

Programas informáticos, lenguajes de programación y conceptos del desarrollo del software

Ciclo: Desarrollo de aplicaciones multiplataforma

Módulo: Entornos de desarrollo

Cristian González Delgado
Profesor/a del Área de Informática

M05: ENTORNOS DE DESARROLLO

UF1. Desarrollo de software (2 clases)

- 1. Reconocer los elementos y las fases que intervienen en el desarrollo de un programa.
- 2. Familiarizarse con un entorno de desarrollo integrado.

UF2. Optimización de software (3 clases)

- 1. Diseñar pruebas para verificar el correcto funcionamiento de un programa.
- 2. Documentar código y identificar los principales puntos en la optimización de código.

UF3. Introducción al diseño orientado a objetos (3 clases)

- 5. Diagramas de clases.
- 6. Diagramas de comportamientos.

Definición de programa informático

```

function(scope, element, attr, ngSwitchController) {
  scope.$watch(attr.ngSwitch, function() {
    selectedTranscludes = [];
    selectedElements = [];
    previousElements = [];
    selectedScopes = [];

    scope.$watchWatchExpr, function ngSwitchWatchAction(value) {
      var ii, ii2;
      for (ii = 0, ii2 = previousElements.length; ii < ii2; ++ii) {
        previousElements[ii].remove();
      }
      previousElements.length = 0;

      for (ii = 0, ii2 = selectedScopes.length; ii < ii2; ++ii) {
        var selected = selectedElements[ii];
        selectedScopes[ii].destroy();
        previousElements[ii] = selected;
        $animate.leave(selected, function() {
          previousElements.splice(ii, 1);
        });
      }

      selectedElements.length = 0;
      selectedScopes.length = 0;

      (selectedTranscludes = ngSwitchController.cases['!' + value]) ?
        scope.$eval(attr.change);
      each(selectedTranscludes, function(selectedTransclude) {
        var selectedScope = scope.$new();
        selectedScopes.push(selectedScope);
      });
    }
  });
}

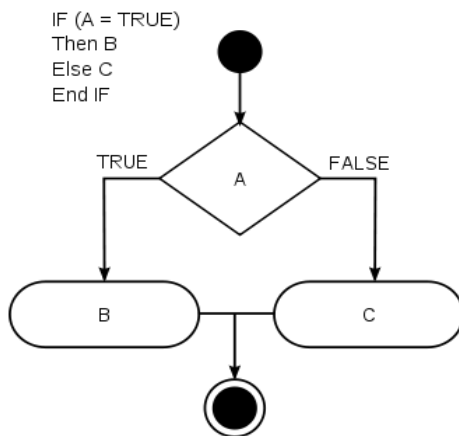
```

- Conjunto de **instrucciones**
- Se ejecutan de forma **secuencial**
- Objetivo: Realizar una o varias **tareas**

Es algo secuencial...

Estas instrucciones en realidad **se dividen subinstrucciones o microinstrucciones**

Se ejecutarán una a una y de forma secuencial en cada ciclo del procesador.




```

Assembler
00428F0F 00      add    %dl, -0x77(%ebp)
unit1.pas:39
00428F10 55      push   %ebp
00428F11 89e5    mov    %esp, %ebp
00428F13 83ec08  sub    $0x8, %esp
00428F16 8945f8  mov    %eax, -0x8(%ebp)
00428F19 8955fc  mov    %edx, -0x4(%ebp)
unit1.pas:40
00428F1C b8f8705700  mov    $0x5770f8, %eax
00428F21 e8ea6c1000  call   0x52fc10 <SHOWMESSAGE>
unit1.pas:41
00428F26 c9      leave
00428F27 c3      ret
00428F28 0000    add    %al, (%eax)
  
```

¿Qué datos procesa un programa informático?

Depende del tipo de programa

- **Un navegador web**: órdenes de un usuario (cliente) y del servidor.
- **Un videojuego**: la ubicación de enemigos y jugadores así como las físicas del juego, los impactos, la puntuación, etcétera,
- **Un programa ofimático**: procesa datos de texto datos numéricos, imágenes.

