

The image features a white square frame centered on a dark blue background. Inside the frame, the letters "UAP" are displayed in a bold, white, sans-serif font. The letter "A" is stylized with three horizontal white lines passing through its center, creating a sense of motion or a modern design.

UAP

ALGORITMOS Y ESTRUCTURAS DE DATOS

ESTRUCTURAS DE DATOS - PYTHON

CONCEPTO

TIPO DE DATO

Atributo del dato. Una forma de clasificar la información para su posterior procesamiento.

Generalmente definida en la documentación de un lenguaje formal.

Todos los lenguajes de programación tratan de alguna forma con tipos de datos.

EJEMPLOS

Numéricos

Una forma de representación de los números, ya sean enteros, decimales o imaginarios. En ocasiones fechas también.

Cadena de caracteres

Strings. Una sucesión de caracteres alfanuméricos. UTF8 incluye simbología de otros idiomas y culturas

Lógicos

Verdadero, falso y negación

Listas

Agrupación o conjunto de datos. Varían en longitud y dimensión

EJEMPLOS

Estructuras personalizadas

Agrupación de varias variables diferentes y definidas bajo un nombre personalizado. Sin funcionalidades.

Clasens y Objetos

IDEM pero sumando funcionalidad y otras características

Literales

Información escrita en el algoritmo o programa que no cambia nunca

Constantes

Variables cuyo contenido no cambia nunca durante la ejecución

CONCEPTO

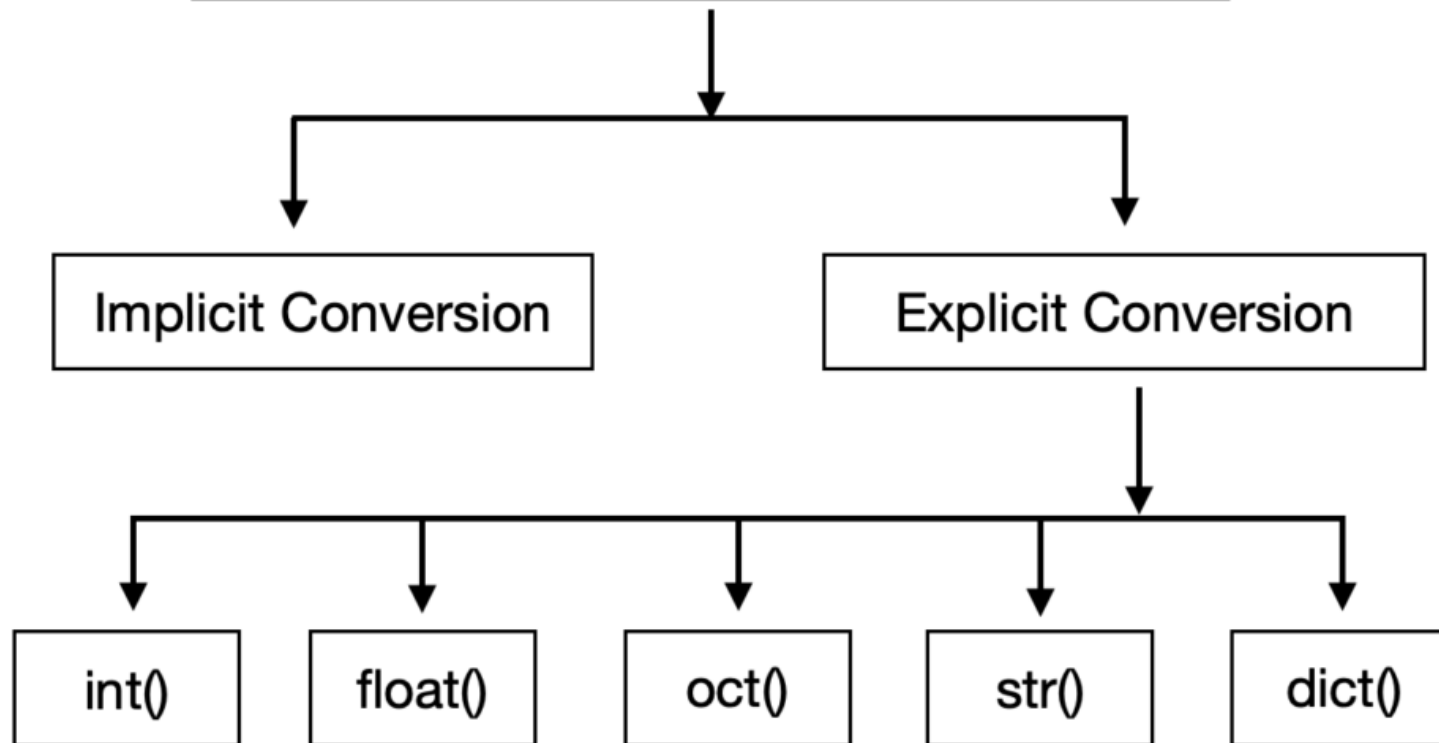
LENGUAJES FUERTEMENTE TIPADOS O DÉBILMENTE TIPADOS

Se dice que un lenguaje es fuertemente tipado cuando no permite el uso de la variable con un tipo definido como si fuera de otro tipo.

Herramientas de conversión o inferencia de tipos (casting)

