

# Algoritmos y Estructuras de Datos

Clase 2



**UTN.BA**

UNIVERSIDAD TECNOLÓGICA NACIONAL  
FACULTAD REGIONAL BUENOS AIRES

# Lógica Proposicional



**UTN.BA**

UNIVERSIDAD TECNOLÓGICA NACIONAL  
FACULTAD REGIONAL BUENOS AIRES

# ¿Para qué sirve?

Estudia la validez de las **proposiciones**

# ¿Para qué sirve?

Estudia la validez de las **proposiciones**



**Enunciados** que tienen una característica particular, pueden ser **verdaderos** o **falsos**.

Se representan como  $p, q, r, s$ , etc.

# ¿Son enunciados?

- Espero ingresar a la Universidad
- ¿Cómo te llamas?
- $X+5 = 7$
- Buenos Aires es la capital de Argentina
- $1+1 = 3$
- Pablo Neruda nació en Chile

# Clases de Proposiciones

SIMPLE

Juan estudia

COMPUESTA

Juan estudia y trabaja

# Clases de Proposiciones

SIMPLE

Juan estudia

COMPUESTA

Juan estudia y trabaja



conector lógico

# Conectores Lógicos

Símbolo	Operación Lógica	Esquema	Significado
$\sim$	negación	$\sim p$	<b>no</b> p
$\wedge$	conjunción	$p \wedge q$	p <b>y</b> q
$\vee$	disyunción	$p \vee q$	p <b>o</b> q
$\rightarrow$	condicional	$p \rightarrow q$	<b>si</b> p, <b>entonces</b> q
$\leftrightarrow$	bicondicional	$p \leftrightarrow q$	p, <b>si y solo si</b> q
$\underline{\vee}$	disyunción exclusiva	$\underline{\vee}$	<b>o</b> p <b>o</b> q



# Negación ( $\sim$ )

$p$	$\sim p$
V	
F	

No

No es verdad que

Es falso que

No ocurre que

No es el caso que

No es cierto que Rodrigo sea maestro.

# Negación ( $\sim$ )

$p$	$\sim p$
V	F
F	V

No

No es verdad que

Es falso que

No ocurre que

No es el caso que

No es cierto que Rodrigo sea maestro.

# Conjunción ( $\wedge$ )

p	q	$p \wedge q$
V	V	
V	F	
F	V	
F	F	

Y

Pero

Sin embargo

Además

No obstante

Aunque

A la vez

También

Juan es futbolista y Ana es voleybolista.

# Conjunción ( $\wedge$ )

p	q	$p \wedge q$
V	V	V
V	F	F
F	V	F
F	F	F

Y

Pero

Sin embargo

Además

No obstante

Aunque

A la vez

También

Juan es futbolista y Ana es voleybolista.

# Disyunción ( $\vee$ )

p	q	$p \vee q$
V	V	
V	F	
F	V	
F	F	

o

Raúl es profesor o Raúl es ingeniero.

# Disyunción ( $\vee$ )

p	q	$p \vee q$
V	V	V
V	F	V
F	V	V
F	F	F

o

Raúl es profesor o Raúl es ingeniero.

# Condicional ( $\rightarrow$ )

p	q	$p \rightarrow q$
V	V	
V	F	
F	V	
F	F	

Si..., entonces...

Porque

Puesto que

Cuando

Cada vez que

Si Carla estudia, entonces ingresará a la universidad.

# Condicional ( $\rightarrow$ )

<b>p</b>	<b>q</b>	<b><math>p \rightarrow q</math></b>
V	V	V
V	F	F
F	V	V
F	F	V

Si..., entonces...

Porque

Puesto que

Cuando

Cada vez que

Si Carla estudia, entonces ingresará a la universidad.



# Bicondicional ( $\leftrightarrow$ )

<b>p</b>	<b>q</b>	<b><math>p \leftrightarrow q</math></b>
V	V	
V	F	
F	V	
F	F	

Si y solo si  
Cuando y solo cuando  
Entonces y solamente entonces

Ana irá a la fiesta si y solo si tiene amigas.

