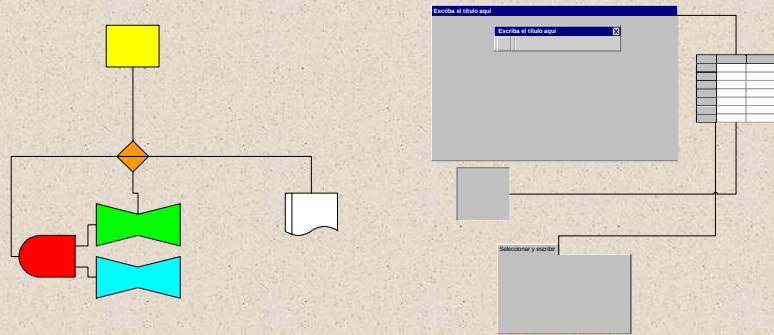


Conceptos Básicos de Lenguajes de Programación



Miguel Ángel Sanz Santos

- ¿Que es un lenguaje de programación?
- ¿Cuantos lenguajes de programación existen?
- ¿Que es un programa?
- ¿Como se escribe un programa?

Componentes estructurales de un programa

- ¿Que son las variables y las constantes?
- ¿Que estructuras de flujo tiene un programa?
- ¿Que son las funciones?
- ¿Que son los objetos y las clases?

Miguel Angel Sanz Santos

¿Que es un lenguaje de programación?



El lenguaje humano lo define el diccionario como **“el conjunto de sonidos articulados con el que el hombre manifiesta lo que piensa o siente”**; a este tipo de lenguaje lo clasificaríamos como natural o propio del ser humano.

El diccionario define los lenguajes de programación como **“un conjunto de caracteres, símbolos, representaciones y reglas que permiten introducir y tratar la información en un ordenador”**, este tipo de lenguaje lo clasificaríamos como un lenguaje artificial.



Miguel Ángel Sanz Santos

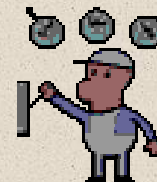
¿Que es un lenguaje de programación?

Dentro de los lenguajes artificiales existen muchos tipos, cada uno de ellos destinado a un propósito concreto; los lenguajes de programación referidos a los ordenadores se pueden clasificar en tres categorías:

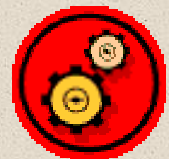
- Lenguajes de bajo nivel.
- Lenguajes de alto nivel.
- Lenguajes de muy alto nivel.



Muy alto nivel



Alto nivel



Bajo nivel

Los primeros son lenguajes que se encuentran muy cerca de la forma interna de comunicación del ordenador, mientras que los terceros son más parecidos al lenguaje humano.

Miguel Ángel Sanz Santos

¿Cuántos lenguajes de programación existen?

Existen numerosos lenguajes de programación y en la última década se han creado para diversos propósitos específicos, teniendo el mayor auge los destinados a Internet.

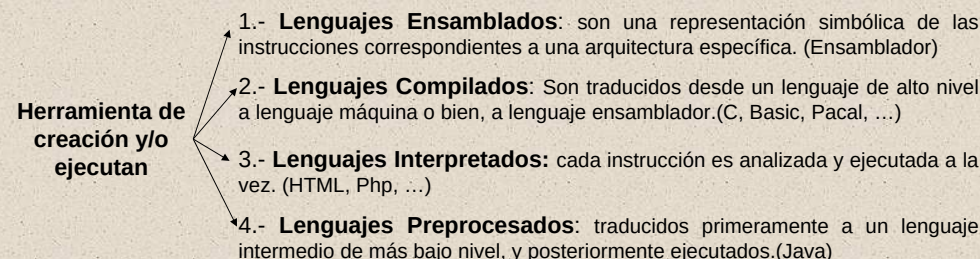
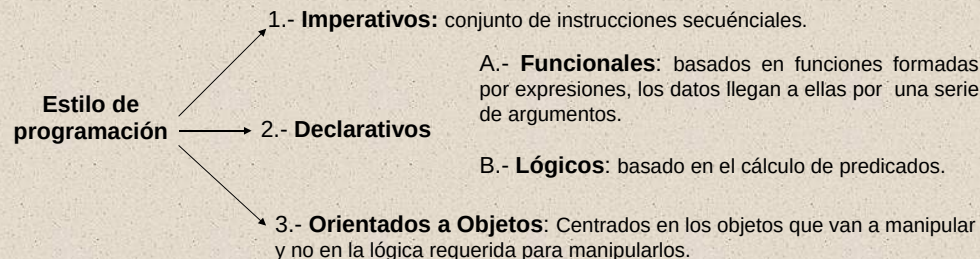
Dentro de los lenguajes de programación los más usados y/o conocidos son:

