|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| escudo | **INSTITUCIÓN EDUCATIVA DEPARTAMENTAL SAN JUAN BAUTISTA DE LA SALLE – ZIPAQUIRÀ**  **DIRIGIDA POR LOS HERMANOS DE LAS ESCUELAS CRISTIANAS**  **2015**  **GUÍA 1**   |  |  |  | | --- | --- | --- | | **C**  **X** | **P** | **N** |   **I PERIODO ACADEMICO** | Código : |  |
| Versión Nº: | 01 |
| Fecha: | 06/02/2015 |
| Página 1 de 10 | |

Conceptos básicos de Programación

Introducción

Se pueden utilizar muchos lenguajes para programar una [computadora](http://www.monografias.com/trabajos15/computadoras/computadoras.shtml). El más básico es [el lenguaje](http://www.monografias.com/trabajos16/desarrollo-del-lenguaje/desarrollo-del-lenguaje.shtml) de máquina, una colección de instrucciones muy detallada que controla la circuitería interna de la maquina. Este es el dialecto natural de la maquina. Muy pocos [programas](http://www.monografias.com/Computacion/Programacion/) se escriben actualmente en [lenguaje](http://www.monografias.com/trabajos35/concepto-de-lenguaje/concepto-de-lenguaje.shtml) de maquina por dos razones importantes: primero, porque el lenguaje de maquina es muy incomodo para trabajar y segundo por que la mayoría de las maquinas se pide programar en diversos tipos de lenguajes, que son lenguajes de alto nivel, cuyas instrucciones son más compatibles con los lenguajes y la forma de pensar humanos como lo es el [lenguaje c](http://www.monografias.com/trabajos4/lenguajec/lenguajec.shtml) que además es de propósito general.

Debido a que los programas diseñados en este lenguaje se pueden ejecutar en cualquier maquina, casi sin modificaciones. Por tanto el uso del lenguaje de alto nivel ofrece tres ventajas importantes, sencillez, uniformidad y portabilidad.

1.1. Lenguaje de [programación](http://www.monografias.com/Computacion/Programacion/): [Sistema](http://www.monografias.com/trabajos11/teosis/teosis.shtml) de [símbolos](http://www.monografias.com/trabajos36/signos-simbolos/signos-simbolos.shtml) y reglas que permite la [construcción](http://www.monografias.com/trabajos35/materiales-construccion/materiales-construccion.shtml) de programas con los que [la computadora](http://www.monografias.com/trabajos15/computadoras/computadoras.shtml) puede operar así como resolver [problemas](http://www.monografias.com/trabajos15/calidad-serv/calidad-serv.shtml#PLANT) de manera eficaz.

Estos contienen un conjunto de instrucciones que nos permiten realizar [operaciones](http://www.monografias.com/trabajos6/diop/diop.shtml) de entrada / salida, calculo, manipulación de textos, [lógica](http://www.monografias.com/trabajos15/logica-metodologia/logica-metodologia.shtml) / comparación y [almacenamiento](http://www.monografias.com/trabajos12/dispalm/dispalm.shtml) / recuperación.

Los lenguajes de programación se clasifican en:

* Lenguaje Máquina: Son aquellos cuyas instrucciones son directamente entendibles por la computadora y no necesitan [traducción](http://www.monografias.com/trabajos32/traductor/traductor.shtml) posterior para que la [CPU](http://www.monografias.com/trabajos12/comptcn/comptcn.shtml#UCP) pueda comprender y ejecutar el [programa](http://www.monografias.com/Computacion/Programacion/). Las instrucciones en lenguaje maquina se expresan en términos de la unidad de [memoria](http://www.monografias.com/trabajos13/memor/memor.shtml) más pequeña el bit (dígito binario 0 ó 1).
* Lenguaje de Bajo Nivel ([Ensamblador](http://www.monografias.com/trabajos14/lenguaje-ensamblador/lenguaje-ensamblador.shtml)): En este lenguaje las instrucciones se escriben en códigos alfabéticos conocidos como mnemotécnicos para las operaciones y direcciones simbólicas.
* Lenguaje de Alto Nivel: Los [lenguajes de programación](http://www.monografias.com/Computacion/Programacion/) de alto nivel (BASIC, [pascal](http://www.monografias.com/trabajos5/estat/estat.shtml), [cobol](http://www.monografias.com/trabajos/marcohistocomp/marcohistocomp.shtml), fortran, etc.) son aquellos en los que las instrucciones o sentencias a la computadora son escritas con palabras similares a los lenguajes humanos (en general en ingles), lo que facilita la [escritura](http://www.monografias.com/trabajos16/metodo-lecto-escritura/metodo-lecto-escritura.shtml) y comprensión del programa.

