UNIVERSIDAD TECNICA
FEDERICO SANTA MARIA

DEPARTAMENTO
DE INFORMÁTICA

LENGUAJES DE PROGRAMACIÓN

Profesor: José Luis Martí Lara
jmarti@inf.utfsm.cl

Unidad 1

Introducción a los Lenguajes de Programación


Programming
Clojure
Assembly
Scheme
PowerShell
CODING
AJAX
.NET
MATLAB
VimL
XML
Groovy
Delphi
WebDNA
Cuda
R
Cobol
OpenEdge ABL
Active Server Page
Visual Basic
Perl
Mathematica
Scala
Ada
Swift
Prolog
JavaScript
Forth
ReXx
Python
Pascal
PHP
Haskell
Dart
Bourne Shell
GO
HTML
Gambas
TCL
BASH
Objective-C
Java
Processing
SAS
Transact-SQL
C
Ruby
ActionScript
ARDUINO
C++
open/source
Erlang
VBScript
Scratch
Liberlab
Logo
C#
Synergy
SQL
Smalltalk
Abap
Elixir
Lisp
CoffeScript
Fortran
AvtiveVFP

¿Porqué estudiar Lenguajes de Programación?

- Incremento de la capacidad de expresar ideas
- Mejor base de conocimiento para elegir lenguajes apropiados
- Incremento de la habilidad de aprender nuevos lenguajes
- Mejor comprensión del significado de la implementación
- Mejor uso de lenguajes ya conocidos
- Mejor visión del avance de la informática y la computación

