Python

Funciones:

En Python, la definición de funciones se realiza mediante la instrucción **def** más un nombre de función 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 de la función finaliza con dos puntos (:) y el algoritmo que la compone, irá identado.

Sintaxis basica:

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

Una función, no es ejecutada hasta tanto no sea invocada. Para invocar una función, simplemente se la llama por su nombre:

```
def funcion():
    print( "Hola Mundo")

funcion()
```

Las funciones que hagan un retorno de datos pueden ser asignadas a variables:

```
def funcion():
    return( "Hola Mundo")

frase = funcion()

print frase
```

Paso de parametros:

Una función puede esperar uno o más parámetros (que irán separados por una coma) o ninguno.

```
def mi_funcion(param1, param2):
        # algoritmo
```

Al llamar a una función, siempre se le deben pasar sus argumentos en el mismo orden en el que los espera.

Parametros por defecto:

En Python, también es posible, asignar valores por defecto a los parámetros de las funciones. Esto significa, que la función podrá ser llamada con menos argumentos de los que espera:

```
def saludar(nombre, mensaje='Hola'):
        print mensaje, nombre
saludar('Pepe Grillo') # Imprime: Hola Pepe Grillo
```

parametros clave-valor:

En Python, también es posible llamar a una función, pasándole los argumentos esperados, como pares de claves=valor:

```
def saludar(nombre, mensaje='Hola'):
    print mensaje, nombre
saludar(mensaje="Buen día", nombre="juan")
```

Parametros arbitrarios:

Al igual que en otros lenguajes de alto nivel, es posible que una función, espere recibir un número arbitrario -desconocido- de argumentos. Estos argumentos, llegarán a la función en forma de tupla. Para definir argumentos arbitrarios en una función, se antecede al parámetro un asterisco (*):

```
def recorrer_parametros_arbitrarios(parametro_fijo, *arbitrarios):
        print parametro_fijo
        # Los parámetros arbitrarios se corren como tuplas
        for argumento in arbitrarios:
                print argumento
recorrer_parametros_arbitrarios('Fixed', 'arbitrario 1', 'arbitrario 2', 'arbitrario 3')
```

Es posible también, obtener parámetros arbitrarios como pares de clave=valor. En estos casos, al nombre del parámetro deben precederlo dos astericos (**):

```
def recorrer_parametros_arbitrarios(parametro_fijo, *arbitrarios, **kwords):
        print parametro_fijo
        for argumento in arbitrarios:
                print argumento
                # Los argumentos arbitrarios tipo clave, se recorren como los diccionarios
        for clave in kwords:
                print "El valor de", clave, "es", kwords[clave]
```

recorrer_parametros_arbitrarios("Fixed", "arbitrario 1", "arbitrario 2", "arbitrario 3", clave1="valor uno", clave2="valor dos")

Ejercicio:

1- Escriba una función usando un numero variable de parámetros para contar el número de ocurrencia de cada carácter en todas las palabras enviadas como parámetros. La salida debe de ser un diccionario.

Ejemplo de llamada y salida:

```
contador = contadorCaracteres("Gary", "Kris", "Cathy", "John", "Alex")
print( contador )
```

Resultado:

{'K': 1, 'I': 1, 'X': 1, 'A': 3, 'R': 2, 'S': 1, 'H': 2, 'L': 1, 'T': 1, 'C': 1, 'O': 1, 'Y': 2, 'G': 1, 'E': 1, 'J': 1, 'N': 1}