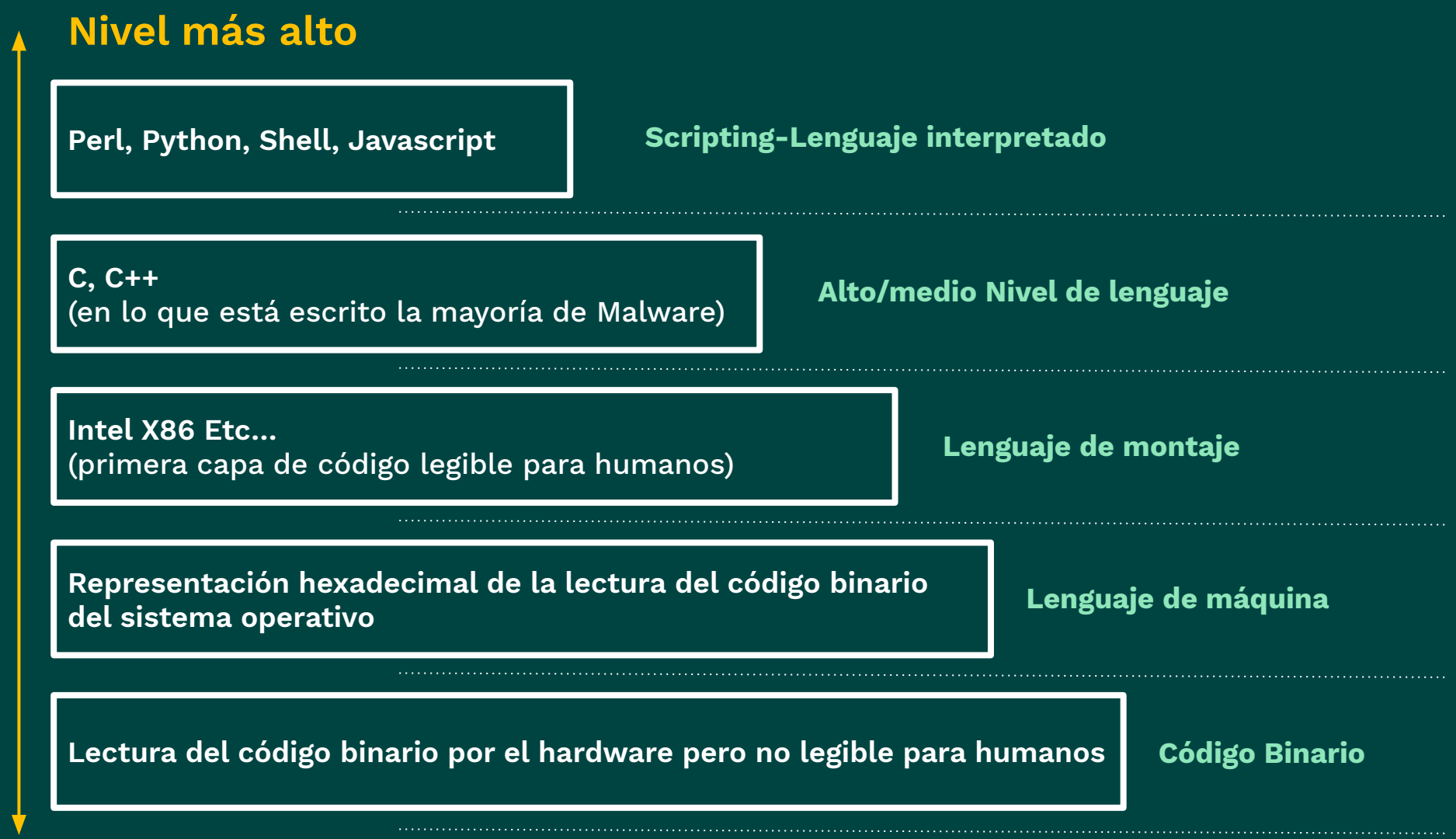
Curso de

Introducción a C

Ricardo Celis

¿Por qué **aprender C?**

C es un Lenguaje de Nivel Intermedio

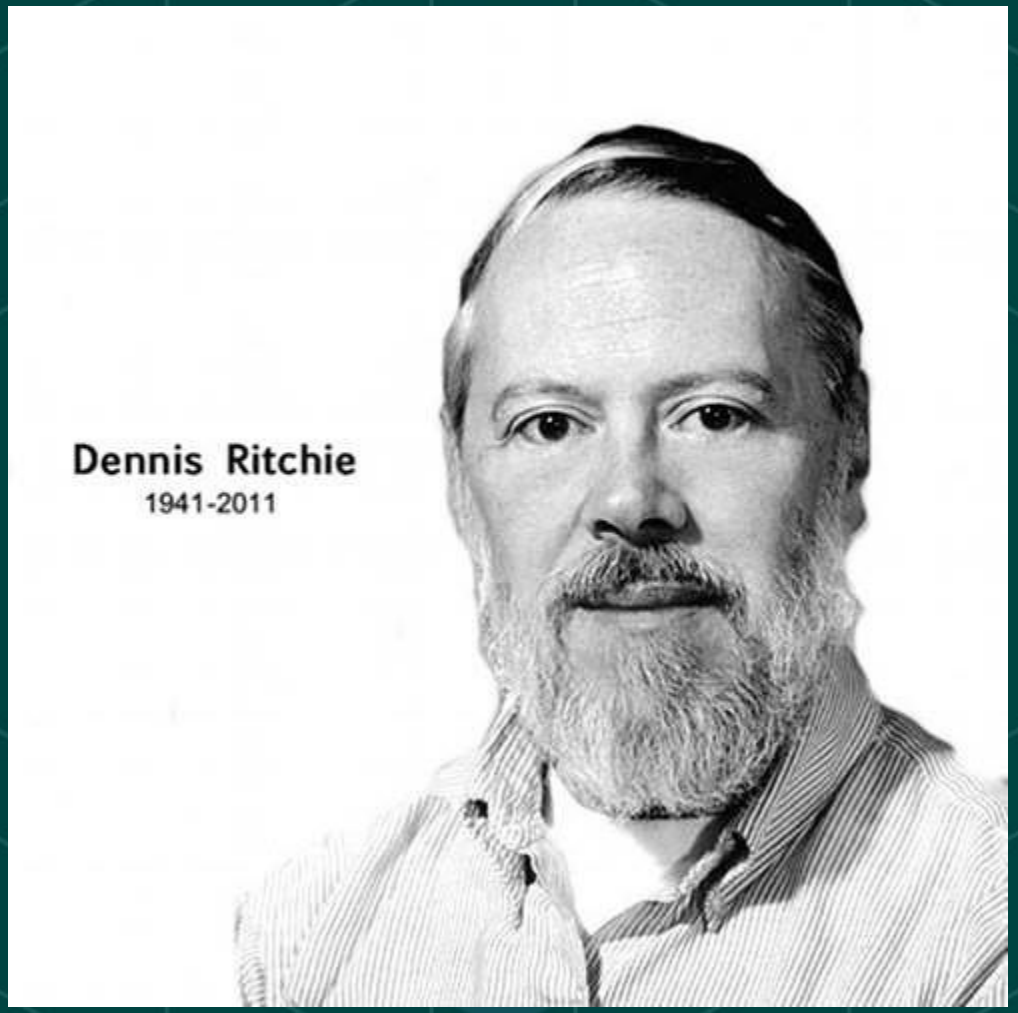

Nivel más alto
Perl, Python, Shell, Javascript
Scripting-Lenguaje interpretado
C, C++
(en lo que está escrito la mayoría de Malware)
Alto/medio Nivel de lenguaje
Intel X86 Etc...
(primera capa de código legible para humanos)
Lenguaje de montaje
Representación hexadecimal de la lectura del código binario del sistema operativo
Lenguaje de máquina
Lectura del código binario por el hardware pero no legible para humanos
Código Binario
Nivel más bajo - Código máquina


Nivel más alto
Alto
Bajo
Alto
Lectura humana
Velocidad de ejecución del programa
Nivel de lenguaje
Perl, Python, Shell, Javascript
C, C++
(en lo que está escrito la mayoría de Malware)
Intel X86 Etc...
(Primera capa de código legible para humanos)
Representación hexadecimal de la lectura del código binario del sistema operativo
Lectura del código binario por el hardware pero no legible para humanos
Bajo
Alto
Bajo
Nivel más bajo - Código máquina

