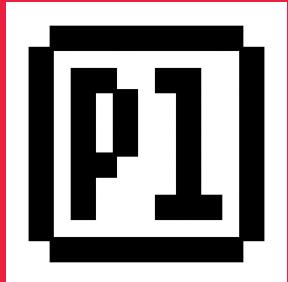


Programación en C#1

UNRN

Universidad Nacional
de Río Negro

r20



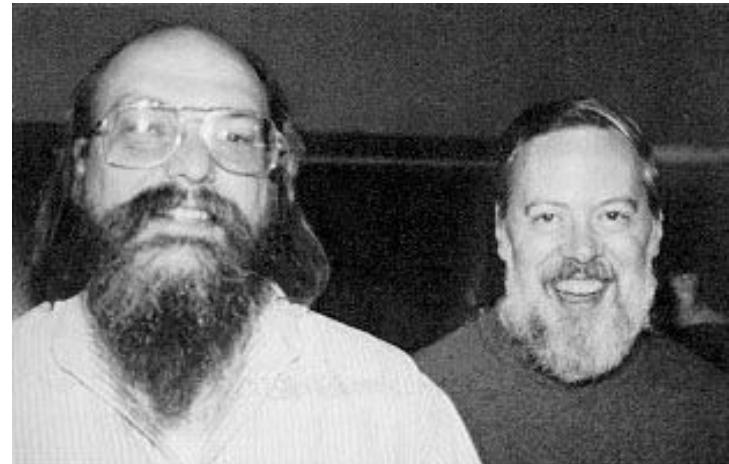


¿Por qué C?

1972

**Creado por
Dennis Ritchie**

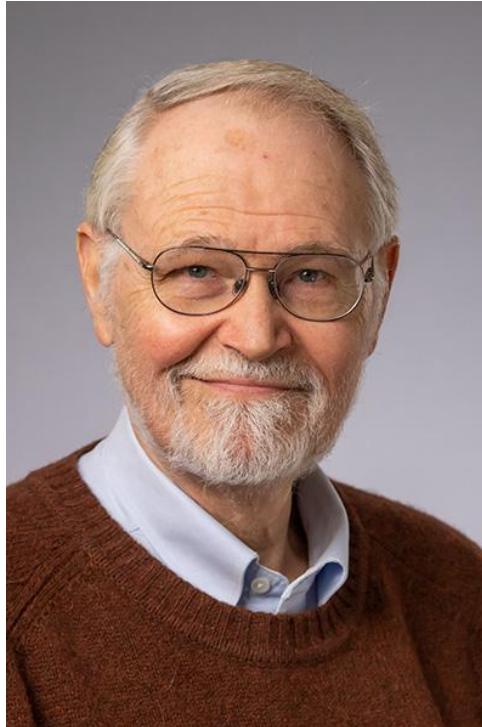
**y
Brian Kernighan**



Bell Labs



Dennis Ritchie
1941-2011



Brian Kernighan

**¿Por qué usamos un
lenguaje que tiene
~50 años?**

– En revisión constante



ISO/IEC 9899



La evolución del lenguaje

UNRN

Universidad Nacional
de Río Negro

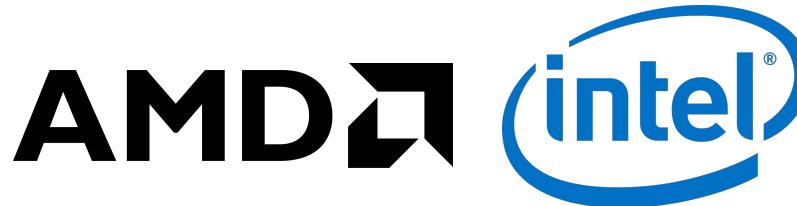
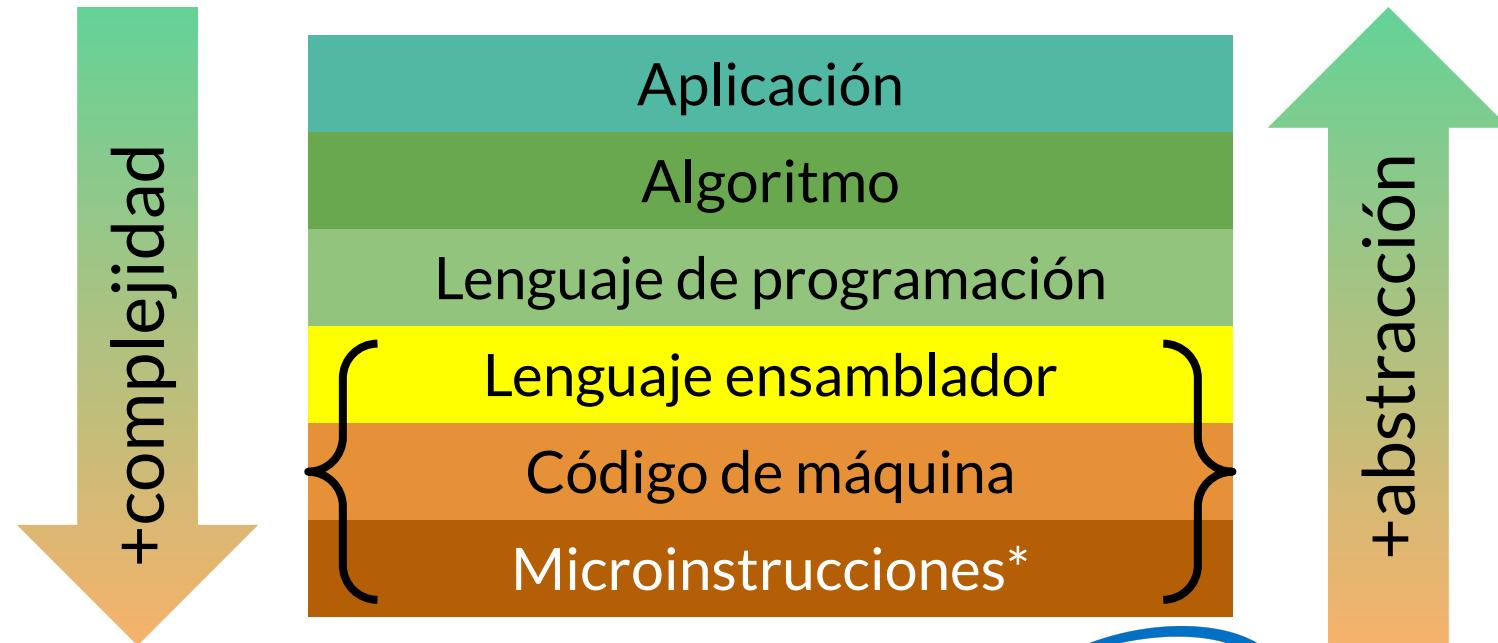
<https://www.iso.org/standard/74528.html>

