Introducción a la Programación

Tipos de Datos

Introducción a Pascal

Datos

Un dato es simplemente la representación de un objeto mediante símbolos manejables por una computadora.

- Los algoritmos se organizan en dos aspectos inseparables: ALGORITMO = ACCIONES + DATOS
  - Entrada: conjunto inicial de datos
  - Salida: conjunto de datos resultantes

- Las entradas antes de convertirse en salidas, pueden pasar por un sinfin de resultados provisionales antes de convertirse en un conjunto de datos resultantes.

Datos - Ejemplo

- Analicemos el siguiente hecho:

  El estudiante de nombre Pedro Velez de 22 años, tiene un promedio de 7.5

- Podemos tomar los siguientes datos:
  - Nombre: Pedro Velez -> texto
  - Edad: 22 -> número entero
  - Promedio: 7.5 -> número real

Tipos de datos

- Los datos se clasifican en TIPOS
- Son los diferentes dominios existentes. Ejemplo:
  - Edad, Año de Nacimiento, Número de multas
    - Tienen dominio **numérico**
  - Nombre, Dirección, Núm. Cedula,
    - Caen en el dominio de la **información tipo texto**
- Y las operaciones permitidas para dicho dominio

Un conjunto de valores **y** operaciones **definidas solo para esos valores**

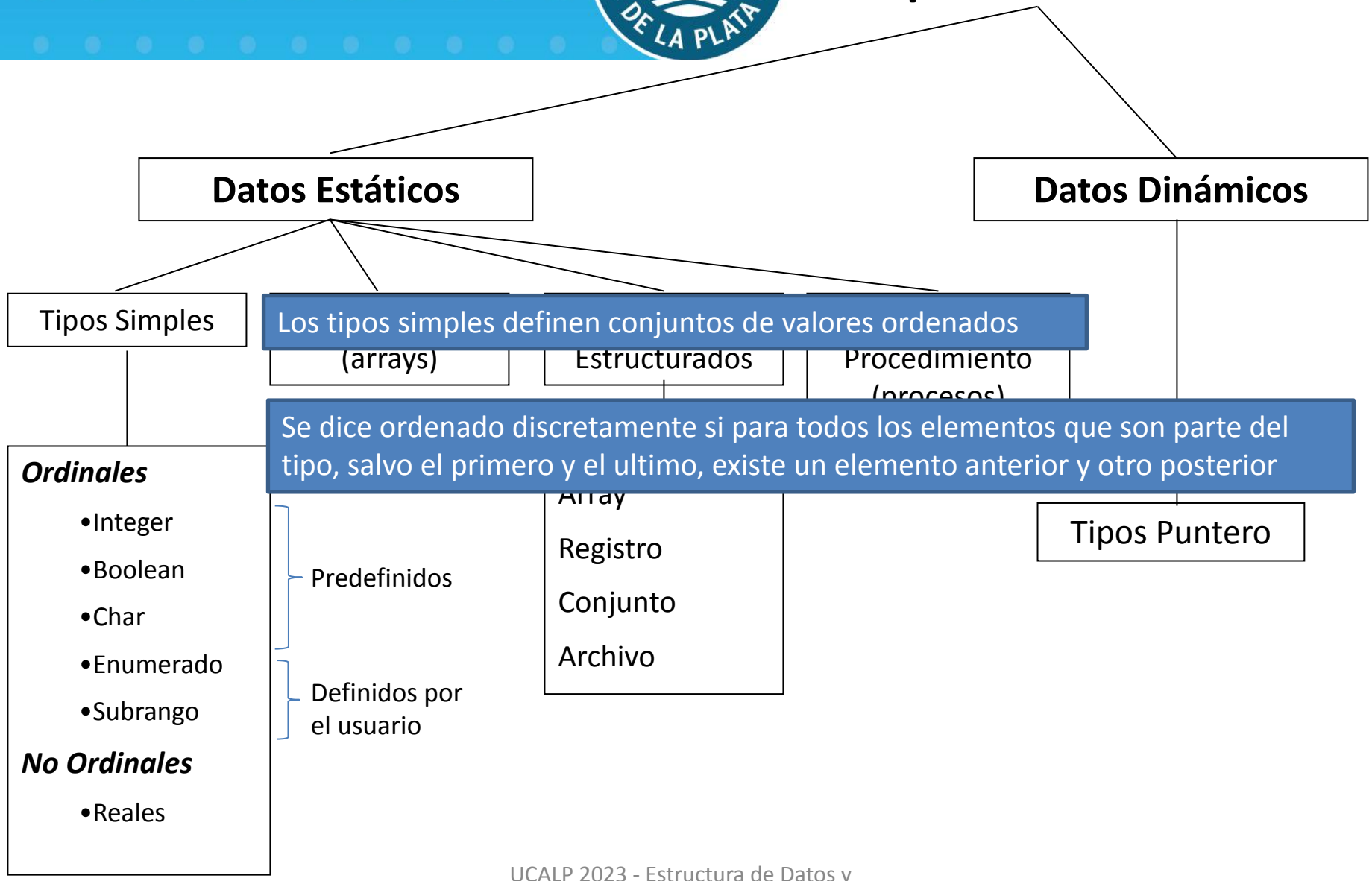
Tipos de datos

Un tipo de datos es el universo de valores con los que un programa trabaja, está dividido en colecciones organizadas llamadas tipos.

- Los tipos de datos se caracterizan por:
  - Un rango de valores posibles (dominio).
  - Un conjunto de operaciones realizables sobre ese tipo.
  - Su representación interna.

- La declaración de tipos de datos presenta las siguientes ventajas:
  - Mejores posibilidades de abstracción.
  - Límites preestablecidos sobre el dominio.
  - Detectar errores de operaciones.
  - Determinar cómo ejecutar las operaciones.

Tipos de datos



Tipo Integer (Entero)

Dominio:

Son todos los números enteros positivos y negativos. El rango de valores {-32.768..32768} (depende de cada lenguaje)

Operaciones:

(+) Suma
(-) Resta
(*) Multiplicación
(DIV) División entera
(MOD) Resto de la division entera

