Módulos

IITA 2023

Primero...un pequeño repaso:

- Python...¿Qué es?
- SOFTWARE
- HARDWARE
- LENGUAJE DE PROGRAMACIÓN
- PROGRAMA
- TIPO DE DATO
- VARIABLE

- Estructuras:
  - Selectivas
    - Simples
    - Múltiples
  - Repetitivas

Temas de hoy:

- **MODULOS**: Dividen al programa en partes más pequeñas
  - Se identifican por un nombre y resuelven alguna necesidad
  - Simplifican la resolución de un problema

- **Estructuras de datos**: Conjunto o colección de valores (variables, por ejemplo) organizados de una manera específica en la memoria de la computadora.

Módulos

- La definición de módulos se realiza mediante la instrucción def más un nombre descriptivo -para el cuál, aplican las mismas reglas que para el nombre de las variables- seguido de paréntesis de apertura y cierre.
- Como toda estructura de control en Python, la definición finaliza con dos puntos (:), y el algoritmo que la compone irá identado (con sangría):

```
#Defino un modulo
def Mi_Modulo():
    #Escribo mi algorimo aqui
    pass

#Defino otro modulo
def Mi_otro_modulo():
    #Escribo otra secuencia de pasos
    pass


Mi_Modulo()    #En esta linea ejecuto la secuencia de pasos de mi modulo

Mi_Modulo()    #Lo vuelvo a hacer

Mi_otro_modulo()   #En esta linea ejecuto la secuencia de pasos de mi otro modulo
```

Ámbitos de las variables

- Cuando definimos módulos, estamos “separando” nuestro programa en programas mas pequeños. En cada módulo las variables que cree dentro de ellos no existen fuera de estos.
  - Las variables LOCALES o de ámbito LOCAL son aquellas que defino dentro de módulos.
  - Las variables de ámbito GLOBAL son aquellas que defino fuera de ellos

```
def Mi_Modulo():
    variable = 2

Mi_Modulo()
print(varible)
```

Ámbitos de las variables

- No se puede acceder a variables LOCALES desde un ámbito GLOBAL.

```
def Mi_Modulo():
    variable = 2

Mi_Modulo()
print(varible)
```

Salida: Error

```
def Mi_Modulo():
    global variable
    variable = 2

variable = 1

Mi_Modulo()
print(variable)
```

Salida: 2

- En Python si puedo acceder a las variables GLOBALES en un ámbito LOCAL. Pero para modificarlas desde un ámbito LOCAL se utiliza la palabra GLOBAL.

```
def Mi_Modulo():
    print(f"Hola {Nombre}")

Nombre = "Seba"
Mi_Modulo()
```

Salida: Hola Seba

```
def Mi_Modulo():
    variable = 2

varaible = 1
Mi_Modulo()
print(varaible)
```

Salida: 1

Funciones: Parámetros

- Un parámetro es un valor que la función espera recibir cuando sea llamada
- Una función puede esperar 0 ,1 o más parámetros

```
def mi_funcion(nombre, apellido):
        print("Nombre: ", nombre)
        print("Apellido: ", apellido)
```

→ Atención!

Los parámetros que una función espera, serán utilizados por ésta dentro de su algoritmo, a modo de variables de **ámbito local**

Retornar valores

- Un modulo puede o no, retornar datos, retornar un valor en un modulo sirve para pasar valores de un ámbito local a un ámbito global.(Es importante almacenar estos datos en el ámbito global)

- Para retornar un valor en un modulo se utiliza la palabra return y una vez el programa lee esa palabra el modulo en el que esta está finaliza. (deja ejecutarse esa parte)

```python
def AptoParaComprarAlchol(edad):
    if edad>= 18:
        return True
    else:
        return False


edad_de_comprador = int(input("Ingresa tu edad: "))

es_apto = AptoParaComprarAlchol(edad_de_comprador)
if (es_apto == True):
    print("Aca esta tu Cerveza")
else:
    print("Espera mientras llega la policia")
```

Funciones y procedimientos

- Un MODULO es una estructura que se puede clasificar de 2 formas según su comportamiento.
  - Funciones: Aquellos módulos que SI retornan valores.
  - Procedimientos: Aquellos módulos que NO retornan valores.

```
def AptoParaComprarAlchol(edad):
    if edad>= 18:
        return True
    else:
        return False
```

```
def Saludar(Persona):
    print(f"Hola querido {Persona}, ¿Como estas?")
```

FUNCIONES

- Atención! → Un Modulo, no se ejecuta hasta que sea invocado. Para invocar un Modulo, simplemente se la llama por su nombre

```
def mi_funcion():
        # aquí el algoritmo

mi_funcion()
```

- Cuando una función haga un retorno de datos, **estos TIENEN que asignarse a una variable si se los quiere poder usar en el futuro:**

```
def funcion():
        return “Hola mundo”

Mensaje=funcion()
Print(Mensaje)
```

Funciones: cuestiones a considerar...

- Python maneja funciones y los procedimientos de la misma forma, pero NO son lo mismo. En la teoría:
  - Función: devuelve un valor
  - Procedimiento: solo ejecuta comandos

- Al llamar a un Modulo, siempre se le deben pasar sus argumentos en el mismo orden en el que los espera

Funciones útiles de Python

- len(): Calcula la cantidad de caracteres de un string
- find(): **Regresa** el índice correspondiente al primer carácter de la cadena original que coincide con lo que estamos buscando, si no encuentra ese carácter regresa -1
- upper() y lower(): convierte a todos los caracteres de una cadena de texto a mayúsculas o a minúsculas
- strip(), lstrip() y rstrip(): para eliminar todos los espacios en blancos, sólo los que aparecen a la izquierda y sólo los que aparecen a la derecha, respectivamente…
- Para obtener substring podemos:
  - String [:valor numerico] → extrae tantos caracteres desde la izquierda como indica el valor num
  - String [valor numérico:] → extrae tatnos caracteres desde el valor numérico hacia el final del texto
  - String [valor numérico1:valor numérico:2] → extrae caracteres desde el valor numérico 1 hasta (valor numerico2 – 1)

Practiquemos sobre lo visto...

- 1. Escriba una función que muestre todos los números primos entre 1 y un número n que es ingresado por parámetro.

- 2. Solicitar al usuario que ingrese su dirección email. Imprimir un mensaje indicando si la dirección es válida o no, valiéndose de una función para decidirlo. Una dirección se considerará válida si contiene el símbolo "@".
- (find) →
- 3. Definir una función que muestre el factorial de un número

- 4. Escriba una función que le pida al usuario ingresar condimentos para un sándwich, hasta que el usuario ingrese ‘salir’. Cada vez que se ingrese un condimento, muestre un mensaje avisando que ya se agregó el condimento al sándwich.

Funciones: cuestiones a considerar...

- Las funciones en Python pueden devolver más de un valor:

```
def funcion(par1,par2):
    return par1,par2

valor1,valor2=funcion('perro','gato')
print(valor1)
print(valor2)
```

Funciones: cuestiones a considerar...

- Se pueden Usar “Parámetros por omisión”:

```
def Saludar(nombre, mensaje="¿Como estas?"):
    print(f"hola {nombre},{mensaje}")

Saludar("Nicolas","¿Hiciste el tp?")
Saludar("Nicolas")
```

Salida:

```
hola Nicolas,¿Hiciste el tp?
hola Nicolas,¿Como estas?
```

Funciones: cuestiones a considerar...

- Se pueden pasar parámetros por posición o por nombre:

```python
def Saludar(nombre,mensaje):
    print(f"{nombre},{mensaje}")

Saludar("¿Como Estas?", "Nicolas")
Saludar(mensaje="¿ComoEstas?",nombre="Nicolas")
```

Salida:

```
¿Como Estas?,Nicolas
Nicolas,¿ComoEstas?
```