Ayuda a **entender** los **fundamentos** de **teoría** de **computación**

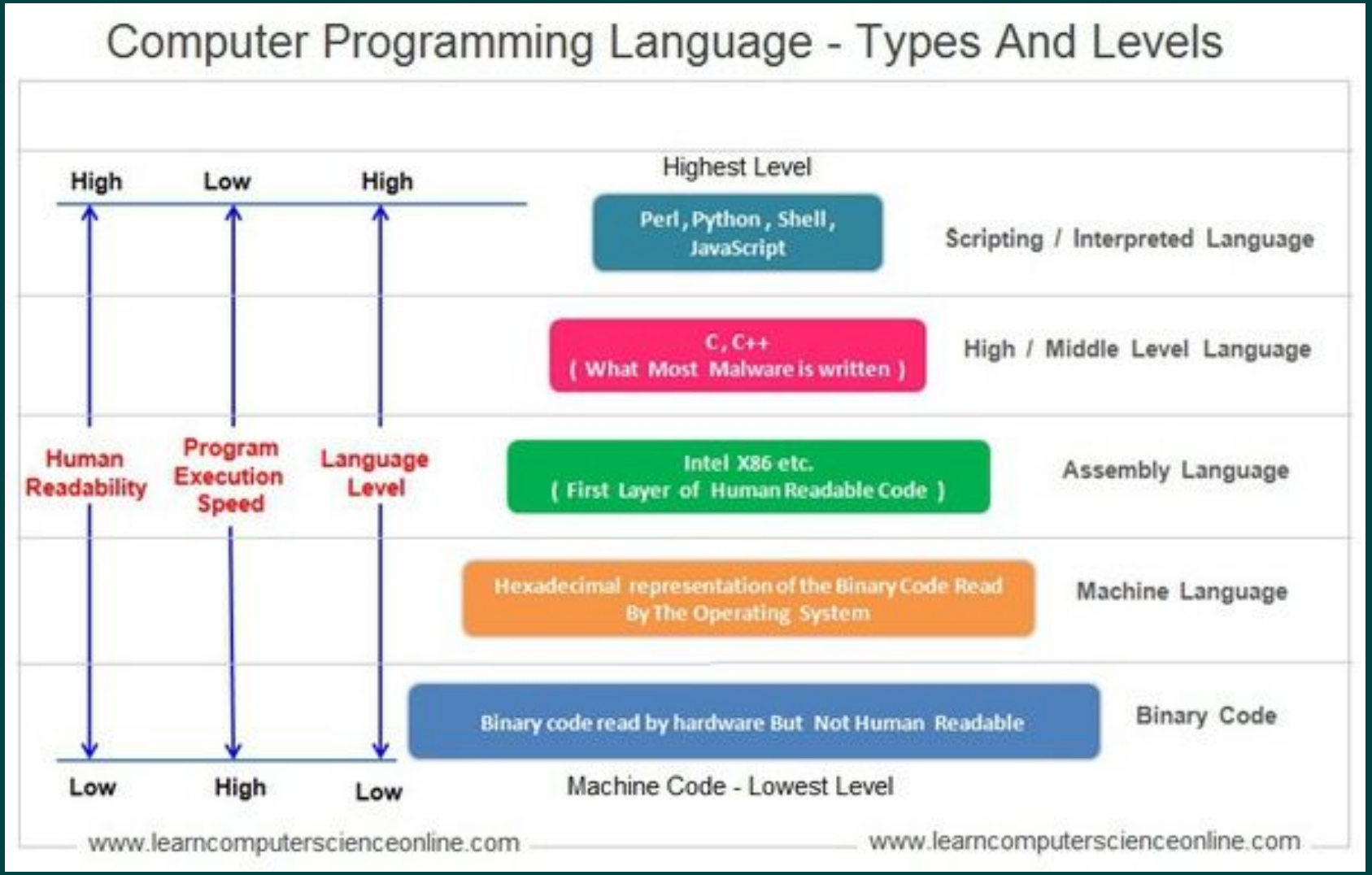
Tienes que **escribir mucho** desde **CERO.** **C** tiene **pocas librerías**