# Bicondicional ( $\leftrightarrow$ )

<b>p</b>	<b>q</b>	<b><math>p \leftrightarrow q</math></b>
V	V	V
V	F	F
F	V	F
F	F	V

Si y solo si  
Cuando y solo cuando  
Entonces y solamente entonces

Ana irá a la fiesta si y solo si tiene amigas.

# Disyunción Exclusiva ( $\vee$ )

p	q	$p \vee q$
V	V	
V	F	
F	V	
F	F	

O...o...

O bien Manuel juega o bien estudia.

# Disyunción Exclusiva ( $\vee$ )

p	q	$p \vee q$
V	V	F
V	F	V
F	V	V
F	F	F

O...o...

O bien Manuel juega o bien estudia.

# En C++

Negación ( $\sim$ )	!
Conjunción ( $\wedge$ )	&&
Disyunción ( $\vee$ )	

# Tipos de Datos



**UTN.BA**

UNIVERSIDAD TECNOLÓGICA NACIONAL  
FACULTAD REGIONAL BUENOS AIRES

# Características

1. Un rango de valores posibles.
2. Un conjunto de operaciones que se pueden realizar.
3. Su representación interna.

# Clasificación

## ESTÁTICOS

Ocupan una posición de memoria en el momento de la definición. Liberan la memoria al finalizar la aplicación.

## DINÁMICOS

Ocupan direcciones de memoria en tiempo de ejecución y se instancian a través de punteros. Las instancias pueden liberarse en tiempo de ejecución.





# Clasificación

ESTÁTICOS

SIMPLES

Son indivisibles en datos más elementales.

CADENAS

Contienen N caracteres tratados como una única variable.

ESTRUCTURAS

Tienen un único nombre para mas de un dato que puede ser del mismo tipo o de distinto. Permiten acceso a cada dato particular. Son divisibles en datos mas elementales.



**UTN.BA**  
UNIVERSIDAD TECNOLÓGICA NACIONAL  
FACULTAD REGIONAL BUENOS AIRES

# Clasificación

ESTÁTICOS

SIMPLES

Son indivisibles en datos más elementales.

Enteros

Lógico o Booleano

Carácter

Reales



**UTN.BA**  
UNIVERSIDAD TECNOLÓGICA NACIONAL  
FACULTAD REGIONAL BUENOS AIRES

# En C++

Enteros	int	4 Bytes	-2147483648 a +2147483647
	unsigned int	4 Bytes	0 a 4294967295
Carácter	char	1 Byte	-128 a 127
Lógico	bool	1 Byte	0, 1
Reales	float	4 Bytes	Positivos: 3.4E-38 a 3.4E38 Negativos: -3.4E-38 a -3.4E38
	double	8 Bytes	Positivos: 1.7E-308 a 1.7E308 Negativos: -1.7E-308 a -1.7E308

...Y hay muchos más!



**UTN.BA**  
UNIVERSIDAD TECNOLÓGICA NACIONAL  
FACULTAD REGIONAL BUENOS AIRES

# Clasificación

Carácter

## Tabla ASCII

Caracteres ASCII de control		
00	NULL	(carácter nulo)
01	SOH	(inicio encabezado)
02	STX	(inicio texto)
03	ETX	(fin de texto)
04	EOT	(fin transmisión)
05	ENQ	(consulta)
06	ACK	(reconocimiento)
07	BEL	(timbre)
08	BS	(retroceso)
09	HT	(tab horizontal)
10	LF	(nueva línea)
11	VT	(tab vertical)
12	FF	(nueva página)
13	CR	(retorno de carro)
14	SO	(desplaza afuera)
15	SI	(desplaza adentro)
16	DLE	(esc.vínculo datos)
17	DC1	(control disp. 1)
18	DC2	(control disp. 2)
19	DC3	(control disp. 3)
20	DC4	(control disp. 4)
21	NAK	(conf. negativa)
22	SYN	(inactividad sinc)
23	ETB	(fin bloque trans)
24	CAN	(cancelar)
25	EM	(fin del medio)
26	SUB	(sustitución)
27	ESC	(escape)
28	FS	(sep. archivos)
29	GS	(sep. grupos)
30	RS	(sep. registros)
31	US	(sep. unidades)
127	DEL	(suprimir)

