

Fundamentos de Programación.

Primer Curso de ASIR.

UT 02.01 - Elementos de un lenguaje informático.

UT 02.01 - Elementos de un lenguaje informático.

1.- El lenguaje de programación Python.



¿Por qué Python?.

- Fácil de aprender.
- Rápido para generar código nuevo.
- Fácil de obtener, instalar e implementar.
- Gratuito, abierto y multiplataforma.

UT 02.01 - Elementos de un lenguaje informático.

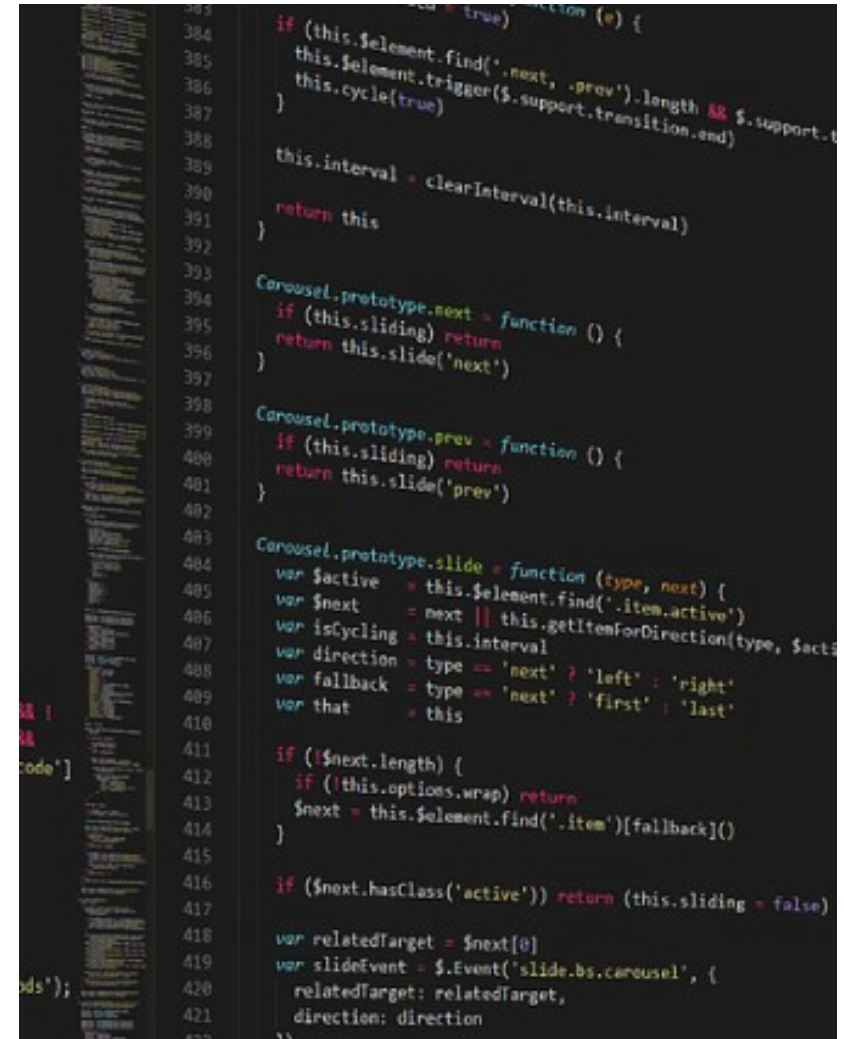
2.- Elementos de un lenguaje de programación.

Elementos de un lenguaje de programación.

- Un lenguaje de programación consta de:
 - **Alfabeto.** Conjunto de **símbolos** utilizados para construir palabras.
 - **Diccionario.** Conjunto de **palabras** que el idioma ofrece a sus usuarios.
 - **Sintaxis.** Conjunto de **reglas** utilizadas para determinar si una determinada cadena de palabras.
 - **Semántica.** Conjunto de reglas que determinan si cierta frase tiene sentido.