1.2. [Algoritmo](http://www.monografias.com/trabajos15/algoritmos/algoritmos.shtml). 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](http://www.monografias.com/trabajos13/sumato/sumato.shtml#SOLUCION) en el siglo IX.

La solución a cualquier problema de cómputo involucra la ejecución de una serie de [acciones](http://www.monografias.com/trabajos4/acciones/acciones.shtml) en orden especifico. Un [procedimiento](http://www.monografias.com/trabajos13/mapro/mapro.shtml) para resolver un problema en términos de: a) Las acciones a ejecutarse y b) el orden en el cual estas acciones deben ejecutarse se llama algoritmo.

Un ejemplo de un algoritmo para llegar a la escuela

1. Salir de la cama
2. Quitarse la pijamas
3. Darse un baño
4. Vestirse
5. Desayunar
6. Utilizar el [transporte](http://www.monografias.com/trabajos/transporte/transporte.shtml) ( autobús, carro , bicicleta, etc)

Programa.

* + Secuencia de instrucciones mediante las cuales se ejecutan diferentes acciones de acuerdo con los [datos](http://www.monografias.com/trabajos11/basda/basda.shtml) que se estén procesando.
  + Es un algoritmo desarrollado para ser utilizado por la computadora
  + Expresión de un algoritmo en un lenguaje preciso que puede llegar a entender una máquina de cómputo.

2. Fases para la creación de un programa.

2.1 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.

2.2 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](http://www.monografias.com/trabajos7/sisinf/sisinf.shtml) que se desea producir (salida)
* Los [métodos](http://www.monografias.com/trabajos11/metods/metods.shtml) y fórmulas que se necesitan para procesar los datos.

Una recomendación muy practica es el que nos pongamos en el lugar de la computadora y analicemos que es lo que necesitamos que nos ordenen y en que secuencia para producir los resultados esperados.

2.3 [Diseño](http://www.monografias.com/trabajos13/diseprod/diseprod.shtml) 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](http://www.monografias.com/trabajos901/evolucion-historica-concepciones-tiempo/evolucion-historica-concepciones-tiempo.shtml) de ejecución.

2.4 Codificación

La [codificación](http://www.monografias.com/trabajos37/codificacion/codificacion.shtml) es la operación de escribir la solución del problema (de acuerdo a la lógica del [diagrama](http://www.monografias.com/trabajos14/flujograma/flujograma.shtml) de flujo o pseudocódigo), en una serie de instrucciones detalladas, en un [código](http://www.monografias.com/trabajos12/eticaplic/eticaplic.shtml) reconocible por la computadora, la serie de instrucciones detalladas se le conoce como código fuente, el cual se escribe en un lenguaje de programación o lenguaje de alto nivel.

2.5 Prueba y Depuración

Los errores humanos dentro de la programación de [computadoras](http://www.monografias.com/trabajos15/computadoras/computadoras.shtml) son muchos y aumentan considerablemente con la complejidad del problema. El [proceso](http://www.monografias.com/trabajos14/administ-procesos/administ-procesos.shtml#PROCE) de identificar y eliminar errores, para dar paso a una solución sin errores se le llama depuración.

La prueba consiste en la captura de datos hasta que el programa no presente errores (los más comunes son los sintácticos y lógicos).

2.6 Documentación

Es la guía o [comunicación](http://www.monografias.com/trabajos12/fundteo/fundteo.shtml) escrita es sus variadas formas, ya sea en enunciados, [procedimientos](http://www.monografias.com/trabajos13/mapro/mapro.shtml), [dibujos](http://www.monografias.com/trabajos13/histarte/histarte.shtml#ORIGEN) o [diagramas](http://www.monografias.com/trabajos12/diflu/diflu.shtml).