```
 7 div  2 ->  3
 7 div -2 -> -3
-7 div  2 -> -3
-7 div -2 ->  3
```

```
 7 mod  2 ->  1
 7 mod -2 ->  1
-7 mod  2 -> -1
-7 mod -2 -> -1
```

Funciones:

Abs: Valor absoluto del entero
Sqr: cuadrado del entero
Pred: entero predecesor
Succ: entero succesor

```
 Abs(-1)   -> 1
Sqr(2)     -> 4
Pred(4)    -> 3
Succ(4)    -> 2
```

Tipo Integer
Precedencia de operadores

Al igual que en las matemáticas la precedencia de las operaciones de multiplicación, división, resta y funciones predefinidas es mayor que las otras.

Cuando una operación se presenta ambigua (igual precedencia) se valoriza la operación asumiendo la asociatividad a la izquierda.

2 + 3 * Sqr(2) -> ?
2 + 3 * 4 ->
2 + 12 ->
14

7 div 2 * 2 -> ?
3 * 2 ->
6 (asociatividad a la izq)

(2 + 3) * 4 ->
5 * 4 ->
20 (Forzar orden de evaluación)

Tipo Real

Incluye valores numéricos con parte decimal. Las limitaciones en la representación son afectadas por dos aspectos:

- La magnitud de los valores incluidos $\pm 1{,}7 * 10^{38}$
- La precisión (hasta 10 cifras significativas)

Operaciones:

(+) Suma
(-) Resta
(*) Multiplicación
(/) División

Funciones:

Abs: Valor absoluto
Sqr: Cuadrado
SqRt: Raíz cuadrada
Sin: Seno
Cos: Coseno
ArcTan: Arcotangente
Ln: Logaritmo natural
Exp: Exponencial

Tipo Real & Tipo Integer

El lenguaje se muestra flexible a la hora de reconocer la inclusión de los números enteros dentro de los reales.

Toda operación que requiera expresamente una cantidad Real aceptará una de tipo entero, realizando automáticamente la conversión.

En caso contrario la conversión no se realiza automáticamente. Para ello se utilizan las funciones:

- Trunc: Truncamiento, eliminación de la parte decimal.
- Round: redondeo al entero más próximo.

Round(-3,6) ->	-4
Trunc(-99,9) ->	-99
-Round(99,9) ->	-100
-Round (-99,9) ->	100

Tipo Char (Carácter)

Dominio:

Incluye el juego de carácteres disponibles en la computadora . La mayoría de los lenguajes reconoce la codificación ASCII de 8 bits. Por lo que existen hasta 256 caracteres que se recogen de la siguiente tabla de códigos.