Ensamblador	1950	
Fortran (FORmula TRANslation)	1957	Laboratorios de IBM bajo la dirección de John Backus
List	1960	John McCarthy pionero de la IA
Cobol (COmmon Business Oriented Language)	1960	Comisión de fabricantes de ordenador de USA
Basic (Beginners All Purpose Symbolic Instruction Code)	1964	J. Kemeny y T. Kurtz en la Universidad de Dartmouth
Pascal (su nombre es homenaje a Blaise Pascal)	1971	Niklaus Wirth
C	1978	Dennis Ritchie en los laboratorios Bell
Prolog	1981	Robert A. Kowalski, de la Universidad de Edimburgo, y Alain Colmerauer, de la Universidad de Aix-Marseille
Ada	1982	Ministerio de Defensa de los EE.UU
C++	1984	Bjarne Stroustrup en los laboratorios Bell
Html	1986	Publicación de la ISO 8879 que presenta el Standard General Markup Language, origen del HTML
Perl (Practical Extraction and Report Language) creado por	1988	Larry Wall
Java	1995	Sun Microsystems para internet y propósito general

Miguel Ángel Sanz Santos

¿Cuántos lenguajes de programación existen?

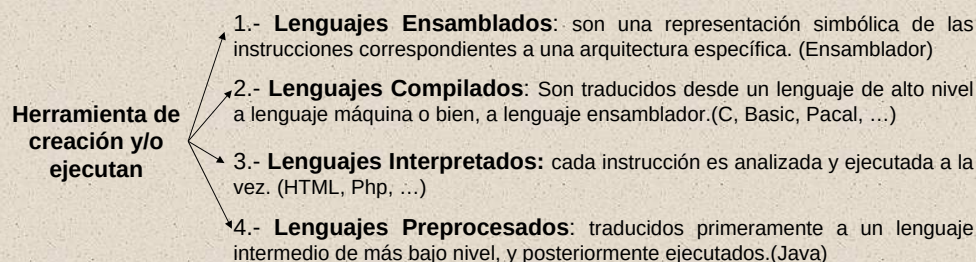
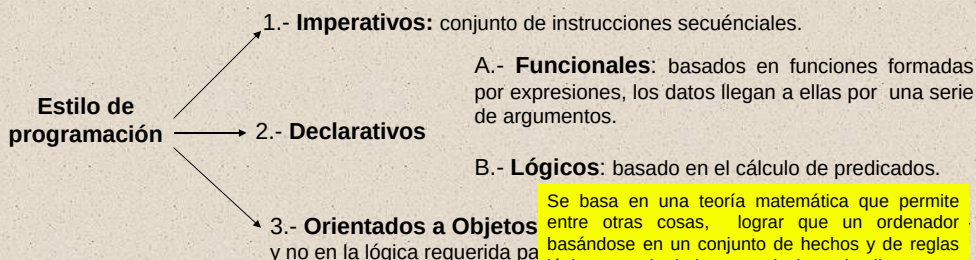
Atendiendo a diferentes criterios podemos clasificar los lenguajes según:



Miguel Angel Sanz Santos

¿Cuántos lenguajes de programación existen?

Atendiendo a diferentes criterios podemos clasificar los lenguajes según:



Miguel Angel Sanz Santos

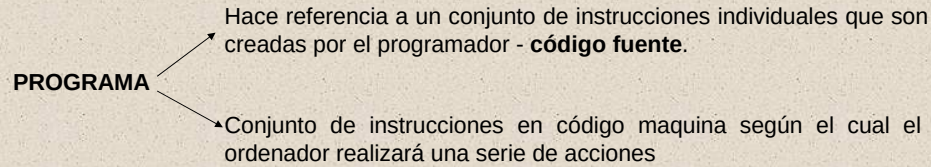
¿Cuántos lenguajes de programación existen?

Atendiendo a su desarrollo histórico

- 1.- **Primera Generación:** asociados a los primeros ordenadores, muy complicados, y en general específicos de cada máquina.
- 2.- **Segunda Generación:** son los lenguajes previos a los más conocidos actualmente y que se encuentran muy cerca del lenguaje máquina. Ejemplo el lenguaje ensamblador.
- 3.- **Tercera Generación:** Son los lenguajes más utilizados actualmente y basados en un funcionamiento procedural (basados en un conjunto de instrucciones que definen la forma de solucionar un problema).
- 4.- **Cuarta Generación:** son los basados en programación visual y orientada a objetos.

Miguel Ángel Sanz Santos

¿Que es un programa?

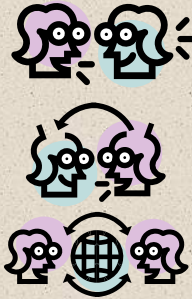


Como en todo lenguaje se diferencian una serie de conceptos que permiten establecer elementos, reglas y significado a nuestro código; así:

Sintaxis: son las reglas con las que deben unirse los elementos de un lenguaje de programación (términos y puntuación) para construir frases (líneas de código).

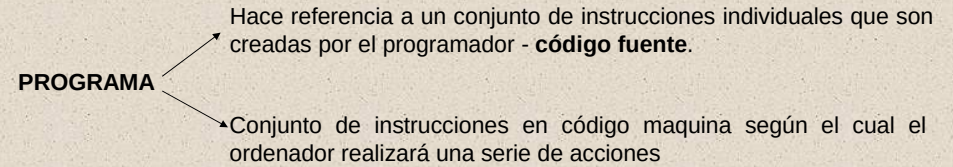
Gramática: es el conjunto de reglas que me permiten implementar instrucciones en un determinado lenguaje.