A menudo un programa escrito por una [persona](http://www.monografias.com/trabajos7/perde/perde.shtml), es usado por otra. Por ello la [documentación](http://www.monografias.com/trabajos11/ladocont/ladocont.shtml) sirve para ayudar a comprender o usar un programa o para facilitar futuras modificaciones ([mantenimiento](http://www.monografias.com/trabajos15/mantenimiento-industrial/mantenimiento-industrial.shtml)).

La documentación se divide en tres partes:

Documentación Interna

Documentación Externa

Manual del Usuario

* Documentación Interna: Son los comentarios o mensaje que se añaden al código fuente para hacer mas claro el entendimiento de un proceso.
* Documentación Externa: Se define en un documento escrito los siguientes puntos:

[Descripción](http://monografias.com/trabajos10/anali/anali.shtml) del Problema

Nombre del Autor

Algoritmo (diagrama de flujo o pseudocódigo)

Diccionario de Datos

Código Fuente (programa)

* Manual del Usuario: Describe paso a paso la manera como funciona el programa, con el fin de que el usuario obtenga el resultado deseado.

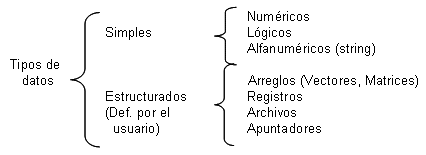
2.7 Mantenimiento

Se lleva a cabo después de terminado el programa, cuando se detecta que es necesario hacer algún [cambio](http://www.monografias.com/trabajos2/mercambiario/mercambiario.shtml), ajuste o complementación al programa para que siga trabajando de manera correcta. Para [poder](http://www.monografias.com/trabajos35/el-poder/el-poder.shtml) realizar este [trabajo](http://www.monografias.com/trabajos34/el-trabajo/el-trabajo.shtml) se requiere que el programa este correctamente documentado.

LOS [DATOS](http://www.monografias.com/trabajos11/basda/basda.shtml) Y [OPERACIONES](http://www.monografias.com/trabajos6/diop/diop.shtml) BÁSICAS.

1. IDENTIFICADOR. Un identificador es una serie de caracteres formados por letras, dígitos y el [carácter](http://www.monografias.com/trabajos34/el-caracter/el-caracter.shtml) subrayado ( \_ ) que no inicie con dígito, asi mismo es el nombre que damos a todo lo que manipulamos dentro de un [programa](http://www.monografias.com/Computacion/Programacion/) ([variables](http://www.monografias.com/trabajos12/guiainf/guiainf.shtml#HIPOTES), constantes, [funciones](http://www.monografias.com/trabajos7/mafu/mafu.shtml), etc). Por ejemplo variables, constantes, funciones, tipos definidos por el usuario etc.

2. TIPOS DE DATOS. Todos los datos tienen un tipo asociado con ellos. Un dato puede ser un simple carácter, tal como b, un [valor](http://www.monografias.com/trabajos14/nuevmicro/nuevmicro.shtml) entero tal como 35. El tipo de dato determina la [naturaleza](http://www.monografias.com/trabajos36/naturaleza/naturaleza.shtml) del conjunto de [valores](http://www.monografias.com/trabajos14/nuevmicro/nuevmicro.shtml) que puede tomar una variable.



3. VARIABLES. Una variable es un identificador que puede tomar diferentes valores dependiendo del tipo que esta se declare.

Una variable es un identificador que puede cambiar de valor durante la ejecución de un programa.

Una variable es una posición de [memoria](http://www.monografias.com/trabajos13/memor/memor.shtml) donde se puede almacenar una valor para uso de un programa.

5. INICIALIZACIÓN DE VARIABLES

Inicializar una variable es el darle un valor después que se ha declarado pero antes de que se ejecuten las sentencias en las que se emplea.

6. CONSTANTES. Constantes son [los valores](http://www.monografias.com/trabajos14/nuevmicro/nuevmicro.shtml) que no pueden ser modificados. En C, pueden ser de cualquier tipo de datos.