Un lenguaje muy influyente

- **C++**
- **C#**
- **Javascript**
- **PHP**
- **Objective-C**
- **Java**

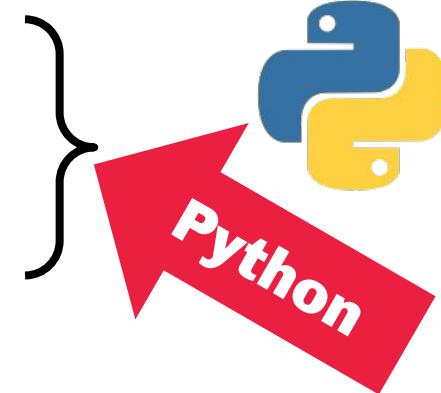
era
de alto nivel
cuando salió

Niveles de abstracción [tradicionales]



**pero considerado
de nivel medio**

Niveles de abstracción



**Esta es la
distinción entre
compilado e
interpretado**

**De propósito
general*
para cuando salió**

simple

Altamente eficiente

Altamente portable

1 - Python

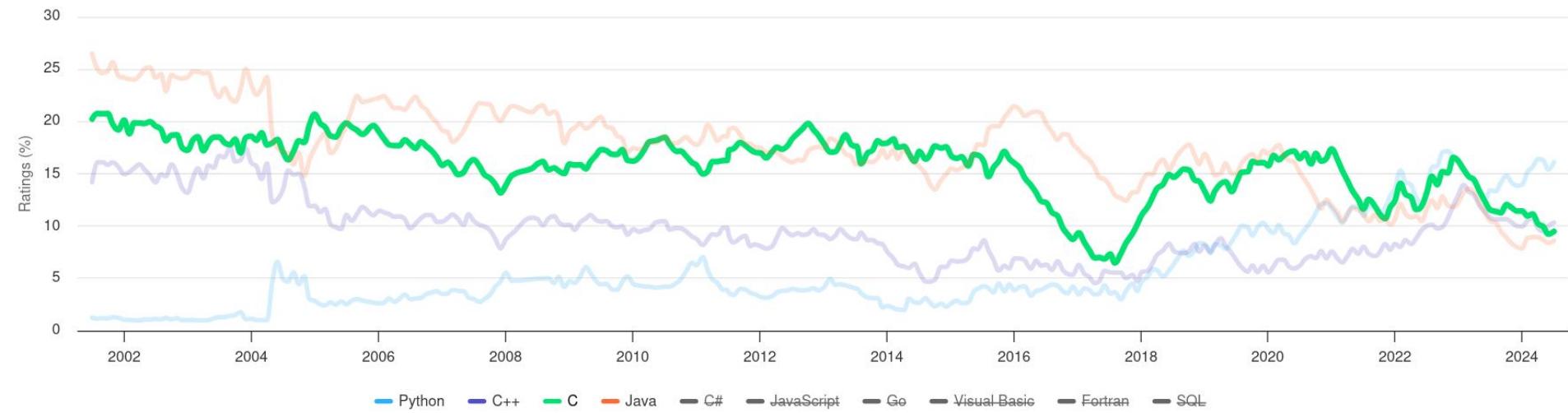
2 - C++

3 - C

4 - Java

TIOBE Programming Community Index

Source: www.tiobe.com



¡Bastante popular! (#3)

UNRN

Universidad Nacional
de Río Negro
<https://www.tiobe.com/tiobe-index/>



¿Preguntas?



Imperativo

**Las instrucciones cambian el
estado del programa**

Imperativo* es:

Que las instrucciones son **secuenciales** con **control de flujo** que lo manipula para modificar el **estado** del programa.

Estructurado

Las instrucciones se agrupan en unidades independientes* entre sí.

Estructurado en

Funciones y procedimientos, que se
diferencian entre sí solo por la forma de
interactuar con el estado del programa.



¿Preguntas?