Semántica: trata el significado y propósito del código



Miguel Ángel Sanz Santos

¿Que es un programa?



Como en todo lenguaje se diferencian una serie de conceptos que permiten establecer elementos, reglas y significado a nuestro código; así:

BASIC

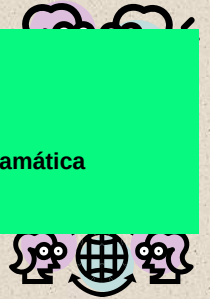
CONTADOR → nivel de semántica

for i=0 to 100 → nivel de sintaxis

contador = contador+1 → nivel de gramática

next i

Semantica: trata el significado y proposito del codigo



Miguel Ángel Sanz Santos

¿Que es un programa?

Miguel Ángel Sanz Santos

La sintaxis difiere enormemente entre los diferentes niveles en los que se diferencian los lenguajes de programación, incluso en un determinado nivel, los diferentes lenguajes presentan tanto formas como procedimientos de construcción distintos.

Por ejemplo:

BASIC

```
for i=0 to 100
    contador = contador+1
next i
```

PASCAL

```
FOR i:=0 to 100 DO
BEGIN
    contador := contador +1;
END;
```

C

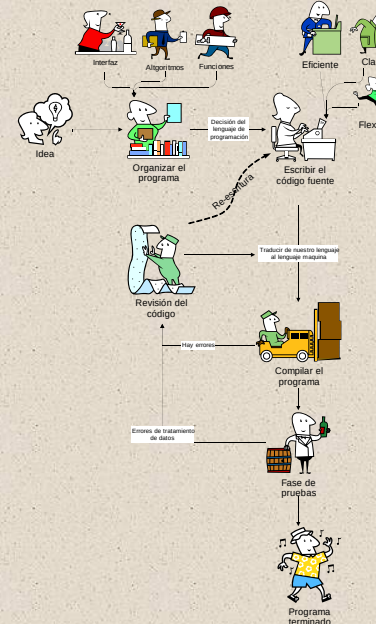
```
for (i=0; i<=100; i++){
    contodor++;
}
```

ENSAMBLADOR

```
MOV CX,0064h
XOR AX,AX
JCXZ yyyy
xxxx ADD AX,CX
LOOP xxxx
yyyy NOP
```

¿Como se escribe un programa?

Los pasos a seguir para crear un programa se esquematizan en:



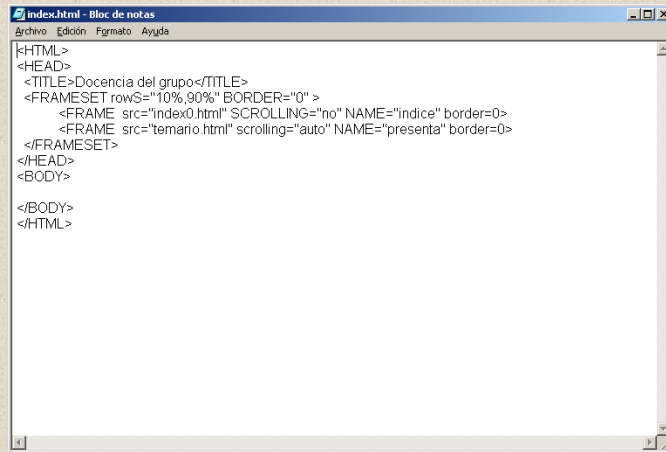
- Tener una idea
- Organizar la idea (interfaz, algoritmos del programa y funciones de tratamiento de datos)
- Selección del lenguaje
- Escritura del código (eficiente, claro y flexible para corregir, modificar y/o ampliar).
- Convertimos a código máquina a través de un **compilar**
 - Si el código da errores deberemos corregirlo y si es necesario rescribirlo y volver a compilar.
 - Si el código no da errores pasa a la siguiente fase
- Fase de prueba (comprobar que no existe fallos conceptuales o de tratamiento de errores)
- Hemos conseguido un programa fiable

Miguel Ángel Sanz Santos

¿Como se escribe un programa?

Miguel Angel Sanz Santos

Las herramientas para crear el código fuente son muy numerosas y van desde los editores de texto más simples, hasta los entornos visuales de programación, que incluyen herramientas de compilación y depuración.



