

# Pseudocódigo y diagramas de flujo

Aldo Sayeg Pasos Trejo

Algoritmos Computacionales  
Facultad de Ciencias  
Universidad Nacional Autónoma de México

17 de julio de 2020

# ¿Cómo especificamos un programa en la arquitectura de von Neumann?

Como mencionamos anteriormente, podemos entender un programa o un algoritmo como una lista finita de pasos a seguir que se ejecutan en un orden específico.

Los pasos básicos que se pueden realizar son, en general, los mismos entre cualquier lenguaje de programación. Algunas cosas pueden variar entre los lenguajes de bajo y alto nivel, pero por ahora no es relevante.

# Sintaxis de los lenguajes de programación

La **sintaxis**, es decir, las reglas y principios básicos de un lenguaje de programación, si varía de manera muy arbitraria entre lenguajes incluso para instrucciones muy básicos.

Esto puede producir problemas pues, para especificar un algoritmo, quisiéramos hacerlo de manera **universal**, sin depender de la sintaxis individual de un lenguaje.

Lenguaje	Sintaxis para imprimir x	Sintaxis para leer y asignar a y
Python	<code>print(x)</code>	<code>y = input()</code>
Julia	<code>println(x)</code>	<code>y = readline() ;</code>
C	<code>printf(x);</code>	<code>y = scanf()</code>
C++	<code>cout &lt;&lt; x &lt;&lt; endl ;</code>	<code>cin &gt;&gt; y ;</code>

**Cuadro 0.1:** Ejemplos de diferencias en la sintaxis

# ¿Cómo describimos un programa o un algoritmo para fines de estudio teórico?

Para evitar las dificultades que presenta usar la sintaxis fija de un programa, podemos describir cada instrucción de un programa en un lenguaje natural (i.e. hablado) tal que el proceso sea legible para una persona y no una máquina.

Aunque la intención sea puramente descriptiva y representativa para otros humanos, podemos dejar algunos detalles relevantes para la implementación computacional. A esta descripción normalmente le llamamos **pseudocódigo**

---

## Programa 1: while para contar todos los múltiplos de 3

---

**Input :**

**Output:**

```
1 Sea  $i = 1$ 
2 Sea  $A$  un arreglo de enteros vacío
3 while  $i < 100$  do
4   if  $i \% 3 = 0$  then
5     |   Añadir( $A, i$ )
6   end
7   Sea  $i = i + 1$ 
8 end
```

---

### MATRIX-CHAIN-ORDER( $p$ )

```
1   $n = p.length - 1$ 
2  let  $m[1..n, 1..n]$  and  $s[1..n - 1, 2..n]$  be new tables
3  for  $i = 1$  to  $n$ 
4       $m[i, i] = 0$ 
5  for  $l = 2$  to  $n$            //  $l$  is the chain length
6      for  $i = 1$  to  $n - l + 1$ 
7           $j = i + l - 1$ 
8           $m[i, j] = \infty$ 
9          for  $k = i$  to  $j - 1$ 
10              $q = m[i, k] + m[k + 1, j] + p_{i-1}p_kp_j$ 
11             if  $q < m[i, j]$ 
12                  $m[i, j] = q$ 
13                  $s[i, j] = k$ 
14  return  $m$  and  $s$ 
```

Figura 0.1: Pseudocódigo del algoritmo para encontrar la mejor asociación para multiplicar matrices [1]

### OPTIMAL-BST( $p, q, n$ )

```
1  let  $e[1..n+1, 0..n]$ ,  $w[1..n+1, 0..n]$ ,  
    and  $root[1..n, 1..n]$  be new tables  
2  for  $i = 1$  to  $n + 1$   
3       $e[i, i - 1] = q_{i-1}$   
4       $w[i, i - 1] = q_{i-1}$   
5  for  $l = 1$  to  $n$   
6      for  $i = 1$  to  $n - l + 1$   
7           $j = i + l - 1$   
8           $e[i, j] = \infty$   
9           $w[i, j] = w[i, j - 1] + p_j + q_j$   
10         for  $r = i$  to  $j$   
11              $t = e[i, r - 1] + e[r + 1, j] + w[i, j]$   
12             if  $t < e[i, j]$   
13                  $e[i, j] = t$   
14                  $root[i, j] = r$   
15 return  $e$  and  $root$ 
```

