



Unidad Azcapotzalco

División de Ciencias y Artes para el Diseño

Posgrado en Diseño y Visualización de la Información

Algoritmos

L. C. C. Eric Omar Torres Velasco

Ensayo de investigación para la UEA Introducción a la programación

Profesora:

Dra. Lizbeth Gallardo López

Ciudad de México, México. A 21 de mayo de 2019

“El conjunto de instrucciones que especifican la secuencia de operaciones a realizar, en orden, para resolver un sistema específico o clase de problemas, se denomina algoritmo” (Joyanes, 2003: 2). En otras palabras, un algoritmo es una secuencia de pasos o instrucciones a seguir para la solución de cualquier problema de la vida cotidiana.

En la actualidad, para realizar un proceso mediante una computadora se le debe proveer al procesador interno un algoritmo, de modo que se exprese en una forma de programa escrito en un lenguaje de programación, la actividad a ejecutarse por medio del ordenador.




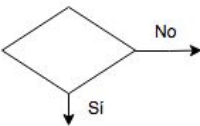
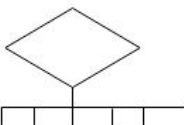



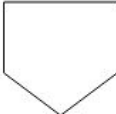

Luis Joyanes (2003) menciona que las características fundamentales que un algoritmo debe cumplir son las siguientes:



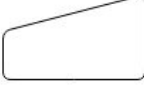

- Ser preciso e indicar el orden de realización de cada paso.
- Estar definido. Esto es, si éste se ejecuta dos veces, las dos veces debe obtener el mismo resultado.
- Ser finito. Es decir, éste debe terminar en algún momento con su operación.

Además, el autor expone que un algoritmo debe describir tres partes: entrada, proceso y salida de datos (Joyanes, 2003: 5).

En cuanto a la representación gráfica de un algoritmo, se pueden utilizar diversos métodos para facilitar el codificado, indistintamente en cualquier lenguaje de programación, de modo que no dependa de ninguna sintaxis en particular. Por ello, un diagrama de flujo es una técnica que utiliza los símbolos (cajas) estándar unidos por flechas denominadas líneas de flujo, para indicar la relación secuencial en que se debe ejecutar dicho algoritmo.

Por lo que, con base en la figura 2.6 en Joyanes (2003) se representa y describe a continuación cada uno de los símbolos de un diagrama de flujo:

Símbolos principales	Descripción
	Terminal (representa el comienzo, «inicio», y el final, «fin», de un programa. Puede representar también una parada o interrupción programada que sea necesario realizar en un programa).
	Entrada/Salida (cualquier tipo de introducción de datos en la memoria desde los periféricos, «entrada», o registro de la información procesada en un periférico, «salida»).
	Proceso (cualquier tipo de operación que pueda originar cambio de valor, formato o posición de la información almacenada en memoria, operaciones aritméticas, de transferencia, etc.).
	Decisión (indica operaciones lógicas o de comparación entre datos –normalmente dos– y en función del resultado de la misma determina cuál de los distintos caminos alternativos del programa se debe seguir; normalmente tiene dos salidas –respuestas SI o NO–, pero puede tener tres o más, según los casos).
	Decisión múltiple (en función del resultado de la comparación se seguirá uno de los diferentes caminos de acuerdo con dicho resultado).
	Conector (sirve para enlazar dos partes cualesquiera de un organigrama a través de un conector de salida y otro conector en la entrada. Se refiere a la conexión en la misma página del diagrama).
	Indicador de dirección o línea de flujo (indica el sentido de ejecución de las operaciones).
	Línea conectora (sirve de unión entre dos símbolos).
	Conector (conexión entre dos puntos del organigrama situado en páginas diferentes).
	Llamada a subrutina o a un proceso predeterminado (una subrutina es un módulo independiente del programa principal, que recibe una entrada procedente de dicho programa, realiza una tarea determinada y regresa, al terminar, al programa principal).

Símbolos secundarios	Descripción
	Pantalla (se utiliza en ocasiones en lugar del símbolo de E/S).
	Impresora (se utiliza en ocasiones en lugar del símbolo de E/S).
	Teclado (se utiliza en ocasiones en lugar del símbolo de E/S).
	Comentarios (se utiliza para añadir comentarios clasificadores a otros símbolos del diagrama de flujo. Se pueden dibujar a cualquier lado del símbolo).

