Q14

Tutorial con los principios básicos de los Algoritmos, pseudocódigo y diagramas de flujo.

Tomado (modificado) de "www.ithinkweb.com.mx/capacita/algoritmo.html", copyright Romeo Sumano - Mexico

ALGORITMOS

En matemáticas, ciencias de la computación, y disciplinas relacionadas, un algoritmo es una lista bien definida, ordenada y finita de operaciones que permite hallar la solución a un problema. Dado un estado inicial y una entrada, a través de pasos sucesivos y bien definidos se llega a un estado final, obteniendo una solución.

En la vida cotidiana se emplean algoritmos en multitud de ocasiones para resolver diversos problemas. Algunos ejemplos se encuentran en los instructivos (manuales de usuario), los cuales muestran algoritmos para usar el aparato en cuestión o inclusive en las instrucciones que recibe un trabajador por parte de su patrón. También existen ejemplos de índole matemático, como el algoritmo de la división para calcular el cociente de dos números o el de Euclides para calcular el máximo común divisor.

Cuando escribimos un programa de computadora, generalmente estamos llevando a cabo un método que se ha inventado para resolver algún problema previamente. Este método es a menudo independiente de la computadora y es probable que sea igualmente apropiado para muchos tipos de computadora y muchos lenguajes. Es el método, en el programa de computación, el que nosotros debemos estudiar para aprender cómo se está tratando de resolver el problema. El término algoritmo se usa en informática para describir un método problema-solución conveniente para la aplicación en un programa de computadora.

Características de los algoritmos

El científico Donald Knuth ofreció una lista de cinco propiedades, ampliamente aceptadas, como requisitos para un algoritmo:

Carácter finito.	"Un algoritmo siempre debe terminar después de un número finito de pasos".
Precisión.	"Cada paso de un algoritmo debe estar precisamente definido; las operaciones a llevar a cabo deben ser especificadas de manera rigurosa y no ambigua para cada caso".
Entrada.	"Un algoritmo tiene cero o más entradas: cantidades que le son dadas antes de que el algoritmo comience, o dinámicamente mientras el algoritmo corre. Estas entradas son tomadas de conjuntos específicos de objetos."
Salida.	"Un algoritmo tiene una o más salidas: cantidades que tienen una relación específica con las entradas".
Eficacia.	"También se espera que un algoritmo sea eficaz, en el sentido de que todas las operaciones a realizar en un algoritmo deben ser suficientemente básicas como para que en principio puedan ser hechas de manera exacta y en un tiempo finito por un hombre usando lápiz y papel".

A partir del carácter finito y de la salida se deduce que ante una misma situación inicial (o valores de entrada) un algoritmo debe proporcionar siempre el mismo resultado (o salida), con excepción de los algoritmos probabilistas.

Medios de expresión de un algoritmo

Los algoritmos pueden ser expresados de muchas maneras, incluyendo al lenguaje natural, pseudocódigo, diagramas de flujo y lenguajes de programación, entre otros. Las descripciones en lenguaje natural tienden a ser ambiguas y extensas. El usar pseudocódigo y diagramas de flujo evita muchas ambigüedades del lenguaje natural. Dichas expresiones son formas más estructuradas para representar algoritmos; no obstante, se mantienen independientes de un lenguaje de programación específico.

La descripción de un algoritmo usualmente se hace en tres niveles:

Descripción de alto nivel.	Se establece el problema, se selecciona un modelo matemático y se explica el algoritmo de manera verbal, posiblemente con ilustraciones y omitiendo detalles.
Descripción formal.	Se usa pseudocódigo para describir la secuencia de pasos que encuentran la solución.
Implementación.	Se muestra el algoritmo expresado en un lenguaje de programación específico (ya "códigos" funcionales, no pseudocódigos) o algún objeto (informático) capaz de llevar a cabo instrucciones.

PSEUDOCÓDIGO

Pseudocódigo es la descripción de un algoritmo que asemeja a un lenguaje de programación pero con algunas convenciones del lenguaje natural. Pseudo significa "falso", imitación, Código se refiere a las instrucciones escritas en el lenguaje de programación; Pseudocódigo no es realmente un código sino una imitación y una versión abreviada de instrucciones reales para la computadora.

Los Pseudocódigos utilizan palabras clave como:

DO (hacer), **IF** – **THEN** – **ELSE** (si – entonces – sino) **ENDIF** (fin de si), **DO UNTIL** (hacer hasta) etc.