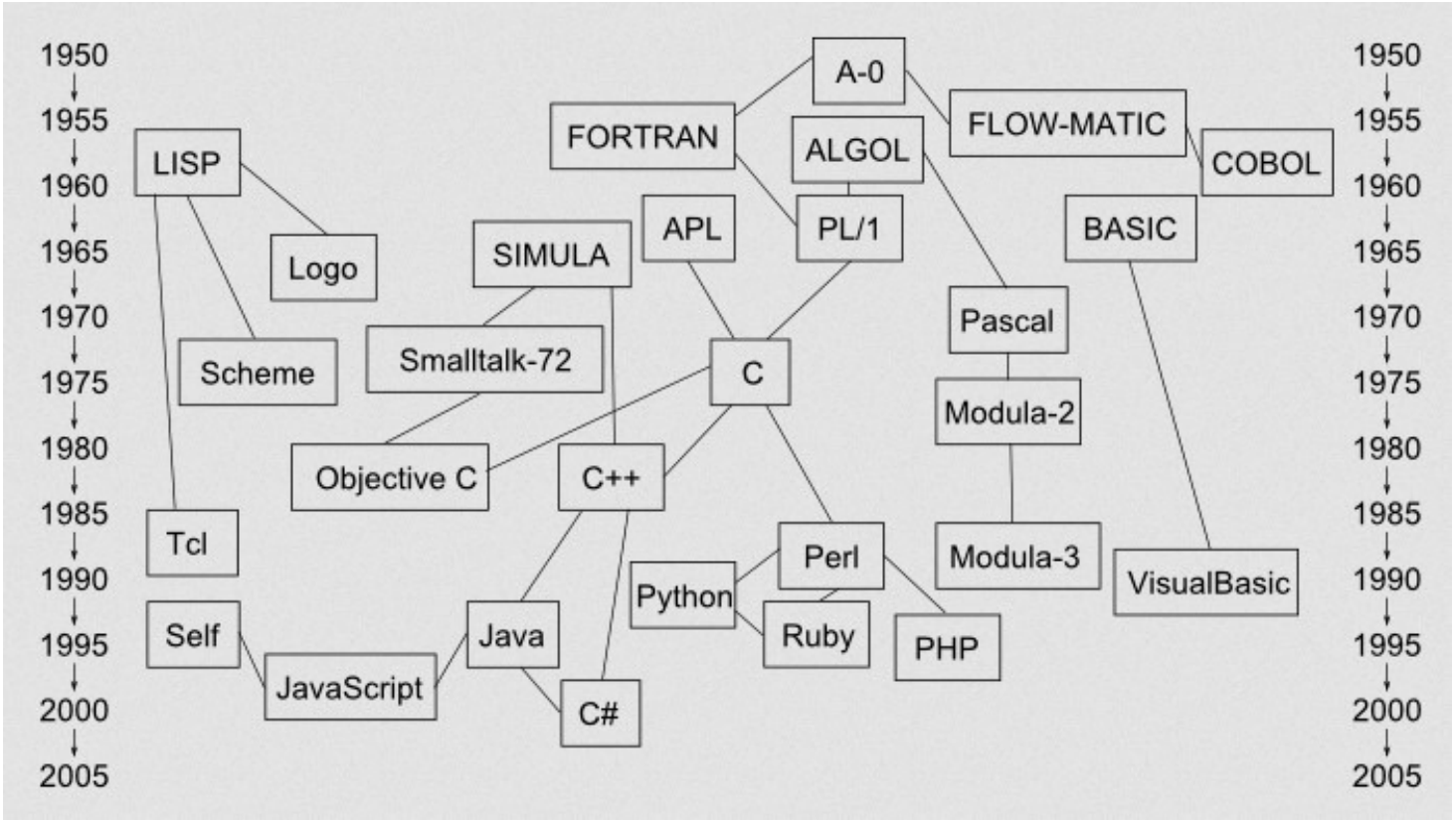
Dominios de Programación

- Aplicaciones de negocio
- Aplicaciones científicas y de ingeniería
- Programación de sistemas
- Ciencia de datos
- Inteligencia artificial
- Aplicaciones web

Evolución de los Lenguajes de Programación

- Lenguajes de Máquina: primeros programas; Babbage (1837), Turing (1936), Zuse (1941), ENIAC (1946)
- Lenguajes de Ensamblaje: código simbólico y herramientas de software; IBM (1954)
- Primeros Lenguajes de Alto Nivel: independiente de máquina; FORTRAN (1957), LISP (1958), COBOL (1959)
- Lenguajes Estructurados: ALGOL (1960), ALGOL 68, PASCAL (1970), C (1972) y ADA (1979)
- Lenguajes Orientados a Objeto: Simula (1967), Smalltalk (1980), C++ (1983), Eiffel (1986), Java (1995)
- Lenguajes de Scripting: JCL (1964), RUNCOM (1964), SH (1971); GREP (1973), AWK (1977); TCL (1988); PERL (1987), Python (1991); JavaScript (1995), PHP (1995), Ruby (1995)

Genealogía de los Lenguajes de Programación



Paradigmas de Programación

- Imperativo: Basado en Máquina de von Neumann. Ejecución secuencial, variables de memoria, asignación y E/S. Típicamente procedural. Dominan principalmente por mejor desempeño. Ejemplos: Fortran, Algol, Pascal y C.
- Funcional: Basado en cálculo Lambda. Usa funciones y recursión. Ejemplos: LISP, Scheme, Haskell.
- Declarativo: Se declara lo que se quiere hacer, no cómo. Es más abstracto al no especificar un algoritmo. Ejemplos: SQL y PROLOG.
  - Lógico (subcategoría): Basado en cálculo de predicados (lógica simbólica). Está fundamentalmente basado en reglas. Ejemplo: PROLOG.
- Orientado a Objetos: Conjunto de objetos (piezas) que interactúan controladamente intercambiando mensajes. Normalmente extienden paradigma imperativo. Ejemplos: Smalltalk, C++ y Java.