El tipo char o carácter se escribe entre apóstrofos. Ej: ‘A’

Operaciones

No existen operaciones interenas entre caracteres.

Funciones

- Pred: carácter anterior.
- Succ carácter sucesor.

Funciones de conversion entre char e integer

- Ord: número de orden del carácter en el juego adoptado.
- Chr: carácter asociado a un número de orden dado.

Pred(‘b’) ->	‘a’
Succ(‘b’) ->	‘c’
Ord(‘z’) ->	122
Ord(‘Z’) ->	90
Chr(64) ->	@

Tipo Char

El código ASCII – www.elCodigoASCII.com.ar

sigla en inglés de American Standard Code for Information Interchange
(Código Estadounidense Estándar para el Intercambio de Información)

Caracteres de control ASCII

DEC	HEX	Simbolo ASCII	
00	00h	NULL	(carácter nulo)
01	01h	SOH	(inicio encabezado)
02	02h	STX	(inicio texto)
03	03h	ETX	(fin de texto)
04	04h	EOT	(fin transmisión)
05	05h	ENQ	(enquiry)
06	06h	ACK	(acknowledgement)
07	07h	BEL	(timbre)
08	08h	BS	(retroceso)
09	09h	HT	(tab horizontal)
10	0Ah	LF	(salto de linea)
11	0Bh	VT	(tab vertical)
12	0Ch	FF	(form feed)
13	0Dh	CR	(retorno de carro)
14	0Eh	SO	(shift Out)
15	0Fh	SI	(shift In)
16	10h	DLE	(data link escape)
17	11h	DC1	(device control 1)
18	12h	DC2	(device control 2)
19	13h	DC3	(device control 3)
20	14h	DC4	(device control 4)
21	15h	NAK	(negative acknowle.)
22	16h	SYN	(synchronous idle)
23	17h	ETB	(end of trans. block)
24	18h	CAN	(cancel)
25	19h	EM	(end of medium)
26	1Ah	SUB	(substitute)
27	1Bh	ESC	(escape)
28	1Ch	FS	(file separator)
29	1Dh	GS	(group separator)
30	1Eh	RS	(record separator)
31	1Fh	US	(unit separator)
127	20h	DEL	(delete)

Caracteres ASCII imprimibles

DEC	HEX	Simbolo	DEC	HEX	Simbolo	DEC	HEX	Simbolo
32	20h	espacio	64	40h	@	96	60h	`
33	21h	!	65	41h	A	97	61h	a
34	22h	"	66	42h	B	98	62h	b
35	23h	#	67	43h	C	99	63h	c
36	24h	$	68	44h	D	100	64h	d
37	25h	%	69	45h	E	101	65h	e
38	26h	&	70	46h	F	102	66h	f
39	27h	'	71	47h	G	103	67h	g
40	28h	(	72	48h	H	104	68h	h
41	29h	)	73	49h	I	105	69h	i
42	2Ah	*	74	4Ah	J	106	6Ah	j
43	2Bh	+	75	4Bh	K	107	6Bh	k
44	2Ch	,	76	4Ch	L	108	6Ch	l
45	2Dh	-	77	4Dh	M	109	6Dh	m
46	2Eh	.	78	4Eh	N	110	6Eh	n
47	2Fh	/	79	4Fh	O	111	6Fh	o
48	30h	0	80	50h	P	112	70h	p
49	31h	1	81	51h	Q	113	71h	q
50	32h	2	82	52h	R	114	72h	r
51	33h	3	83	53h	S	115	73h	s
52	34h	4	84	54h	T	116	74h	t
53	35h	5	85	55h	U	117	75h	u
54	36h	6	86	56h	V	118	76h	v
55	37h	7	87	57h	W	119	77h	w
56	38h	8	88	58h	X	120	78h	x
57	39h	9	89	59h	Y	121	79h	y
58	3Ah	:	90	5Ah	Z	122	7Ah	z
59	3Bh	;	91	5Bh	[	123	7Bh	{
60	3Ch	<	92	5Ch	\	124	7Ch	|
61	3Dh	=	93	5Dh	]	125	7Dh	}
62	3Eh	>	94	5Eh	^	126	7Eh	~
63	3Fh	?	95	5Fh	_	elCodigoASCII.com.ar		