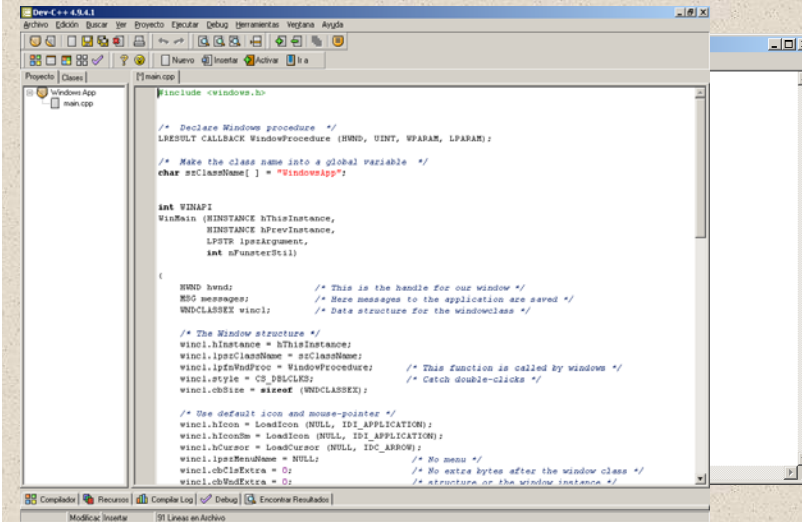
```
index.html - Bloc de notas
Archivo Edición Formato Ayuda

<HTML>
<HEAD>
<TITLE>Docencia del grupo</TITLE>
<FRAMESET rows="10%,90%" BORDER="0">
  <FRAME src="index0.html" SCROLLING="no" NAME="indice" border=0>
  <FRAME src="temario.html" scrolling="auto" NAME="presenta" border=0>
</FRAMESET>
</HEAD>
<BODY>
</BODY>
</HTML>
```

¿Como se escribe un programa?

Miguel Angel Sanz Santos

Las herramientas para crear el código fuente son muy numerosas y van desde los editores de texto más simples, hasta los entornos visuales de programación, que incluyen herramientas de compilación y depuración.



```
Dev-C++ 4.9.4.1
Archivo Edición Buscar Herramientas Ventana Ayuda

#include <windows.h>

/* Declare Window procedure */
LRESULT CALLBACK WindowProc(HWND, UINT, WPARAM, LPARAM);

/* Make the class name into a global variable */
char szClassName[] = "WindowsApp";

int WINAPI
WinMain(HINSTANCE hThisInstance,
        HINSTANCE hPrevInstance,
        LPSTR lpstrArgument,
        int nFunctionStyle)
{
    HWND hwnd; /* This is the handle for our window */
    MSG messages; /* Here messages to the application are saved */
    WNDCLASSEX wincl; /* Data structure for the windowclass */

    /* The Window structure */
    wincl.hInstance = hThisInstance;
    wincl.lpszClassName = szClassName;
    wincl.lpfnWndProc = WindowProc; /* This function is called by windows */
    wincl.style = CS_DBLCLKS; /* Catch double-clicks */
    wincl.cbSize = sizeof(WNDCLASSEX);

    /* Use default icon and mouse-pointer */
    wincl.hIcon = LoadIcon(NULL, IDI_APPLICATION);
    wincl.hIconSm = LoadIcon(NULL, IDI_APPLICATION);
    wincl.hCursor = LoadCursor(NULL, IDC_ARROW);
    wincl.lpszMenuName = NULL; /* No menu */
    wincl.cbClsExtra = 0; /* No extra bytes after the window class */
    wincl.cbWndExtra = 0; /* structure or the window instance */

    /* Register the window class */
    if (!RegisterClassEx(&wincl))
    {
        MessageBox(NULL, "Window Class Registration Failed!", "Error",
            MB_ICONEXCLAMATION);
        return 0;
    }

    hwnd = CreateWindowEx(0, szClassName, "WindowsApp", WS_OVERLAPPED,
        CW_USEDEFAULT, CW_USEDEFAULT, CW_USEDEFAULT, CW_USEDEFAULT,
        hThisInstance, NULL, NULL, hThisInstance);

    if (!hwnd)
    {
        MessageBox(NULL, "CreateWindowEx Failed!", "Error",
            MB_ICONEXCLAMATION);
        return 0;
    }

    ShowWindow(hwnd, nFunctionStyle);
    UpdateWindow(hwnd);

    while (GetMessage(&messages, NULL, 0, 0))
    {
        TranslateMessage(&messages);
        DispatchMessage(&messages);
    }

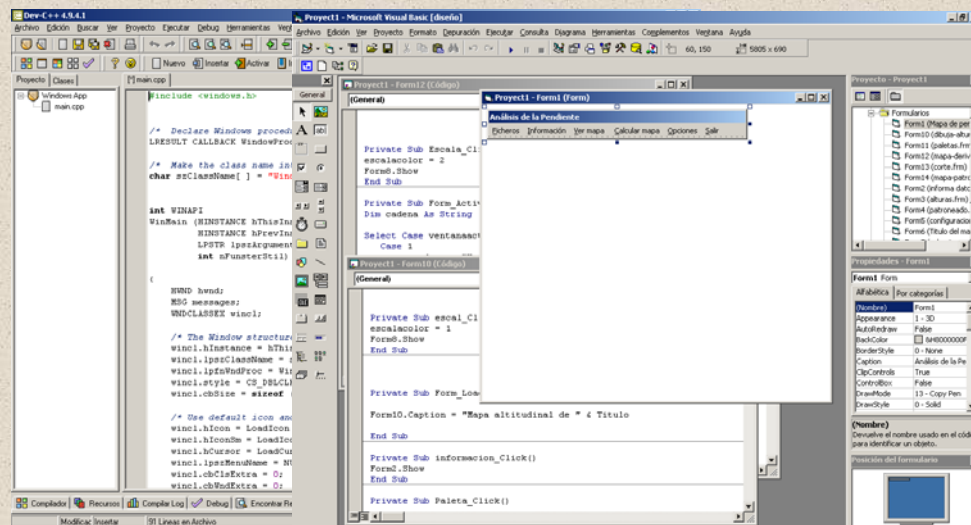
    return 0;
}

LRESULT CALLBACK WindowProc(HWND hwnd, UINT uMsg, WPARAM wParam, LPARAM lParam)
{
    switch (uMsg)
    {
        case WM_DESTROY:
            PostQuitMessage(0);
            return 0;
        default:
            return DefWindowProc(hwnd, uMsg, wParam, lParam);
    }
}
```