Tenemos que ver

sintaxis

La estructura válida de un programa

gramática

Un programa correctamente escrito

estilo

La prolijidad

A yellow cube with two large white question marks on its faces, resembling a power-up item from the Super Mario video game series.

¿Preguntas?



Empecemos por el
hola mundo



Un “Hola Mundo C.”

```
#include <stdio.h>

int main(int argc, char *argv[])
{
    printf("Hola mundo C. \n");
    return 0;
}
```

Las piezas básicas

#include <stdio.h>

```
int main(int argc, char *argv[])
{
    printf("Hola mundo . \n");
    return 0;
}
```

Directiva

Punto de entrada

Instrucción de salida

¿Como terminamos?

Compilando y ejecutando

```
$> gcc hola.c  
$> ./a.out    (linux)  
$> ./a.exe    (windows)
```

**¡A la
terminal!**

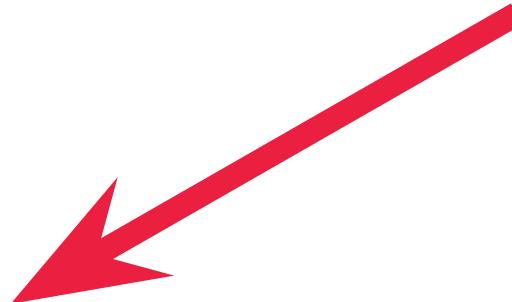


Paso a paso

1 - El punto de entrada

```
#include <stdio.h>

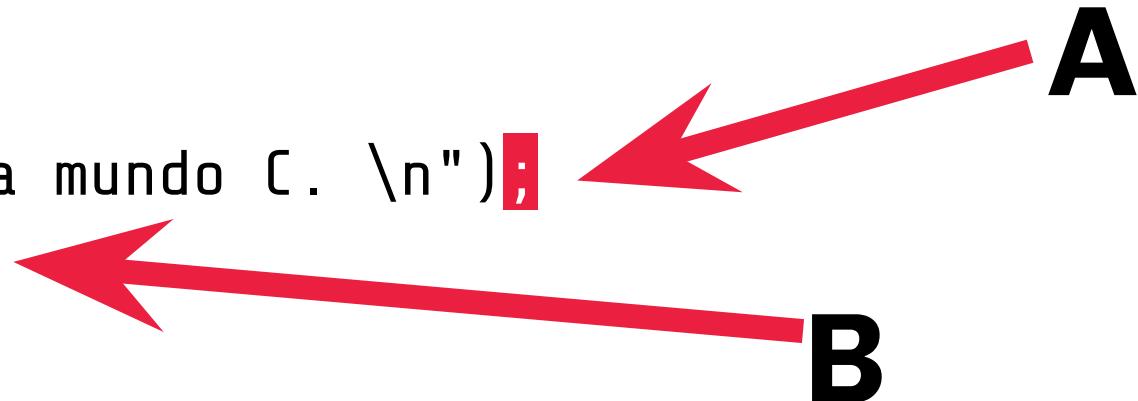
int main(int argc, char *argv[])
{
    printf("Hola mundo (. \n");
    return 0;
}
```



2 - Sentencias

```
#include <stdio.h>
```

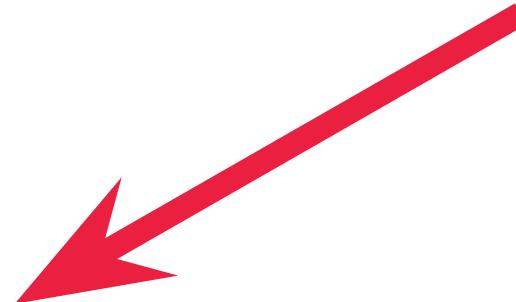
```
int main()
{
    printf("Hola mundo c. \n");
    return 0;
}
```



3 - Bloques

```
#include <stdio.h>

int main(int argc, char *argv[])
{
    printf("Hola mundo \n");
    return 0;
}
```



**¡A la
terminal!**



4 - ¡esto compila!

¡Cuestiones de
estilo!

```
#include <stdio.h>
int main(){printf("Hola mundo C. \n");return 0;}
```

A yellow cube with two large white question marks on its faces, resembling a power-up item from the Super Mario video game series.

¿Preguntas?



Sintaxis

Identificadores

o coloquialmente, nombres

Deben empezar con azAZ_ y
pueden contener luego 09.

Identificadores correctos y no válidos

valido

incorrecto

suma_cuadrados

sumaCuadrados

SumaCuadrados

SUMA CUADRADOS

SUMA2

2SumaCuadrados

2Suma-2

int

Pero si cumple...

Palabras reservadas

auto	register	sizeof	
double	typedef	volatile	
int	char	do	static while
struct	extern	if	_Bool
break	return	continue	_Imaginary
else	union	for	restrict
long	const	signed	_Complex
switch	float	void	inline
case	short	default	
enum	unsigned	goto	

Palabras reservadas en el Hola Mundo C.

```
#include <stdio.h>

int main(int argc, char *argv[])
{
    printf("Hola mundo C. \n");
    return 0;
}
```

1

Los identificadores deben ser descriptivos*

Variables

A diferencia de Python

```
int identificador;  
int identificador = 0;
```

Las variables van
con tipo.