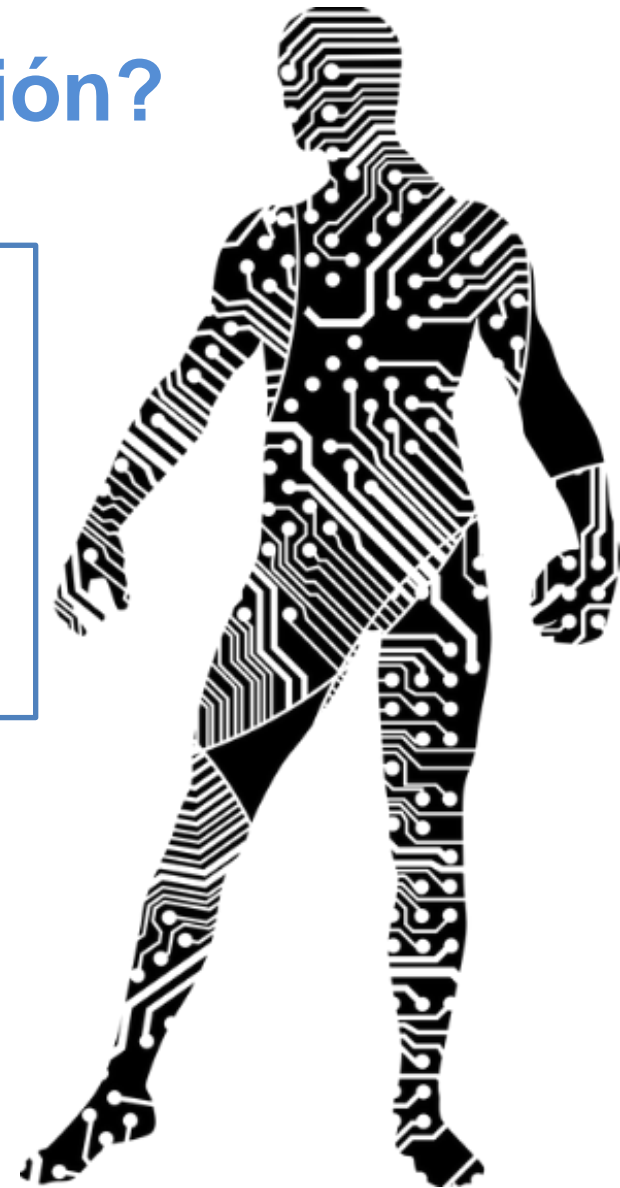
Conceptos de los lenguajes de programación

¿Qué es un Lenguaje de Programación?

- Conjunto de:
 - **Instrucciones**
 - **Operadores**
 - **Reglas de sintaxis**
 - **Reglas semánticas**

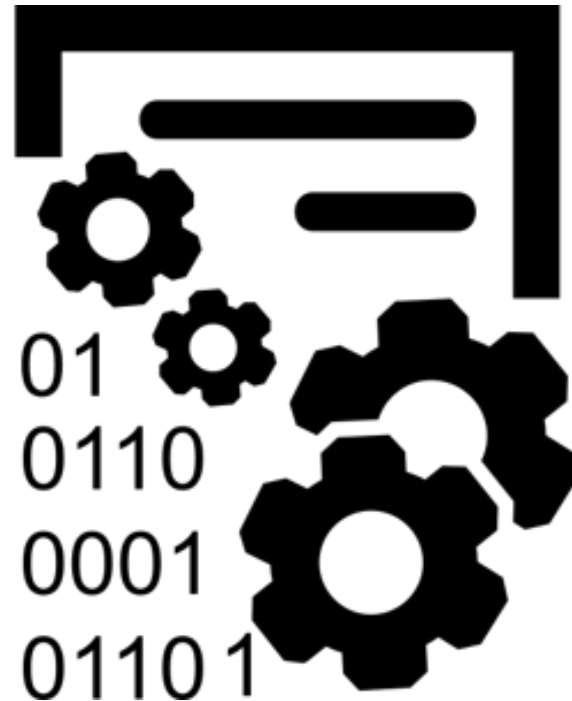
Pensado para que el programador pueda comunicarse con el software y el hardware

