Sesion 31

Reconoce los conceptos, aplicaciones e importancia de las funciones en Python, para el desarrollo de programas.

1. Introducción:

El tema propuesto es relevante, ya que los alumnos van a reconocer como se diseñan funciones y su aplicación básica, para una mejor distribución del código de programación

MÓDULO 4: ¿Cómo definir y utilizar funciones?, ¿Cómo pasar argumentos y las distintas formas de hacerlo?, El alcance de los nombres, Tuplas y diccionarios, Procesamiento de datos.

2. Funciones.

a. Definiciones: ¿Por qué necesitamos funciones?, descomposición, creación de funciones.

Cuando desarrollamos una aplicación existe código que se ejecuta de manera repetida y que genera algún resultado, este código que se utiliza en diversas partes del programa es denominado código reutilizable, es en ese momento que nace la idea de generar funciones.

Una función entonces es un bloque de código que se le asigna un nombre y requiere de algunos valores para que pueda operar, retornando un resultado; a estas funciones se les puede invocar desde cualquier parte del programa.

En general, las funciones en Python son componentes importantes que cuentan con una estructura que consta de dos principios.

- **La reutilización**: indica que la función se puede reutilizar varias veces y en distintos programas.
- **La modularización**: indica que puedes segmentar programas complejos con módulos más simples para depurar y programar con mayor facilidad, el principio de “Divide y Vencerás”.

Python nos ha mostrado al momento algunas funciones: cuando se desea mostrar o imprimir algo en consola se utiliza print(), cuando se desea leer el valor de una variable se emplea input(), combinados posiblemente con int() o float(); de esa manera podemos observar como ejemplo como es que operan las funciones y el alcance que pueden tener si son bien diseñadas.

Cuando descomponemos(dividimos) el trabajo, logramos que todos estén involucrados en el desarrollo de un sistema, las funciones generan ese efecto dentro de las aplicaciones

¿De dónde provienen las funciones?

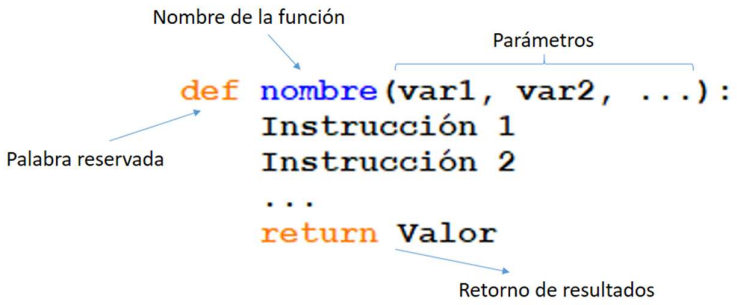
En general, las funciones provienen de al menos tres lugares:

- **De Python**: varias funciones (como print()) son una parte integral de Python, y siempre están disponibles sin algún esfuerzo adicional del programador; se les llama a estas funciones ***funciones integradas***.
- **De los módulos preinstalados de Python**: muchas de las funciones, las cuales comúnmente son menos utilizadas que las integradas, están disponibles en módulos instalados juntamente con Python; para poder utilizar estas funciones el programador debe importarlas o en su defecto hacer las instalaciones respectivas, para luego importarlas.
- **Directamente del código**: son las funciones que el programador diseña, las coloca dentro del código, y usa libremente.

Creación de Funciones

Para crear funciones en Python se debe considerar lo siguiente:

- Uso de la palabra clave def.
- Definir un nombre para la función a crear.
- El uso de paréntesis, dentro de estos deberá ubicar los parámetros de entrada, considerar que estos pueden llegar a ser opcionales.
- Uso de la sentencia pass.
- Uso de la sentencia return.