Figura 0.2: Pseudocódigo del algoritmo para un árbol binario de búsqueda [1]



### Metropolis importance sampling Monte Carlo scheme

- (1) Choose an initial state
- (2) Choose a site  $i$
- (3) Calculate the energy change  $\Delta E$  which results if the spin at site  $i$  is overturned
- (4) Generate a random number  $r$  such that  $0 < r < 1$
- (5) If  $r < \exp(-\Delta E/k_B T)$ , flip the spin
- (6) Go to the next site and go to (3)

Figura 0.3: Pseudocódigo del algoritmo de Metrópolis para resolver el modelo de Ising [2]

## Wang-Landau Monte Carlo scheme

- (1) Set  $g(E) = 1$ ; choose a modification factor (e.g.  $f_0 = e^1$ )
- (2) Choose an initial state
- (3) Choose a site  $i$
- (4) Calculate the ratio of the density of states

$$\eta = \frac{g(E_1)}{g(E_2)}$$

which results if the spin at site  $i$  is overturned

- (5) Generate a random number  $r$  such that  $0 < r < 1$
- (6) If  $r < \eta$ , flip the spin
- (7) Set  $g(E_i) \rightarrow g(E_i) * f$
- (8) If the histogram is not 'flat', go to the next site and go to (4)
- (9) If the histogram is 'flat', decrease  $f$ , e.g.  $f_{i+1} = f_i^{1/2}$
- (10) Repeat steps (3)–(9) until  $f = f_{\min} \sim \exp(10^{-8})$
- (11) Calculate properties using final density of states  $g(E)$

Figura 0.4: Pseudocódigo del algoritmo de Wang-Landau para resolver el modelo de Ising [2]

---

## Programa 2: Algoritmo para hacer huevos a la mexicana

---

**Input** : Dos jitomates, una cebolla, un chile serrano, dos huevos

**Output:** Huevo a la mexicana

- 1 Cortar el jitomate, cebolla y chile
  - 2 Saltear todo en un sartén con aceite
  - 3 Añadir huevos crudos y batidos
  - 4 **while** *No este cocinado el huevo* **do**
  - 5     |   Mover la mezcla en el sartén
  - 6 **end**
-

---

### Programa 3: Algoritmo para hacer huevos a la mexicana

---

**Input** : Dos jitomates, una cebolla, un chile serrano, dos huevos

**Output:** Huevo a la mexicana

- 1 Cortar el jitomate
- 2 Cortar la cebolla
- 3 Cortar el chile serrano
- 4 Saltear la cebolla en un sartén con aceite
- 5 Añadir el chile y el tomate
- 6 Añadir huevos crudos y batidos
- 7 **while** *No este cocinado el huevo* **do**
- 8     |   Mover la mezcla en el sartén
- 9 **end**

## Programa 4: Algoritmo para bañarse

---

**Input :**

**Output:**

- 1 Abrir el agua y entrar a la regadera
- 2 Mojarse el cuerpo
- 3 Apagar el agua
- 4 Enjabonarse el cabello con shampoo
- 5 Enjabonarse el cuerpo
- 6 Abrir el agua
- 7 **while** *Haya jabón en el el cuerpo o cabello* **do**
- 8     |   Enjuagarme el cuerpo y el cabello
- 9 **end**
- 10 **if** *Uso acondicionador* **then**
- 11     |   Ponerse acondicionador
- 12     |   Enjuagarse el acondicionador
- 13 **end**
- 14 Cerrar el agua

# ¿Existe alguna otra manera de especificar un algoritmo?

Por su naturaleza ordenada y con estructuras de control, es posible especificar un programa o algoritmo mediante un diagrama.

