PLANTEMAIENTO DEL PROBLEMA

Identificación de entradas y salidas

Un algoritmo puede ser definido como la **secuencia ordenada** de pasos, sin ambigüedades, que conducen a la resolución de un problema dado y expresado en lenguaje natural, por ejemplo el castellano, Todo algoritmo debe ser:

1. **Preciso:** Indicando el orden de realización de cada uno de los pasos.
2. - **Definido:** Si se sigue el algoritmo varias veces proporcionándole **(consistente)** los mismos datos, se deben obtener siempre los mismos resultados.
3. - **Finito:** Al seguir el algoritmo, este debe terminar en algún momento, es decir tener un número finito de pasos.

Para diseñar un algoritmo se debe comenzar por identificar las tareas más importantes para resolver el problema y disponerlas en el orden en el que han de ser ejecutadas. Los pasos en esta primera descripción pueden requerir una revisión adicional antes de que podamos obtener un algoritmo claro, preciso y completo.

Este método de diseño de algoritmos en etapas, yendo de los conceptos generales a los de detalle, se conoce como método descendente (top-down).

En un algoritmo se deben de considerar tres partes:

- **Entrada:** Información dada al algoritmo.
- **Proceso:** Operaciones o cálculos necesarios para encontrar la solución del problema.
- **Salida:** Respuestas dadas por el algoritmo o resultados finales de los procesos realizados.

Como ejemplo supongamos que desea desarrollar un algoritmo que calcule la superficie de un rectángulo proporcionándole su base y altura. Lo primero que debemos hacer es plantearnos las siguientes preguntas:

Especificaciones de entrada

¿Qué datos son de entrada?
¿Cuántos datos se introducirán?
¿Cuántos son datos de entrada válidos?

Especificaciones de salida

¿Cuáles son los datos de salida?
¿Cuántos datos de salida se producirán?
¿Qué formato y precisión tendrán los resultados?

El algoritmo que podemos utilizar es el siguiente:

Paso 1. Entrada desde el teclado, de los datos de base y altura.
Paso 2. Cálculo de la superficie, multiplicando la base por la altura.
Paso 3. Salida por pantalla de base, altura y superficie calculada.

El lenguaje algorítmico debe ser independiente de cualquier lenguaje de programación particular, pero fácilmente traducible a cada uno de ellos.

Alcanzar estos objetivos conducirá al empleo de **métodos** normalizados para la representación de algoritmos, tales como los diagramas de flujo o **pseudocódigo**.

TIPOS DE DATOS Y CONCEPTOS DE VARIABLES

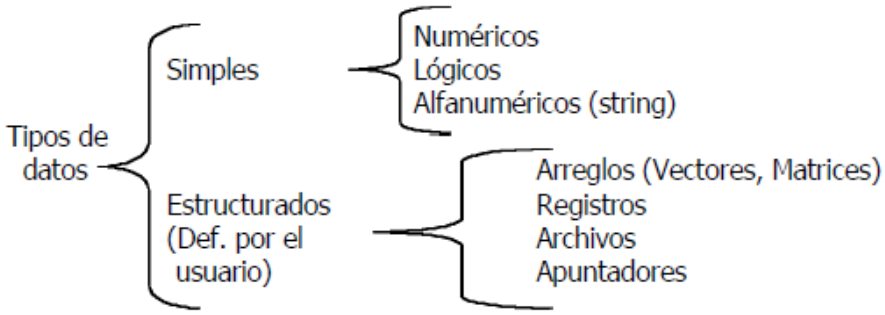
En el ejemplo del cálculo del área de un rectángulo podemos observar que en la resolución de programas nos encontramos con datos que pueden ser números como por ejemplo la base o la altura y otros que pueden ser los mensajes que aparecen por pantalla (" La superficie es ...."). Quiere esto decir que previamente necesitamos conocer **qué TIPOS de datos** puede manejar un ordenador en un programa.

¿Por qué es importante este apartado? Podríamos pensar en lo siguiente:
¿Qué tipo de número es el 7? → Entero
¿Qué tipo de número es el 3? → Entero
¿Qué tipo de número es 7/3 ? → **???**

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.



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 sólo pueden tener dos valores (verdadero 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.

Ejemplo:

"Instituto Tecnológico"

"2002"

Entero: Subconjunto finito de los números enteros, cuyo rango dependerá del lenguaje en el que posteriormente codifiquemos el algoritmo y del ordenador. El rango depende de cuantos bits utilice para codificar el número, normalmente **2 bytes**, Para números **positivos,** con 16 bits se pueden almacenar 216= 65536 números enteros diferentes: de 0 al 65535, y de -32768 al 32767 para números con signo.

Por ejemplo 2, 14, -199,....

Operaciones asociadas al tipo entero: Como norma general las operaciones asociadas a un tipo cualquiera serán aquellas cuyo **resultado** sea un **elemento del mismo tipo**, por tanto:

+, - , *, división → div (cociente), modulo → mod (resto),

sucesor, predecesor, es_par *

(*) es_par(n) → si n es par devuelve un 0 sino, un 1.

Real: Subconjunto de los números reales limitado no sólo en cuanto al tamaño, sino también en cuanto a la precisión. Suelen ocupar 4, 6 ó 10 bytes. Se representan por medio de la ***mantisa***, y un ***exponente*** (1E-3 = 0'001), utilizando *24 bits* para la mantisa (1 para el signo y 23 para el valor) y *8 bits* para el exponente (1 para el signo y 7 para el valor). El orden es de 10-39 hasta 1038.