Programar es más fácil al conseguir un lenguaje más cercano al lenguaje humano



El procesador tan sólo es capaz de interpretar
instrucciones a muy bajo nivel

es decir con ceros y unos

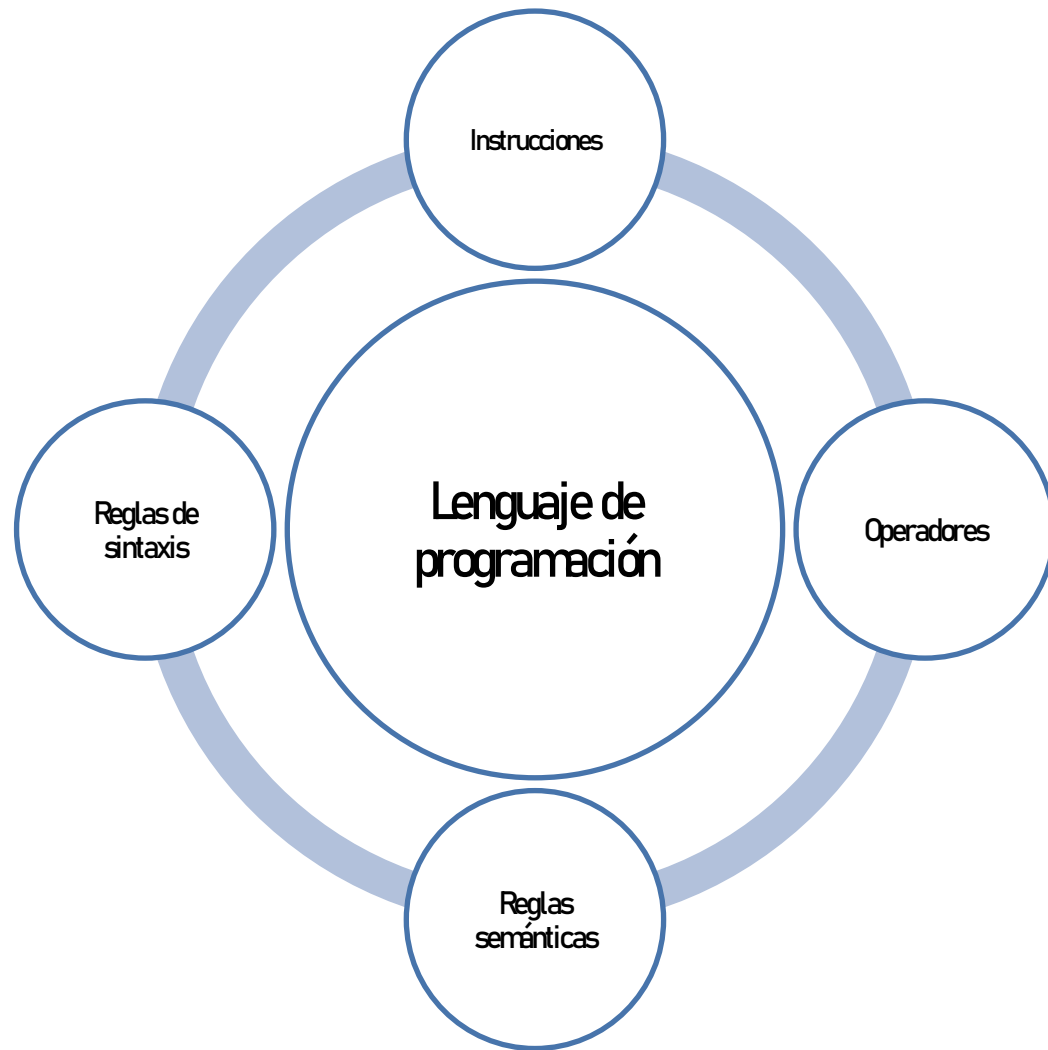


Debido a esta dificultad, aparecen los **lenguajes de programación**.

- Objetivo de **facilitar** la tarea a los programadores
- Permite escribir programas utilizando un **mayor nivel de abstracción** en el código.