Caracteres ASCII imprimibles			
32	espacio	64	@
33	!	65	A
34	"	66	B
35	#	67	C
36	\$	68	D
37	%	69	E
38	&	70	F
39	'	71	G
40	(	72	H
41	)	73	I
42	*	74	J
43	+	75	K
44	,	76	L
45	-	77	M
46	.	78	N
47	/	79	O
48	0	80	P
49	1	81	Q
50	2	82	R
51	3	83	S
52	4	84	T
53	5	85	U
54	6	86	V
55	7	87	W
56	8	88	X
57	9	89	Y
58	:	90	Z
59	;	91	[
60	<	92	\
61	=	93	]
62	>	94	^
63	?	95	_
96	`		
97	a		
98	b		
99	c		
100	d		
101	e		
102	f		
103	g		
104	h		
105	i		
106	j		
107	k		
108	l		
109	m		
110	n		
111	o		
112	p		
113	q		
114	r		
115	s		
116	t		
117	u		
118	v		
119	w		
120	x		
121	y		
122	z		
123	{		
124			
125	}		
126	~		

ASCII extendido					
128	Ç	160	á	192	Ł
129	ü	161	í	193	ł
130	é	162	ó	194	Ł
131	â	163	ú	195	ł
132	ä	164	ñ	196	—
133	à	165	Ñ	197	†
134	â	166	°	198	ä
135	ç	167	°	199	Ä
136	è	168	¿	200	ℒ
137	ë	169	©	201	ℓ
138	ê	170	¬	202	ℓ
139	ï	171	½	203	ℓ
140	î	172	¼	204	ℓ
141	ì	173	¡	205	=
142	À	174	«	206	≠
143	Á	175	»	207	≠
144	Ê	176	⋮	208	δ
145	æ	177	⋮	209	Ð
146	Æ	178	⋮	210	Ê
147	ô	179		211	É
148	ö	180		212	È
149	ò	181	À	213	Ì
150	û	182	Á	214	Í
151	ù	183	Â	215	Î
152	ÿ	184	©	216	Ï
153	Ö	185	ℓ	217	Ɔ
154	Ü	186	ℓ	218	ℓ
155	ø	187	ℓ	219	ℓ
156	£	188	ℓ	220	ℓ
157	Ø	189	¢	221	ℓ
158	×	190	¥	222	ℓ
159	f	191	ℓ	223	ℓ
224	Ó				
225	ô				
226	Ô				
227	Õ				
228	ö				
229	Ö				
230	μ				
231	ρ				
232	ρ				
233	Ù				
234	Ú				
235	Û				
236	Ý				
237	Ý				
238	—				
239	—				
240	≡				
241	±				
242	—				
243	¼				
244	¶				
245	§				
246	÷				
247	—				
248	—				
249	—				
250	—				
251	—				
252	—				
253	—				
254	—				
255	nbsp				