Python es fuertemente tipado

Type Conversion in Python

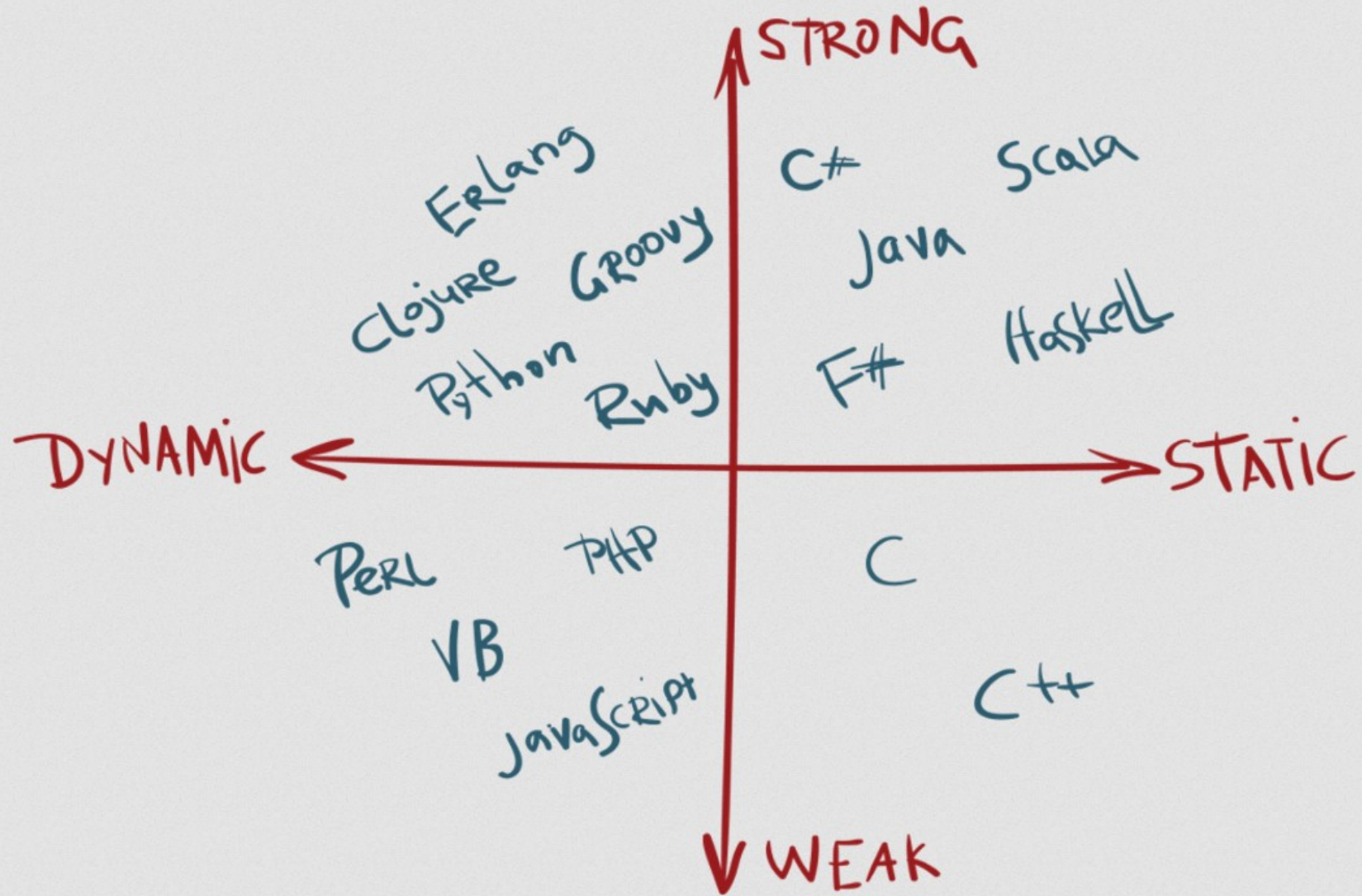


CONCEPTO

LENGUAJES TIPADO DINÁMICO o ESTÁTICO

Análisis y asignación automático del tipo de dato en la instanciación de una variable.

Python es de tipado dinámico



CONCEPTO

LENGUAJES INTERPRETADOS o COMPILADOS