Variaciones *

int a;  sin inicializar

int b = 10;  inicializada en '10'

int c, d, e;  tres variables sin inicializar

int f = 1, g = 4;  dos variables inicializadas

int es para número entero (ahorita pasamos a ese tema)

2

Una declaración de variable por línea

3

Siempre inicializar la variable a un valor conocido

¿Qué tipo de datos hay?

Números enteros

opcionalmente
unsigned

short int

[16-bit]

int

[32-bit]

long int

[64-bit]

long long int

[128-bit]

Varían de una plataforma a otra

Podemos saber el tamaño de los valores en <limits.h>

Rango para `int` esta entre

`INT_MIN` y `INT_MAX` más `UINT_MAX` sin signo

Rango para `long`, entre

`LONG_MIN` y `LONG_MAX` más `ULONG_MAX` sin signo

Hay constantes para cada tipo de número entero

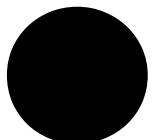
Números enteros de tamaño específico en <stdint.h>

con signo

sin signo

intmax_t	uintmax_t
int8_t	uint8_t
int16_t	uint16_t
int32_t	uint32_t
int64_t	uint64_t

Para tipos de un tamaño específico



**Como le
damos un
valor**

Con los *literales*

0b1010

- binario (0b)

0173

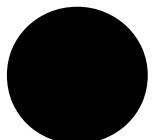
- octal (0)

1234

- decimal

0xCAFE

- hexadecimal (0x)



**¿
hay
decimales?**

Números reales

float (4-bytes)

1.0f

double (8-bytes)

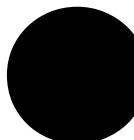
1.0

long double

1.0F

(16-bytes)

**La aritmética de
punto flotante es
mucho más compleja
de lo que parece**



**Algún otro
tipo de dato**

Caracteres individuales

tipo: char

literal: 'a'

Valores lógicos

tipo: bool

literal: true / false



**¡A la
terminal!**



A yellow cube with two large white question marks on its faces, resembling a power-up item from the Super Mario video game series.

¿Preguntas?



-

No hay strings



**Son algo diferente
y vamos a ver más
adelante**

PERO

Antes de seguir, necesitamos ver *algo de cadenas*

Salida por consola



printf: <stdio.h>

```
printf("cadena literal");  
printf("cadena de formato", variables);
```

Forma básica de códigos de formato

«**%C**»

C puede ser

- d ➔ int
- c ➔ char
- f ➔ double (y float)
- c ➔ char

Sin los «» , por cada variable a mostrar en la consola

**Hay un apunte
de este tema**

Un ejemplo

```
int valor = 10;  
printf("numero en valor, %d\n", valor);
```

Se reemplaza en donde están los códigos de formato

printf admite varios valores!

```
int valor = 10;  
double otro = 3.1456;  
printf("v=%d, o=%f\n", valor, otro);
```

Se reemplazan en el mismo orden

**¡A la
terminal!**





¿Preguntas?



Expresiones



Asignación

No se olviden

variable = expresion;



Formalmente, son las partes izquierda y derecha

L-VALUE = R-VALUE ;



Solo variables

L-VALUE

es el destino

Siempre y exclusivamente una variable

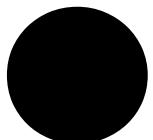
R-VALUE

es el “origen”

**¡Tienen que ser
del mismo tipo!!**

Por ahora





**¿Qué puede
ser un
R-VALUE?**

R-VALUE

- 1. Variables**
- 2. Literales**
- 3. Operaciones**
- 4. una mezcla de lo anterior**

Operadores

Aritméticos binarios

$n + m$

$n - m$

n / m

$n * m$

$n \% m$

Aritméticos unarios

```
variable++;      //incremento  
variable--;      //decremento  
-variable;      //inversión de  
signo
```

**¡A la
terminal!**



4

Un espacio antes y después de cada operador



¿Preguntas?



Porque la idea es que se pueden combinar

```
variable = 1 + suma(3) * variable;
```

Y agrupar instrucciones con paréntesis

¡No hacen lo mismo! [cuidado con la precedencia]

```
variable = (1 + suma(3)) * variable;
```

Uno agrupa instrucciones y el otro los argumentos

**Cada operación
individual, tiene
una entrada y
salida**

**En donde ambos
tienen un tipo**

¿Qué resultado obtenemos?

```
int a = 10;  
float b = 2.33;  
int c = a * b;
```

1. 23,3
2. 23,9999
3. 23
4. 0

Para empezar, mezclar tipos **no es una buena idea**

¿Qué resultado obtenemos?

```
int a = 10;  
float b = 2.33;  
double c = a * b;
```

1. 23,3
2. 23,9999
3. 23
4. 0

A yellow cube with two large white question marks on its faces, resembling a power-up item from the Super Mario video game series.

¿Preguntas?