Definición de lenguaje de programación



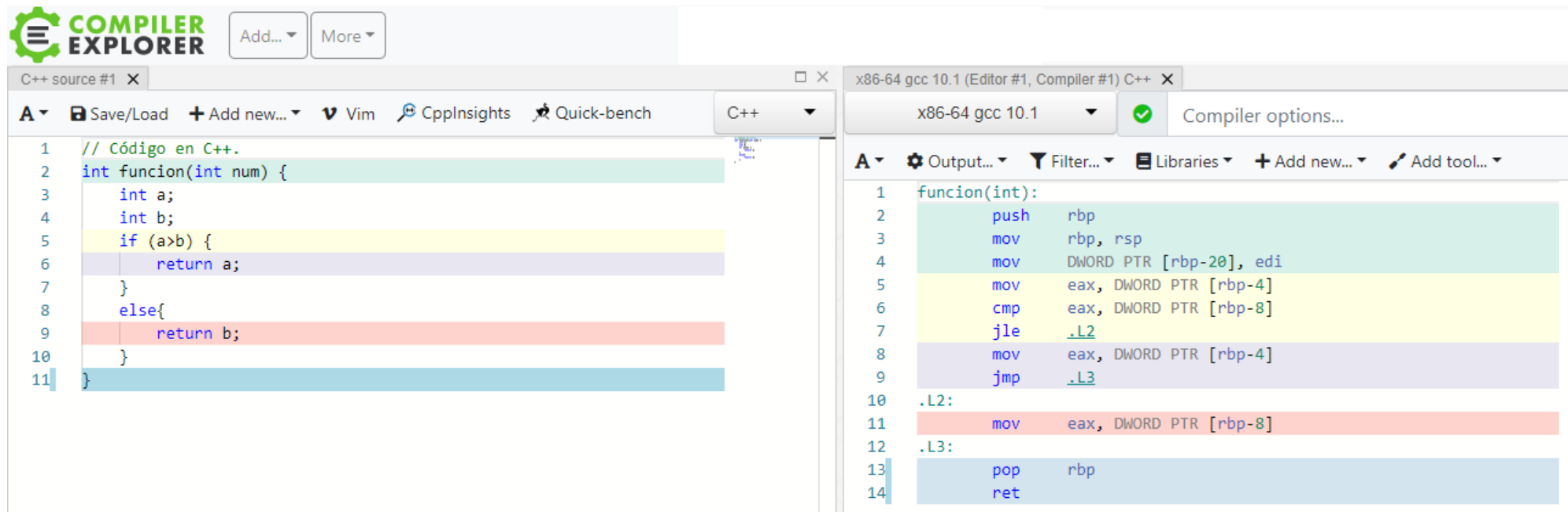
**Conjunto de instrucciones +
Operadores +
Reglas de sintaxis y semánticas**

Se ponen a disposición de los programadores para que se comuniquen con el hardware y el software.

- Tendencia a usar **lenguajes de alto nivel**
- Son los más **cercanos al lenguaje humano**.



Hay programadores que todavía desarrollan en **lenguaje assembler**.
Para desarrollar los **drivers** de ciertos periféricos.



The image shows a side-by-side comparison of C++ source code and its compiled assembly code. The left pane, titled 'C++ source #1', shows a simple C++ function 'funcion' that takes an integer 'num' and returns either 'a' or 'b' based on a comparison. The right pane, titled 'x86-64 gcc 10.1 (Editor #1, Compiler #1) C++', shows the corresponding assembly code generated by the compiler. The assembly code includes stack frame setup, variable storage, comparison, conditional jumps, and return instructions.

```

1 // Código en C++.
2 int funcion(int num) {
3     int a;
4     int b;
5     if (a>b) {
6         return a;
7     }
8     else{
9         return b;
10    }
11 }

```

```

1 funcion(int):
2     push    rbp
3     mov     rbp, rsp
4     mov     DWORD PTR [rbp-20], edi
5     mov     eax, DWORD PTR [rbp-4]
6     cmp     eax, DWORD PTR [rbp-8]
7     jle     .L2
8     mov     eax, DWORD PTR [rbp-4]
9     jmp     .L3
10 .L2:
11     mov     eax, DWORD PTR [rbp-8]
12 .L3:
13     pop     rbp
14     ret

```

Ocupan muy poco

Óptimos para estos pequeños programas

que comunican el dispositivo con nuestro ordenador.



