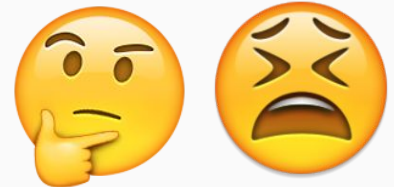


# Lenguajes de programación

# Qué es un lenguaje de programación?

Es un **lenguaje formal** computacional diseñado para comunicar **instrucciones** a una máquina, particularmente una computadora. Pueden utilizarse para crear **programas** que controlan el comportamiento de máquinas o para expresar **algoritmos**.

[https://en.wikipedia.org/wiki/Programming\\_language](https://en.wikipedia.org/wiki/Programming_language)



# Lenguaje formal

Combinaciones de símbolos que se rigen bajo un conjunto finito de reglas

- Letras y símbolos
  - a, b, c, d, e, /, <, |, &
- Palabras
  - while, for, if, int, double, do
- Sintaxis/Gramática
  - if something then  
    other  
end

# Instrucción

Una operación específica realizada por un agente, en este caso una computadora

- ADD (suma)
- JUMP (salto a otra parte del programa)
- COMPARE (comparación)
- MOVE (asignación)
- etc...

# Programa

Conjunto de instrucciones que controlan el comportamiento de una máquina

```
#include <stdio.h>

int main() {
    printf("Hello, world!");
    return 0;
}
```



A terminal window with a black background. The text "Hello World!" is displayed in white at the top. Below it is a white cursor line.

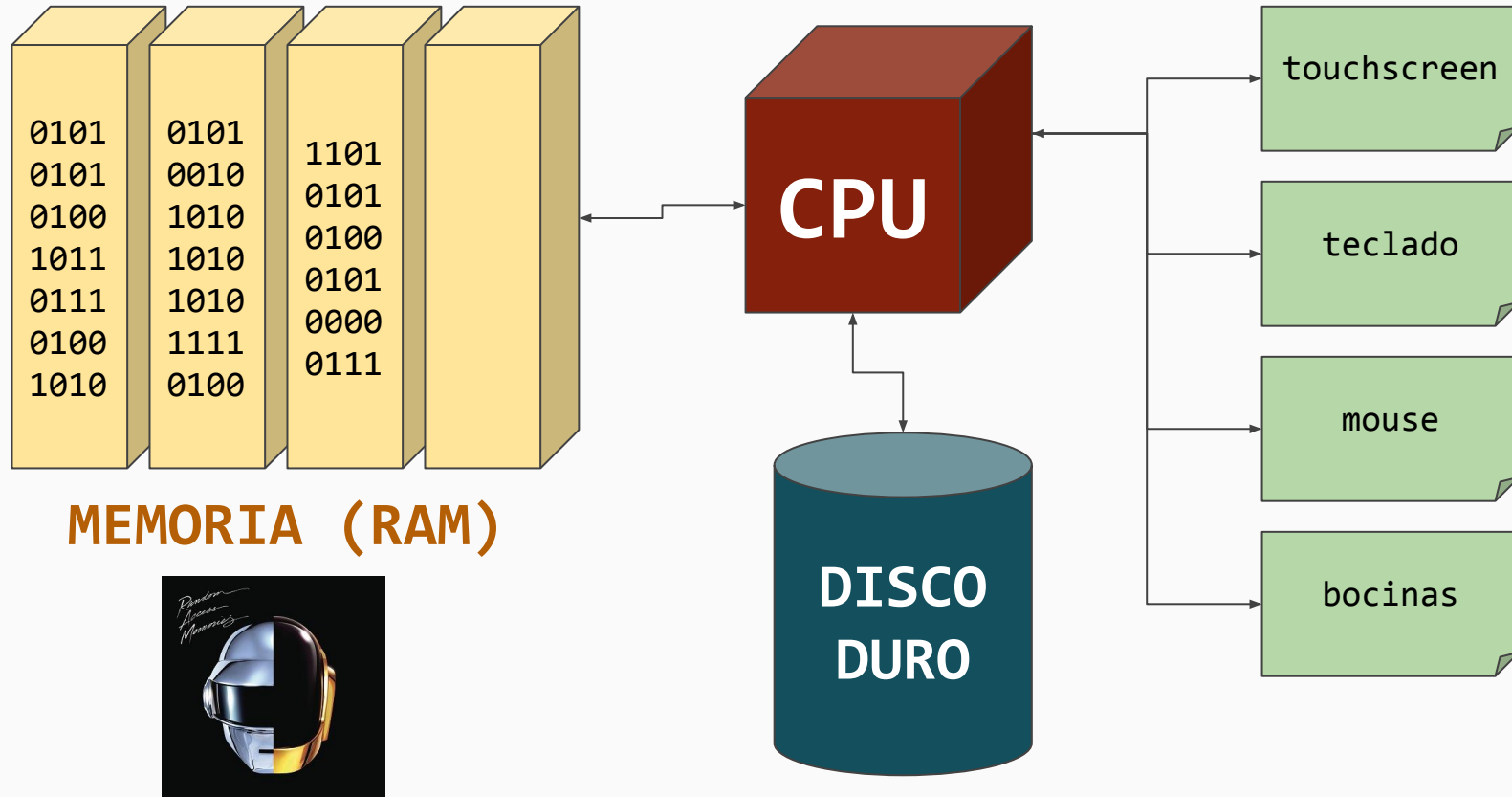
# Algoritmo

Serie de operaciones paso a paso para lograr un cálculo específico, procesamiento de datos, o tarea automatizada.