¿Como se escribe un programa?

Miguel Angel Sanz Santos

Las herramientas para crear el código fuente son muy numerosas y van desde los editores de texto más simples, hasta los entornos visuales de programación, que incluyen herramientas de compilación y depuración.



```
Dev-C++ 4.9.4.1
Archivo Edición Buscar Herramientas Ventana Ayuda

#include <windows.h>

/* Declare Window procedure */
LRESULT CALLBACK WindowProc(HWND, UINT, WPARAM, LPARAM);

/* Make the class name into a global variable */
char szClassName[] = "WindowsApp";

int WINAPI
WinMain(HINSTANCE hThisInstance,
        HINSTANCE hPrevInstance,
        LPSTR lpstrArgument,
        int nFunctionStyle)
{
    HWND hwnd; /* This is the handle for our window */
    MSG messages; /* Here messages to the application are saved */
    WNDCLASSEX wincl; /* Data structure for the windowclass */

    /* The Window structure */
    wincl.hInstance = hThisInstance;
    wincl.lpszClassName = szClassName;
    wincl.lpfnWndProc = WindowProc; /* This function is called by windows */
    wincl.style = CS_DBLCLKS; /* Catch double-clicks */
    wincl.cbSize = sizeof(WNDCLASSEX);

    /* Use default icon and mouse-pointer */
    wincl.hIcon = LoadIcon(NULL, IDI_APPLICATION);
    wincl.hIconSm = LoadIcon(NULL, IDI_APPLICATION);
    wincl.hCursor = LoadCursor(NULL, IDC_ARROW);
    wincl.lpszMenuName = NULL; /* No menu */
    wincl.cbClsExtra = 0; /* No extra bytes after the window class */
    wincl.cbWndExtra = 0; /* structure or the window instance */

    /* Register the window class */
    if (!RegisterClassEx(&wincl))
    {
        MessageBox(NULL, "Window Class Registration Failed!", "Error",
            MB_ICONEXCLAMATION);
        return 0;
    }

    hwnd = CreateWindowEx(0, szClassName, "WindowsApp", WS_OVERLAPPED,
        CW_USEDEFAULT, CW_USEDEFAULT, CW_USEDEFAULT, CW_USEDEFAULT,
        hThisInstance, NULL, NULL, hThisInstance);

    if (!hwnd)
    {
        MessageBox(NULL, "CreateWindowEx Failed!", "Error",
            MB_ICONEXCLAMATION);
        return 0;
    }

    ShowWindow(hwnd, nFunctionStyle);
    UpdateWindow(hwnd);

    while (GetMessage(&messages, NULL, 0, 0))
    {
        TranslateMessage(&messages);
        DispatchMessage(&messages);
    }

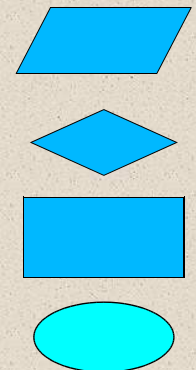
    return 0;
}

LRESULT CALLBACK WindowProc(HWND hwnd, UINT uMsg, WPARAM wParam, LPARAM lParam)
{
    switch (uMsg)
    {
        case WM_DESTROY:
            PostQuitMessage(0);
            return 0;
        default:
            return DefWindowProc(hwnd, uMsg, wParam, lParam);
    }
}
```

¿Componentes estructurales de un programa?

A la hora de escribir el código fuente para un programa se diferencian una serie de elementos que nos van a permitir estructurar los datos, realizar una serie de operaciones y organizar el flujo de "decisión" del programa. Estos elementos son:

- Las variables y las constantes en un programa, así como sus tipos y estructuras.
- Las estructuras de flujo que van a facilitar la ejecución de diferentes cálculos u operaciones dependiendo de unos condicionantes.
- Funciones que conforman una forma de ahorro de código y hace más legible el código.
- Los objetos y clases que nos permiten incluir dentro de los programas elementos ya construidos.



El uso adecuado de este conjunto de elementos nos permitirá realizar códigos eficiente y fáciles de revisar.

