Ayudantía 4: Importación y Definición de Funciones

21 de Octubre

- **Profesor:** Cristian Sepúlveda cristian.sepulvedas@usach.cl
- **Ayudante:** John Serrano john.serrano@usach.cl

1 ¿Que es una Función en Python?

Similar a las funciones de Cálculo I y Álgebra I, una función en Python es un trozo de código que recibe algo como **entrada** y devuelve algo como **salida**.

$$f(x) = x \cdot 2$$

Una función definida en el código no sirve por si sola. Es importante llamarla dentro del código

2 Importación de Funciones

Ya hemos utilizado Funciones en Python. A estas funciones se les conocen como **"Funciones Nativas"**, pues no es necesario definirlas ni importalas, ya vienen creadas y listas para utilizar por defecto en Python.

Algunos ejemplos son:

- print()
- input()
- str()
- float()
- bool()
- list()
- sum()
- max()
- len()

Debemos tener en consideración que no nos interesa como estan definidas estas funciones. Solo nos interesa que recibe como entrada, que devuelve como salida y que es lo que hace.

Función	Entradas	Salida	Proceso
abs(x)	Un número	Un número	Devuelve el valor absoluto de un número
pow(x, y)	Dos números	Un número	Es equivalente a realizar x ** y
max(x)	Un elemento iterable	Un elemento	Devuelve el elemento de mayor valor del elemento iterable
min(x)	Un elemento iterable	Un elemento	Devuelve el elemento de menor valor del elemento iterable
round(x,y)	Un flotante (x) y un entero (y)	Un flotante	Devuelve el elemento x redondeado a la cantidad de decimales informada en y

Las **funciones importadas** provienen de paquetes, modulos o librerias que son llamadas dentro de un código. Por buenas prácticas es buena idea seguir la siguiente estructura de código ahora que estamos hablando de funciones:

[1]:
```
# BLOQUE DE DEFINICION
# IMPORTACION DE FUNCIONES
# DEFINICION DE FUNCIONES
# DEFINICION DE CONSTANTES

# BLOQUE PRINCIPAL
# ENTRADA
# PROCESAMIENTO
# SALIDA
```

Hay dos formas de realizar estos llamados. **La primera no es muy recomendada** y consiste en llamar a todo el paquete / modulo / libreria. Como ejemplo, vamos a utilizar la libreria "math" (Funciones matemáticas).

Esta forma sigue el siguiente formato: **import nombre_libreria**, donde **nombre_libreria** es el nombre de la libreria que queremos importar. Las funciones se llaman usando el formato **nombre_libreria.funcion**

[2]:
```
# BLOQUE DE DEFINICION
# IMPORTACION DE FUNCIONES
import math # Importamos toda la libreria math

# BLOQUE PRINCIPAL
# PROCESAMIENTO
resultado = math.cos(math.pi)    # Se llama a cos de la forma math.cos
# SALIDA
print(resultado)
```

```
-1.0
```

Nótese que estas importaciones no solo traen funciones, si no que a veces traen constantes tam-

bién.

La segunda forma es mas recomendada y consiste en llamar solo lo que necesitamos de la libreria. Sigue el siguiente formato: **from nombre_libreria import nombre_funcion**, donde **nombre_libreria** es el nombre de la libreria que queremos importar y **nombre_funcion** es el nombre de la funcion a importar. Con esta forma solo llamamos a las funciones con su nombre.

[3]:
```
# BLOQUE DE DEFINICION
# IMPORTACION DE FUNCIONES
from math import pi          # Importamos pi de la libreria math
from math import sqrt        # Importamos sqrt de la libreria math
from math import sin

# BLOQUE PRINCIPAL
# PROCESAMIENTO
resultado = sqrt(pi)         # Se llama a la funcion sin mencionar a la libreria
# SALIDA
print(resultado)
```

```
1.7724538509055159
```

Existen librerias / paquetes que no vienen con python. Estos deben instalarse mediante **pip**. Si tenemos pip instalado, entonces debemos abrir una consola (CMD) y escribir: **pip install nombre_paquete**, donde **nombre_paquete** es el nombre del paquete a instalar.