# Clasificación

ESTÁTICOS

ESTRUCTURAS

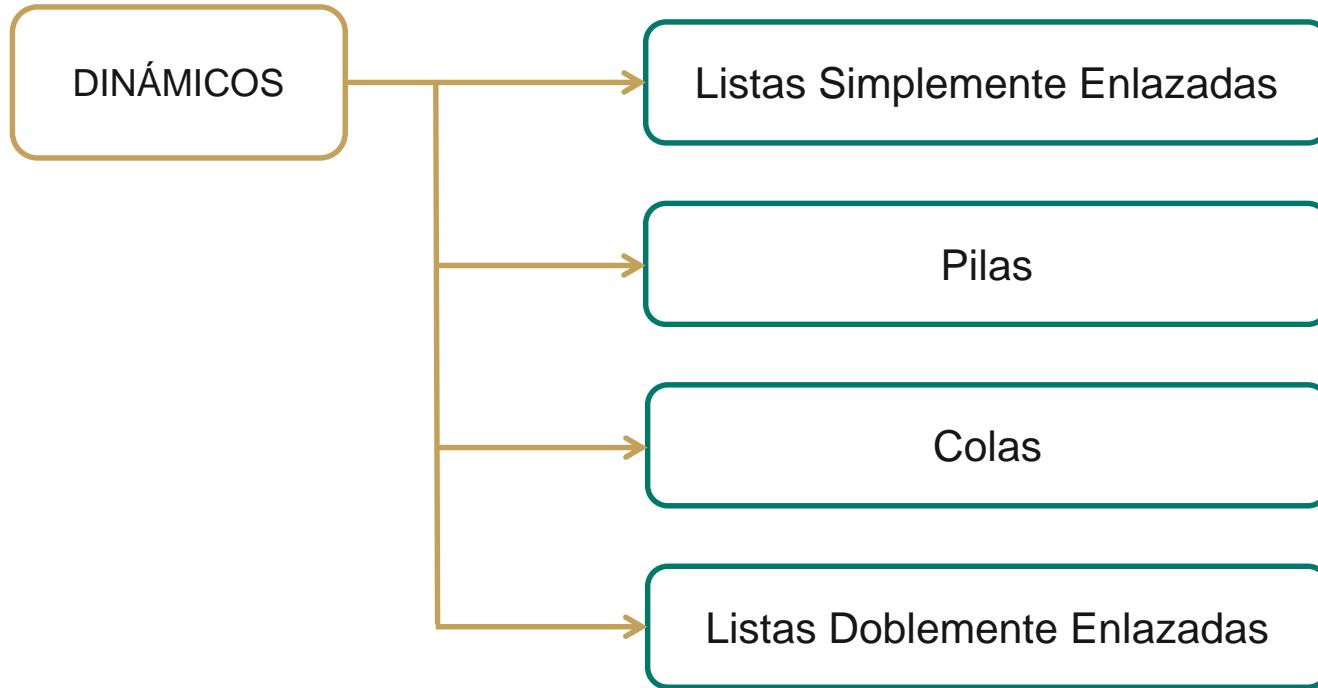
Tienen un único nombre para mas de un dato que puede ser del mismo tipo o de distinto. Permiten acceso a cada dato particular. Son divisibles en datos mas elementales.

Registro

Vector

Archivos

# Clasificación



# Identificadores, Variables, Constantes y Asignación



**UTN.BA**

UNIVERSIDAD TECNOLÓGICA NACIONAL  
FACULTAD REGIONAL BUENOS AIRES

# Identificadores

Conjunto de caracteres alfanuméricos que sirve para identificar las entidades del programa (variables, constantes, funciones, procedimientos, etc.)



Reglas en C++:

- Solo está permitido usar letras, dígitos y guión bajo.
- El identificador no puede comenzar con un dígito.
- Las palabras reservadas no pueden usarse como identificador.
- Los caracteres especiales no pueden utilizarse.
- Las minúsculas y las mayúsculas son diferentes.



# ¿Están permitidos?

- `_stock`
- `Stock`
- `stock`
- `stock_1`
- `INT`
- `int`
- `stock_deposito`
- `stockDeposito`
- `$stock`
- `pow`
- `stock 1`
- `2stock`

# ¿Están permitidos?

- `_stock`
- `Stock`
- `stock`
- `stock_1`
- `INT`
- `int`
- `stock_deposito`
- `stockDeposito`
- `$stock`
- `pow`
- `stock 1`
- `2stock`

# Variables

Los datos están representados simbólicamente por un nombre que se asocia con una dirección única de memoria.