El más **utilizado** en **programación** de **circuitos embebidos**

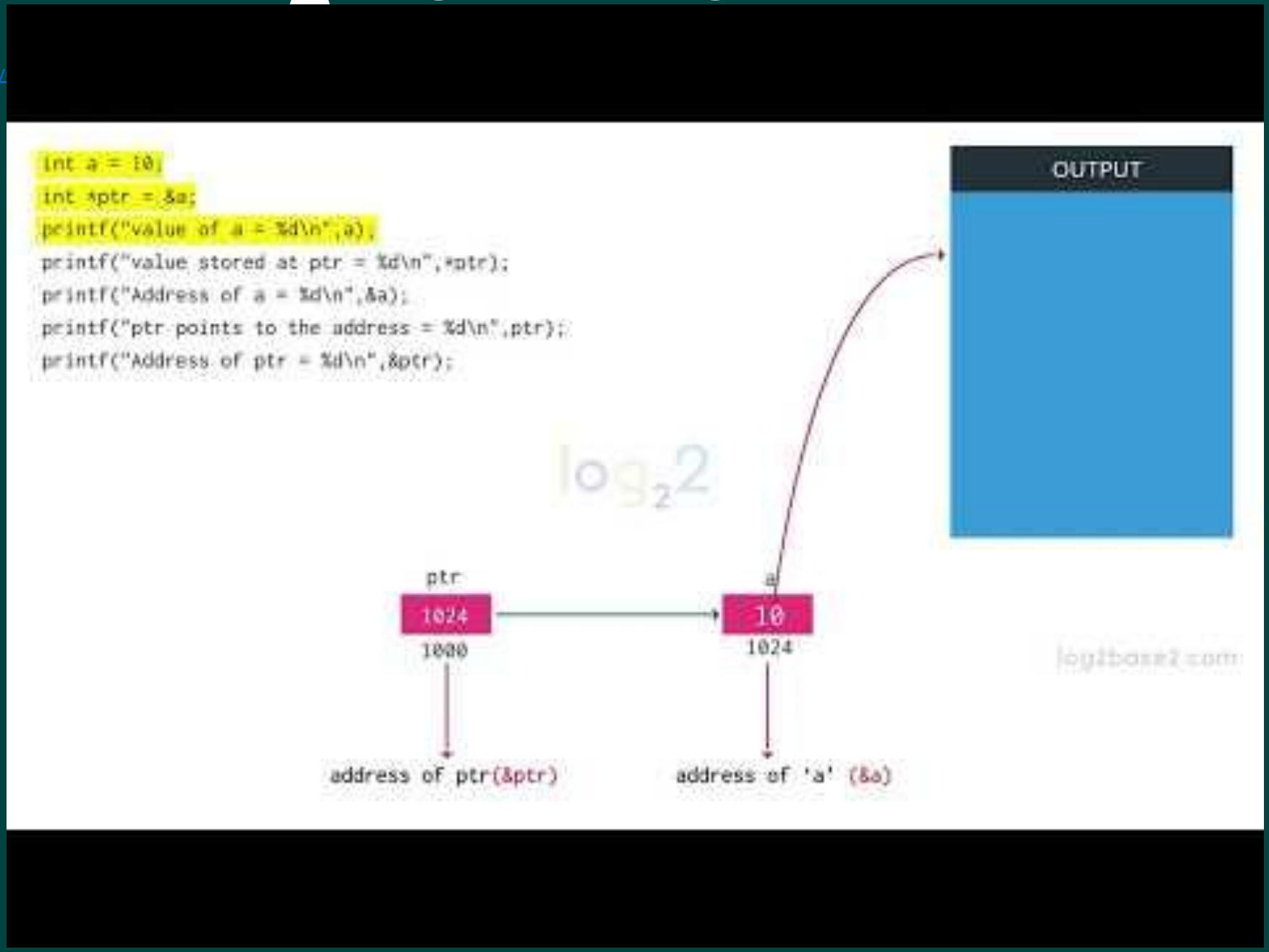

Dennis Ritchie
1941-2011

Muy rápido en **tiempo** de **ejecución**

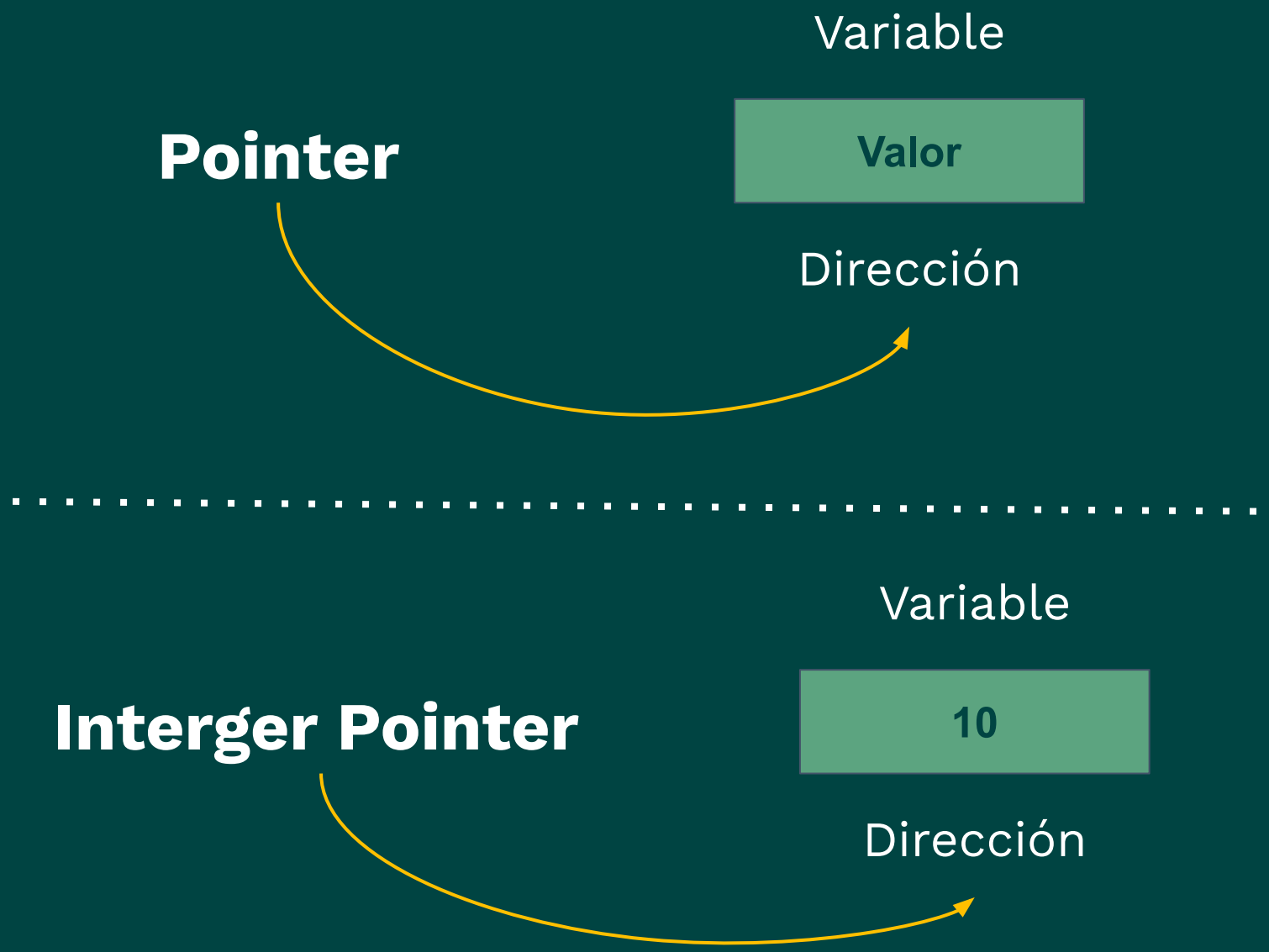
Animación


Computer Programming Language - Types And Levels
High
Low
High
Highest Level
Perl, Python, Shell, JavaScript
Scripting / Interpreted Language
C, C++
(What Most Malware is written)
High / Middle Level Language
Human Readability
Program Execution Speed
Language Level
Intel X86 etc.
(First Layer of Human Readable Code)
Assembly Language
Hexadecimal representation of the Binary Code Read By The Operating System
Machine Language
Binary code read by hardware But Not Human Readable
Binary Code
Low
High
Low
Machine Code - Lowest Level
www.learncomputerscienceonline.com
www.learncomputerscienceonline.com