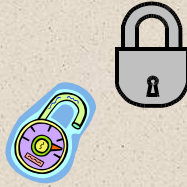
Miguel Angel Sanz Santos

¿Componentes estructurales de un programa? Variables

Los datos con los que trabaja un programa se almacenan por este en dos tipos básicos de estructuras, las variables y las constantes.

Constante: su valor permanece inalterable a lo largo del programa.

Variable: su valor cambia con el flujo del programa.



Dependiendo del lenguaje de programación que estemos utilizando, tanto variables como constantes deberán ser o no definidas en algún TIPO. En el caso de tener que establecer el tipo, éste indicará las características de la variable o constante (tamaño, datos que contendrá, etc.) y se encuentra muy relacionado con la ocupación de memoria de esa variable o constante. La forma que por lo general se usa para establecer el tipo es:

```
tipo [nombre_variable]      float datos1
const tipo [nombre_constante] const float PI = 3.141592653589793238463
```

Miguel Ángel Sanz Santos

¿Componentes estructurales de un programa? Variables

Los datos con los que trabaja un programa se almacenan por este en dos tipos básicos de estructuras, las variables y las constantes.

En los lenguajes más usuales (C, Basic, etc) los tipos de variables más comunes son:

- Bool (booleano) : 1Byte -> 0 ó 1
- Integer/Int (entero): 2/4 Byte -> -32768 a 32767 / -2.147.484.648 a 2.147.483.647
- Single/Float (coma flotante sencillo) 4 Byte -3,402823E38 a -1,401298E-45 y 1.401298E-45 a 3.402823E38
- Double (doble coma flotante) 8 Byte -1,79769313486232E308 a -4,94065645841247E-324 y 4,94065645841247E-324 a 1,79769313486232E308
- Char (carácter) 1 Byte -> 256 caracteres
- String (cadena) Estáticas/Dinámicas 0 a 63000 caracteres / 0 a 2.000.000 caracteres
- Variant: variables especiales que almacenan cualquiera de los tipos anteriores.
- Objet (objeto) 4 Byte hace referencia a objetos de la aplicación
- Null (nulo) 1/2 Byte -> valor nulo
- Etc

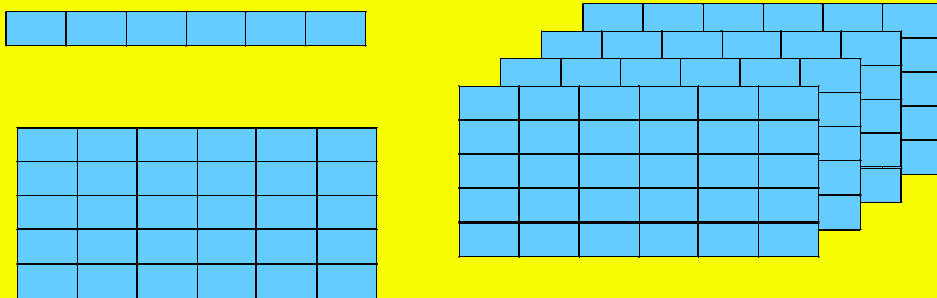
Existe algunos modificadores como *unsigned* que trata a las variables solamente como números positivos.

Miguel Ángel Sanz Santos

¿Componentes estructurales de un programa? Variables

Los datos con los que trabaja un programa se almacenan por este en dos tipos básicos de

Las variables pueden existir de forma individual o agruparse en conjuntos ordenados que reciben el nombre de **matrices**. Las matrices pueden presentar desde 1 dimensión (vectores) hasta un número de dimensiones n que depende del lenguaje de programación, lo más normal son matrices de dos y tres dimensiones.



Miguel Ángel Sanz Santos

¿Componentes estructurales de un programa? Operadores

Los operadores nos van a permitir realizar cálculos, comparaciones y asignaciones entre las variables y constantes de un programa. Los operadores más comunes en los lenguajes de programación son:

- Aritméticos (+, -, *, /, %, _mod, ^)
- Lógicos (and, or, xor, not)
- Asignación(=, +=, -=, ...)
- Comparación (=, <, >, <=, >=, <=)

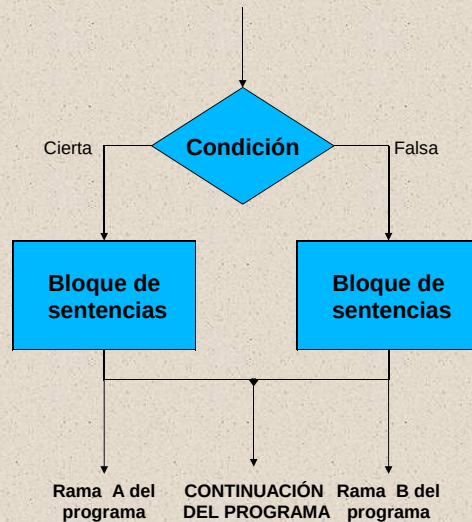
No existe un acuerdo en la simbología utilizada para cada operador entre los diferentes lenguajes, por lo que deberemos en cada uno de los casos consultar las ayudas pertinente. Tampoco existe consenso en los tipos de operadores y cada lenguaje ofrece un surtido de operadores diferente al resto de los otros lenguajes.