Compilar: proceso por el cual se traduce el código fuente de alto nivel a un código de bajo nivel que luego es ejecutado por el ordenador.

Interpretar: proceso de tomar el código fuente de alto nivel y de ejecutar en un ordenador.

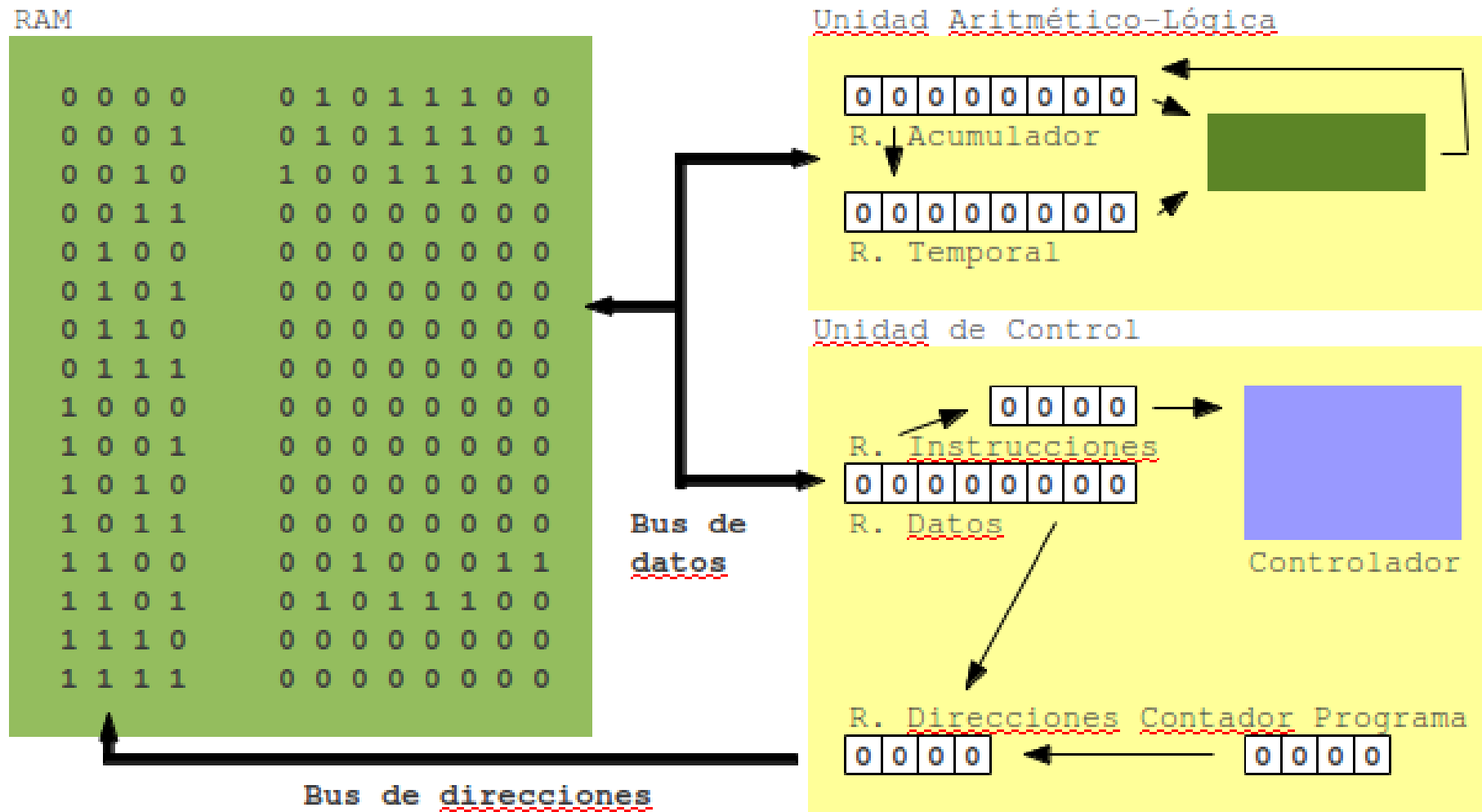
Discusión de pro y contras...