https://www.y

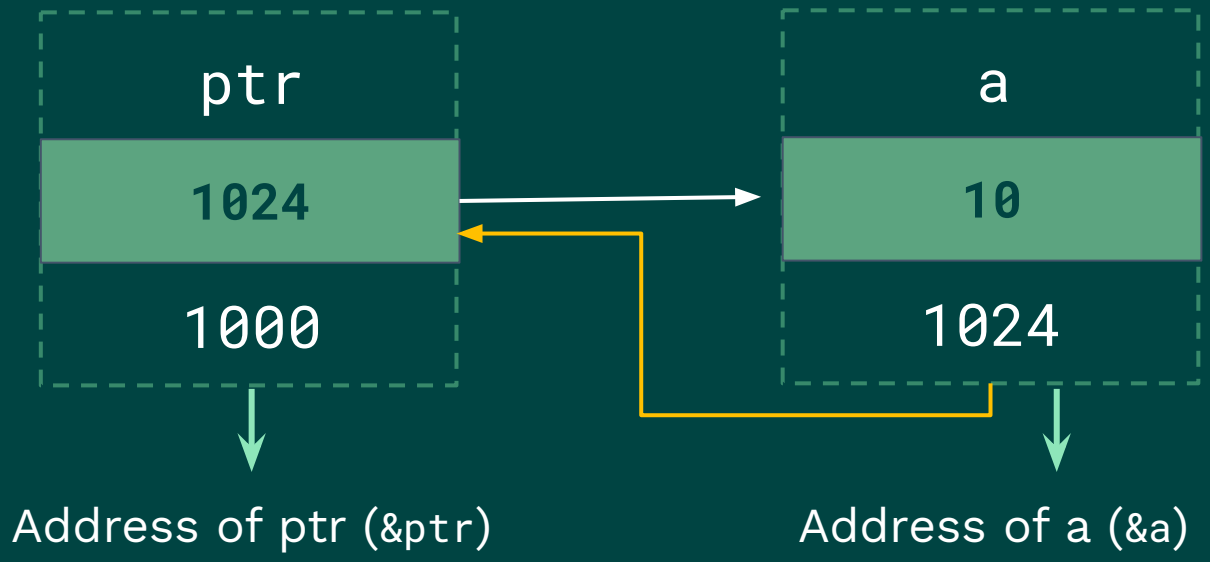

int a = 10;
int *ptr = &a;
printf("value of a = %d\n",a);
printf("value stored at ptr = %d\n",*ptr);
printf("Address of a = %d\n",&a);
printf("ptr points to the address = %d\n",ptr);
printf("Address of ptr = %d\n",&ptr);
OUTPUT
ptr
1024
1000
address of ptr(&ptr)
a
10
1024
address of 'a' (&a)

Punteros


Variable
Pointer
Valor
Dirección
Variable
Interger Pointer
10
Dirección


Variable
Character Pointer
a
Dirección

```
int a = 10
int *ptr = &a;
printf (“value of a = % d/n” , *ptr);
printf (“value stored at ptr = % d/n” , *ptr);
printf (“Address of a = % d/n” , &a );
printf (“ptr points to the address = % d/n” , ptr);
printf (“Address of ptr = % d/n” , &ptr );
```



```
int a = 10
int *ptr = &a;
printf (“value of a = % d/n” , *ptr);
printf (“value stored at ptr = % d/n” , *ptr);
printf (“Address of a = % d/n” , &a );
printf (“ptr points to the address = % d/n” , ptr);
printf (“Address of ptr = % d/n” , &ptr );
```

```
Value of a = 10
Value stored at ptr = 10
Address a = 1024
Ptr points to the address
= 1024
```