Operadores relacionales

n > m
n < m
n >= m
n <= m
n == m
n != m



int



**Dan como resultado
un número entero
int**

**Los booleanos son
tecnicamente
nuevos**

¿Pero que significa?

numero > 4

verdadero



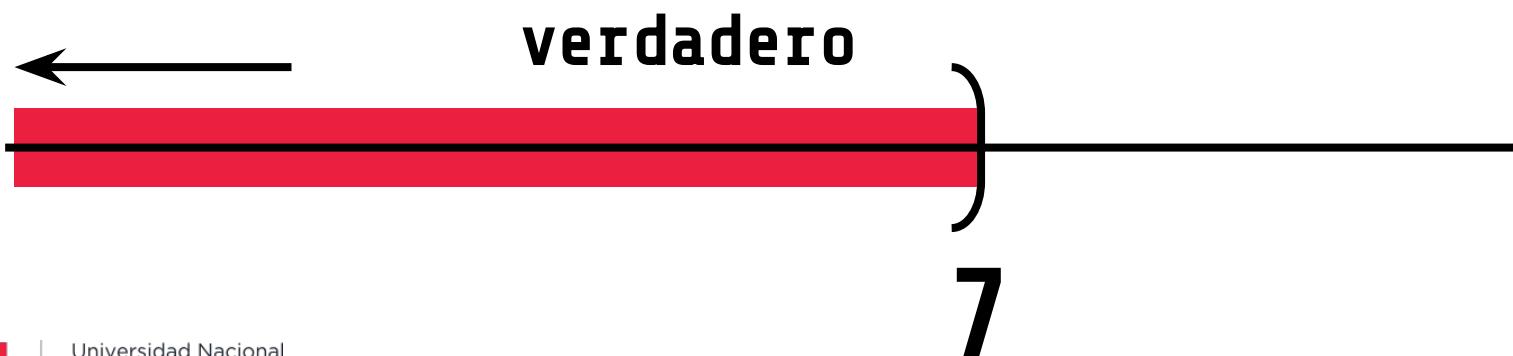
4

numero >= 4

verdadero



numero < 7

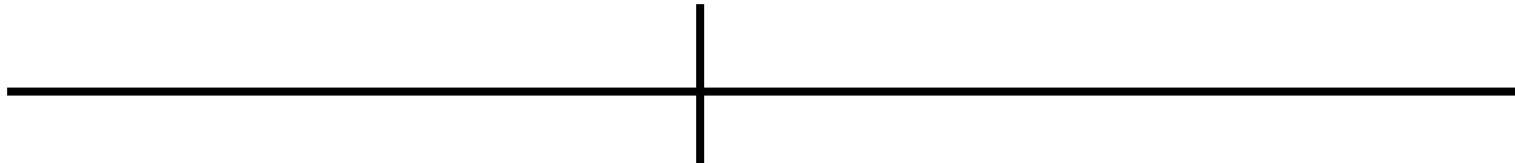


numero <= 7



numero == 5

verdadero



5

numero != 5

verdadero



verdadero



5

0 es falso

1 es verdadero*

***Cualquier otra cosa que no sea cero es verdadero**

Operadores lógicos

para condiciones más
complejas

!
&&
||

negación y lógico o lógico

Un ejemplo rápido

al que volveremos en instantes nada más

`(nota >= 4) && (nota < 7)`



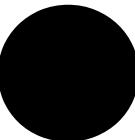
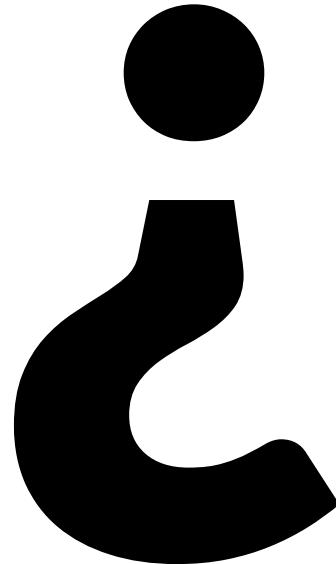
A yellow cube with two large white question marks on its faces, resembling a power-up item from the Super Mario video game series.

¿Preguntas?



Y esto...

Para qué



Tomar decisiones

Estructuras de control

y el estilo a usar

```
if (valor_booleano)
{
    bloque verdadero;
}
```

```
if (nota >= 4)
{
    printf("Aprobado\n");
}
```

`(nota >= 4) && (nota < 7)`



```
if ((nota >= 4) && (nota < 7))  
{  
    printf("Aprobado\n");  
}
```

El valor _booleano puede ser una variable

```
bool comparar = (nota >= 4) && (nota < 7);
if (compara)
{
    printf("Aprobado\n");
}
```

```
if (condición)
{
    bloque verdadero;
}
else
{
    bloque falso;
}
```

Al **valor_booleano** , mejor llamémoslo...

```
if (condición)  
{  
    bloque condición verdadero;  
}  
else if(otra_condición)  
{  
    bloque otra_condición verdadero;  
}
```

Cuestiones a tener en cuenta:

Sin olvidarnos qué if acepta int

```
#include <stdio.h>
#include <stdbool.h>

int main ()
{
    bool b = true;

    if ( b )
    {
        printf ( "Si!\n" );
    }
    else
    {
        printf ( "No!\n" );
    }
    return 0;
}
```

Los bool no forman parte del C original

¡Cuidado con las superposiciones!

```
int numero = scanf("%d");

if (numero >= 0)
{
    printf("Es cero");
}
else if (numero < 4)
{
    printf("menor a 4");
}
else
{
    printf("que va aca?");
}
```