El entorno de desarrollo es deseable que tenga.

- Editor para escribir código con asistencia contextual.
- Consola para ejecutar código y controlar su ejecución.
- Depurador para lanzar el código paso a paso e inspeccionarlo.



```
384 if (this.$element.find('.next, .prev').length && $.support.transition.end) {
385   this.$element.trigger($.support.transition.end)
386   this.cycle(true)
387 }
388
389 this.interval = clearInterval(this.interval)
390
391 return this
392 }
393
394 Carousel.prototype.next = function () {
395   if (this.sliding) return
396   return this.slide('next')
397 }
398
399 Carousel.prototype.prev = function () {
400   if (this.sliding) return
401   return this.slide('prev')
402 }
403
404 Carousel.prototype.slide = function (type, next) {
405   var $active = this.$element.find('.item.active')
406   var $next = next || this.getItemForDirection(type, $active)
407   var isCycling = this.interval
408   var direction = type == 'next' ? 'left' : 'right'
409   var fallback = type == 'next' ? 'first' : 'last'
410   var that = this
411
412   if (!$next.length) {
413     if (!this.options.wrap) return
414     $next = this.$element.find('.item')[fallback]()
415   }
416
417   if ($next.hasClass('active')) return (this.sliding = false)
418
419   var relatedTarget = $next[0]
420   var slideEvent = $.Event('slide.bs.carousel', {
421     relatedTarget: relatedTarget,
422     direction: direction
423   })
424   this.$element.trigger(slideEvent)
425   if (slideEvent.isDefaultPrevented()) return
426
427   this.sliding = true
428   $(this.$element).trigger('sliding.bs.carousel', {
429     relatedTarget: relatedTarget,
430     direction: direction
431   })
432   if (isCycling) clearInterval(this.interval)
433   this.interval = setInterval($.proxy(this.cycle, this), this.interval)
434   this.$element.trigger('slid.bs.carousel')
435 }
```

UT 02.01 - Elementos de un lenguaje informatico.

3.- Las palabras claves de Python.

- Las palabras reservadas son las **instrucciones y expresiones recogidas en el diccionario de Python.**
- **No se pueden usar como nombres de variables o funciones.**
- En la consola de Python podemos ver las palabras reservadas de la forma siguiente:

and	del	from	not	while
as	elif	global	or	with
assert	else	if	pass	yield
break	except	import	print	
class	exec	in	raise	
continue	finally	is	return	
def	for	lambda	try	

```
Python 3.8.10 (default, Jun  2 2021, 10:49:15)
[GCC 9.4.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import keyword
>>> keyword.kwlist
['False', 'None', 'True', 'and', 'as', 'assert', 'async', 'await', 'break', 'class',
'continue', 'def', 'del', 'elif', 'else', 'except', 'finally', 'for', 'from',
'global', 'if', 'import', 'in', 'is', 'lambda', 'nonlocal', 'not', 'or', 'pass',
'raise', 'return', 'try', 'while', 'with', 'yield']
```

UT 02.01 - Elementos de un lenguaje informático.

4.- Tipos de Datos.

4.1.- Tipos de datos simples.

4.1.1.- Números.

- **int.** Entero de tamaño pequeño y medio.
- **long.** Entero de tamaño grande.
- **float.** Coma flotante de doble precisión.
- **complex.** Números complejos.

```
Python 3.8.10 (default, Jun  2 2021, 10:49:15)
[GCC 9.4.0] on linux
Type "help", "copyright", "credits" or "license"
for more information.
>>> entero = 238
>>> entero
238
>>> numeroGrande = 346328746567
>>> numeroGrande
346328746567
>>> decimal = 23.76
>>> decimal
23.76
>>> decimal = 3E4
>>> decimal
30000.0
>>> pi = 3.141592
>>> pi
3.141592
```

UT 02.01 - Elementos de un lenguaje informatico.