Cabe mencionar que el diagrama de flujo no es el único modo de representación pictórico para un algoritmo, sino que también existen diversos métodos como, por ejemplo: el diagrama N-S (Nassi-Schneiderman), pseudocódigo, lenguaje español y las fórmulas (Joyanes, 2003: 46).

Ahora bien, para la resolución de un problema dado, a través de un algoritmo, existe la posibilidad de diseñar la solución mediante el análisis o la descomposición del problema a partir de una forma sencilla y una perspectiva estructurada. Esta perspectiva se le conoce como paradigma de programación estructurado, debido a al conjunto de técnicas que aumentan la legibilidad de los algoritmos y la facilidad de comprender la escritura, verificación, lectura y el mantenimiento de su flujo lógico a seguir. Por lo que, “en un programa estructurado el flujo lógico se gobierna por las estructuras de control básicas: secuenciales, repetitivas y de selección” (Joyanes, 2003: 95)

Con respecto a las estructuras de control básicas, Böhm y Jacopini (1966) demostraron que un programa propio puede ser escrito utilizando solamente tres tipos de estructuras de control: secuenciales, selectivas y repetitivas (Joyanes, 2003). Las estructuras selectivas o alternativas se utilizan para tomar decisiones lógicas, con base en la evaluación de una condición. Del mismo modo, estas estructuras pueden ser: simples, dobles o múltiples. Las estructuras simples ejecutan una determinada acción cuando se cumple una determinada condición. Esto es, cuando la condición es evaluada sí y solo sí verdadera, realiza una acción. Pero si la condición evaluada es falsa, no realiza nada.

En tanto que, las estructuras dobles permiten “elegir entre dos opciones o alternativas posibles (si-entonces-si_no / if-then-else), en función del cumplimiento o no de una determinada condición” (Joyanes, 2003: 105).

Actualmente, existen lenguajes de alto nivel que su código fuente es traducido, desde su análisis léxico y semántico, por programas a un lenguaje de tipo máquina para su ejecución. A estos lenguajes se les conoce como lenguajes de programación interpretados o compilados. En tanto que, “un intérprete es un traductor que toma un programa fuente, lo traduce y a continuación lo ejecuta” (Joyanes, 2003: 10)

Python es un lenguaje de programación, publicado en 1991 por Guido van Rossum. Se usa para desarrollo web por parte del servidor, desarrollo de software, matemáticas y sistema de scripting. Es un lenguaje interpretado.

A modo de síntesis, un dato es la expresión mínima que describe los objetos, fenómenos o entidades en el mundo real. De modo que, una computadora procesa varios tipos de datos por medio de algoritmos y programas interno. Esto significa,

los datos entran, se procesan y salen en forma de algo, por ejemplo: una representación.

El lenguaje Python es provisto, como en cualquier lenguaje de programación, de tipos datos simples o sin estructura. Estos datos son: numéricos (integer, float, double), lógicos (boolean) y de carácter (char, string).

Los de tipo numérico son el conjunto de los valores numéricos positivos y negativos. Éstos pueden representarse, asimismo en enteros (son valores enteros o completos), flotantes (son valores con 5 decimales) y double (son valores con dos decimales). En relación con los de tipo lógico, también se les conoce como booleano y son aquellos datos que solo puede tomar uno de dos valores, es decir: verdadero (true) o falso (false). Esto es para representar las alternativas sí o no a determinadas condiciones. Por último, en lo que se refiere a datos de tipo carácter, una cadena (string) de caracteres es una sucesión de caracteres que se encuentran delimitados por una comilla (apóstrofo) o dobles comillas, según el tipo de lenguaje de programación. La longitud de una cadena de caracteres es el número de ellos comprendidos entre los separadores o limitadores (Joyanes, 1996: 15)

En resumen, desde los antiguos matemáticos persas ya se creía en la noción de resolver problemas a través del principio del algoritmo. Pero, conforme el avance en esta noción se desarrollaron formas más sencillas para estructurar un problema y así, solucionarlo dependiendo la complejidad de éste.

Bibliografía:

Joyanes, Luis. (2003). Fundamentos de programación: algoritmos y estructuras de datos. México: McGraw-Hill.

Python tutorial. W3Schools. Recuperado el 21 de mayo de 2019 de [dehttps://www.w3schools.com/python/](https://www.w3schools.com/python/)