Pueden cambiar el significado del condicional

numero >= 0

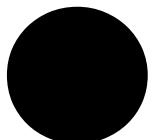
verdadero





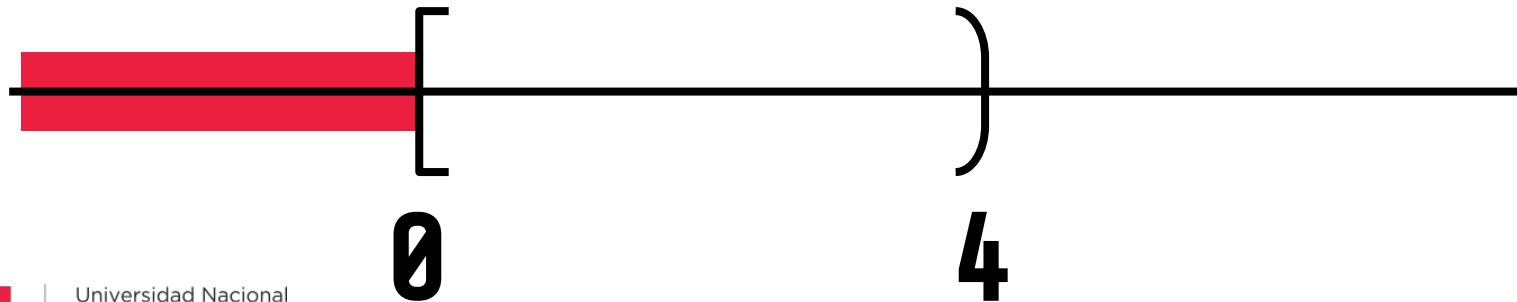
¿Preguntas?





Como solucionarlo

`!(numero >= 0)`



numero < 4



A yellow cube with two large white question marks on its faces, resembling a power-up item from the Super Mario video game series.

¿Preguntas?



¿Cuantos caminos nos da una condición individual?

Con la condición compuesta y más compleja

```
if ((numero >= 4) && (numero < 7)){
```

Solo obtenemos un resultado

A yellow cube with two large white question marks on its faces, resembling a power-up item from the Super Mario video game series.

¿Preguntas?



Repetiendonos



```
for (inicio; consulta; vuelta)
{
    bloque;
}
```

Lazo definido for

```
for (int i = 0; i < 10; i++)
{
    1
    bloque;
}
2
```

```
while (condición)
{
    bloque;
}
```

Lazo indefinido while

```
int i = 0;  
while(i < 10A)  
{  
    1  
    i++;  
}  
2
```

```
do
{
    bloque;
}
while (condición);
```

Lazo indefinido do . . while

```
int i = 0;  
do  
{  
    1  
    i++;  
}  
while (i < 10A);  
2
```

A classic Looney Tunes scene. Wile E. Coyote is shown in mid-stride, running towards the right. He has a determined expression, his mouth is open as if he's shouting, and his arms are pumping. He wears his signature brown suit and bow tie. In the upper left, Road Runner stands on a rocky ledge, looking back over his shoulder at Wile E. with a smug, knowing smile. He has his signature long blue tail and white body. The background is a simple yellow gradient. Two speech bubbles are present: one above Wile E. containing code for a while loop, and one above Road Runner containing code for a do-while loop.

```
while (not edge) {  
    run();  
}
```

```
do {  
    run();  
} while (not edge);
```

5

Las llaves de los bloques van en una línea propia

Lazos más comunes



De menor a mayor

```
int numero = 0;
while (numero < 10)
{
    printf("numero:%d", numero);
    numero = numero + 1;
}
```

Este es uno de los más comunes.

De menor a mayor

```
int numero = 0;  
do  
{  
    printf("numero:%d", numero);  
    numero = numero + 1;  
}  
while (numero < 10);
```

Este es uno de los más comunes.

De mayor a menor

```
int numero = 10;
while (numero > 0)
{
    printf("numero:%d", numero);
    numero = numero - 1;
}
```

Una variación simple

Cuando tenemos **menos** idea de cuantas veces ejecutar

```
int numero = 10;
bool bandera = true;
while (bandera)
{
    printf("numero:%d", numero);
    numero = numero - 1;
    if (numero > 0)
    {
        bandera = false;
    }
}
```

¡El if interno puede ser mucho más complejo!

¡La condición puede ser compuesta!

```
int numero = 10;
while (((numero > 0) && (numero < 20)))
{
    printf("numero:%d", numero);
    numero = numero + 1;
    if (numero % 2 == 0)
    {
        numero = -numero * 2;
    }
}
```

Aunque este ejemplo sea *un poco rebuscado* :-)

Suma de una cantidad abierta de números

```
int numero;
int cantidad = 0;
int suma = 0;
bool activo = true;
while (activo)
{
    printf("numero %d", cantidad);
    scanf("%d", &numero);
    cantidad = cantidad + 1;
    if ((cantidad > 100) && (numero < 0))
    {
        activo = false;
    }
}
```

¡El if interno puede ser mucho más complejo!

Manipulaciones de lazos

break

```
int i = 0;  
while(i < 10)  
{  
    break;  
    i++;  
}  
printf("i es %d\n", i); 1
```

break;

para cortar el lazo

continue ;

para pasar a la siguiente iteración

6

Los lazos sin break y continue en su lugar, lazos con bandera

7

**Prefieran usar
while en lugar de
for**

Deciciones multiples



NINTENDO
SWITCH

```
switch (expresión)
{
    case valor:
        sentencia
    break;
    default:
        sentencia;
}
```

```
switch (expresión)
{
    case valor:
        sentencia
    break;
    default:
        sentencia;
}
```

¡son opcionales!