Miguel Ángel Sanz Santos

¿Componentes estructurales de un programa? Flujo del programa

Las estructuras que van a ser explicadas a continuación permiten la variación de los procesos en función de datos introducidos por el usuario o resultados de algún cálculo. Este conjunto de estructuras pue encontrarse de forma individual o anidadas (una dentro de otra).

Sentencia **IF ... ELSE ...** corresponde a un *SI condicional*. Esta sentencia comprobará una o varias condiciones controladas por operadores de igualdad y/o lógicos, si la condición se cumple realizará una serie de operaciones y si no se cumple realizará otro conjunto de operaciones. La sentencia puede carecer de la segunda parte (ELSE) y ejecutar un conjunto de sentencias si se cumple la condición impuesta.

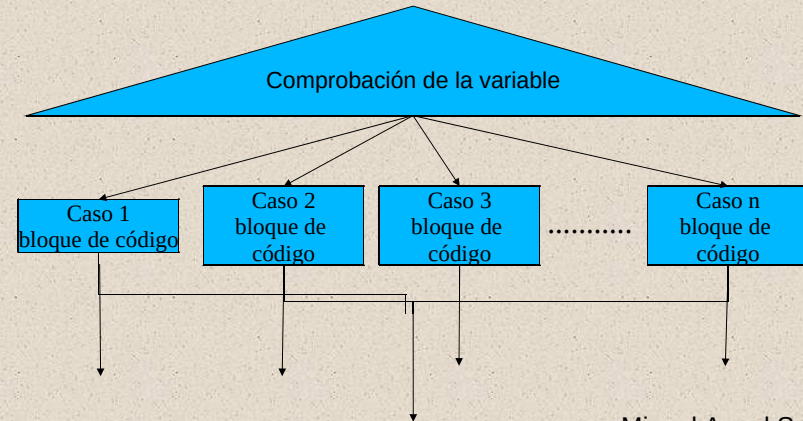


Miguel Angel Sanz Santos

¿Componentes estructurales de un programa? Flujo del programa

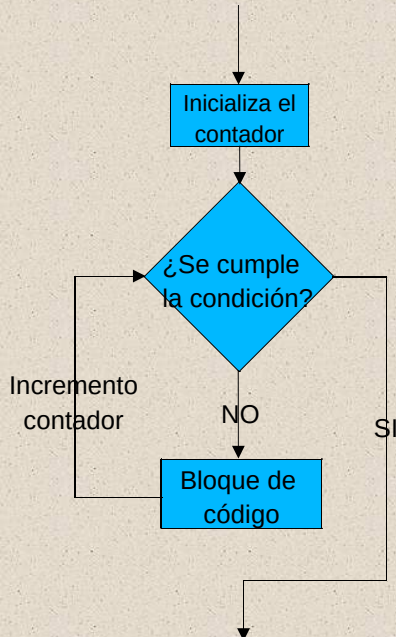
La estructura **CASE o SWITCH** funciona como un conjunto de sentencias IF. El comando funciona según el valor de una variable, otorgando bloques de código según los posibles valores.

Igual que en el caso de IF pueden encontrarse estructuras anidadas, pero en este caso la sentencias anidadas afectarán al bloque donde se encuentran y no podrán cruzarse.



Miguel Angel Sanz Santos

¿Componentes estructurales de un programa? Flujo del programa

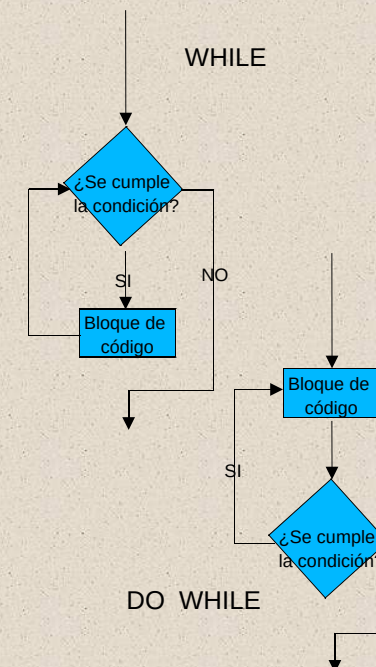


Bucles son estructuras en las que un bloque de código se repite un número determinado de veces o hasta que una condición se cumple.

El bucle FOR ejecuta un bloque de sentencias mientras se cumpla una condición, aumentando en cada ejecución un contador en un determinado valor. Es una sentencia que se utiliza mucho, sobre todo cuando conocemos las dimensiones de lo que queremos analizar.

Miguel Angel Sanz Santos

¿Componentes estructurales de un programa? Flujo del programa



El bucle WHILE y su variante DO ... WHILE ejecutarán un bloque de datos mientras una condición lógica resulte verdadera.

La diferencia entre ambos es que el primero comprueba la condición antes de ejecutar el código, mientras el segundo primero ejecuta el código y después comprueba la condición.

En contraposición al bucle FOR, este tipo de bucles no llevan contador de control.