DIAGRAMA DE FLUJO

Los diagramas de flujo son descripciones gráficas de algoritmos; usan símbolos conectados con flechas para indicar la secuencia de instrucciones y están regidos por ISO (International Standard Organization) y ANSI (American National Standards Institute)

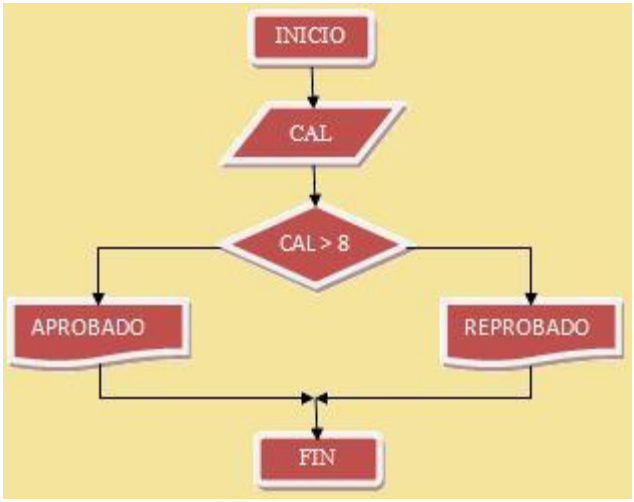
Los diagramas de flujo son usados para representar algoritmos pequeños, ya que abarcan mucho espacio y su construcción es laboriosa. Por su facilidad de lectura son usados como introducción a los algoritmos, descripción de un lenguaje y descripción de procesos a personas ajenas a la computación. Un diagrama de flujo debe ilustrar gráficamente los pasos o procesos a seguir para alcanzar la solución de un problema.

Etapas en la construcción de un diagrama de flujo.

1. Todo diagrama de flujo debe tener un Inicio y un Fin.
2. Las líneas utilizadas para indicar la dirección del flujo del diagrama deben ser Rectas, Verticales y Horizontales. NO pueden ser inclinadas o cruzadas.
3. Todas las líneas que indiquen la dirección del flujo deberán estar conectadas por medio de un símbolo que exprese lectura, proceso, decisión, impresión o fin.
4. La notación utilizada en el diagrama de flujo debe ser independiente del lenguaje de programación.
5. El diagrama de flujo debe ser construido de arriba hacia abajo y de izquierda a derecha.
6. Si el diagrama requiere más de una hoja, debemos utilizar los conectores adecuados y enumerar las páginas convenientemente.
7. No puede llegar más de una línea a un símbolo.

Ejemplo.

Diseñar un algoritmo correspondiente que dado como dato la calificación de un alumno en un examen escriba “Aprobado” si su calificación es mayor que 8 y “Reprobado” en caso contrario.



Solución:

Dato: CAL donde CAL es una variable de tipo real que expresa la calificación del alumno.

1. Leer CAL
2. Si CAL > 8

   ENTONCES escribir "Aprobado"

   SINO Escribir "Reprobado"
3. FIN SI

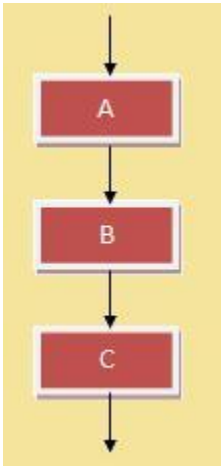
Estructuras de control.

Las estructuras lógicas básicas necesarias para confeccionar un programa se reduce en tres:

SECUENCIALES, SELECTIVAS Y REPETITIVAS

Estructuras secuenciales

Estructura DO - END (INICIO - FIN)



DO

acción A

acción B

acción C

END

Estructuras selectivas.

Las estructuras algorítmicas selectivas que se utilizan para la toma de decisiones lógicas las podemos clasificar de la siguiente manera:

SI ENTONCES que es una estructura selectiva simple.

SI ENTONCES / SINO que es una estructura selectiva doble

SI MÚLTIPLE que es una estructura selectiva múltiple.

Aquí podemos observar una estructura SI ENTONCES/SINO la cual permite que el flujo del diagrama se bifurque por dos ramas diferentes en el punto de la toma de decisiones.

En lenguaje algorítmico se expresa así:

SI condición (verdadera)