Ejemplo: Máximo común denominado (MCD)

```
int mcd(int a, int b) {    // C
   while (a != b) {
      if (a > b) a = a - b;
      else b = b - a;
   }
   return a;
}
```

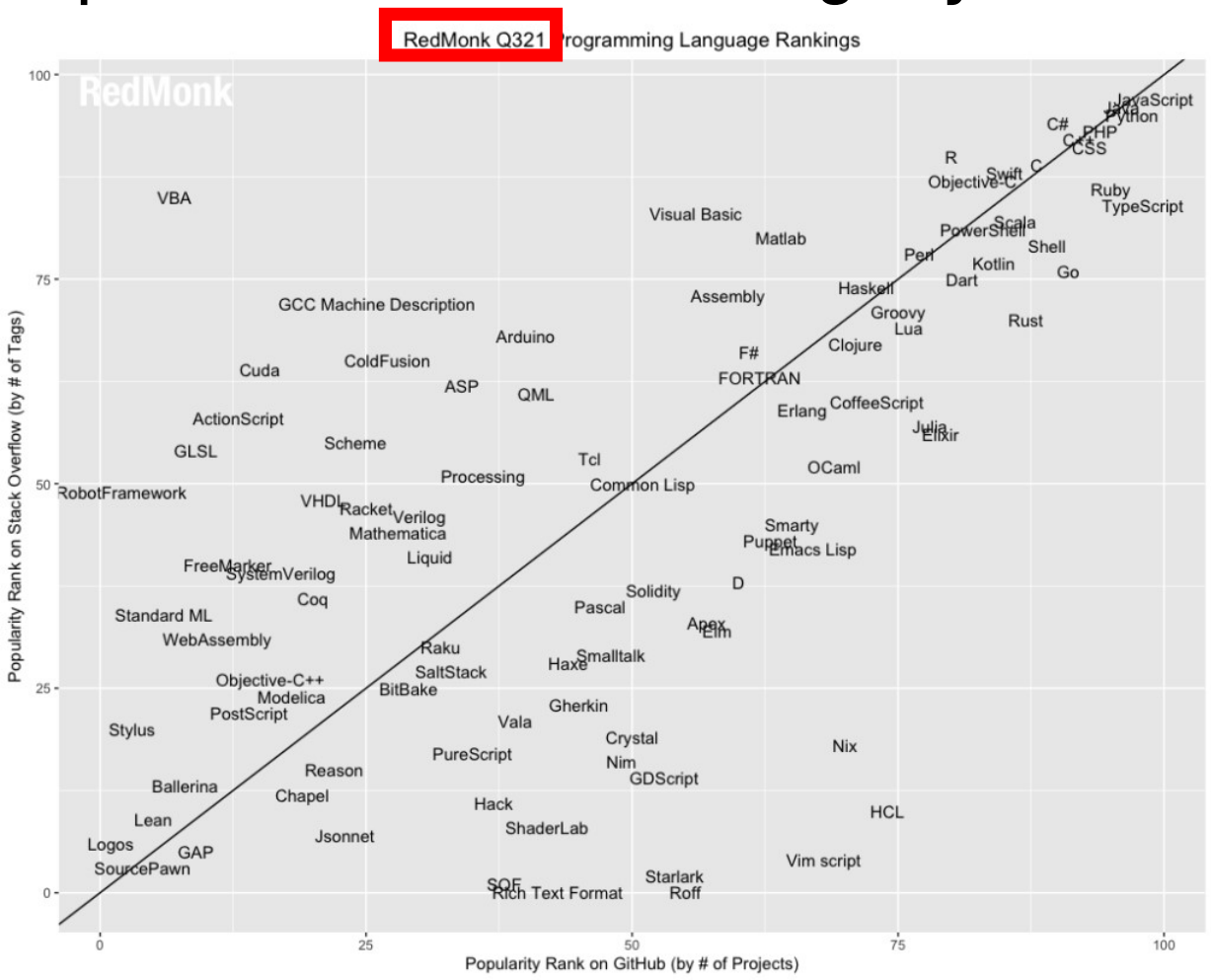
```
(define mcd            ; Scheme
 (lambda (a b)
  (cond ((= a b) a)
        ((> a b) (mcd (- a b) b))
        (else (mcd (- b a) a)))))
```

```
mcd(A,B,G) :- A = B, G = A.          %  Prolog
mcd(A,B,G) :- A > B, C is A-B, mcd(C,B,G).
mcd(A,B,G) :- B > A, C is B-A, mcd(C,A,G).
```

