



El futuro digital
es de todos

MinTIC

Ciclo 1

Fundamentos de programación con Python

Sesión 16: Uso de Funciones en Python

Programa Ciencias de la Computación e Inteligencia Artificial
Escuela de Ciencias Exactas e Ingeniería
Universidad Sergio Arboleda
Bogotá



UNIVERSIDAD
SERGIO ARBOLEDA

Mision
TIC 2022



Agenda

1. Elementos de un programa en Python (Repaso)
2. Funciones integradas
3. Funciones con un parámetro de entrada
4. Funciones para cadenas en Python
5. Funciones varias en Python
6. Ejemplo
7. Ejercicios





1. Elementos de un programa en Python (repaso)

Un programa en Python como ya hemos visto es un archivo de texto, el cual contiene las expresiones y las sentencias o instrucciones propias para el lenguaje Python.

Las expresiones y las instrucciones o sentencias las vamos a crear utilizando y combinando los elementos básicos del lenguaje.





1. Elementos de un programa en Python (repaso)

Los elementos que forman este lenguaje son de diferentes tipos:

- Palabras reservadas (keywords)
- Funciones integradas (built-in functions)
- Literales
- Operadores
- Delimitadores
- Identificadores





1. Elementos de un programa en Python (repaso)

Recordemos que para que un programa se pueda ejecutar, el programa debe ser sintácticamente correcto, es decir, utilizar los elementos del lenguaje Python respetando su reglas de "ensamblaje". Dentro de esta reglas esenciales tenemos:

- Líneas
- Espacios





1. Elementos de un programa en Python (repaso)

➤ Palabras reservadas (keywords)

Las palabras reservadas de Python son las que forman el núcleo del lenguaje Python. Son las siguientes:

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





1. Elementos de un programa en Python (repaso)

➤ Literales o datos simples

Los datos simples que Python es capaz de manejar:

- números: valores lógicos, enteros, decimales y complejos, en notación decimal, octal o hexadecimal
- Cadenas de texto





1. Elementos de un programa en Python (repaso)

➤ Operadores

Los operadores son los caracteres que definen operaciones matemáticas (lógicas y aritméticas). Son los siguientes:

+	-	*	**	/	//	%	@
<<	>>	&		^	~		
<	>	<=	>=	==	!=		





1. Elementos de un programa en Python (repaso)

➤ Delimitadores

Los delimitadores son los caracteres que permiten delimitar, separar o representar expresiones. Son los siguientes:

'	"	#	\			
()	[]	{	}	
,	:	.	;	@	=	->
+=	-=	*=	/=	//=	%=	@=
&=	=	^=	>>=	<<=	**=	





1. Elementos de un programa en Python (repaso)

➤ Identificadores

Los identificadores son las palabras que se utilizan para nombrar elementos creados por el usuario u otros usuarios. Esos elementos pueden ser variables u objetos funciones, clases que combinan ambos o módulos que agrupan los elementos anteriores, etc.

Están formados por letras (mayúsculas y minúsculas), números y el carácter guion bajo (_).





2. Funciones integradas

➤ Funciones integradas (built-in functions)

Una función es un bloque de instrucciones agrupadas, que permiten reutilizar partes de un programa y Python incluye varias funciones de forma predeterminada.

Una función (en este contexto) es una parte separada del código de computadora el cual es capaz de:

- Efecto
- Resultado
- Argumento





2. Funciones integradas

<code>abs()</code>	<code>dict()</code>	<code>help()</code>	<code>min()</code>	<code>setattr()</code>
<code>all()</code>	<code>dir()</code>	<code>hex()</code>	<code>next()</code>	<code>slice()</code>
<code>any()</code>	<code>divmod()</code>	<code>id()</code>	<code>object()</code>	<code>sorted()</code>
<code>ascii()</code>	<code>enumerate()</code>	<code>input()</code>	<code>oct()</code>	<code>staticmethod()</code>
<code>bin()</code>	<code>eval()</code>	<code>int()</code>	<code>open()</code>	<code>str()</code>
<code>bool()</code>	<code>exec()</code>	<code>isinstance()</code>	<code>ord()</code>	<code>sum()</code>
<code>bytearray()</code>	<code>filter()</code>	<code>issubclass()</code>	<code>pow()</code>	<code>super()</code>
<code>bytes()</code>	<code>float()</code>	<code>iter()</code>	<code>print()</code>	<code>tuple()</code>
<code>callable()</code>	<code>format()</code>	<code>len()</code>	<code>property()</code>	<code>type()</code>
<code>chr()</code>	<code>frozenset()</code>	<code>list()</code>	<code>range()</code>	<code>vars()</code>
<code>classmethod()</code>	<code>getattr()</code>	<code>locals()</code>	<code>repr()</code>	<code>zip()</code>
<code>compile()</code>	<code>globals()</code>	<code>map()</code>	<code>reversed()</code>	<code>__import__()</code>
<code>complex()</code>	<code>hasattr()</code>	<code>max()</code>	<code>round()</code>	
<code>delattr()</code>	<code>hash()</code>	<code>memoryview()</code>	<code>set()</code>	





2. Funciones integradas

Dentro del listado anterior de funciones las más conocidas son:

- Función `print()`: imprime en pantalla el argumento.
- Función `input()`: Permite la entrada de datos al usuario

```
print("como te llamas compañero")
nombre=input()
print("hola:", nombre, "bienvenid@ a este curso de Python...")
```

```
Python 3.9.2 (tags/v3.9.2:1a79785, Feb 19 2021, 13:44:55) [MSC v.1928 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:/Users/Liliana/Desktop/ejemplo función print e input.py =====
como te llamas compañero
Liliana Contreras
hola: Liliana Contreras bienvenid@ a este curso de Python...
>>> |
```





3. Función con un parámetro de entrada

¿Qué sucede cuando Python encuentra una invocación como la que está a continuación?



nombreFunción(argumento)





3. Función con un parámetro de entrada

En programación, así como en matemáticas, para las funciones definidas como:

$$f : A \rightarrow B,$$

- Al conjunto A se le denomina dominio y al conjunto B como rango.
- A partir de estos objetos se construye el encabezado de las funciones de programación.
- Sobre esta función se tiene que f corresponde al nombre de la función, el conjunto A corresponde al tipo de los argumentos de dicha función y el conjunto B que es el rango corresponderá al valor de retorno de dicha función.





4. Funciones para cadenas en Python

Función	Utilidad	Ejemplo	Resultado
len()	Determina la longitud en caracteres de una cadena.	<code>len("Hola Python")</code>	11
join()	Convierte en cadena utilizando una separación	<code>Lista = ['Python', 'es'] '-'.join(Lista)</code>	'Python-es'
split()	Convierte una cadena con un separador en una lista	<code>a = ("hola esto sera una lista") Lista2 = a.split() print (Lista2)</code>	['hola', 'esto', 'sera', 'una', 'lista']
replace()	Reemplaza una cadena por otra	<code>texto = "Manuel es mi amigo" print (texto.replace ('es', 'era'))</code>	Manuel era mi amigo
upper()	Convierte una cadena en Mayúsculas	<code>texto = "Manuel es mi amigo" texto.upper()</code>	'MANUEL ES MI AMIGO'
lower()	Convierte una cadena en Minúsculas	<code>texto = "MaNueL eS mI AmIgo" texto.lower()</code>	'manuel es mi amigo'





5. Funciones varias en Python

Función	Utilidad	Ejemplo	Resultado
<code>range()</code>	Crea un rango de números	<code>x = range (5)</code> <code>print (list(x))</code>	[0, 1, 2, 3, 4]
<code>str()</code>	Convierte un valor numérico a texto	<code>str(22)</code>	'22'
<code>int()</code>	Convierte a valor entero	<code>int('22')</code>	22
<code>float()</code>	Convierte un valor a decimal	<code>float('2.22')</code>	2.22
<code>max()</code>	Determina el máximo entre un grupo de números	<code>x = [0, 1, 2]</code> <code>print (max(x))</code>	2
<code>min()</code>	Determina el mínimo entre un grupo de números	<code>x = [0, 1, 2]</code> <code>print (min(x))</code>	0
<code>sum()</code>	Suma el total de una lista de números	<code>x = [0, 1, 2]</code> <code>print (sum(x))</code>	3