Por ejemplo 6.9, 33.00123, 3E-34.....

Las operaciones asociadas serian +, -, *, /, etc.

Lógico: Conjunto formado por los valores Cierto y Falso. '1' y '0'.

Operaciones: todas las lógicas y relacionales® AND, OR,

> , < == (igual), >= , <= , ¹ ...

Carácter: Conjunto finito y ordenado de los caracteres que el ordenador reconoce, se almacenan en un ***byte***. Con 8 bits se podrán almacenar 28= 256 valores diferentes (normalmente entre 0 y 255; con ciertos compiladores entre -128 y 127). Un carácter *(letra)* se guarda en un solo byte como un número entero, el correspondiente en el **código ASCII**, que se muestra en la Tabla para los caracteres estándar (existe un código ASCII extendido que utiliza los 256 valores:

Entero, Real, Carácter, Cadena y Lógico son tipos predefinidos en la mayoría de los lenguajes de programación. Además el usuario podrá definir sus propios tipos de datos, si estos facilitan de alguna manera la resolución del problema.

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.

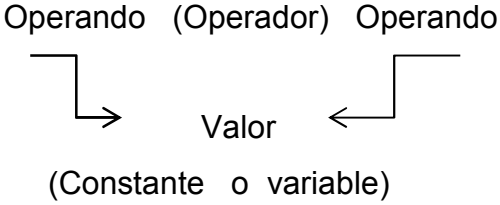
Una expresión consta de operadores y operandos. Según sea el tipo de datos que manipulan, se clasifican las expresiones en:

- Aritméticas
- Relacionales
- Lógicas

Operadores y Operandos

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

A) 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 46 / 5 = 9.2

3 + 5 * (10 - (2 + 4)) = 23 3 + 5 * (10 - 6) = 3 + 5 * 4 = 3 + 20 = 23

3.5 + 5.09 - 14.0 / 40 = 5.09 3.5 + 5.09 - 3.5 = 8.59 - 3.5 = 5.09

2.1 * (1.5 + 3.0 * 4.1) = 28.98 2.1 * (1.5 + 12.3) = 2.1 * 13.8 = 28.98

B) Operadores Relacionales:

· Se utilizan para establecer una relación entre dos valores.

· Compara estos valores entre si y esta comparación produce un resultado de certeza o falsedad (verdadero o falso).

· Los operadores relaciónales comparan valores del mismo tipo (numéricos o cadenas)

· Tienen el mismo nivel de prioridad en su evaluación.

· Los operadores relacionales tiene menor prioridad que los aritméticos.

4.4.1.1 Operadores Relacionales

> Mayor que **<** Menor que **> =** Mayor o igual que **< =** Menor o igual que **< >** Diferente **=** Igual	Ejemplos lógicos: 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" < 30 (no es lógico porque tiene diferentes operandos)

C) Operadores Lógicos: Estos operadores se utilizan para establecer relaciones entre valores lógicos y pueden ser resultado de una expresión relacional.

Operadores Lógicos

And significa **Y**
Or significa **O**
Not significa **Negación**

Operador **And (Y)**				Operador **Or (O)**				Operador **Not (negación)**	
Op1	Operador	Op2	Res	Op1	Operador	Op2	Res	Operando	Resultado
T	**AND**	T	T	T	**OR**	T	T	T	F
T		F	F	T		F	T	F	T
F		T	F	F		T	T		
F		F	F	F		F	F		

Constantes y Variables

Constante: Una constante es un dato numérico o alfanumérico que no cambia durante la ejecución del programa.

Ejemplo:

Pi ← 3,1416

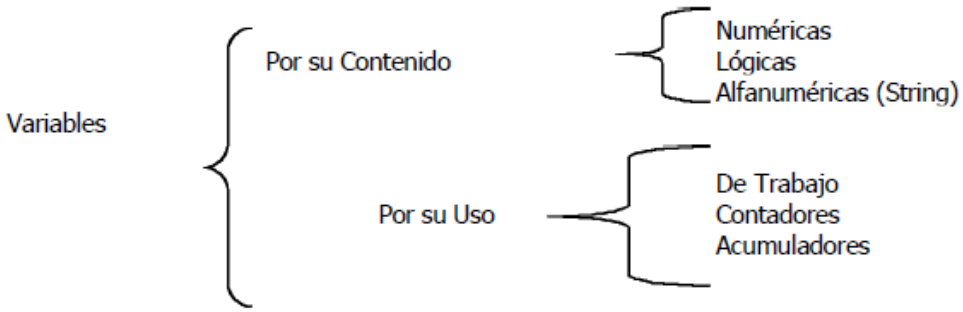
Variable: Es un espacio en la memoria de la computadora que permite almacenar temporalmente un dato durante la ejecución de un proceso, su contenido puede cambia durante la ejecución del programa. Para poder reconocer una variable en la memoria de la computadora, es necesario darle un nombre con el cual podamos identificarla dentro de un algoritmo.

Ejemplo:

Área ← pi * radio ^ 2

Las variables son: radio, área y 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 (verdadero o falso) estos representan el resultado de una comparación entre otros datos.

Variables Alfanuméricas: Están formada por caracteres alfanuméricos (letras, números y caracteres especiales).

Ejemplo:

Letra ← 'a' Apellido ← "López" Dirección ← '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