ASCII extendido

DEC	HEX	Simbolo	DEC	HEX	Simbolo	DEC	HEX	Simbolo	DEC	HEX	Simbolo
128	80h	Ç	160	A0h	á	192	C0h	└	224	E0h	Ó
129	81h	ü	161	A1h	í	193	C1h	┴	225	E1h	ß
130	82h	é	162	A2h	ó	194	C2h	┬	226	E2h	Ô
131	83h	â	163	A3h	ú	195	C3h	├	227	E3h	Ò
132	84h	ä	164	A4h	ñ	196	C4h	─	228	E4h	õ
133	85h	à	165	A5h	Ñ	197	C5h	┼	229	E5h	Õ
134	86h	å	166	A6h	ª	198	C6h	ã	230	E6h	µ
135	87h	ç	167	A7h	º	199	C7h	Ã	231	E7h	þ
136	88h	ê	168	A8h	¿	200	C8h	╚	232	E8h	Þ
137	89h	ë	169	A9h	®	201	C9h	╔	233	E9h	Ú
138	8Ah	è	170	AAh	¬	202	CAh	╩	234	EAh	Û
139	8Bh	ï	171	ABh	½	203	CBh	╦	235	EBh	Ù
140	8Ch	î	172	ACh	¼	204	CCh	╠	236	ECh	ý
141	8Dh	ì	173	ADh	¡	205	CDh	═	237	EDh	Ý
142	8Eh	Ä	174	AEh	«	206	CEh	╬	238	EEh	¯
143	8Fh	Å	175	AFh	»	207	CFh	¤	239	EFh	´
144	90h	É	176	B0h	░	208	D0h	ð	240	F0h	≡
145	91h	æ	177	B1h	▒	209	D1h	Ð	241	F1h	±
146	92h	Æ	178	B2h	▓	210	D2h	Ê	242	F2h	‗
147	93h	ô	179	B3h	│	211	D3h	Ë	243	F3h	¾
148	94h	ò	180	B4h	┤	212	D4h	È	244	F4h	¶
149	95h	ò	181	B5h	Á	213	D5h	ı	245	F5h	§
150	96h	û	182	B6h	Â	214	D6h	Í	246	F6h	÷
151	97h	ù	183	B7h	À	215	D7h	Î	247	F7h	¸
152	98h	ÿ	184	B8h	©	216	D8h	Ï	248	F8h	°
153	99h	Ö	185	B9h	╣	217	D9h	┘	249	F9h	¨
154	9Ah	Ü	186	BAh	║	218	DAh	┌	250	FAh	·
155	9Bh	ø	187	BBh	╗	219	DBh	▀	251	FBh	¹
156	9Ch	£	188	BCh	╝	220	DCh	▄	252	FCh	³
157	9Dh	Ø	189	BDh	¢	221	DDh	¦	253	FDh	²
158	9Eh	×	190	BEh	¥	222	DEh	Ì	254	FEh	■
159	9Fh	ƒ	191	BFh	┐	223	DFh	▀	255	FFh	

Tipo Boolean

Dominio:

Este tipo proviene del álgebra de Bool. Su dominio obtiene dos valores lógicos: True & False (Verdadero y Falso). Es útil para las tareas de control de bucles y selecciones.

Operaciones:

Not: Negación lógica (con la precedencia más alta)

And: conjunción lógica (precedencia multiplicativa)

Or: disyunción lógica (precedencia aditiva)

Su funcionamiento viene dado por la sigueinte tabla de verdad:

A	B	A and B	A or B	not A
False	False	False	False	True
False	True	False	True	True
True	False	False	True	False
True	True	True	True	False

Tipo Boolean
Operadores relacionales

Son las operaciones binarias de comparación

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

Las operaciones relacionales permiten comparar cualquiera de los tipos básicos, resultando de ello un valor lógico:

A	B	A = B	A <> B	A <= B
False	False	True	False	True
False	True	False	True	True
True	False	False	True	False
True	True	True	False	True

En cuanto a la comparación de datos no numéricos el símbolo < o > significa anterior o posterior.

En el caso del tipo char, se considera el orden establecido por la tabla ASCII.

En el caso del tipo bool, se considera false anterior a true.

Tipo Strings (Cadenas)

Un String (cadena) es una secuencia de caracteres que tiene una longitud máxima de 255 caracteres. Los caracteres que componen el String se delimitan con apostrofes.