5. Funciones varias en Python

Función	Utilidad	Ejemplo	Resultado
list()	Crea una lista a partir de un elemento	<code>x = range (5) print (list(x))</code>	[0, 1, 2, 3, 4]
tuple()	Crea o convierte en una tupla	<code>print(tuple(x))</code>	(0, 1, 2, 3, 4)
open()	Abre, crea, edita un elemento (archivo)	<code>with open("Ejercicios/Ejercicio.py", "w") as variables: variables.writelines("Eje")</code>	Crea el archivo "Ejercicio.py" con el contenido "Eje"
ord()	Devuelve el valor ASCII de una cadena o carácter.	<code>print(ord('A'))</code>	65
round()	Redondea después de la coma de un decimal	<code>print (round(12.723))</code>	13
type()	Devuelve el tipo de un elemento	<code>type(x)</code>	<class 'range'>





6. Ejemplo

- Realizar un programa que capture dos números enteros ingresados por el usuario y realice las operaciones aritméticas básicas (suma, resta, multiplicación y división)





6. Ejemplo

```
operacionesBasicas.py - C:/Users/Liliana/Desktop/operacionesBasicas.py (3.9.2)
File Edit Format Run Options Window Help
print("digite el primer numero")
numero1=float(input())
print("digite el segundo numero")
numero2=float(input())
resultado=numero1+numero2
print("La suma de", numero1,"+",numero2,"=",resultado)
resultado=numero1-numero2
print("La resta de", numero1,"-",numero2,"=",resultado)
resultado=numero1*numero2
print("La producto de", numero1,"*",numero2,"=",resultado)
resultado=numero1/numero2
print("La división de", numero1,"/",numero2,"=",resultado)
```





6. Ejemplo

```
IDLE Shell 3.9.2
File Edit Shell Debug Options Window Help
Python 3.9.2 (tags/v3.9.2:1a79785, Feb 19 2021, 13:44:55) [MSC v.1928 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:/Users/Liliana/Desktop/operacionesBasicas.py =====
digite el primer numero
4.5
digite el segundo numero
3.2
La suma de 4.5 + 3.2 = 7.7
La resta de 4.5 - 3.2 = 1.2999999999999998
La producto de 4.5 * 3.2 = 14.4
La división de 4.5 / 3.2 = 1.40625
>>> |
```





6. Ejemplo

- Realizar un programa que calcule la siguiente expresión:

$$((5 * ((25 \bmod 13) + 100) / (2 * 13)) // 2)$$

```
print((5 * ((25 % 13) + 100) / (2 * 13)) // 2)

===== RESTART: C:/Users/Liliana/Desktop/operacionesBasicas.
10.0
>>> |
```





6. Ejemplo

- Crea un programa que pida al usuario un número real y muestra su raíz cuadrada.

```
import math
print("Digite el número al que desea calcularle la raíz cuadrada")
numero=float(input())
raizC=math.sqrt(numero)
print("La raíz cuadrada de ",numero, "es",raizC)
```

```
===== RESTART: C:\Users\Liliana\Desktop\operacionesBasicas.py =====
Digite el número al que desea calcularle la raíz cuadrada
7.66
La raíz cuadrada de 7.66 es 2.7676705006196096
>>> |
```





6. Ejemplo

- Crear un programa que permita solucionar un problema a través del teorema de Pitágoras:

```
print ("Ingrese el valor del cateto adyacente")
cA=float(input())
print ("Ingrese el valor del cateto opuesto")
cO=float(input())
Hipotenusa = (cA ** 2 + cO ** 2) ** 0.5
print("el valor de la Hipotenusa es=", Hipotenusa)
```

```
===== RESTART: C:\Users\Liliana\Desktop\operacionesBasicas.py =====
Ingrese el valor del cateto adyacente
5
Ingrese el valor del cateto opuesto
2
el valor de la Hipotenusa es= 5.385164807134504
>>> |
```





7. Ejercicios propuestos

- Crea un programa que pida al usuario un número real y muestre su raíz cuarta (la raíz cuadrada de la raíz cuadrada).
- Crear un programa que calcule y escriba el área para un círculo.
- Elaborar un programa que calcule y escriba el precio de venta para un artículo, se tiene, el nombre del producto y el costo de producción; el precio de venta se calcula añadiendo el 120 % como utilidad y 15 % de impuestos.





Preguntas