El no usar break agrupa casos

```
switch (numero)
{
    case 1:
    case 2:
    case 3:
    case 4:
    case 5:
    case 6:
        printf ("numero es >= 1 y <= 6\n");
        break;
}
```

El no usar break agrupa casos

```
int veces = 0;
switch (numero)
{
    case 1:
        veces = veces + 10;
    case 2:
        veces = veces + 10;
    case 3:
        veces = veces + 10;
        break;
}
printf ("veces es %d\n", veces);
```

¿Que valores acepta?
¿Cuál es la salida?

**¡A la
terminal!**



A yellow cube with two large white question marks on its faces, resembling a power-up item from the Super Mario video game series.

¿Preguntas?



scanf

Tomando de la consola

De a una variable por vez

```
int variable;
```

```
scanf( "%d" , &variable );
```



**Ya vamos a ver que
es el
&variable**

**Usa los mismos
marcadores de tipo
que printf**

Los más importantes (lista no exhaustiva)

%d %i	int	Números enteros
%u	unsigned int	Números enteros sin signo
%e %f %g	float/double	Números decimales
%c	char	Caracter individual

No exactamente igual a printf.

A yellow cube with two large white question marks on its faces, resembling a power-up item from the Super Mario video game series.

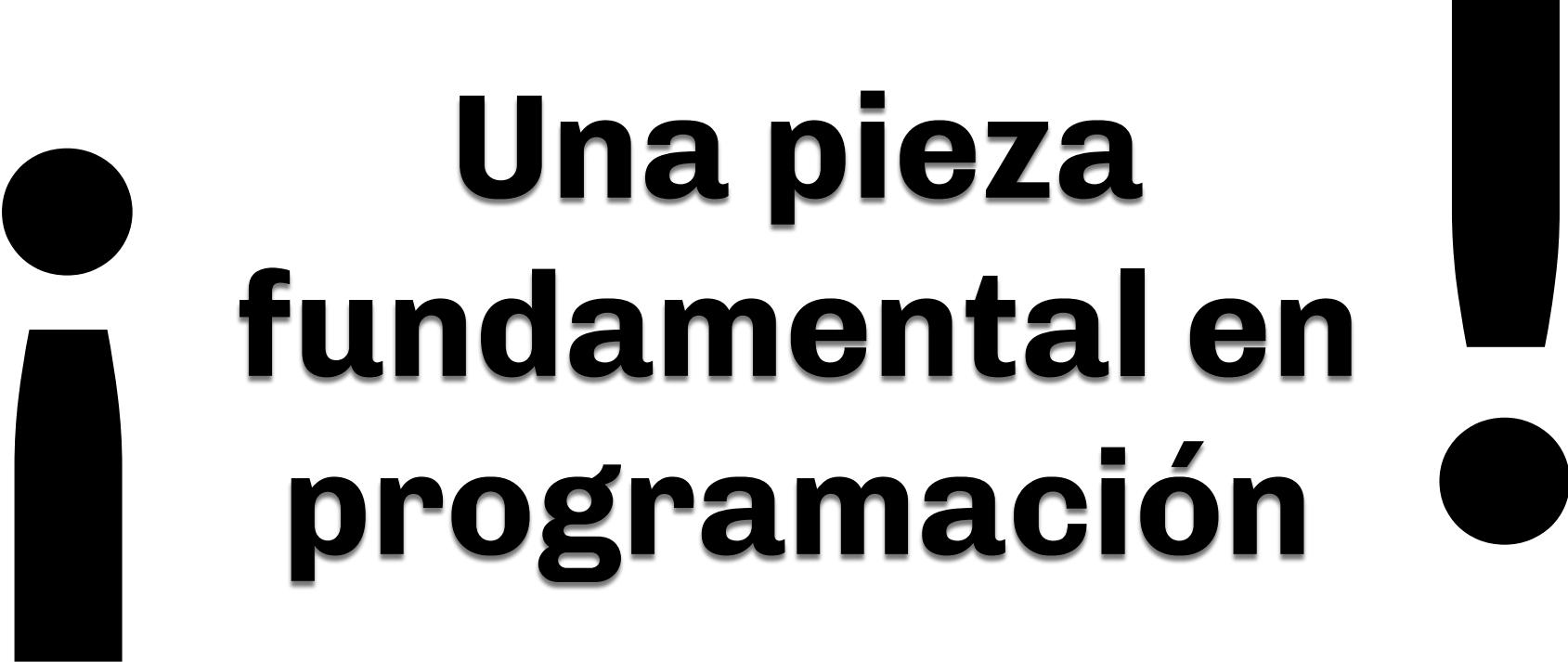
¿Preguntas?