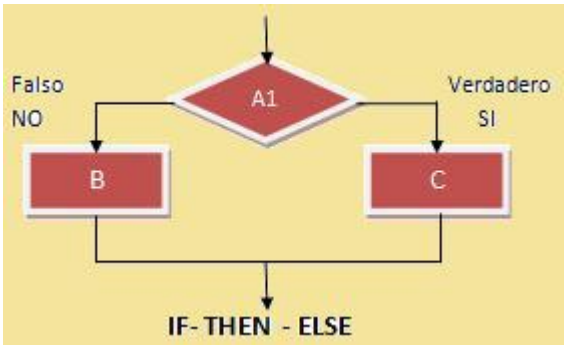
entonces

Hacer operación 1

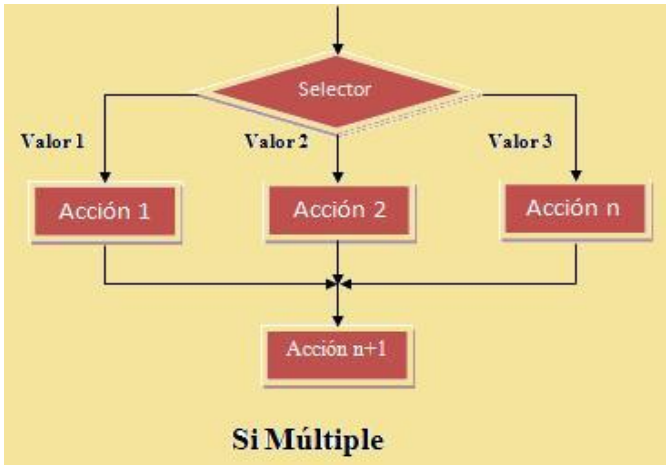
sino

Hacer operación 2

Fin de la condicional

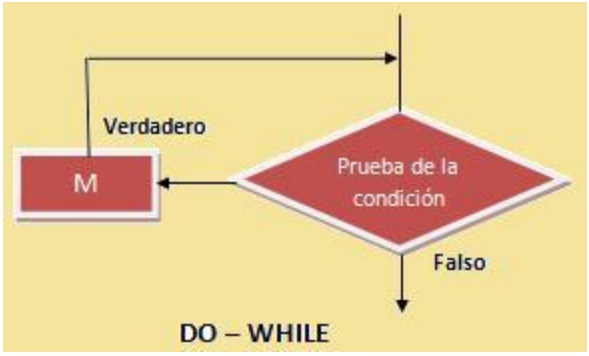


A continuación se muestran las otras estructuras selectivas



Estructuras repetitivas

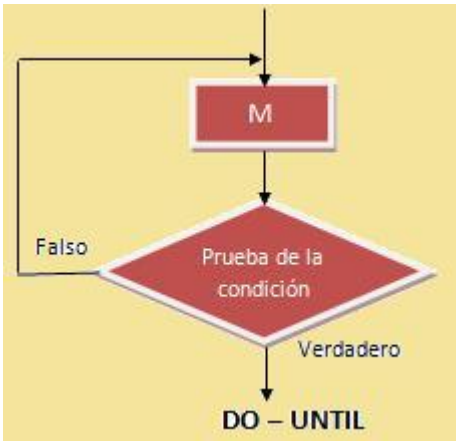
Estructura repetitiva DO WHILE (mientras - hacer)




```
DO WHILE condición
        acción1
        acción2
        .....
END DO
```

El seudocódigo significa: "Mientras la condición sea verdadera hacer la o las acciones; cuando sea falsa, terminar el bucle". Puede ocurrir que el bucle no se ejecute ni una sola vez en el caso de que la condición no se cumpla inicialmente.

Estructura repetitiva DO UNTIL (repetir - hasta)



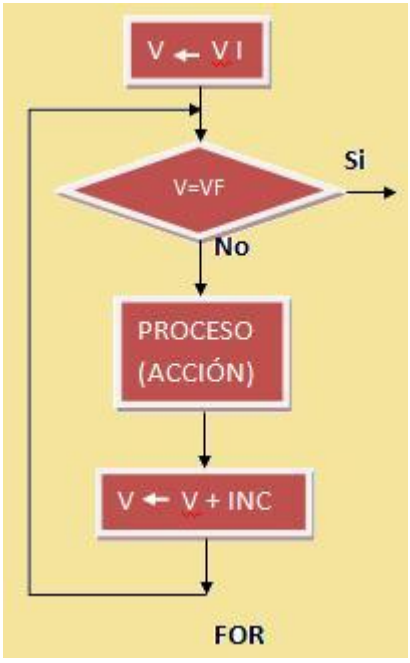

```
DO UNTIL condición
        acción1
        acción2
        .....
END DO
```

El seudocódigo significa: "Repetir la condición y hacer la o las acciones hasta que la condición sea verdadera y terminar el bucle".

Tanto en DOWHILE como en DOUNTIL se necesita que el bucle contenga al menos una instrucción que cambie la condición que controla el bucle. Si no hubiera el bucle continuaría indefinidamente. Estas estructuras se usan cuando No sabemos el número de veces que se repetirá el ciclo.

La estructura REPETIR

La estructura REPETIR es la estructura algorítmica adecuada para utilizar en un ciclo que se ejecutará un número definido de veces. Por ejemplo cuando calculamos las nóminas de una empresa, tenemos que sumar los sueldos de N empleados de la misma. Podemos calcular el promedio de calificaciones de un grupo de alumnos sumando todas las calificaciones y dividir entre el número de alumnos o también sacar el promedio de cada alumno según sus calificaciones mensuales. En todos los casos sabemos de antemano cuántas veces tenemos que repetir una determinada operación.



- V es una variable de control
- VI es el valor inicial
- VF es el valor final
- INC es el incremento

El formato es:

```
FOR variable = expresión1 TO expresión2 [STEP expresión3]
        DO acción
END FOR
```

STEP expresión3 es opcional y es un incremento (positivo o negativo)