Operaciones

(+): Concatenación

=, >, <, <>: Comparación por orden lexicográfico

Tipo Strings (Cadenas)

Funciones

- **concat(x1, x2, xn):** concatena string.
- **length(X):** longitud del string.
- **copy(X, 2, 5):** extrae un string de otro string.
- **pos(X, Y):** determina si un string esta incluido dentro de otro.
- **delete(X, 2, 5):** suprime el número de caracteres que le digamos de un string a partir de la posición que le indiquemos.
- **insert(X, Y, 3):** inserta un string dentro de otro a partir de la posición que le indicamos.
- **upcase(X):** devuelve el string todo en mayúscula
- **strToInt(X):** devuelve el valor de X en tipo Integer. (Idem strToFloat)
- **intToStr(valor):** devuelve el string del número. (Idem FloatToStr)

Práctica

Expresar cuáles de las siguientes afirmaciones son verdaderas o falsas:

- 2 * 2 div 4 = 2 * (2 div 4)
- Suma: integer
  Suma := 2 * 240 + 10000 * 3
- Resultado: integer
  Resultado := 2 / 240 + 10000 * 3
- Resul: real
- Resul := 35000 + 375
- 5 + (5 mod 2) = 7
- Succ('1') + 276 = 277
- Pred('J') > Succ('I')
- A := true
  A = Succ(False)
- variableA: real
  variableA := 10;
- Reaultado: integer
  Resultado := Ord('Z');

Resumen

Hasta aquí vimos:

- Datos
- Tipos de Datos
- Tipo Integer
- Tipo Boolean
- Tipo Char
- Tipo Float
- Tipo String

Teorema de la programación estructurada:

Todo algoritmo propio puede ser expresado en sólo tres tipos de estructuras: secuencial, condicional y repetitiva.

Un algorimo se define como propio si:

- Tiene un único punto de entrada y un único punto de salida.
- Todas las sentencias son alcanzables: no hay código muerto.
- No hay bucles infinitos.
- No hay ambigüedades.
- Todos los posibles caminos llevan desde el punto de entrada al de salida.
- El algoritmo debe producir al menos una salida o efecto.

Constantes y Variables

Una constante es un tipo de dato (valor numérico o texto) que permanece invariable, sin posibilidad de cambiar su valor durante el curso de un programa.

(Por convención las constantes se escriben con mayusculas)

```
Const
    TEMPCONGELAAGUA = 0;
    TEMPHIERVEAGUA = 100;
    PI = 3,14;
    MENOSPI = -Pi;
    PRIMERALETRA = 'A';
```

Una variable es un tipo de dato, que como su nombre lo indica, puede cambiar su valor durante el transcurso de un programa.

```
Var
    indice: integer;
    altura, peso: real;
    esPrimo: boolean;
    Inicial: chat;
```

Una variable corresponde a un área reservada de la memoria principal de la computadora.

- Una variable está asociada a un tipo de datos determinado.
- En función al tamaño del tipo de datos se determina la cantidad de bytes necesarios para almacenarla.

Introducción a Pascal

El lenguaje Pascal fue concebido por Niklaus Wirth en 1968 y definido en 1970 en el instituto Politécnico de Zurich para enseñar la programación a sus alumnos.

Actualmente es uno de los lenguajes más usados en las universidades de muchos países del mundo. Gracias a esta difusión, junto con los compiladores de este lenguaje, se han desarrollado potentes entornos de programación de gran calidad y bajo precio.

Pascal toma el nombre del matemático frances Blaise Pascal (1623-1662) que en 1642 inventó la primera máquina de calcular para ayudar a su padre en su trabajo de tasador de impuestos.

Características:

- Lenguaje de programación estructurado.
- Lenguaje fuertemente tipado (valores y expresiones).
- El código está dividido en funciones y procedimientos.
- El tipo de todas las variables debe ser declarado antes de poder utilizarse.

Estructura de un programa en Pascal

Program ProgramaDeEjemplo

Cabecera del programa

Const
    constante1 = valor;
    constanteN = otroValor;

Definición de constantes

Type
    nombre1=tipo;
    nombre2=otroTipo;

Definición de tipos de datos definidos por el usuario

Var
    variable1: tipo;
    variableN: tipo;

Definición de variables

Procedure
Function

Definición de procedimientos y funciones

Begin
….
End.