3 Definición de Funciones

Ahora vamos a crear nuestras propias funciones. Para ello debemos utilizar algunas palabras reservadas: * **def:** Sirve para definir una función. * **return:** Sirve para indicar que devuelve la función

Un ejemplo es el siguiente:

[4]:
```
# BLOQUE DE DEFINICION
# IMPORTACION DE FUNCIONES
# DEFINICION DE FUNCIONES

'''
Entrada: Funcion que recibe un valor entero
Salida: Retorna el resultado de x * 2, donde x es la entrada
Descripcion: Funcion que recibe un valor entero y realiza la multiplicacion
de x * 2. Luego guarda ese resultado en una variable y retorna esa variable.
'''
def f(x):
    resultado = x * 2
    return resultado # ACA SE ACABA LA FUNCION

# BLOQUE PRINCIPAL
# PROCESAMIENTO
```

```
llamado = f(4)      # SE LLAMA a la funcion f, la cual recibe el numero 4
# SALIDA
print("El resultado del llamado a la funcion con x = 4 es: ", llamado)
```

```
El resultado del llamado a la funcion con x = 4 es:  8
```

Veamos otro ejemplo:

[5]:
```
# BlOQUE DE DEFINICION
# IMPORTACION DE FUNCIONS
from math import pi
# DEFINICION DE FUNCIONES

def obtener_maximo(lista):
    '''
    Entrada: Una lista de numeros enteros
    Salida: Un numero entero correspondiente al maximo de una lista
    Descripcion: Funcion que recorre una lista de numeros enteros y obtiene
    su maximo sin utilizar la funcion nativa max().
    '''
    i = 0
    maximo = lista[0]
    while i < len(lista):
        if lista[i] > maximo:
            maximo = lista[i]
        i += 1
    return maximo


def operar_cada_elemento(lista):
    '''
    Entrada: Una lista de numeros enteros
    Salida: Una lista de numeros flotantes
    Descripcion: Funcion que recorre una lista de numeros enteros y
    multiplica cada elemento con el valor de PI. Devuelve la lista modificada.
    '''
    i = 0
    while i < len(lista_numeros):
        lista.append(lista_numeros[i] * PI)
        i += 1
    return lista

# DEFINICION DE CONSTANTES
PI = pi       # Se define PI como una constante

# BLOQUE PRINCIPAL
# ENTRADA
lista_numeros = eval(input("Ingrese una lista de numeros enteros: "))
```

```
# PROCESAMIENTO
i = 12
maximo = obtener_maximo(lista_numeros)  # Se llama a la funcion obtener_maximo
lista_numeros2 = operar_cada_elemento([])  # Se llama a la funcion␣
 ↪operar_cada_elemento
maximo2 = obtener_maximo(lista_numeros2)   # Se llama a la funcion obtener_maximo

# SALIDA
print("La lista original es: ", lista_numeros)
print("El maximo de esta lista es: ", maximo)
print("\n")
print("La lista al llamar a la funcion operar_cada_elemento es: ",␣
 ↪lista_numeros2)
print("El maximo de esta lista es: ", maximo2)
```

```
Ingrese una lista de numeros enteros: [3,1,12,-2,4,-9,8,10,-10,2,9,-12,-6]
La lista original es:  [3, 1, 12, -2, 4, -9, 8, 10, -10, 2, 9, -12, -6]
El maximo de esta lista es:  12


La lista al llamar a la funcion operar_cada_elemento es:  [9.42477796076938,
3.141592653589793, 37.69911184307752, -6.283185307179586, 12.566370614359172,
-28.274333882308138, 25.132741228718345, 31.41592653589793, -31.41592653589793,
6.283185307179586, 28.274333882308138, -37.69911184307752, -18.84955592153876]
El maximo de esta lista es:  37.69911184307752
```

- **Variable local:** Variable que solo existe dentro de la función, por lo que una vez que se acaba el procedimiento de la función, se eliminan.
- **Variable global:** Variable que existe para todo el código y las funciones son capaces de acceder a estas variables.

También es posible importar funciones definidas en un archivo Python para utilizar en otro archivo Python.

4 Ejercicios

1. ¿Que es lo que hace la siguiente funcion? ¿Cual es su resultado?

[6]:
```
def funcion_unica(x):
    return 0
    i = 0
    while i < len(x):
        elemento_actual = x[i]
        if elemento_actual % 2 == 0:
            x[i] = x[i] * 0
        i += 1
    print("Hola gente")
```

```
    return x
```

```
x = [3,4,2,1,4,5]
x = funcion_unica(x)
print(x)
```

```
0
```

2. Construya un programa en Python que imprima por pantalla la raíz cuadrada de los números múltiplos de 3 de la siguiente lista de valores. La lista es: [10,33,9,14,18,14,12,21,50,55,60]

3. Construya un programa en Python que reciba como entrada un string y determine cuáles la letra que más se repite. Restricción: no utilice el método count()

[ ]: