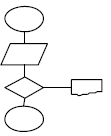
**QUÉ ES ALGORITMO?**   
  
La palabra algoritmo se deriva de la traducción al latín de la palabra árabe alkhowarizmi, nombre de un matemático y astrónomo árabe que escribió un tratado sobre manipulación de números y ecuaciones en el siglo IX.   
  
Un algoritmo es una serie de pasos organizados que describe el proceso que se debe seguir, para dar solución a un problema específico.   
  
**¿TIPOS DE ALGORITMOS…?**   
  
Existen dos tipos y son llamados así por su naturaleza:

* Cualitativos: Son aquellos en los que se describen los pasos utilizando palabras.
* Cuantitativos: Son aquellos en los que se utilizan cálculos numéricos para definir los pasos del proceso.

**Lenguajes Algorítmicos**   
  
Un Lenguaje algorítmico es una serie de símbolos y reglas que se utilizan para describir de manera explícita un proceso.   
  
**Tipos de Lenguajes Algorítmicos**

* Gráficos: Es la representación gráfica de las operaciones que realiza un algoritmo (diagrama de flujo).



* No Gráficos: Representa en forma descriptiva las operaciones que debe realizar un algoritmo (pseudocodigo).   
    
  INICIO   
     Edad: Entero   
     ESCRIBA “cual es tu edad?”   
     Lea Edad   
     SI Edad >=18 entonces   
        ESCRIBA “Eres mayor de Edad”   
     FINSI   
     ESCRIBA “fin del algoritmo”   
     FIN

**Metodología para la solución de un problema mediante un ordenador. Creación de un algoritmo.**

El computador es una máquina que por sí sola no puede hacer nada, necesita ser programada, es decir, introducirle instrucciones u ordenes que le digan lo que tiene que hacer. Un programa es la solución a un problema inicial, así que todo comienza allí: en el Problema. El proceso de programación es el siguiente: Dado un determinado problema el programador debe idear una solución y expresarla usando un algoritmo (aquí es donde entra a jugar); luego de esto, debe codificarlo en un determinado lenguaje de programación y por último ejecutar el programa en el computador el cual refleja una solución al problema inicial. Esto es a grandes rasgos lo que hace el programador de computadores.   
  
La parte que corresponde a este manual es la de: “Dado un determinado problema debemos idear una solución y expresarla usando un ALGORITMO!”.   
  
**Metodología para la solución de problemas por medio de computadora**

* DEFINICIÓN DEL PROBLEMA   
    
  Esta fase está dada por el enunciado del problema, el cual requiere una definición clara y precisa. Es importante que se conozca lo que se desea que realice la computadora; mientras esto no se conozca del todo no tiene mucho caso continuar con la siguiente etapa.
* ANÁLISIS DEL PROBLEMA   
    
  Una vez que se ha comprendido lo que se desea de la computadora, es necesario definir:
  + Los datos de entrada.
  + Cual es la información que se desea producir (salida)
  + Los métodos y fórmulas que se necesitan para procesar los datos.

Una recomendación muy práctica es el de colocarse en el lugar de la computadora y analizar qué es lo que se necesita que se ordene y en qué secuencia para producir los resultados esperados.

* DISEÑO DEL ALGORITMO   
    
  Las características de un buen algoritmo son:
  + Debe tener un punto particular de inicio.
  + Debe ser definido, no debe permitir dobles interpretaciones.
  + Debe ser general, es decir, soportar la mayoría de las variantes que se puedan presentar en la definición del problema.
  + Debe ser finito en tamaño y tiempo de ejecución.
  + Diseño del Algoritmo
  + Prueba de escritorio o Depuración

Se denomina prueba de escritorio a la comprobación que se hace de un algoritmo para saber si está bien hecho. Esta prueba consiste en tomar datos específicos como entrada y seguir la secuencia indicada en el algoritmo hasta obtener un resultado, el análisis de estos resultados indicará si el algoritmo está correcto o si por el contrario hay necesidad de corregirlo o hacerle ajustes.

**Para el proceso de Algoritmos es necesario aprender a desarrollar un conjunto de elementos.**

Todo estos elementos con los cuales se construyen dichos algoritmos se basan en una disciplina llamada: Programación Estructurada.   
  
Empecemos por conocer las reglas para cambiar fórmulas matemáticas a expresiones válidas para la computadora, además de diferenciar constantes e identificadores y tipos de datos simples.   
  
