisión simple

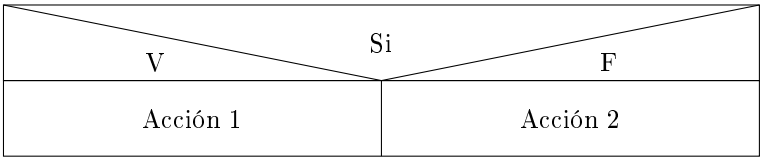

Figura 4: Decisión Binaria

Valor =				
Valor 1	Valor 2	Valor 3	Valor 4	Valor 5
Acción 1	Acción 2	Acción 3	Acción 4	Acción 5

Figura 5: Selección

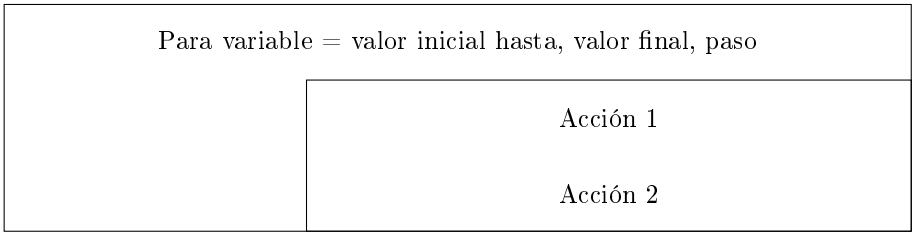

Figura 6: Para

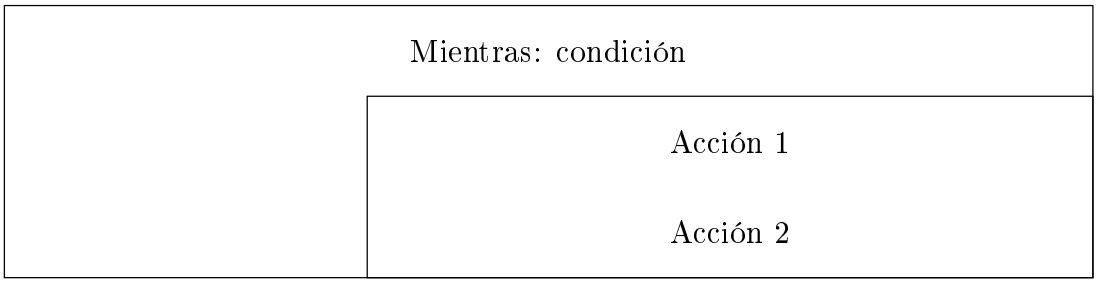

Figura 7: Mientras

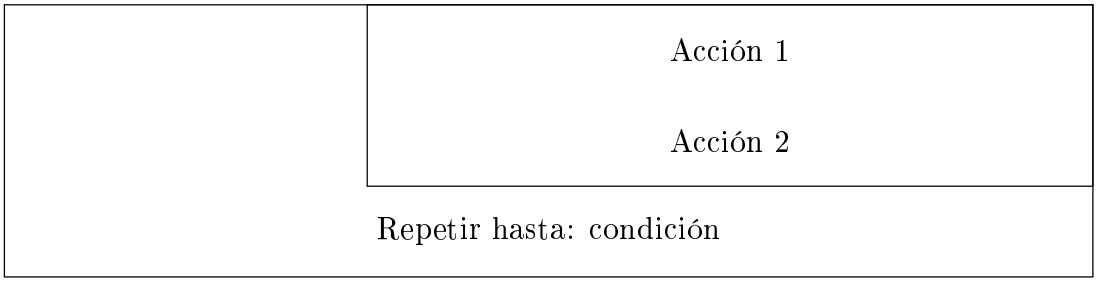

Figura 8: Repetir

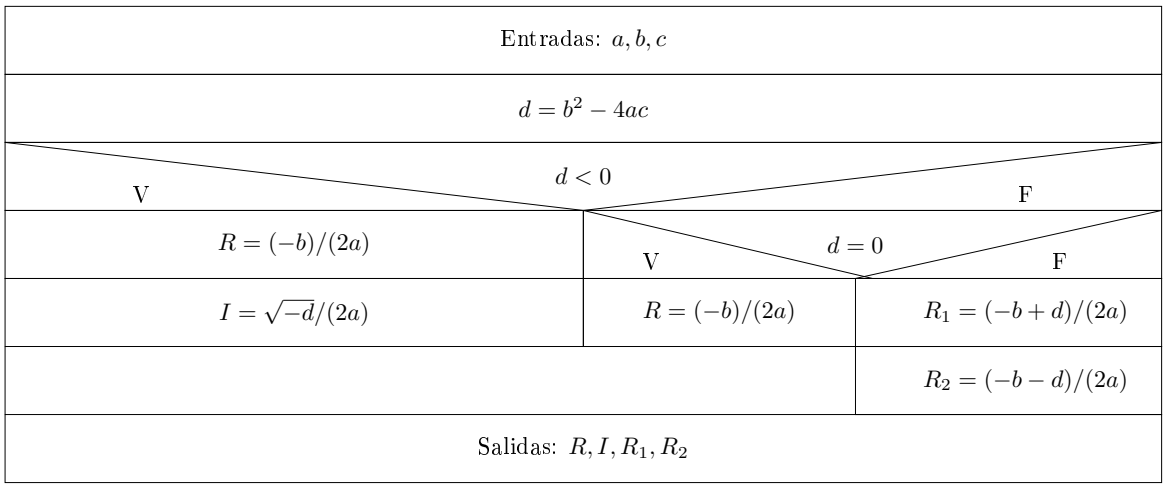

Figura 9: Diagrama N-S proceso principal. Raíces de una ecuación de 2^o