Además de los ejemplificados anteriormente, Podemos crear constantes de caracteres con barra invertida. Estos corresponden a los caracteres que son imposibles introducir desde el [teclado](http://www.monografias.com/trabajos5/sisope/sisope2.shtml#tecla).

7. OPERADORES

Un operador es un símbolo que indica al compilador que realice manipulaciones lógicas o [matemáticas](http://www.monografias.com/Matematicas/index.shtml) específicas.

Los operadores del mismo nivel de precedencia son evaluados por el compilador de izquierda a derecha. Por supuesto, se puede utilizar paréntesis para ordenar la [evaluación](http://www.monografias.com/trabajos11/conce/conce.shtml).

También, conviene utilizar paréntesis para hacer más claro el orden en que se producen las evaluaciones, tanto para la [persona](http://www.monografias.com/trabajos7/perde/perde.shtml) que lo elabora o para los que después tengan que seguir el programa.

Operadores Lógicos:

Estos operadores se utilizan para establecer relaciones entre valores lógicos.

Estos valores pueden ser resultado de una expresión relacional.

Operadores Lógicos

And Y

Or O

Not Negación

Prioridad de los Operadores Lógicos

Not

And

Or

Operadores de Asignación. Los operadores de asignación se utilizan para formar expresiones de asignación, en las que se asigna el valor de una expresión a un identificador. \*\*\*\*\* Por definir el operador de asignación\*\* .

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
* Relaciónales
* Lógicas

9. PALABRAS RESERVADAS.

Son palabras que tienen un significado especial para [el lenguaje](http://www.monografias.com/trabajos16/desarrollo-del-lenguaje/desarrollo-del-lenguaje.shtml) y no se pueden utilizar como identificadores.

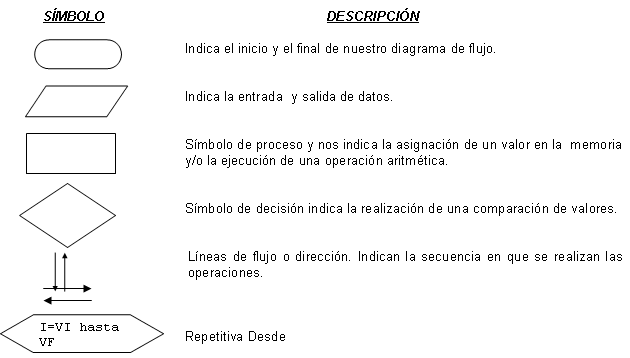
10. COMENTARIOS.

Los comentarios pueden aparecer en cualquier parte del programa, mientras estén situados entre los delimitadores /\* comentario \*/. Los comentarios son útiles para identificar los elementos principales de un programa o para explicar la [lógica](http://www.monografias.com/trabajos15/logica-metodologia/logica-metodologia.shtml) subyacente de estos.

Un [diagrama](http://www.monografias.com/trabajos14/flujograma/flujograma.shtml) de flujo es la representación gráfica de un [algoritmo](http://www.monografias.com/trabajos15/algoritmos/algoritmos.shtml). También se puede decir que es la representación detallada en forma gráfica de como deben realizarse los pasos en la [computadora](http://www.monografias.com/trabajos15/computadoras/computadoras.shtml) para producir resultados.

Esta representación gráfica se da cuando varios [símbolos](http://www.monografias.com/trabajos36/signos-simbolos/signos-simbolos.shtml) (que indican diferentes [procesos](http://www.monografias.com/trabajos14/administ-procesos/administ-procesos.shtml#PROCE) en [la computadora](http://www.monografias.com/trabajos15/computadoras/computadoras.shtml)), se relacionan entre sí mediante líneas que indican el orden en que se deben ejecutar los procesos.

Los símbolos utilizados han sido normalizados por el instituto norteamericano de [normalización](http://www.monografias.com/trabajos5/norbad/norbad.shtml) (ANSI).



Recomendaciones para el [diseño](http://www.monografias.com/trabajos13/diseprod/diseprod.shtml) de [Diagramas](http://www.monografias.com/trabajos12/diflu/diflu.shtml) de Flujo.