**Tipos De Datos**   
  
Todos los datos tienen un tipo asociado con ellos. Un dato puede ser un simple carácter, tal como ‘b’, un valor entero tal como 35. El tipo de dato determina la naturaleza del conjunto de valores que puede tomar una variable.



**Tipos de Datos Simples**   
  
Datos Numéricos:   
Permiten representar valores escalares de forma numérica, esto incluye a los números enteros y los reales. Este tipo de datos permiten realizar operaciones aritméticas comunes.   
  
Datos lógicos:   
Son aquellos que solo pueden tener dos valores (cierto o falso) ya que representan el resultado de una comparación entre otros datos (numéricos o alfanuméricos).   
  
Datos alfanuméricos (string):   
Es una secuencia de caracteres alfanuméricos que permiten representar valores identificables de forma descriptiva, esto incluye nombres de personas, direcciones, etc. Es posible representar números como alfanuméricos, pero estos pierden su propiedad matemática, es decir no es posible hacer operaciones con ellos. Este tipo de datos se representan encerrados entre comillas.   
  
**Identificadores**   
  
Los identificadores representan los datos de un programa (constantes, variables, tipos de datos). Un identificador es una secuencia de caracteres que sirve para identificar una posición en la memoria de la computadora, que permite acceder a su contenido.   
Ejemplo:

» Nombre

» Num\_hrs

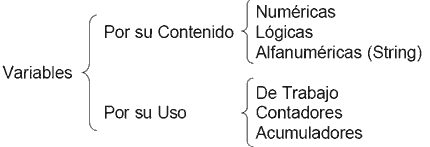
» Calif2

**Reglas para formar un identificador**

* Debe comenzar con una letra (A a Z, mayúsculas o minúsculas) y no deben contener espacios en blanco.
* Letras, dígitos y caracteres como la subraya ( \_ ) están permitidos después del primer carácter.
* La longitud de identificadores puede ser de varios caracteres. Pero es recomendable una longitud promedio de 8 caracteres.
* El nombre del identificador debe dar una idea del valor que contiene.

**Qué son las constantes, las variables y las expresiones en la programación así como su clasificación.**

**Constantes**   
  
Una constante es un dato numérico o alfanumérico que no cambia durante la ejecución del programa.   
Ejemplo:   
pi = 3.1416   
  
**Variable**   
  
Es un espacio en la memoria de la computadora que permite almacenar temporalmente un dato durante la ejecución de un proceso, su contenido puede cambiar durante la ejecución del programa.   
Para poder reconocer una variable en la memoria de la computadora, es necesario darle un nombre con el cual podamos identificarla dentro de un algoritmo.   
Ejemplo:   
area = pi \* radio ^ 2   
Las variables son : el radio, el area y la constate es pi   
  
**Clasificación de las Variables**



**Por su contenido**

* Variables Numéricas: Son aquellas en las cuales se almacenan valores numéricos, positivos o negativos, es decir almacenan números del 0 al 9, signos (+ y -) y el punto decimal.   
  Ejemplo:   
  iva = 0.15 pi = 3.1416 costo = 2500
* Variables Lógicas: Son aquellas que solo pueden tener dos valores (cierto o falso) estos representan el resultado de una comparación entre otros datos.
* Variables Alfanuméricas: Esta formada por caracteres alfanuméricos (letras, números y caracteres especiales).   
  Ejemplo:   
  letra = ’a’ apellido = ’lopez’ direccion = ’Av. Libertad #190’

**Por su uso**

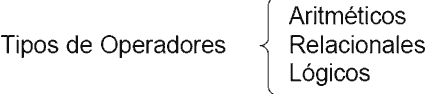
* Variables de Trabajo: Variables que reciben el resultado de una operación matemática completa y que se usan normalmente dentro de un programa.   
  Ejemplo:   
  Suma = a + b /c
* Contadores: Se utilizan para llevar el control del numero de ocasiones en que se realiza una operación o se cumple una condición. Con los incrementos generalmente de uno en uno.
* Acumuladores: Forma que toma una variable y que sirve para llevar la suma acumulativa de una serie de valores que se van leyendo o calculando progresivamente.

**Expresiones**   
  
Las expresiones son combinaciones de constantes, variables, símbolos de operación, paréntesis y nombres de funciones especiales.   
Por ejemplo:   
a + (b + 3) / c   
  
Cada expresión toma un valor que se determina tomando los valores de las variables y constantes implicadas y la ejecución de las operaciones indicadas.   
  
Una expresión consta de operadores y operandos. Según sea el tipo de datos que manipulan, se clasifican las expresiones en:

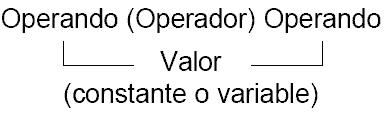
* Aritméticas
* Relacionales
* Lógicas

**Qué son los operadores y los operando, sus tipos y las prioridades de ejecución de los mismos.**

**Operadores**   
  
Son elementos que relacionan de forma diferente, los valores de una o mas variables y/o constantes. Es decir, los operadores nos permiten manipular valores.



**Operadores Aritméticos**  
  
Los operadores aritméticos permiten la realización de operaciones matemáticas con los valores (variables y constantes).   
  
Los operadores aritméticos pueden ser utilizados con tipos de datos enteros o reales. Si ambos son enteros, el resultado es entero; si alguno de ellos es real, el resultado es real.



*Operadores Aritméticos*   
+ Suma   
- Resta   
\* Multiplicación   
/ División   
mod Modulo (residuo de la división entera)   
  
Ejemplos:

|  |  |  |
| --- | --- | --- |
| Expresión |  | Resultado |
| 7 / 2 |  | 3.5 |
| 12 mod 7 |  | 5 |
| 4 + 2 \* 5 |  | 14 |

**Prioridad de los Operadores Aritméticos**   
  
Todas las expresiones entre paréntesis se evalúan primero. Las expresiones con paréntesis anidados se evalúan de dentro a fuera, el paréntesis más interno se evalúa primero.   
Dentro de una misma expresión los operadores se evalúan en el siguiente orden:

1. ^ Exponenciación
2. \*, /, mod Multiplicación, división, modulo.
3. +, - Suma y resta.

Los operadores en una misma expresión con igual nivel de prioridad se evalúan de izquierda a derecha.   
  
Ejemplos:

|  |  |  |
| --- | --- | --- |
| 4 + 2 \* 5 = 14 |  | 23 \* 2 / 5 = 9.2 |
| 3 + 5 \* (10 - (2 + 4)) = 23 |  | 2.1 \* (1.5 + 12.3) = 2.1 \* 13.8 = 28.98 |

**Operadores Relacionales**   
  
Se utilizan para establecer una relación entre dos valores. Luego compara estos valores entre si y esta comparación produce un resultado de certeza o falsedad (verdadero o falso).   
Los operadores relacionales comparan valores del mismo tipo (numéricos o cadenas). Estos tienen el mismo nivel de prioridad en su evaluación.   
Los operadores relaciónales tiene menor prioridad que los aritméticos.   
  
Tipos de operadores Relacionales

* > Mayor que
* < Menor que
* > = Mayor o igual que
* < = Menor o igual que
* < > Diferente
* = Igual

Ejemplos:   
Si a = 10, b = 20, c = 30

|  |  |  |
| --- | --- | --- |
| a + b > c |  | Falso |
| a - b < c |  | Verdadero |
| a - b = c |  | Falso |
| a \* b < > c |  | Verdadero |

Ejemplos no lógicos:   
a < b < c   
10 < 20 < 30   
T > 5 < 30   
(no es lógico porque tiene diferentes operandos)   
  
**Operadores Lógicos**   
  
Estos operadores se utilizan para establecer relaciones entre valores lógicos. Estos valores pueden ser resultado de una expresión relacional.   
  
Tipos de operadores Lógicos   
And Y   
Or O   
Not Negación   
  
Ejemplo:   
Para los siguientes ejemplos T significa verdadero y F falso.



**Operador Not Operador Not**

|  |  |  |
| --- | --- | --- |
| Operando |  | Resultado |
| T |  | F |
| F |  | T |

**Operador And Operador And**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Operando1 |  | Operador |  | Operando2 |  | Resultado |
| T |  | AND |  | T |  | T |
| T | | | | F |  | F |
| F | | | | T |  | F |
| F | | | | F |  | F |

**Operador Or Operador Or**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Operando1 |  | Operador |  | Operando2 |  | Resultado |
| T |  | Or |  | T |  | T |
| T | | | | F |  | T |
| F | | | | T |  | T |
| F | | | | F |  | F |

Prioridad de los Operadores Lógicos

1. Not
2. And
3. Or

Prioridad de los Operadores en General

1. ( )
2. ^
3. \*, /, Mod, Not
4. +, -, And
5. >, <, > =, < =, < >, =, Or

Ejemplos:   
Sea: a = 10 b = 12 c = 13 d =10