Funciones

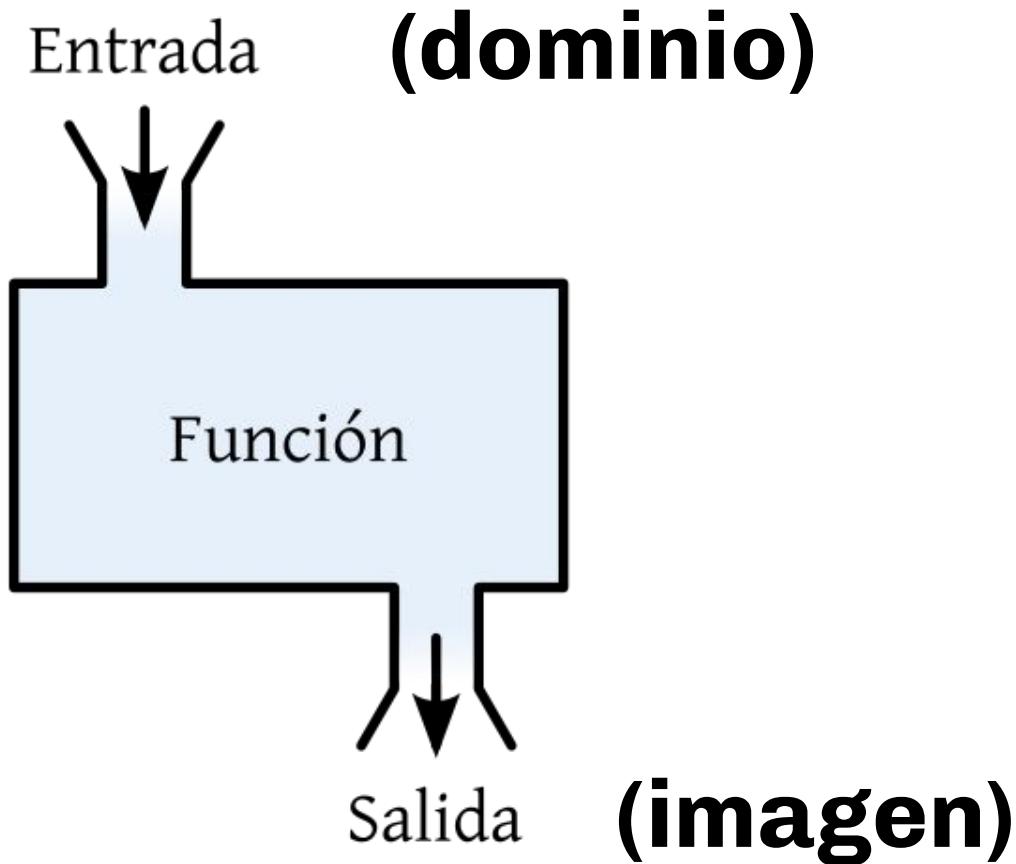
parte 1





Una pieza fundamental en programación

¿Cómo es una función?



Definición de funciones

```
tipo_retorno nombre(tipo_argumento nombre_argumento)
{
    return valor_generado;
}
```



*Ya vimos como llamar a una

Una función simple

```
int sumar(int valor_a, int valor_b)
{
    return valor_a + valor_b;
}
```

8

Una sola instrucción return por función

**Una función con
retorno de valor es un**

R-VALUE

Formas de llamada a función

```
int resultado = suma(10, 45);
```

```
int otro = suma(resultado, 10);
```

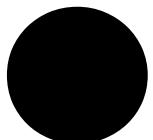
Con clases de argumentos diferentes

A yellow cube with two large white question marks on its faces, resembling a power-up item from the Super Mario video game series.

¿Preguntas?

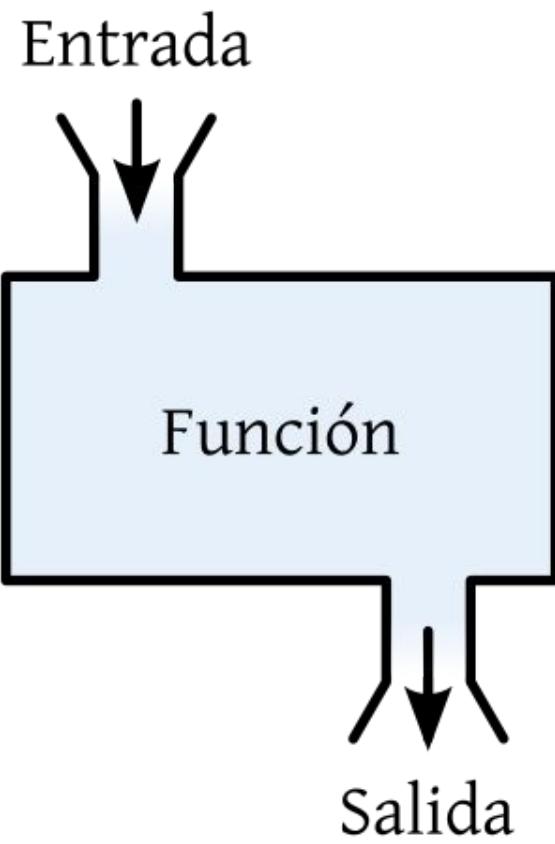


**Una observación
muy importante**



**cuál es el
propósito de
una función**

¿Cómo es una función?



9

Las funciones **no van**
con **printf** o **scanf**

Consideraciones generales

¿Como es un comentario documentación completo?

```
/**  
 * La suma lenta es una forma rebuscada  
 * de reinventar la rueda en pos del  
 * aprendizaje.  
 * @param n es el primer sumando  
 * @param m es el segundo sumando  
 * @return la suma de los sumandos  
 */  
int suma_lenta(int n, int m);
```

El miércoles vamos a ver un par más de detalle de esto

1

0

**Todas las funciones
van con comentario
de documentación**

A yellow cube with two large white question marks on its faces, resembling a power-up item from the Super Mario video game series.

¿Preguntas?



Hasta la próxima





unrn.edu.ar



Universidad Nacional
de Río Negro



| unrnionegro