Dichos diagramas se llaman normalmente **diagramas de flujo** o diagrama de actividades. Existe un estándar internacional para hacer dichos diagramas, llamado **Unified Modeling Language (UML)**

# Símbolos de un diagrama de flujo

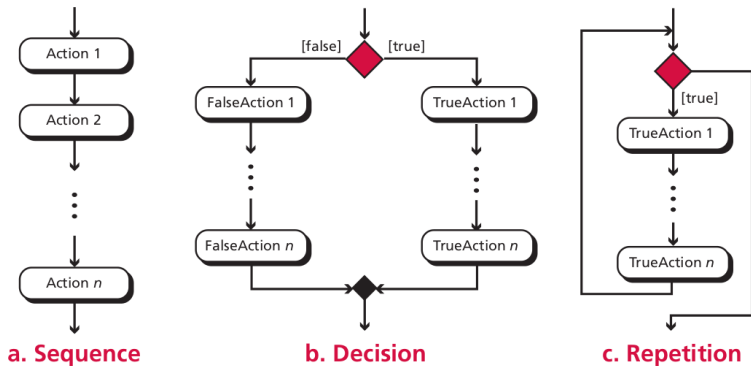
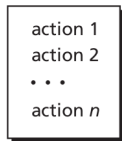
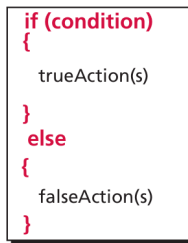


Figura 0.5: Símbolos representantes en el UML [3]

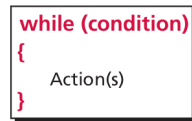
# Símbolos de un diagrama de flujo



**a. Sequence**



**b. Decision**



**c. Repetition**

Figura 0.6: Pseudocódigo equivalente [3]



# Símbolos de un diagrama de flujo

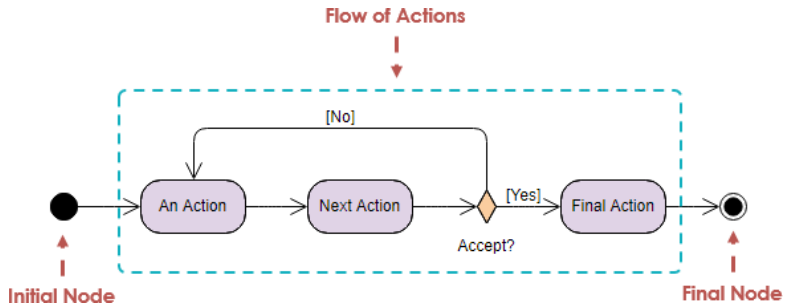


Figura 0.7: Símbolos del UML

# Ejemplo: sumar una lista de enteros

---

## Programa 5: Sumar lista de enteros

---

**Input** : Un arreglo de enteros  $A$

**Output:** Un entero  $suma$

```
1 Sea  $suma = 0$ 
2 Sea  $n = longitud(A)$ 
3 Sea  $i = 1$ 
4 while  $i \leq n$  do
5   |    $suma = suma + A[i]$ 
6   |    $i = i + 1$ 
7 end
```