- ✓ Un nombre.
- ✓ Un tipo de dato.
- ✓ El valor que tiene en un momento determinado.

En C++: `int edad;`

# Asignación

El operador de asignación (=), sirve para almacenar un valor en una variable.



La asignación de un **valor inicial** se llama **inicialización**.

# Asignación

En diagrama de Lindsay:

```
codigoPostal ← 1602
```

En C++:

*Inicialización:*

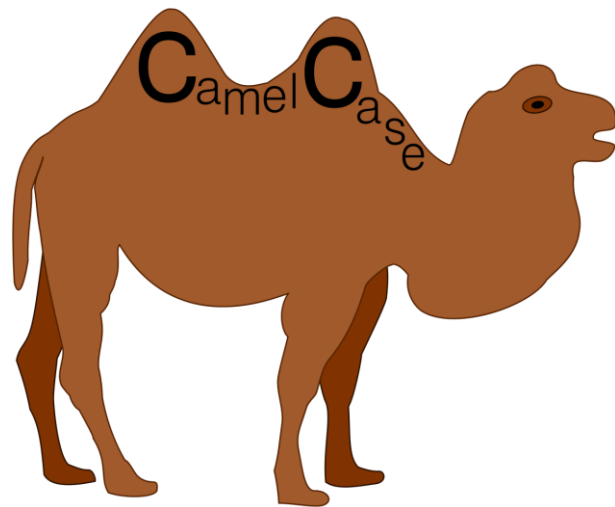
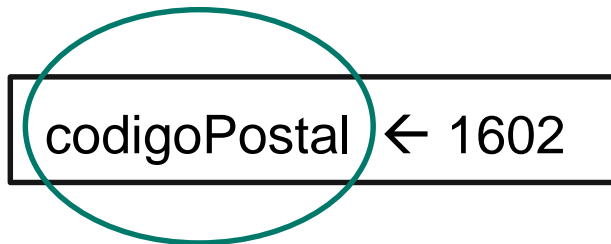
```
int edad = 27;  
int contador = 0;  
bool activo = 1;  
float precio = 10.5;
```

*Asignación:*

```
nota = 10;  
legajo = 1383073;  
nombre = "Natalia";
```

# Asignación

En diagrama de Lindsay:



En C++:

**lowerCamelCase**

*Inicialización:*

```
int edad = 27;  
int contador = 0;  
bool activo = 1;  
float precio = 10.5;
```

*Asignación:*

```
nota = 10;  
legajo = 1383073;  
nombre = "Natalia";
```

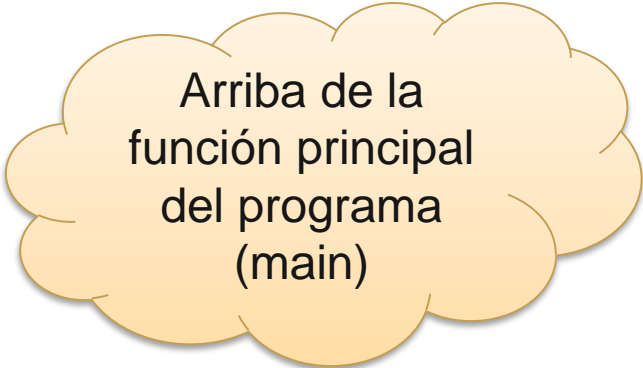
# Constantes

Nombre asociado a un valor que permanece sin modificación durante toda la ejecución del programa.



Solo puede ser modificado en una nueva compilación.

En C++: `const double pi = 3.14159;`  
`#define PI 3.14159`



Arriba de la  
función principal  
del programa  
(main)

# Operadores, Expresiones y Sentencias



**UTN.BA**  
UNIVERSIDAD TECNOLÓGICA NACIONAL  
FACULTAD REGIONAL BUENOS AIRES



# Operadores

Un **operador** actúa sobre variables para realizar una determinada **operación** con un determinado **resultado**.