Características de los lenguajes de programación más usados

La programación ha ido evolucionando

Con cada evolución se han ido creando más lenguajes de programación

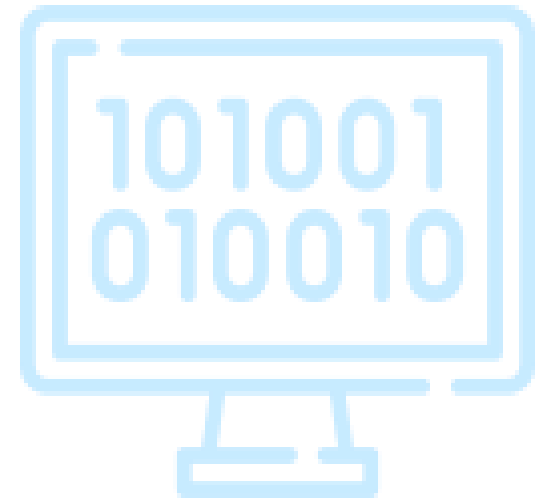
Cada vez más eficientes y fáciles de hacer servir.

Primera generación – El lenguaje máquina

Es el lenguaje que entiende el ordenador directamente, estamos hablando a **nivel de procesador**.

Sólo hay uno y es el **código máquina**.

Se obtienen programas **muy eficientes**.



Lenguajes de segunda generación – Lenguaje ensamblador

Programas muy optimizados que permiten aprovechar al máximo el hardware.

Estos lenguajes de programación **dependen del hardware** en el que se ejecuta

No se pueden trasladar a otro de forma fácil.

Requiere conocer a fondo la **arquitectura del sistema y del procesador.**

Assembler

El primer lenguaje de programación qué ha hecho servir **códigos memotécnicos**.

- Instrucciones muy **básicas**.
- Depende directamente del **procesador**
- Se utiliza para programar controladores (**drivers**) o aplicaciones que requieran un uso muy eficiente de la velocidad y la memoria.

```
; Ejemplo sencillo tipo Hello World
; Escribe un string en la salida

        JMP start
hello: DB ";Bienvenidos a Entornos de desarrollo!" ; Variable
        DB 0      ; String terminator

start:
        MOV C, hello      ; Point to var
        MOV D, 232        ; Point to output
        CALL print
        HLT                ; Stop execution

print:                                     ; print(C:*from, D:*to)
        PUSH A
        PUSH B
        MOV B, 0

.loop:
        MOV A, [C]         ; Get char from var
        MOV [D], A         ; Write to output
        INC C
        INC D
        CMP B, [C]         ; Check if end
        JNZ .loop          ; jump if not

        POP B
        POP A
        RET
```

Lenguajes de tercera generación. Los lenguajes de alto nivel

- **Fáciles de entender.**
- Permite expresar **flujos de control** de una forma bastante intuitiva.
- Para desarrollar **grandes aplicaciones**.
- Son lenguajes de programación **independientes de la máquina** en la que se van a ejecutar.
- **Fáciles de aprender** porque están formados por elementos del lenguaje natural, normalmente palabras en inglés.
- En contrapartida **ejecutar un programa escrito en alto nivel será más lento** que haberlo hecho en un lenguaje de bajo nivel

```

1  # Ejemplo en Python
2  x = 1
3  if x == 1:
4      print("x vale 1.")

```


Lenguajes de cuarta generación o de propósito específico

- Permiten desarrollar **aplicaciones sofisticadas** en poco tiempo.
- Estos lenguajes permiten muchas **acciones que antes se tenían que hacer manualmente**.
 - Por ejemplo una query SELECT en una línea.
- Estos lenguajes de programación están **orientados básicamente al manejo de base de datos**

SQL Statement:

```
SELECT NombreCliente,Ciudad FROM Clientes;
```

Lenguajes de quinta generación

Son lenguajes específicos para tratar problemas relacionados con la **inteligencia artificial**.

```
% La sintaxis es factorial(N, F) -> Factorial de N es F (el resultado se guarda en F)
factorial(0, 1).
factorial(1, 1).
factorial(N, F) :- N>0, N1 is N - 1, factorial(N1, F1), F is N * F1.

%el factorial se llama recursivamente dejando el resultado en F
```

Clasificación de los lenguajes de programación – Según el nivel de abstracción

El nivel de abstracción de un lenguaje implica **lo alejado que está del código máquina**.

Cuando más parecido sea a nuestro lenguaje y menos al código máquina mayor será nivel del lenguaje.