Cuerpo del programa

```
Program AreaCirculo

Const
        Pi = 3,14;

Var
        radio, area: real;

Begin
        Write("Cual es el radio?:  ");
        Readln(radio);
        area = Pi * Sqr(radio);
        WriteLn('Area = ', area);
End.
```

De pseudocódigo a Pascal

Se desea realizar un algoritmo que pida ingresar dos numeros enteros por teclado y muestre en pantalla la suma de ambos

Pseudocódigo

```
Inicio
   Escribir (“Ingrese el primer número”)
   Leer(numero1)
   Escribir(“Ingrese el segundo número”)
   Leer(numero2)
   Suma = numero1 + numero2
   Escribir(suma)
Fin
```

Pascal

```
Program Suma
Var
   numero1, numero2, suma: integer;
Begin
   Write(“Ingrese el primer número”)
   Read(numero1);
   Write(“Ingrese el primer número”)
   Read(numero2);
   suma := numero1 + numero2;
   Write(suma);
End.
```

Leer(X)	Read(X)
Escribir(X)	Write(X)

De pseudocódigo a Pascal

Asignación

Pseudocódigo

```
Variable = 10
```

Pascal

```
Variable := 10;
```

Ejemplos

```
base := 10;
altura := 11,4;
area := base * altura / 2;
contador := contador + 1;
acumulador = acumulador + valor;
```

De pseudocódigo a Pascal

Repetición

Supongamos las siguientes instrucciones:

Sumarle 10 veces a X el valor de Y, luego mostrar en pantalla el valor de la variable X

Pseudocódigo

```
X = 0
Y = 1
para I = 1 hasta 10 hacer
  X = X + Y
fin para
escribir(X)
```

Pascal

```
X := 0;
Y := 1;
for i := 1 to 10 do
begin
  X := X + Y;
end;
write(X);
```

De pseudocódigo a Pascal

Decisión

Supongamos las siguientes instrucciones:

Leer un número por teclado e imprimir en pantalla si es negativo o positivo

Pseudocódigo

```
leer(X)
si X <> 0 entonces
  si X>0 entonces
     escribir (“Es positivo”)
   sino
     escribir (“Es negativo”)
   fin si
sino
    escribir (“El número es 0”)
fin si
```

Pascal

```
Read(X);
if X <> 0 then
  if X>0 then
     escribir (“Es positivo”)
   else
     escribir (“Es negativo”);
   end;
else
    write (“El numero es 0”);
end;
```

De pseudocódigo a Pascal

Selección

Supongamos las siguientes instrucciones:

Leer una edad por teclado e imprimir en pantalla si es niño, adolecente, adulto o mayor

Pseudocódigo

```
leer(edad)
seleccionar (edad)
caso edad < 10
  escribir (“Niño”)
caso (edad > 10 y edad < 21)
  escribir (“Adolescente”)
caso (edad > 20 y edad < 65)
  escribir (“Adulto”)
caso edad < 64
  escribir (“Mayor”)
en otro caso
  escribir (“No esta definido”)
fin seleccionar
```

Pascal

```
read (edad)
case (edad) of
0..9:
  write(“Niño”)
10..20:
  write(“Adolescente”)
21..64:
  write(“Adulto”)
65..100:
  write(“Mayor”)
else
  write(“No esta definido”)
end;
```

De pseudocódigo a Pascal

Iteración

Supongamos la siguiente instrucción:

Calcular las potencias de números ingresados por teclado siempre que sean positivos e imprimir en pantalla

Pseudocódigo

```
leer(numero)
mientras numero > 0
   escribir (numero * numero)
   leer(numero)
fin mientras
```

Pascal

```
read(numero);
while (numero > 0) do
begin
   write(numero * numero);
   read(numero);
end;
```

Práctica

Antes de codificar, leer atentamente el enunciado apuntado en lenguaje natural y especifique cuáles son las entradas y salidas del problema y qué proceso de transformación se realiza.

1. Calcular el factorial de un número leído por teclado.
2. Calcular el índice de masa corporal de una persona (IMC = peso (kg) / talla² (mts)²).
3. Imprima los números del 0 al 100 en pantalla en orden creciente.
4. Imprimir la suma de los números de 1 a 100 en pantalla.
5. Imprimir los números que hay desde 1 hasta un número ingresado por t eclado.
6. Imprimir los números del 0 al 100 en pantalla en orden decreciente.
7. Imprimir la cantidad de palabras ingresadas por teclado hasta ingresar l a palabra “fin”.
8. Imprimir una frase en pantalla leyendo palabra por palabra hasta ingresar la palabra “fin”