A continuación, veremos un ejemplo simple, en esta última parte la programación será dividida en dos bloques: Zona de Funciones y la Zona del

Programa Principal:

```
## Zona de Funciones
def saludo():
    print("Hola a Todos...!!!")

## Programa MAIN
print("Creando una función")
print("Invocaremos a la función:")
saludo()
```

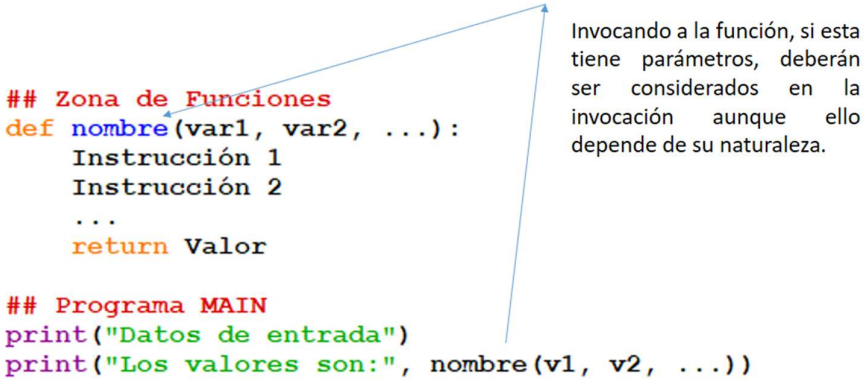
Al ejecutar el código se muestra:

```
Creando una función
Invocaremos a la función:
Hola a Todos...!!!
```

Cuando se invoca o llama a una función se le debe nombrar, tal como fue diseñada en caso de tener parámetros estos deberán incluirse.

b. ¿Cómo trabajan las funciones?

En el ejemplo anterior hemos observado que, al invocar a una función, debemos realizarlo por su nombre, de esa manera se ejecuta; debemos considerar entonces lo siguiente:




```
## Zona de Funciones
def nombre(var1, var2, ...):
    Instrucción 1
    Instrucción 2
    ...
    return Valor

## Programa MAIN
print("Datos de entrada")
print("Los valores son:", nombre(v1, v2, ...))
```

Cuando una función es invocada desde el programa principal, se ejecutarán las instrucciones internas de la función, dependerá de los parámetros si es que los tiene para que el retorno de información sea el adecuado; los parámetros tienen diversa naturaleza, por lo que se debe considerar en el momento del diseño.

3. Como las funciones se comunican con su entorno

a. Los parámetros en las funciones, el paso de parámetros y sus variantes

Toda función en Python requiere de generar información, y en consecuencia siempre deben devolver un valor, en caso de que no se genere nada, la función devuelve la palabra reservada None.

En otros lenguajes de programación existen los procedimientos, los cuales realmente no devuelven valor alguno, en Python no existen.

Al tener que generarse un valor (pueden generarse varios también), necesita de parámetros o argumentos, los cuales serán entregados dependiendo la naturaleza de la función que se está diseñando.

Observe el siguiente ejemplo, la función opera tiene 2 parámetros, los cuales dependiendo la condición se sumarán o se restarán; desde el programa principal se invoca a la función pasándole los parámetros adecuados, en el orden que se indicó.

```
## Zona de Funciones
def opera(a,b):
    if(a==b):
        X=a+b
    else:
        X=a-b
    return X

## Programa MAIN
print("Función de Igualdad...!!!")
N1=int(input("Ingrese 1er valor:"))
N2=int(input("Ingrese 2do valor:"))
print("El resultado es:",opera(N1,N2))
```

Al ejecutar el código se muestra:

```
Función de Igualdad...!!!
Ingrese 1er valor:8
Ingrese 2do valor:3
El resultado es: 5
```

Lo que ha observado se denomina: **paso de parámetros posicionales**, donde se respeta el orden de acuerdo al diseño entregado.

Existe otra manera de realizar el paso de parámetros y es denominado **con palabras clave**, en otras palabras, al momento de colocar el parámetro, se debe indicar el valor que deberá ser entregado a cada uno de los parámetros, observar:

```
## Zona de Funciones
def sumar(a,b,c):
    if(a==0):
        X=a+b+c
    else:
        X=a-b-c
    return X

## Programa MAIN
N1=int(input("Ingrese 1er valor:"))
N2=int(input("Ingrese 2do valor:"))
N3=int(input("Ingrese 3er valor:"))
print("El resultado es:",sumar(b=N1,c=N3, a=N2))
```

Al ejecutar el código se muestra:

```
Ingrese 1er valor:1
Ingrese 2do valor:0
Ingrese 3er valor:2
El resultado es: 3
```

Si observa, al momento de invocar a la función, en esta se indica los valores que deberán asumir cada uno de los parámetros.