Python es interpretado

Comentarios en Python

- Un comentario es una texto que no se procesa como código Python.
- Es muy útil para documentar en el código mismo
- Comentarios de una sola línea comienzan con # hasta el final de la misma.
- Comentarios de bloques 3 comillas dobles de apertura y cierre: `"""`
- Una buena práctica es que todo archivo tenga un comentario como encabezado sobre el autor, fecha y contenido del mismo.

Variable



Variables Python

- Una variable es el nombre que se le da a un espacio de memoria en donde guardar datos.
- Siempre se deben inicializar en algún valor antes de usarlas.
- Su contenido cambia a lo largo de un programa
- Debe comenzar con una letra o _
- Luego se puede mezclar letras y números
- Convención de estilo: snake_case (no CamelCase)
- Sensible a mayúsculas y minúsculas

edad_alumno
Edad_alumno

Literales

- Un literal es un dato que se escribe en el código del programa y su contenido no cambia a lo largo de la ejecución

```
edad_alumno = 20  
nombre_alumno = "Juan"
```

Constantes

- Es una variable que por convención su contenido no cambia a lo largo de la ejecución y se escribe con todas mayúsculas

$\text{PI} = 3.1416$

Números enteros

$Z = \{..., -2, -1, 0, 1, 2, ...\}$

`a = 1234 // número base decimal`

`a = -123 // un número negativo base10`

`a = 0o123 // número octal (equivale a 83 decimal)`

`a = 0x1A // número hexadecimal (equivale a 26 decimal)`

`a = 0b11111111 // número binario (equivale al 255 decimal)`

`a = int('33')`

Números coma flotante

$R = \{..., -2, -1, 0, 1, 2, ...\}$

`a = 12.34`

`a = 10e-23`

`a = float('1.66')`

Operaciones matemáticas básicas

+	suma	$X + Y$
-	resta	$X - Y$
*	multiplicación	$X * Y$
/	división	X / Y
//	división entera	$X // Y$
%	módulo	$X \% Y$
**	potencia	$X ** Y$
abs	valor absoluto	abs(x)

CONCEPTO

ORDEN DE PRECEDENCIA DE OPERADORES

Es el orden de prioridad y secuencia en que se ejecutan o calculan los operadores de una expresión.

Operator	Description
()	Parentheses (grouping)
**	Exponentiation
~x	Bitwise nor
+x,-x	Positive , negative (unary +,-)
*,/,//,%	Multiplication , division , floor division , remainder
+,-	Addition , subtraction
&	Bitwise and
^	Bitwise XOR
	Bitwise OR
<,<=,>,>=,<>,!','=', ,is,isnot	Comparisons (Relational operators),identity operators
not x	Boolean NOT
And	Boolean AND
or	Boolean OR

Highest

Lowest

Valores lógicos

`B = bool(False/True)`

`B = True/False`

False: `bool(0), bool(0j), bool(''),
bool(None), bool([])`

True: `bool(1), bool(-1), bool('casa'),
bool([2])`

Operaciones lógicas

or	Disyunción	X or Y
and	Conjunción	X and Y
not	Negación	Not X

CONCEPTO

CORTOCIRCUITO LÓGICO

Método de optimización que evita la evaluación de toda una expresión de forma completa

Se corta una evaluación de expresiones al primer and que tenga por resultado false y no se evalúa el resto.

Se corta una evaluación de expresiones al primer or que tenga por resultado true y no se evalúa el resto.

Ejemplo: `if student and student.name.length > 50`

CONCEPTO

LEY de MORGAN

Reglas de transformación de disyunción a conjunción o viceversa manteniendo la misma tabla de verdad de la función booleana.

Leyes de De Morgan

"La negación de la conjunción es la disyunción de las negaciones."

"La negación de la disyunción es la conjunción de las negaciones."

Expresadas en lenguaje formal

$$\neg(P \wedge Q) \Leftrightarrow (\neg P) \vee (\neg Q)$$

$$\neg(P \vee Q) \Leftrightarrow (\neg P) \wedge (\neg Q)$$

Donde:

- \neg es el operador de negación
- \wedge es el operador de conjunción
- \vee es el operador de disyunción
- \Leftrightarrow es un símbolo que significa "puede ser reemplazado en una prueba lógica"



Comparaciones

>	Mayor	$X > Y$
>=	Mayor o igual	$X \geq Y$
<	Menor	$X < Y$
<=	Menor o igual	$X \leq Y$
==	Igual	$X == Y$
!=	Distinto	$X != Y$
is	Es objeto idéntico	$X \text{ is } Y$
is not	No es objeto idéntico	$X \text{ is not } Y$

Cadenas de caracteres

A = 'un texto'

B = 'el nro 22 aquí es texto y no número entero'

C = "se puede usar con doble comilla también"

D = "en especial si quiero 'poner' comillas simples dentro"

E = 'sino hay que usar así para escapar una comilla
"O\'Reilly" Books'

Caracteres especiales:

\n: salto de línea

\t: tabulador

\\: barra invertida

CONCEPTO

CONCATENAR CADENAS DE CARACTERES

El resultado de sumar 2 cadenas será una cadena con todos los caracteres de ambas.

Se puede concatenar combinando variables y literales

Todos los operandos de la concatenación deben ser del mismo tipo de dato. De ser necesario hay que efectuar conversión de tipos de dato a `str()`

`nombre_completo = apellido + ", " + nombre`

CONCEPTO

INTERPOLAR CADENAS DE CARACTERES

El resultado de combinar una cadena literal o variable del mismo tipo en lugares específicos de una cadena.

Se puede concatenar combinando variables y literales

Cadenas de caracteres

Ejemplos de interpolación

`"{} pueden ser {}".format("strings", "interpolados")`


`"{0} sé ligero, {0} sé rápido, {0} brinca sobre la {1}".format("Jack", "vela")`
#=> "Jack sé ligero, Jack sé rápido, Jack brinca sobre la vela"

`"{nombre} quiere comer {comida}".format(nombre="Bob", comida="lasaña")`
#=> "Bob quiere comer lasaña"

`nombre = 'Bob'`

`comida = 'Lasaña'`

`f'{nombre} quiere comer {comida}' #=> "Bob quiere comer lasaña"`



```
>>> help('keywords')
```

Here is a list of the Python keywords. Enter any keyword to get more help.

False	def	if	raise
None	del	import	return
True	elif	in	try
and	else	is	while
as	except	lambda	with
assert	finally	nonlocal	yield
break	for	not	
class	from	or	
continue	global	pass	



UAP.EDU.AR | @**UAP**ARGENTINA