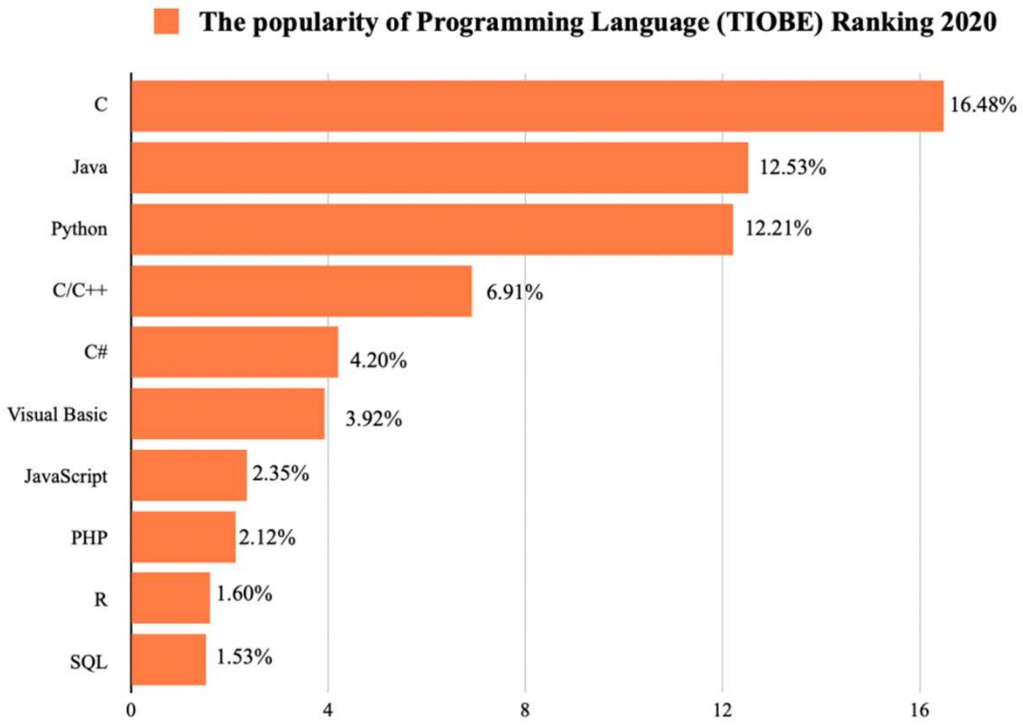
Otros Modelos de Programación

- Programación basada en Eventos: flujo del control está determinado por eventos que procesa el manejador de eventos. Ejemplos: Interfaces gráficas, manejo de interrupciones, sistema de sensores.
- Programación Concurrente: Conjunto de procesos cooperativos que se pueden ejecutar en paralelo. Se requiere sincronización en el acceso a recursos compartidos. Ejemplos: Sistemas operativos, sistemas distribuidos.
- Programación Visual: Permite crear programa manipulando objetos gráficos. Normalmente se integra con otros lenguajes. Ejemplos: Ingeniería de software, Kodu (juegos), LabVIEW (ingeniería).
  - No se debe confundir con un ambiente de programación visual (ej.: Visual Studio)

Popularidad de los Lenguajes de Programación



1 JavaScript
2 Java
3 Python
4 C#
5 PHP
6 C++
7 CSS
8 C
9 R
10 Swift
11 Objective-C
12 Ruby
13 TypeScript
14 Scala



https://darly.solutions/the-most-popular-programming-languages-in-2021/.