Clasificación de los lenguajes de programación - Según el nivel de abstracción

- **Bajo nivel** - Sólo hay uno el código máquina ceros y unos.
- **Medio nivel** - El lenguaje ensamblador que hace servir instrucciones sencillas para trabajar con datos simples y posiciones de memoria.
- **Alto nivel** - Todos los demás lenguajes de programación son los que son más cercanos a nuestro lenguaje

Clasificación de los lenguajes de programación

Según la forma de ejecución

• Compilados

Son los lenguajes que **deben ser compilados** antes de poder ejecutarse.

La compilación:

proceso que consigue que el lenguaje de programación **baje de nivel** hasta el código máquina y sea capaz de ejecutarse.

Clasificación de los lenguajes de programación

Según la forma de ejecución

- **Interpretados**

Estos lenguajes se ejecutan **línea a línea**.

No necesitamos compilar el programa completo para ejecutarlo.

El código interpretado no lo ejecuta directamente el sistema operativo, sino que lo hace un intérprete.

El sistema tiene su propio intérprete.

Clasificación de los lenguajes de programación

Según el paradigma de programación

El paradigma de programación de un lenguaje de programación se basa en:

- El **método** para llevar a cabo los cálculos en el proceso.
- La **forma** en la que deben estructurarse las tareas que debe realizar el programa.

Clasificación de los lenguajes de programación Según el paradigma de programación

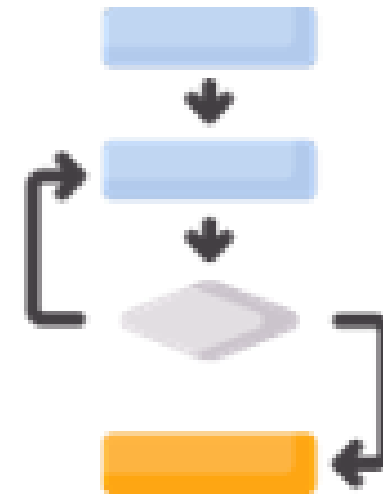
• Lenguajes imperativos o estructurados

Se basan en **sentencias imperativas**.

Operaciones una tras otra.

Estas operaciones van modificando los datos de la memoria.

Técnica de la programación estructurada, es decir de un programa grande y complejo se divide y se representa con secuencias, selecciones, iteraciones, etc.



Clasificación de los lenguajes de programación Según el paradigma de programación

•Orientado a objetos

Lenguajes que intentan **abstraer conceptos de la vida real y representarlos con objetos.**

Un objeto es una combinación de **datos y métodos** diseñados para interactuar entre objetos.



Clasificación de los lenguajes de programación

Según el paradigma de programación

• Funcional

Son lenguajes basados en **modelos matemáticos**.

Funcionan teniendo en cuenta en que **el resultado de un cálculo es la entrada del siguiente**, siempre de forma sucesiva hasta que se produce un resultado.

Normalmente estos lenguajes se usan en ámbitos de **investigación y aplicaciones matemáticas**.



Clasificación de los lenguajes de programación Según el paradigma de programación

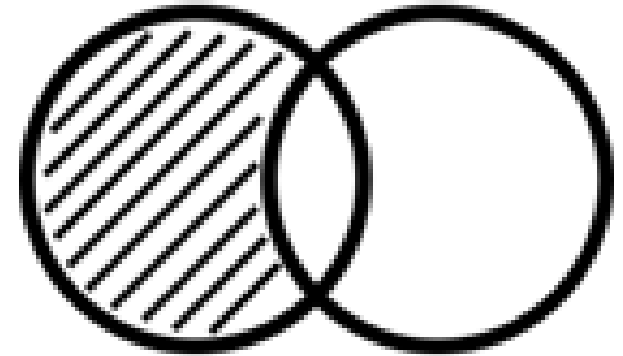
• Lógico

Son lenguajes cuya finalidad es la de acabar respondiendo preguntas planteadas al sistema para resolver problemas.

Necesita una **base de conocimientos** formadas por hechos (la información)

También necesitan **reglas lógicas** que permitan deducir las consecuencias de combinar los hechos.

Se hacen servir para **investigación**.