4.- Tipos de Datos.

4.1.- Tipos de datos simples.

4.1.2.- Cadenas de texto.

- Conjunto de caracteres almacenados en una variable.
- Se delimita mediante comillas dobles o simples.
- Se admiten caracteres especiales tales como:
 - `\n`. Retorno no carro.
 - `\t`. Tabulador.

```
Python 3.8.10 (default, Jun  2 2021, 10:49:15)
[GCC 9.4.0] on linux
Type "help", "copyright", "credits" or "license"
for more information.
>>> nombre = "Carlos"
>>> nombre
'Carlos'
>>> nombre = 'Jose Perez'
>>> nombre
'Jose Perez'
```

UT 02.01 - Elementos de un lenguaje informatico.

4.- Tipos de Datos.

4.1.- Tipos de datos simples.

4.1.3.- Variables booleanas.

- Toman valores binarios definidos como **true** (verdadero) o **false** (falso).
- Nos permite caracterizar estados de las condiciones de programa indicando si se cumplen determinadas condiciones.

```
Python 3.8.10 (default, Jun  2 2021, 10:49:15)
[GCC 9.4.0] on linux
Type "help", "copyright", "credits" or "license"
for more information.
>>> condicion = True
>>> condicion
True
>>> condicion = False
>>> condicion
False
>>> 3>2
True
```

UT 02.01 - Elementos de un lenguaje informático.

4.- Tipos de Datos.

4.2.- Tipos de datos compuestos.

- Conjunto de elementos que pueden ser accesibles de forma individual o colectiva.
- Pueden ser del mismo de dato o de datos diferentes.
- Los elementos están **Indexados**. Se pueden acceder mediante el nombre la lista y su número de orden **empezando en cero**.
También se puede acceder a un **rango de elementos** (sublista).
- Hay tres tipos de datos compuestos: **listas**, **tuplas** y **diccionarios**.

```
Python 3.8.10 (default, Jun 2 2021, 10:49:15)
[GCC 9.4.0] on linux
Type "help", "copyright", "credits" or "license"
for more information.
>>> lista = [7,1,5,2,8]
>>> lista[2]
5
>>> lista = lista + [6,9]
>>> lista
[7, 1, 5, 2, 8, 6, 9]
>>> lista[1:3]
[1, 5]
>>> lista[1:4]
[1, 5, 2]
>>> lista[-2]
6
```


UT 02.01 - Elementos de un lenguaje informático.

5.- Operadores.

5.1.- Aritméticos.

Operador	Resultado
+	Suma.
-	Resta.
*	Producto.
/	División.
//	Div. entera (truncado).
%	Resto de la división entera.
**	Potencia.

```
Python 3.8.10 (default, Jun  2 2021, 10:49:15)
[GCC 9.4.0] on linux
Type "help", "copyright", "credits" or "license"
for more information.
>>> 8/3
2.6666666666666665
>>> 8//3
2
>>> 15%9
6
>>> 2**3
8
```

UT 02.01 - Elementos de un lenguaje informatico.

5.- Operadores.

5.2.- Aritméticos.

Operador	Resultado
==	Igual que
!=	Distinto.
<	Menor.
>	Mayor.
<=	Menor o igual.
>=	Mayor o igual.

```
Python 3.8.10 (default, Jun  2 2021, 10:49:15)
[GCC 9.4.0] on linux
Type "help", "copyright", "credits" or "license"
for more information.
>>> print(4<6)
True
>>> print(3==3)
True
>>> print(2<=2)
True
>>> print(2>=3)
False
```

UT 02.01 - Elementos de un lenguaje informatico.

5.- Operadores.

5.3.- Lógicos.