Ejemplos cotidianos:

- Dividir la cuenta al ir a cenar con amigos
- Saber quién es el más joven de mis amigos

# Repasando la arquitectura de una computadora... (diagrama muy simplificado)



# Ejecución

Para ejecutar un programa:

1. Se copian todas las instrucciones del programa desde el **Disco duro** hasta la **Memoria RAM**
2. El **CPU** lee y procesa cada instrucción en orden hasta terminar el programa, coordinando **entradas y salidas** (teclado, pantalla, etc.) según sea necesario.



# Instrucciones que entiende el CPU

Muchos tipos de procesadores (CPU), cada uno con una lista finita de instrucciones que entiende:

- i386, x86, x86\_64 (PCs, MacBook, etc.)
- ARM (iPhone, smartphones)
- PowerPC (iBooks viejas)
- etc...

Un procesador no entiende código en Ruby, C, C++, Java... cómo lo logra?

# Cómo entiende mi programa la computadora?

La computadora (el CPU) sólo entiende instrucciones específicas, no?

Si yo escribo mi código en un lenguaje de programación, qué debe suceder para ejecutar el programa?

# Compilación

# Compilador

Un programa que traduce código escrito en un lenguaje de programación (**código fuente**) a una forma binaria de bajo nivel (**código objeto**) para producir un **programa ejecutable**.

# Código objeto

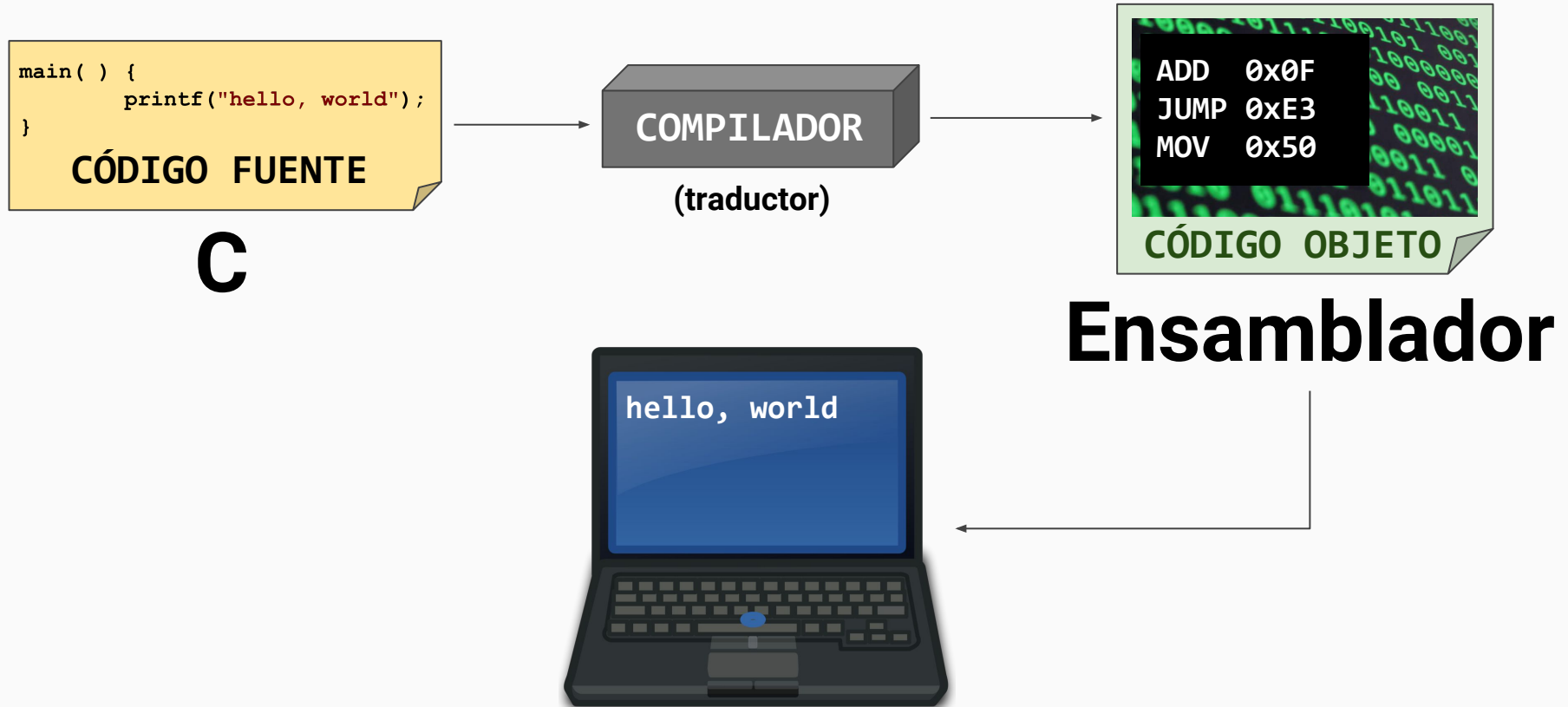
También se le conoce generalmente como **código ensamblador**