Se puede indicar a los parámetros de una función que convine lo visto anteriormente, y ello se vería de la siguiente manera:

```
## Zona de Funciones
def restar(a,b,c):
    if(a!=0):
        X=a+b+c
    else:
        X=a-b-c
    return X

## Programa MAIN
N1=int(input("Ingrese 1er valor:"))
N2=int(input("Ingrese 2do valor:"))
N3=int(input("Ingrese 3er valor:"))
print("El resultado es:",restar(N1,c=N3, b=N2))
```

Al ejecutar el código se muestra:

```
Ingrese 1er valor:6
Ingrese 2do valor:2
Ingrese 3er valor:3
El resultado es: 11
```

Se puede crear parámetros **con valores por defecto**, ello puede ocurrir cuando el valor a usar es constante o en su defecto es un valor que puede en algún momento no tener un ingreso en el programa principal.

De ocurrir ello, el valor por defecto será quien asuma el valor no enviado.

```
## Zona de Funciones
def promediar(nota1=0,nota2=0):
    Pr=(nota1+nota2)/2
    return Pr

## Programa MAIN
print("El alumno falto a las Evaluaciones SI/NO?:")
Op=input()
if(Op=="NO"):
    N1=int(input("Ingrese 1era nota:"))
    N2=int(input("Ingrese 2da nota:"))
    print("El promedio es:",promediar(N1,N2))
else:
    print("El promedio es:",promediar())
```

Al ejecutar el código se muestra, en caso de que **NO** haya faltado:

```
El alumno falto a las Evaluaciones SI/NO?:
NO
Ingrese 1era nota:15
Ingrese 2da nota:18
El promedio es: 16.5
```

Al ejecutar el código se muestra, en caso de que **SI** haya faltado:

```
El alumno falto a las Evaluaciones SI/NO?:
SI
El promedio es: 0.0
```

En este último caso, podemos observar que el valor que asumió la función por las 2 notas no ingresadas es de cero.

4. Regresando el resultado de una función.

a. La instrucción return

Esta instrucción usada dentro de una función y tiene como finalidad el devolver el resultado generado dentro de la función, si esta no retorna nada no es necesario su uso.

Por otro lado, el uso de la instrucción return indica la finalización de la función en el lugar donde esta se encuentre y además implica la devolución de los resultados dela función sobre su mismo nombre, de ahí que al invocar a la función se use su nombre.

b. El valor None y su uso en programación.

Este valor None, resulta un dato que se debe aprender a manipular, ya que en realidad no tiene un valor especifico, es una palabra reservada de Python que indica nada.

Existe dos tipos de circunstancias en las que None se puede usar de manera segura:

- Cuando se le asigna a una variable (o se devuelve como el resultado de una función).
- Cuando se compara con una variable para diagnosticar su estado interno.

Por ejemplo:

```
Op=input("Ingresar un valor SI o NO?:")
if(Op=="SI"):
    Var=int(input())
else:
    Var=None
if(Var==None):
    print("La variable no tiene un valor alguno...")
```

Al ejecutar el código se muestra:

```
Ingresar un valor SI o NO?:NO
La variable no tiene un valor alguno...
```

Finalmente, cuando trabajamos en una función y este devuelve un valor None, es señal de que el diseño de la función no es el correcto, observar:

```
## Zona de Funciones
def analizando(N):
    if(N>=0):
        return True

## Programa MAIN
Num=int(input("Ingrese un Número:"))
print(analizando(Num))
```

Al ejecutar el código se muestra:

```
Ingrese un Número:-9
None
```

La función no sabe que devolver si N es negativo, tiene un return.

5. Desarrollo de ejercicios tipo

Los ejercicios que veremos a continuación se desarrollan usando funciones conjuntamente con todas las estructuras desarrolladas anteriormente.

1. Problema Prg1

Desarrolle un algoritmo que permita ingresar 2 números, luego mediante una función calcule una operación que se indique en base a estos números: puede ser suma, resta, multiplicación o división.

Solución:
La Función a crear, debe tener un parámetro que indique la opción a generar para indicar la operación que se desea realizar, diseñarla de la mejor manera.