Un diagrama de flujo es la representación gráfica de un algoritmo. También se puede decir que es la representación detallada en forma gráfica de como deben realizarse los pasos en la computadora para producir resultados.

Esta representación gráfica se da cuando varios símbolos (que indican diferentes procesos en la computadora), se relacionan entre si mediante líneas que indican el orden en que se deben ejecutar los procesos.

Los símbolos utilizados han sido normalizados por el instituto norteamericano de normalización (ANSI).

* Todo diagrama debe tener un inicio y un fin.
* Se deben se usar solamente líneas de flujo horizontales y/o verticales.
* Se debe evitar el cruce de líneas utilizando los conectores.
* Se deben usar conectores solo cuando sea necesario.
* No deben quedar líneas de flujo son conectar.
* Se deben trazar los símbolos de manera que se puedan leer de arriba hacia abajo y de izquierda a derecha.
* Todo [texto](http://www.monografias.com/trabajos13/libapren/libapren.shtml) escrito dentro de un símbolo deberá ser escrito claramente, evitando el uso de muchas palabras.
* Evitar la terminología de un [lenguaje](http://www.monografias.com/trabajos35/concepto-de-lenguaje/concepto-de-lenguaje.shtml) de [programación](http://www.monografias.com/Computacion/Programacion/) o maquina.
* Utilizar comentarios ya sea al margen o mediante el símbolo grafico comentarios para que este sea entendible por cualquier persona que lo consulte.
* Si el diagrama abarca más de una hoja es conveniente enumerarlo e identificar de donde viene y a donde se dirige.

**Pseudocódigo**

Mezcla de lenguaje de programación y [español](http://www.monografias.com/trabajos5/oriespa/oriespa.shtml) (o ingles o cualquier otro idioma) que se emplea, dentro de la programación estructurada, para realizar el diseño de un programa. En esencial, el pseudocódigo se puede definir como un lenguaje de especificaciones de [algoritmos](http://www.monografias.com/trabajos15/algoritmos/algoritmos.shtml).

Es la representación narrativa de los pasos que debe seguir un algoritmo para dar solución a un problema determinado. El pseudocódigo utiliza palabras que indican el [proceso](http://www.monografias.com/trabajos14/administ-procesos/administ-procesos.shtml#PROCE) a realizar.

El pseudocódigo se concibió para superar las dos principales desventajas del [Diagrama de Flujo](http://www.monografias.com/trabajos12/diflu/diflu.shtml): el diagrama de flujo es lento de crear y difícil de modificar sin un nuevo redibujo. Por otra parte el pseudocódigo es mas fácil de utilizar ya que es similar al lenguaje natural.

Al contrario de los lenguajes de programación de alto nivel como [Pascal](http://www.monografias.com/trabajos5/estat/estat.shtml) o Basic no existe un conjunto de reglas que definan con precisión lo que es y lo que no es pseudocódigo. Varia de un programador a otro.

El pseudocódigo requiere de ciertos símbolos que ya tienen significado preciso y establecido a los que se les conoce como palabras clave. Es necesario que exista una palabra clave para la [selección](http://www.monografias.com/trabajos5/selpe/selpe.shtml) y otra para la iteración condicional, así como para las instrucciones adicionales y otras [estructuras](http://www.monografias.com/trabajos15/todorov/todorov.shtml#INTRO) de [control](http://www.monografias.com/trabajos14/control/control.shtml).

Programación estructurada

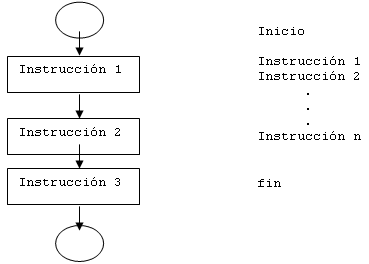
Método disciplinado de escribir [programas](http://www.monografias.com/Computacion/Programacion/) que sean claros, que se demuestren que son correctos y fáciles de modificar

Un programa se compone de:

1. Estructuras de datos.- Los hechos reales, representación en forma de datos, manera en que se organizan los datos.
2. Operaciones primitivas elementales.- Son [acciones](http://www.monografias.com/trabajos4/acciones/acciones.shtml) que se ejecutan sobre los datos para transformarlos en [información](http://www.monografias.com/trabajos7/sisinf/sisinf.shtml).
3. Estructuras de control.- Son los [métodos](http://www.monografias.com/trabajos11/metods/metods.shtml) que existen para dirigir el flujo de acciones que la computadora deberá ejecutar sobre los datos manejados por el programa.