---

# Ejemplo: sumar un arreglo de enteros

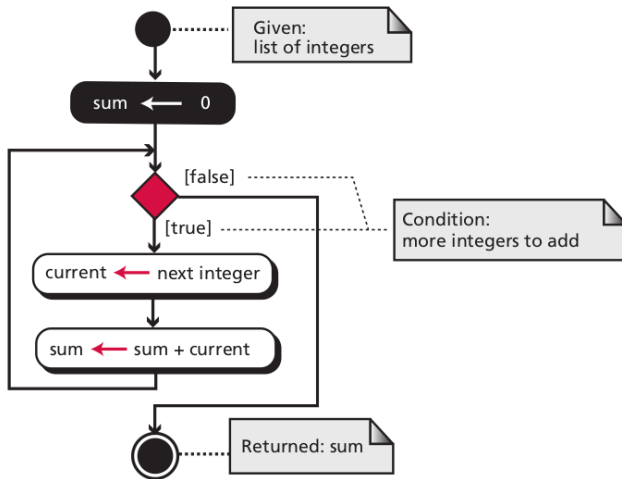


Figura 0.8: Diagrama de flujo del algoritmo

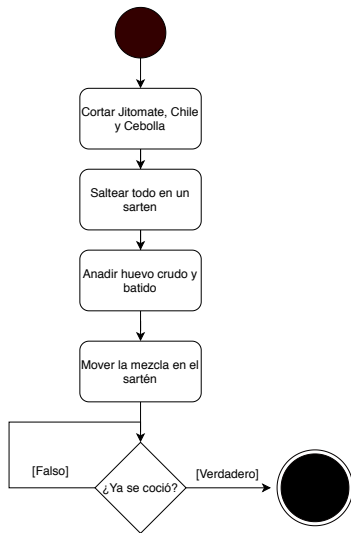


Figura 0.9: Diagrama de flujo para hacer huevo

# Baño

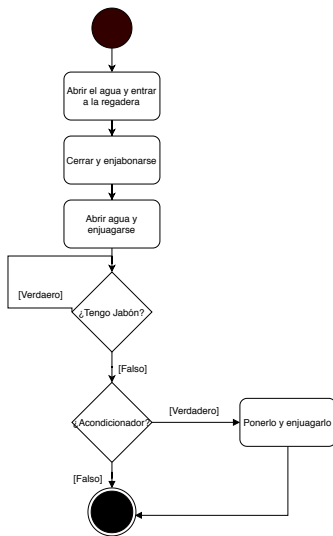


Figura 0.10: Diagrama de flujo para bañarse

# Vida real: química computacional

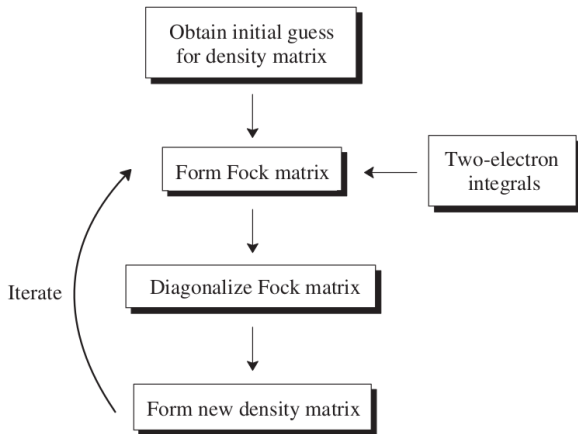


Figura 0.11: Algoritmo para resolver Hartree-Fock mediante orbitales autoconsistentes [4]

# Vida real: hidrodinámica suavizada de partículas

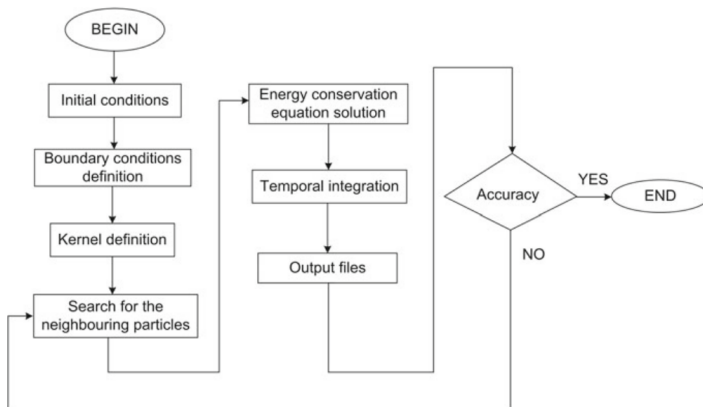


Figura 0.12: Algoritmo para difusión en una placa plana y homogénea [5]

# Referencias



Thomas H Cormen, Charles E Leiserson, Ronald L Rivest, and Clifford Stein.  
*Introduction to Algorithms.*  
MIT press, third edition, 2009.



David Landau and Kurt Binder.  
*A Guide to Monte Carlo Simulations in Statistical Physics.*  
Cambridge University Press, USA, 2005.



Behrouz A Forouzan and Firouz Mosharraf.  
*Foundations of Computer Science.*  
Cengage Learning EMEA, 2007.



Frank Jensen.  
*Introduction to Computational Chemistry.*  
John Wiley and Sons, Inc., Hoboken, NJ, USA, 2006.



C.A.D.F. Filho.  
*Smoothed Particle Hydrodynamics: Fundamentals and Basic Applications in Continuum Mechanics.*  
Springer International Publishing, 2018.