# Lenguaje ensamblador

Lenguaje de programación en el cual todas las instrucciones son **exactamente las mismas** que las que puede entender el procesador.

```
mov    edx,len          ;third argument: message length
mov    ecx,msg          ;second argument: pointer to message to write
mov    ebx,1            ;first argument: file handle (stdout)
mov    eax,4            ;system call number (sys_write)
int    0x80             ;call kernel
```

# Proceso de compilación



# Antes que existieran los lenguajes modernos...

[https://github.com/chrislgarry/Apollo-11/blob/master/Luminary099/ORBITAL\\_INTEGRATION.agc](https://github.com/chrislgarry/Apollo-11/blob/master/Luminary099/ORBITAL_INTEGRATION.agc)



# Ciclo de desarrollo de software

1. Escribir código (hack)
2. Compilar
3. Ejecutar
4. Revisar errores
5. Repetir

# Paradigmas de lenguajes de programación

# Paradigma de programación

Una forma de clasificar lenguajes de programación de acuerdo a su estilo de programación.

Diferente manejo de instrucciones, flujo de control, operaciones, etc.

# Los más comunes

- Estructurados
  - Procedurales
  - Orientados a objetos
- Declarativos
  - Funcionales

# Lenguajes estructurados

- Estatutos

- `int x = 1337;`

- Ciclos

- `while (i < 5) {  
 print "hola";  
 i++;  
}`

- Sub-rutinas

- `void suma(int a, int b) {  
 return (a + b);  
}`

## Estructurados >

# Lenguajes procedurales

Lenguajes estructurados que hacen uso exclusivo de sub-rutinas para organizar el flujo del código.

C, Pascal, Fortran, BASIC, etc...

Estructurados >

# Lenguajes orientados a objetos

Lenguajes estructurados que intentan describir todo el flujo del programa como objetos interactuando con otros objetos.

El paradigma más popular del momento.

**Ruby**, Java, C++, Swift, Objective-C, Python, PHP, etc...

Estructurados >

# Dinámicos vs estáticos

Los lenguajes estructurados pueden ser **dinámicos** o **estáticos**, dependiendo de cómo traten el manejo de **tipos de dato**.



# Estructurados > Dinámicos vs estáticos

## Estáticos

```
int x = 200;  
x = 5;  
x = 10;  
x = "hola"; // Error!
```

C, C++, Java, etc...

## Dinámicos

```
x = 200; // Sin declaración de tipo.  
x = 5;  
x = 10;  
x = "hola"; // Todo bien!
```

**Ruby**, Python, PHP...

Declarativos >

# Lenguajes funcionales

Lenguajes declarativos que buscan describir todo el programa a partir de funciones matemáticas.

No permiten el cambio permanente de estado de ningún aspecto del programa.

Comúnmente utilizados en ámbitos científicos por su poder de cálculo matemático.

Scheme, Erlang, Lisp, Haskell, etc...

# Historia de los lenguajes de programación

<http://rigaux.org/language-study/diagram.png>

# Repaso

# Preguntas de repaso

1. Qué es un lenguaje de programación?
2. Qué es un lenguaje formal?
3. Qué es un programa?
4. Qué es una instrucción?
5. Qué es un algoritmo?
6. Qué es un compilador?
7. A qué tipo de código traduce un compilador?
8. Cómo se carga y ejecuta un programa en la computadora?

# Preguntas de repaso

9. Cómo se carga y ejecuta un programa en la computadora?
- 10.Cuál es la diferencia entre la Memoria RAM y el Disco duro?
11. Qué es el lenguaje ensamblador? Para qué sirve?
12. Qué es un Paradigma de programación?
13. Qué distingue a los lenguajes procedurales?
14. Qué distingue a los lenguajes orientados a objetos?
15. Qué distingue a los lenguajes funcionales?
- 16.Cuál es la diferencia entre un lenguaje estático y dinámico?