## Aritméticos: Operadores Binarios

Suma	+
Resta	-
Multiplicación	*
División	/
Resto	%

Resto de la división entera: Se aplica solamente a variables, constantes y expresiones de tipo **int**.



# Operadores

## Asignación (=)



Una variable puede aparecer a la izquierda y a la derecha del operador de asignación.



A la izquierda del operador no puede haber nunca una expresión.

$a + b = c;$



# Operadores

Asignación (=) Hay que operadores simplifican algunas operaciones recurrentes sobre una misma variable.

edad += 1;	edad = edad + 1;
distancia -= 1;	distancia = distancia - 1;
rango /= 2.0;	rango = rango / 2.0;
x *= 3.0 * y - 1.0	x = x * (3.0 * y - 1.0)

variable **operador**= expresión → variable = variable **operador** expresión



# Operadores

## Incrementales: Operadores Unarios

<code>i++;</code>	<code>i = i + 1;</code>
<code>i--;</code>	<code>i = i - 1;</code>

```
int i = 2;  
int j = 2;  
m = i++;  
n = ++j;
```

# Operadores

## Incrementales: Operadores Unarios

i++;	i = i + 1;
i--;	i = i - 1;

int i = 2;

int j = 2;

m = i++; // después de ejecutarse esta sentencia m=2 e i=3

n = ++j; // después de ejecutarse esta sentencia n=3 y j=3



# Operadores

## Relacionales: Operadores Binarios

Igual que	==
Menor que	<
Mayor que	>
Menor o igual que	<=
Mayor o igual que	>=
Distinto que	!=



# Operadores

## Relacionales:

Si la condición representada por el operador relacional se cumple, el resultado es 1; si la condición no se cumple, el resultado es 0.

- $2+1 == 1$
- $3+1 \leq 2+2$
- $3 < 3$
- $1 \neq 1$

expresión1 **operador** expresión2



**UTN.BA**  
UNIVERSIDAD TECNOLÓGICA NACIONAL  
FACULTAD REGIONAL BUENOS AIRES

# Operadores

## Relacionales:

Si la condición representada por el operador relacional se cumple, el resultado es 1; si la condición no se cumple, el resultado es 0.

- $2+1 == 1$  // resultado = 0
- $3+1 <= 2+2$  // resultado = 1
- $3 < 3$  // resultado = 0
- $1 != 1$  // resultado = 0

expresión1 operador expresión2



**UTN.BA**  
UNIVERSIDAD TECNOLÓGICA NACIONAL  
FACULTAD REGIONAL BUENOS AIRES



# Operadores

Lógicos:

Y	&&
O	
Negación	!

- `!((2==1) || (-1== -1))`
- `(2==2) && (3== -1)`
- `((2==2) && (3==3)) || (4==0)`
- `((6==6) || (8==0)) && ((5==5) && (3==2))`



# Operadores

Lógicos:

Y	&&
O	
Negación	!

- `!((2==1) || (-1== -1))` // resultado = 0
- `(2==2) && (3== -1)` // resultado = 0
- `((2==2) && (3==3)) || (4==0)` // resultado = 1
- `((6==6) || (8==0)) && ((5==5) && (3==2))` // resultado = 0

# Operadores

Lógicos:

Y	&&
O	
Negación	!

- `!((2==1) || (-1== -1))` // resultado = 0
- `(2==2) && (3== -1)` // resultado = 0
- `((2==2) && (3==3)) || (4==0)` // resultado = 1
- `((6==6) || (8==0)) && ((5==5) && (3==2))` // resultado = 0

# Expresiones

Una **expresión** es un conjunto de variables y constantes – y también de otras expresiones más sencillas – relacionadas mediante distintos operadores.

Ejemplo:  $x = (-b + \sqrt{(b*b) - (4*a*c)}) / (2*a);$

# Sentencias

Las expresiones son unidades o componentes elementales de unas entidades de rango superior que son las sentencias.



Las sentencias son unidades completas, ejecutables en sí mismas.