Estructura de control secuencial

La computadora ejecutará automáticamente enunciados uno después del otro, en el orden en el cual se han escrito de inicio a fin.



Ejemplo Sumar dos números:

Pseudocódigo

Inicio

Entero a,b,c declara las variables a utilizar

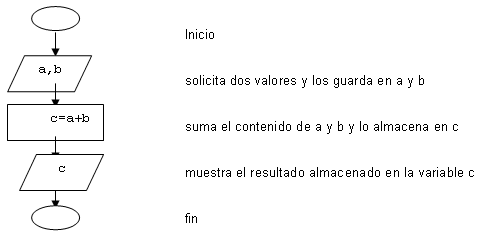
Leer a,b solicita el valor de a y b

c= a+b suma a y b y lo almacena en c

Imprimir c [muestra](http://www.monografias.com/trabajos11/tebas/tebas.shtml) el resultado almacenado en la variable c

fin

Diagrama de flujo

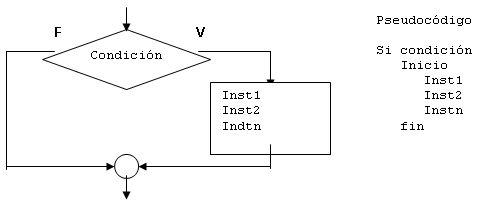


ESTRUCTURA DE CONTROL SELECTIVA

Existen tres tipos de estructuras de control selectivas, estas se basan en una condición o en una opción para decidir la parte del programa por el que pasará.

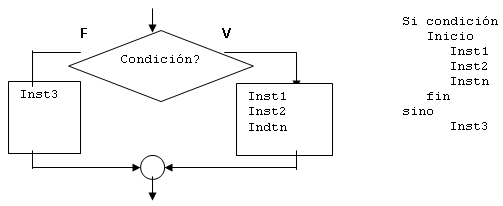
1. Simple b)Doble o compuesta c)Múltiple

Selectiva simple.- evalúa una condición, si esta es verdadera ejecuta la [acción](http://www.monografias.com/trabajos35/categoria-accion/categoria-accion.shtml) o acciones especificadas, si es falsa no realiza ninguna acción.



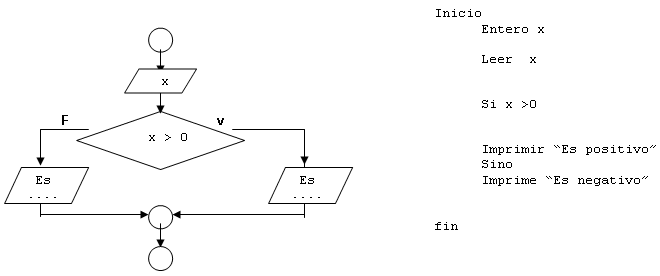
Nota: Si existe sola una instrucción o sentencia dentro de la condición no es necesario marcarlos con inicio y fin, en caso contrario si, como se muestra en el diagrama anterior.

Selectiva doble o compuesta.- evalúa una condición, si esta es verdadera ejecuta la acción o acciones especificadas, si es falsa ejecuta otra acción o acciones.



Nota: Si existe sola una instrucción o sentencia dentro de la condición no es necesario marcarlos con inicio y fin como en este caso que la condición fue falsa, en caso contrario si, en este ejemplo cuando la condición fue verdadera.

Ejemplo: Imprimir si un número es positivo o negativo



Nota: las variables no se especifican en el diagrama de flujo, pero si en el pseudocódigo .

BIBLIOGRAFÍA

* D Appleby, y J.J. VandeKopple, Lenguajes de programación: [paradigma](http://www.monografias.com/trabajos16/paradigmas/paradigmas.shtml#queson) y práctica, McGraw-Hill Interamericana, 1998