Clasificación de los lenguajes de programación

Según el paradigma de programación

- **Virtuales**

Son parecidos a los lenguajes compilados

No generan un ejecutable propiamente para la máquina en la que se está compilando.

Usan una **máquina virtual**.

Elementos que intervienen en el desarrollo de aplicaciones



Código fuente

Son las instrucciones que codifican los programadores.

El código fuente debe cumplir con las normas del lenguaje de programación en el que estemos codificando

Elementos que intervienen en el desarrollo de aplicaciones



Código objeto

El resultado de compilar el código fuente.

Al traducir el código fuente se obtiene este código objeto, eso si, previamente se tiene que validar a nivel sintáctico y semántico.

Elementos que intervienen en el desarrollo de aplicaciones



Código ejecutable

El que podemos ejecutar.

Es el que resulta de después de enlazar el código objeto con las librerías



Herramientas implicadas para la obtención de código ejecutable

Compilador:

Programa que traduce el código fuente
De alto nivel a lenguaje máquina

Máquina virtual:

Genera código para diferentes processadores

Bytecode

Bytecode

El bytecode es en realidad un código máquina de bajo nivel.
Sólo que este se puede ejecutar en la máquina virtual de Java



¿Qué es la compilación?

La **traducción** de un programa escrito en lenguaje de programación en un código fuente a otro lenguaje de programación de código intermedio para que posteriormente pueda ejecutarse

Etapas del proceso de obtención de código ejecutable

Código fuente

Líneas de **texto** con los pasos que se deben seguir la para ejecutar un programa

Análisis lexicográfico

Valida palabra por palabra - produce una salida compuesta de tokens (componentes léxicos)

Análisis sintáctico semántico

Valida la gramática

Generador de código intermedio

Se transforma el código. Se consigue un **código intermedio** en un lenguaje de una máquina abstracta

Optimizador de código

Transformaciones de mejora de código, se obtiene un **código optimizado**

Generador de código

El compilador convierte un programa sintácticamente correcto en una serie de instrucciones a ser interpretadas por una máquina. Se obtiene un **código objeto**

Enlazador

Programa que a partir del código generado, las bibliotecas (recursos necesarios), quita aquellos recursos que no necesita, y enlaza el código objeto las bibliotecas y genera un fichero con **código ejecutable**

Fases de desarrollo de una aplicación

Análisis

Define los **requisitos del software**

Diseño

Determinar **como funcionará el software** de una forma global , sin entrar en detalles.

Codificación

Reducir el diseño a **código**.

Pruebas

Confirmar que la codificación ha sido exitosa y que el software no contiene errores, a la vez que se **comprueba que el software hace lo que debe hacer**.

Documentación

Crear la información que ilustre cómo manejar la aplicación. También la documentación técnica destinada a ser leída por otros desarrolladores.

Explotación

Se prepara el software para su **distribución**. Se implementa el software en el sistema elegido.

Mantenimiento

Se **arreglan errores** que suceden cuando el programa ya ha sido implementado, también se realizan ampliaciones o modificaciones.

Personal que interviene en el desarrollo de software

Analista de
sistemas

Realizar un **estudio del sistema** para dirigir el proyecto de manera que garantice las expectativas del cliente.

Diseñador de
software

En función del análisis, **diseña la solución** que hay que desarrollar

Analista
programador

Con una visión de todo el proyecto más detallada **diseña soluciones para la codificación**, participa activamente en ella.

Programador


Codifica según la solución aportada por el analista programador. Crea el código fuente.

Arquitecto de
software

Investiga acerca de las tecnologías y frameworks. Revisa que todo los procedimientos del proyecto se llevan a cabo de la mejor forma y con los recursos optimos.

Ideas clave

- Conceptos de los lenguajes de programación
- Lenguajes de programación
 - Diferentes clasificaciones
 - Por generaciones
 - Por la forma de ejecución
 - Por el paradigma de ejecución
- Elementos que intervienen en el desarrollo de aplicaciones
 - Elementos:
 - Código fuente
 - Código objeto
 - Ejecutable
 - Herramientas
 - Compilador
 - Máquina virtual
 - Etapas del proceso de obtención del código
 - La compilación
- Fases del desarrollo de una aplicación
- Personal que interviene en el desarrollo



No olvidéis hacer la evaluación
de la clase 1