Sentencias Simples: Es una expresión de algún tipo terminada con un carácter (;)

float precio;

posicion = posicionInicial + velocidad \* tiempo;



**UTN.BA**  
UNIVERSIDAD TECNOLÓGICA NACIONAL  
FACULTAD REGIONAL BUENOS AIRES

# Sentencias

Sentencia vacía o nula:

;

Sentencias compuesta o bloques:

```
{  
    double posicion;  
    double posicionInicial = 30.5;  
    double velocidad = 25.4;  
    double tiempo;  
    tiempo = 3.1;  
    posicion = posicionInicial + velocidad * tiempo;  
}
```



**UTN.BA**  
UNIVERSIDAD TECNOLÓGICA NACIONAL  
FACULTAD REGIONAL BUENOS AIRES

# E/S (Entrada/Salida)



**UTN.BA**  
UNIVERSIDAD TECNOLÓGICA NACIONAL  
FACULTAD REGIONAL BUENOS AIRES

# Entrada

La entrada estándar es el teclado.



En diagrama de Lindsay:

```
int edad = 0;
```



En C++:

```
int edad = 0;  
cin >> edad;
```



# Salida

La salida estándar es la pantalla.

En diagrama de Lindsay:



En C++:

```
cout << edad;
```



# Estructuras de Selección

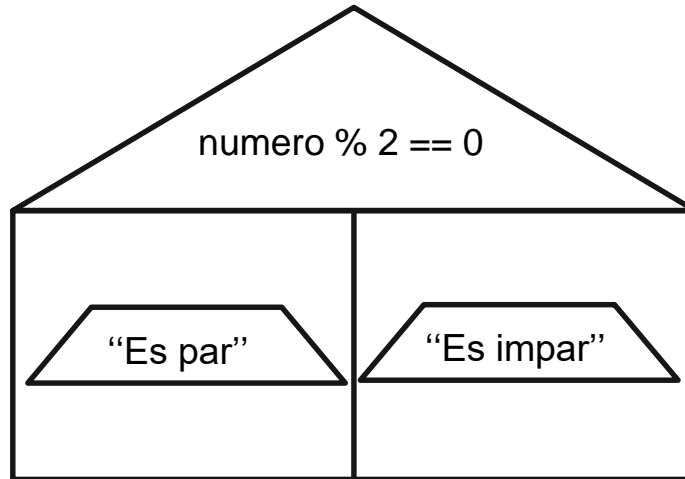
## Toma de Decisiones



**UTN.BA**  
UNIVERSIDAD TECNOLÓGICA NACIONAL  
FACULTAD REGIONAL BUENOS AIRES

# Selección simple: if

En diagrama de Lindsay:



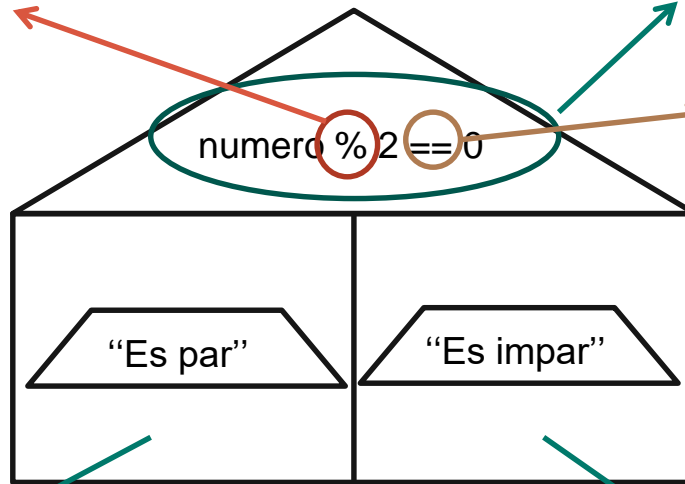
# Selección simple: if

En diagrama de Lindsay:

Resto de la división entera

Condición

Comparador de igualdad



Parte verdadera

Parte falsa



**UTN.BA**  
UNIVERSIDAD TECNOLÓGICA NACIONAL  
FACULTAD REGIONAL BUENOS AIRES

# Selección simple: if

En C++:

```
if ( numero % 2 == 0 ) {  
    cout << "Es par" << endl;  
} else {  
    cout << "Es impar" << endl;  
}
```



# Selección simple: if

En C++:

```
if ( numero % 2 == 0 ) {  
    cout << "Es par" << endl;  
} else {  
    cout << "Es impar" << endl;  
}
```

Las llaves se utilizan para delimitar un bloque de código.



# Selección simple: if

**En este caso**, también se puede escribir de esta manera:

```
if ( numero % 2 == 0 )  
    cout << "Es par" << endl;  
else  
    cout << "Es impar" << endl;
```

¿Qué falta?



**UTN.BA**  
UNIVERSIDAD TECNOLÓGICA NACIONAL  
FACULTAD REGIONAL BUENOS AIRES

# Selección simple: if

Y de esta manera:

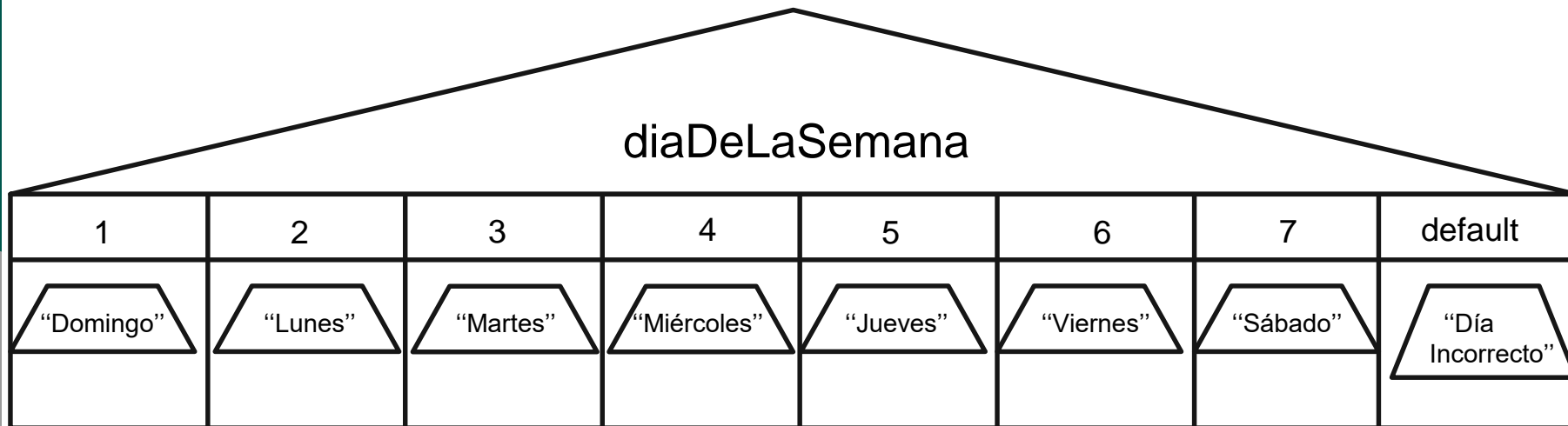
```
bool esPar;
```

```
esPar = numero % 2 == 0 ? 1 : 0;
```



# Selección múltiple: switch

En diagrama de Lindsay:



# Selección múltiple: switch

En C++:

```
switch (diaDeLaSemana) {  
  
    case 1: cout << "Domingo"; break;  
    case 2: cout << "Lunes"; break;  
    case 3: cout << "Martes"; break;  
    case 4: cout << "Miércoles"; break;  
    case 5: cout << "Jueves"; break;  
    case 6: cout << "Viernes"; break;  
    case 7: cout << "Sábado"; break;  
    default: cout << "Día Incorrecto"; break;  
  
}
```