Programa:

```
## Zona de Funciones
def operando(a,b,sw):
    if(sw==1):
        return a+b
    elif(sw==2):
        return a-b
    elif(sw==3):
        return a*b
    else:
        if(b!=0):
            return a/b
        else:
            return "Div*Cero"
```

```
## Programa MAIN
print("Ingrese 1er número:")
Num1=int(input())
print("Ingrese 2do número:")
Num2=int(input())
Op=True
while(Op):
    print("Ingrese la operación que "
          "desea realizar:\n"
          "1: Suma \n"
          "2: Resta \n"
          "3: Multiplicación \n"
          "4: División")
    Op=int(input())
    if(Op==1)or(Op==2)or(Op==3)or(Op==4):
        break
print("La operación es:",operando(Num1, Num2, Op))
```

Al ejecutar el programa, se muestra lo siguiente:

```
Ingrese 1er número:
7
Ingrese 2do número:
0
Ingrese la operación que desea realizar:
1: Suma
2: Resta
3: Multiplicación
4: División
4
La operación es: Div*Cero
```

2. Problema Prg2

Desarrolle un algoritmo que permita ingresar 2 notas, luego mediante una función calcule el promedio, teniendo en cuenta que se obtiene duplicando la mayor nota; finalmente muestre un mensaje que diga si aprobó o no el curso.

Solución:

Este ejercicio, no tiene complejidad en el desarrollo, solo debe tener cuidado al momento de generar la función, los parámetros a entregar son las dos notas. Se debe validar el ingreso de datos.

Programa:

```
## Programa MAIN
Op=True
while(Op):
    print("Ingrese 1er nota:")
    N1=int(input())
    if(N1>=0)and(N1<=20):
        break
Op=True
while(Op):
    print("Ingrese 2da nota:")
    N2=int(input())
    if(N2>=0)and(N2<=20):
        break
if(N1>N2):
    Pr=(N1*2+N2)/3
else:
    Pr=(N2*2+N1)/3
print("El promedio es:",Pr)
```

En esta ocasión empezamos creando el programa completo dentro del MAIN, para luego transformarlo en funciones, observar:

```
## Zona de Funciones
def valida(Not):
    if(Not>=0)and(Not<=20):
        return False
    else:
        return True

def promedia(a, b):
    if(a>b):
        Pr=(a*2+b)/3
    else:
        Pr=(b*2+a)/3
    return Pr
```

```
## Programa MAIN
Op=True
while(Op):
    print("Ingrese 1er nota:")
    N1=int(input())
    Op=valida(N1)
Op=True
while(Op):
    print("Ingrese 2da nota:")
    N2=int(input())
    Op=valida(N2)
print("El promedio es:",promedia(N1,N2))
```

Al ejecutar el programa, se muestra lo siguiente:

```
Ingrese 1er nota:
13
Ingrese 2da nota:
18
El promedio es: 16.333333333333332
```

3. Problema Prg3

Se pretende autogenerar 100 números, mediante un algoritmo indicar la media aritmética de los números que son positivos y negativos, por separado.

Solución:

Considerar el uso de bucles, listas y tener en observación que el cero no se toma para nada en la solución.

Programa:

```
## Zona de Funciones
def media(Lista):
    Ac1=0;Ac2=0
    c1=0;c2=0
    for valor in Lista:
        if(valor>0):
            Ac1=Ac1+valor
            c1=c1+1
        else:
            if(valor<0):
                Ac2=Ac2+valor
                c2=c2+1
    p1=Ac1/c1
    p2=Ac2/c2
    return p1,p2

## Programa MAIN
import random
Lst=[]
for i in range(100):
    Lst.append(random.randint(-50,50))
mp,mn=media(Lst)
print("Media de Positivos:",mp)
print("Media de Negativos:",mn)
```

Al ejecutar el programa, se muestra lo siguiente:

```
Media de Positivos: 28.053571428571427
Media de Negativos: -25.74418604651163
```

4. Problema Prg4

Desarrolle un algoritmo que permita calcular el interés compuesto generado en un año por un capital que se ingresa, si se sabe que el interés mensual es de 3.7% (no usar formulas financieras).

Solución:
Se debe generar la formula respectiva para calcular el interés que se acumula cada mes, teniendo en cuenta el % que cobra el banco al mes.