Worldwide, Apr 2021 compared to a year ago:

Rank	Change	Language	Share	Trend
1		Python	29.5 %	-1.0 %
2		Java	17.51 %	-0.6 %
3		JavaScript	8.19 %	+0.2 %
4		C#	7.05 %	-0.2 %
5	↑	C/C++	6.73 %	+1.0 %
6	↓	PHP	6.23 %	+0.0 %
7		R	3.86 %	+0.0 %
8		Objective-C	2.77 %	+0.3 %
9	↑	TypeScript	1.87 %	-0.0 %
10	↓	Swift	1.85 %	-0.3 %
11	↑	Kotlin	1.78 %	+0.3 %
12	↓	Matlab	1.77 %	-0.1 %
13	↑	Go	1.37 %	+0.1 %
14	↓	VBA	1.33 %	-0.0 %
15		Ruby	1.21 %	-0.1 %
16	↑↑	Rust	1.1 %	+0.4 %

https://pypl.github.io/PYPL.html

Abstracciones (1/2)

Datos

- Primitivos: Tipos de datos básicos y variables (ej.: enteros y reales, caracteres).
- Simples: Tipo de datos no estructurado, que puede ser primitivo o definido por el usuario en base a uno primitivo.
- Estructurados: permite agrupar/componer conjuntos de datos en una unidad (ej.: Arreglo, registro, archivo de texto). Define nuevos tipos de datos.

Abstracciones (2/2)

Control

- Sentencias: abstrae conjunto de instrucciones.
- Estructuras de control: secuenciación, condición y repetición (ej.: if, while e iterador).
- Abstracción procedural: permite invocar un procedimiento con un nombre y parámetros (ej.: procedimientos, subprogramas y funciones).
- Concurrencia: permite computación paralela (ej.: procesos, hebras y tareas). Se introducen también abstracciones de comunicación.

Tipos de Datos Abstractos

- Agrupa datos y operaciones relacionadas en una unidad (ej.: módulos y clases). Abstrae el tipo de dato con sus operaciones, de la implementación del tipo.

Modelos de Implantación

Compilación

- Se traduce a lenguaje de máquina para su posterior ejecución (interpretación directa por el procesador). Ejemplos: C, C++.

Interpretación

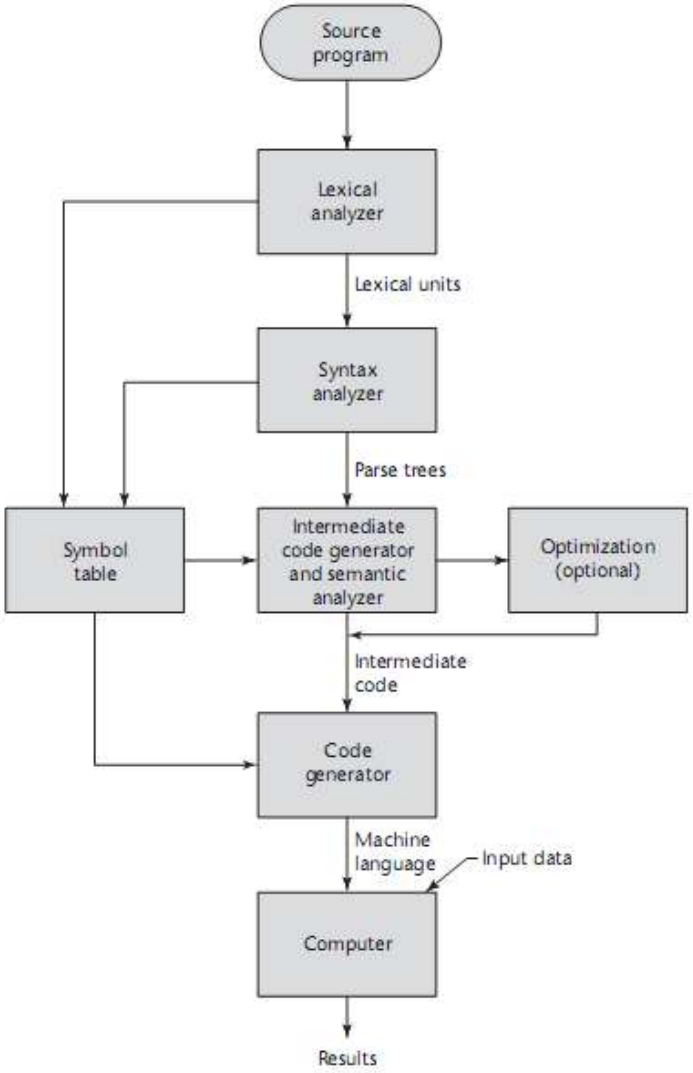
- Una máquina virtual interpreta directamente el código fuente durante la ejecución. Ejemplos: LISP, Python.

Esquema Híbrido

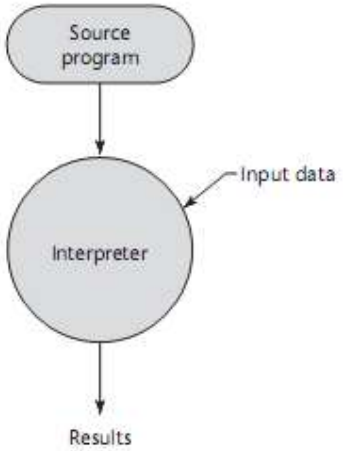
- Se compilan a un lenguaje intermedio, que luego es interpretado por una máquina virtual. Ejemplos: Java, C#.

Observación: No confundir preprocesamiento con híbrido.

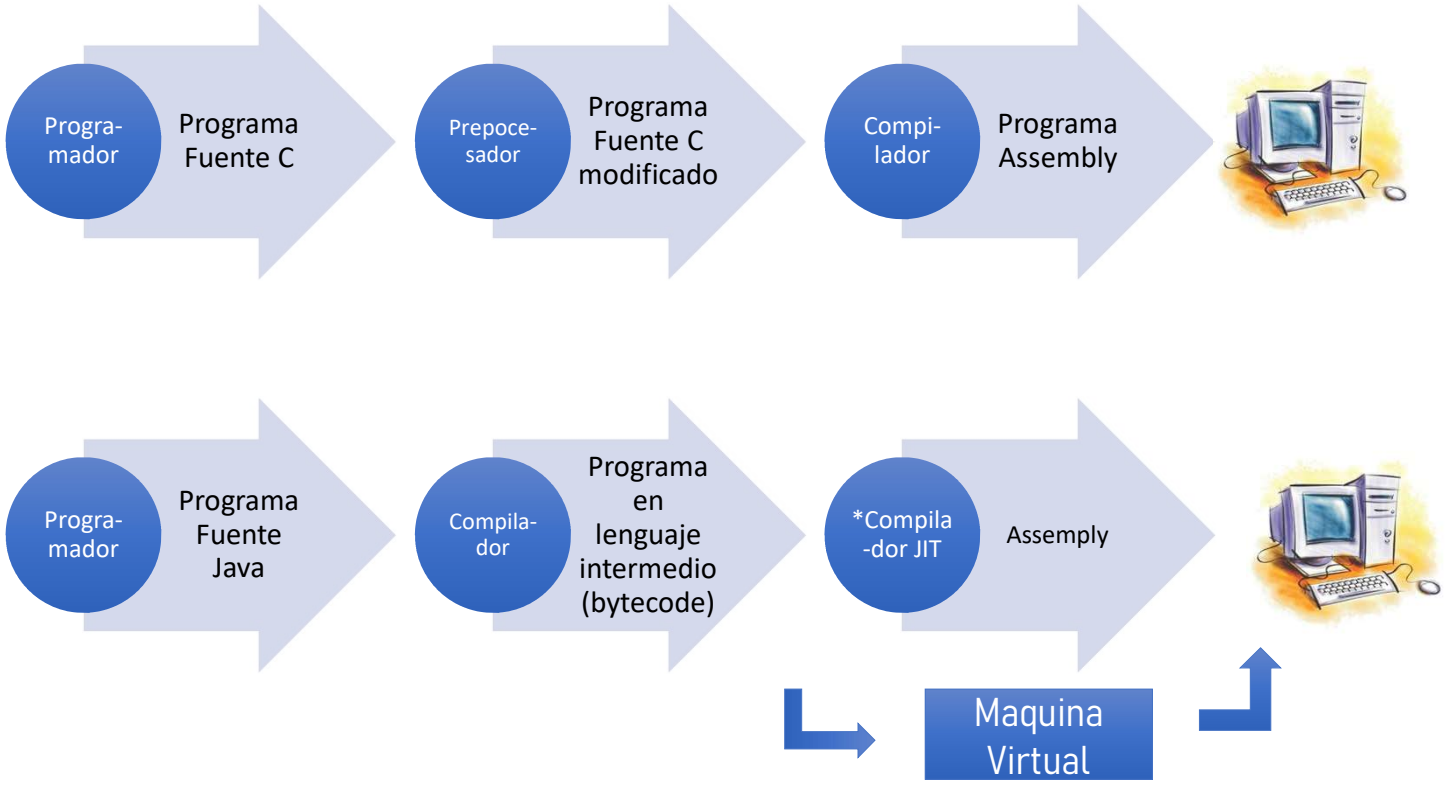
Compilación



Interpretación



Ejemplos de C y de Java



Programación “en grande”

- Modularización: Necesidad de descomponer los programas en unidades de desarrollo más pequeñas (piezas o módulos de software). Apoya el concepto de *ocultamiento de información*, que reduce carga cognitiva.
- Compilación Separada o Independiente: Módulos de un programa se pueden traducir apartes. Facilita mantención.
- Reutilización: Módulos se pueden reutilizar para diferentes programas (ej.: Bibliotecas, módulos y paquetes).
- Ambientes de Desarrollo: Se tienen ambientes de desarrollo con diferentes tipos de herramientas y facilidades para apoyar el proceso en todo el ciclo de vida del software.

Aspectos de Diseño

- Arquitectura: Máquina objetivo donde se ejecuta. Mayoría de computadores siguen basados en modelo de von Neumann, lo que a veces los lenguajes no calzan con su modelo.
- Estándares: Lenguajes populares tienden a estandarizarse, haciéndolos más portable. Se incorpora la estandarización de los tipos de datos primitivos (ej.: enteros, caracteres) y bibliotecas (ej.: STL). Hace más pesado el proceso de definición e innovación.
- Sistemas legados: Mantener compatibilidad hacia atrás. Permite mantener código legado (ej.: Cobol y C++). Hace más complejo el diseño de los lenguajes.

Criterios de Evaluación

Característica \ Criterio	Facilidad de Lectura	Facilidad de Escritura	Fiabilidad
Simplicidad	√	√	√
Ortogonalidad	√	√	√
Tipos de datos	√	√	√
Diseño de sintaxis	√	√	√
Soporte para abstracción		√	√
Expresividad		√	√
Prueba de tipos			√
Manejo de excepciones			√
Restricción de alias			√

Tendencias

- Lenguajes tienden a ser multiparadigmas y a especializarse en determinados tipos de problemas.
- Fuerte auge de lenguajes de programación para el área de ciencia de datos.
- Existe un gran movimiento en el desarrollo de lenguajes para programación de aplicaciones Web y móviles.
- Lenguajes basados en SQL siguen dominando el área de base de datos.