Operador	Resultado
and	Y lógica.
or	O lógica.
not	No lógica.

```
Python 3.8.10 (default, Jun  2 2021, 10:49:15)
[GCC 9.4.0] on linux
Type "help", "copyright", "credits" or "license"
for more information.
>>> True and True
True
>>> True and False
False
>>> True or False
True
>>> not True
False
```

UT 02.01 - Elementos de un lenguaje informatico.

6.- Variables de programa.



- **Las variables permiten acceder a los datos almacenados por el programa mediante un nombre.**
- Las variables son de un determinado tipo; ya sea de tipo de dato simple o compuesto.
- Los nombres de las variables deben cumplir las siguientes reglas:
 - Pueden contener **mayúsculas, minúsculas, dígitos y el guion bajo (“_”)**.
 - **Deben comenzar con una letra o “_”**.
 - **Son sensibles a mayúsculas.** No es lo mismo el nombre de variable **resultado** que **RESULTADO**.
 - No pueden ser una palabra reservada.
 - Se permiten varios alfabetos.
 - Se crean en realidad cuando se les da un valor.
 - No se deben usar antes de crearlas.
- **Las variables se crean en el momento de que se le asigna un valor, tampoco es necesario especificar su tipo.** Este tipo de creación de variables se denomina **declaración al vuelo**.

UT 02.01 - Elementos de un lenguaje informático.

7.- Conversiones de tipo (casting).

- El tipo de una variable se selecciona por el programa a la hora de definirla.
- No obstante, hay casos en que el resultado debe de tener un tipo de dato diferente para sucesivas operativas del programa.
- Esta operación se denomina **Casting**.
- Se definen tres funciones de casting:
 - **int()**. Convierte (si es posible realizar la conversión) el valor dado como parámetro a un número entero.
 - **str()**. Convierte el valor dado como parámetro a una cadena de texto.
 - **float()**. Convierte (si es posible realizar la conversión) el valor introducido como parámetro a un número decimal.

```
Python 3.8.10 (default, Jun 2 2021, 10:49:15)
[GCC 9.4.0] on linux
Type "help", "copyright", "credits" or "license"
for more information.
>>> int(5.2)
5
>>> str(3.141592)
'3.141592'
>>> float("459.23")
459.23
```

UT 02.01 - Elementos de un lenguaje informático.

8.- Introducción de datos teclado. La función `input()`.

- La función **`input()`** permite solicitar un dato por teclado al usuario del programa.
- La sintaxis de uso es la siguiente:
`variable = input("mensaje")`
- Por defecto **devuelve una cadena de caracteres**. En el caso de que se necesite un valor de otro tipo es necesario realizar un **casting**.

```
Python 3.8.10 (default, Jun 2 2021, 10:49:15)
[GCC 9.4.0] on linux
Type "help", "copyright", "credits" or "license"
for more information.
>>> a = input("Valor de la variable a: ")
Valor de la variable a: 23.45
>>> a
'23.45'
>>> b = int(input("Valor de la variable a: "))
Valor de la variable a: 23
>>> b
23
```

UT 02.01 - Elementos de un lenguaje informatico.

9.- Comentarios del código.

- Un comentario está destinado a añadir anotaciones en el código fuente de un programa.
- Son ignorados por los compiladores e intérpretes.
- Los comentarios hacen más fácil la tarea de entender el código con vistas a su mantenimiento.
- Los comentarios tienen una amplia gama de posibles usos:
 - Mejora del código fuente.
 - Generación de documentación externa.
 - Integración con sistemas de control de versiones y otras herramientas.
- En Python se pueden introducir comentarios de dos formas:
 - **Un sola línea.** Empezando por #.
 - **Multilínea.** Encerrados entre tres comillas ya sean simples o dobles.

```
Python 3.8.10 (default, Jun 2 2021, 10:49:15)
[GCC 9.4.0] on linux
Type "help", "copyright", "credits" or "license"
for more information.
>>> # Este es un comentario de una linea.
>>> """
... Este es un comentario
... que ocupa varias lineas.
... Asi se puede extender mas.
... """
>>>
```