Programa:

```
## Zona de Funciones
def intcompuesto(monto):
    for i in range(12):
        monto=monto+0.037*monto
    return monto

## Programa MAIN
print("Ingrese el Capital a depositar:")
Kp=float(input())
print("El interés equivale a :",intcompuesto(Kp)-Kp)
```

Al ejecutar el programa, se muestra lo siguiente:

```
Ingrese el Capital a depositar:
1000
El interés equivale a : 546.4827377937811
```

5. Problema Prg5

Desarrolle un algoritmo que genere una función denominada MAX, esta deberá mostrar siempre el mayor valor de un conjunto de números que se ingresan, no podrá usar otras funciones ni instrucciones especiales en su diseño, solo lógica de programación.

Solución:

Los valores que se analizarán deberán ingresarse a una lista, a partir de ese momento en una función deberá generar el código respectivo para hallar el mayor componente.

Por otro lado, el hecho de que la función a crear (MAX) no tenga ninguna otra función interna o instrucción, es para poder emplear estrategias lógicas que permitan ubicar el mayor valor.

Programa:

```
## Zona de Funciones
def MAX(lista,n):
    My=lista[0]
    for i in range(1,n):
        if(My<lista[i]):
            My=lista[i]
    return My

## Programa MAIN
Op="SI"
Lst=[]
c=0
while(Op=="SI"):
    print("Ingrese un número:")
    Lst.append(int(input()))
    c=c+1
    Sw=True
    while(Sw):
        print("Desea continuar SI/NO?")
        Op=input()
        if(Op=="SI")or(Op=="NO"):
            break
print(Lst)
print("El mayor valor es:",MAX(Lst,c))
```

Al ejecutar el programa, se muestra lo siguiente:

```
Ingrese un número:
4
Desea continuar SI/NO?
SI
Ingrese un número:
6
Desea continuar SI/NO?
SI
Ingrese un número:
14
Desea continuar SI/NO?
SI
Ingrese un número:
1
Desea continuar SI/NO?
SI
Ingrese un número:
50
Desea continuar SI/NO?
SI
Ingrese un número:
34
Desea continuar SI/NO?
SI
Ingrese un número:
22
Desea continuar SI/NO?
NO
[4, 6, 14, 1, 50, 34, 22]
El mayor valor es: 50
```

6. Actividad

A) Ejercicios en general

Desarrolle los siguientes ejercicios usando todas las sentencias e instrucciones desarrolladas a lo largo del curso.

a) Desarrolle una función que simule a la función MIN del Excel, en este caso, la función deberá devolver el menor valor disponible de un conjunto de valores entregados.

b) Desarrolle un algoritmo que permita ingresa un día y un mes del año (validar), de modo que nos muestre a que signo zodiacal pertenecen los datos ingresados.

c) Realiza una función separar(lista), que tome una lista de números enteros y devuelva dos listas ordenadas. La primera con los números pares y la segunda con los números impares.

d) El siguiente ejercicio, fue planteado en la unidad anterior, en esta oportunidad deberá implementar las funciones que considere necesarias para mejorar el código:

   Se pretende simular la tabla clientes de una base de datos, para ello se debe considerar los siguientes campos que se deben ingresar en la lista bidimensional que debe contener 10 registros:

   Código/Nombre/Edad/Nhijos/SBasico

   Deberá calcular el Sueldo Neto a pagar por cada trabajador, para ello considerar que si tiene más de 3 hijos se le debe considerar un 20% adicional sobre el Sbasico en otro caso no aplica; por otro lado, si la edad del trabajador es mayor a 50 años del mismo modo deberá considerarse un 10% adicional sobre el Sbasico.

   Finalmente, deberá generar el código de los clientes: C001, C002, C003…; así mismo validar que la edad debe ser mayor a 18 años y el sueldo básico es como mínimo 1100 soles.

e) Desarrolle una función que permita definir si un número ingresado es capicúa; considerar que puede ser de cualquier número de dígitos.

7. Fuentes consultadas:

A) https://edube.org/learn/programming-essentials-in-python-part-1-spanish
B) Edison Zavaleta C. (2005). *Fundamentos de Programación*. Perú, Editorial Abaco-Lima.