Hay que tener cuidado a la hora de establecer la condición, ya que si la condición establecida nunca se puede conseguir el programa entrará en un ciclo infinito de interacciones

Miguel Angel Sanz Santos

¿Componentes estructurales de un programa? Flujo del programa

Junto a estos comandos de flujo existen una serie de comando que permiten romper dicho flujo continuando la ejecución del programa tras la ultima sentencia del bucle. Dentro de este conjunto de sentencias se pueden diferenciar cuatro, que son :

- **Break:** puede utilizarse en sentencias de tipo condicional (switch/if) o en bucles; en el primero de los casos produce un salto incondicional al final de la sentencia, mientras que en un bucle produce la salida del mismo.
- **Continue:** solo puede ejecutarse en el interior de un bucle y produce un salto tras la ultima línea del bloque sentencia del bucle, pero no sale del bucle.
- **Exit:** es una terminación forzada del programa.
- **Return:** produce un salto incondicional al final de una función.

Además de estas sentencias en muchos lenguajes aún se mantiene el comando GOTO (etiqueta) que realiza un salto incondicional a una posición del programa representada por una etiqueta. Es poco aconsejable utilizar este comando por la complejidad que crea en la estructura del programa.

Miguel Angel Sanz Santos

¿Componentes estructurales de un programa? Funciones

Las funciones son conjuntos de sentencias (una o más) que realizan una determinada tarea. La utilización de funciones dentro de un programa nos va ha permitir:

- Estructurar el programa como si fueran módulos, lo que facilita su revisión y ampliación.
- Ahorrar código ya que una misma función puede ser llamada desde distintas partes del programa, lo que permite no tener que repetir el mismo código varias veces.

En realidad son pequeñas partes del programa que pueden contener tanto datos como definiciones de variables propias, llamar a otras funciones e incluso a si mismas (recursividad).

Las funciones constan de dos partes: la **declaración** que hace referencia a como será la función y la **definición** que es la propia función

TIPO nombre_función (parámetros)

Su definición es igual que la de una variables y toda función debe de tener un tipo, que indicara el tipo de valor que la función nos devolverá tras ejecutarse.

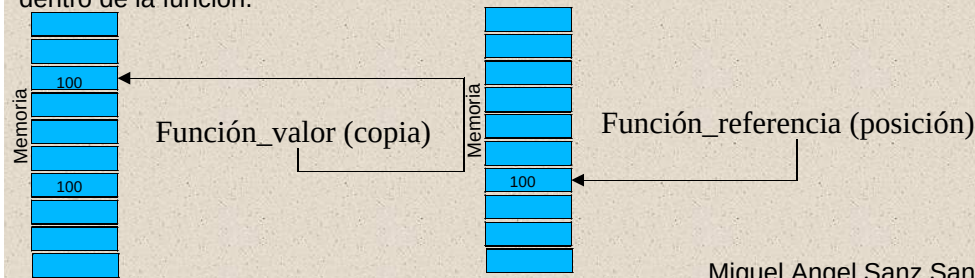
Miguel Angel Sanz Santos

¿Componentes estructurales de un programa? Funciones

A las funciones les podemos enviar datos a través de los parámetros de la función, es tos parámetros son variables a las que se les asigna un valor en cada llamada a la función y como variables que son tienen un tipo. Existen dos formas de pasar valores a una función:

Por valor: en este caso el valor que se pasa es una copia del valor de una variable, por tanto una vez procesada la función y devuelto el valor, el contenido de la variable que ha pasado el dato no se ve afectada.

Por referencia: en este caso no se pasa el contenido de una variable, si no su posición de almacenamiento en la memoria, por tanto su contenido puede ser variado dentro de la función.



Miguel Angel Sanz Santos

¿Componentes estructurales de un programa? Clases y objetos

Las tendencias más actuales en programación es el diseño de objetos a los cuales se les asigna una tarea especifica dentro del programa y que son reutilizables por diferentes programas.

Las clases son los tipos de los objetos, es decir al definir una clase lo que estamos establecimiento es un tipo y cuando se crean estancias de este tipo lo que desarrollamos son objetos.

En la definición de la clase establecemos las variables miembro (variables generales y publicas) y las definiciones de funciones, también denominadas métodos miembros, que constituirían las funciones del objeto El archivo objeto lo que contendrá será el código de la funciones definidas en la clase.

```
Clase gato {
    métodos:
        ronroneo();
        comer();
        brincar();
        (Variables)
        int edad;
        long peso;
}

....
(constructor)
gato::ronroneo():
    (inicialización)
    {
        hacer_ruido (mientras_caricias == 0){
            ruido=rooon
        }
    }
....
```

Miguel Angel Sanz Santos

Bibliografía

- Jesse Liberty (2000) “C++ para principiantes” Edit Pearson Educación 448 pg, México.
- Microsoft “Ayuda de Visual Basic 6” Microsoft.
- Historia de la programación:
 - <http://leo.worldonline.es/rodolfov/>
 - <http://galeon.hispavista.com/ultimopunto/>
 - <http://www.cs.buap.mx/%7Ejcom/>