Funciones

Clase #07

IIC1103 – Introducción a la Programación

Marcos Sepúlveda (marcos@ing.puc.cl)

Veremos hoy …

- Definición y uso de **Funciones**
- Casos especiales
  - Funciones que ya hemos usado
  - Funciones que no retornan nada
  - Funciones que retornan múltiples valores
  - Funciones pre-definidas

Motivación

- Queremos crear un programa que nos permita calcular el coeficiente binomial (corresponde al número de formas en que se pueden extraer subconjuntos de tamaño **n** a partir de un conjunto de **m** elementos) dado por:

$$C(m,n) = \frac{m!}{(m-n)!\ n!}$$

- Donde **n!** (factorial) es:

$$n! = 1 \times 2 \times \cdots \times (n-1) \times n$$

¿Cómo calculamos el factorial de un número?

$n!$

```
print("Calcular factorial de un número")

numero = int(input("Dime un número: "))

i = 1
factorial = 1

while i <= numero:
    factorial *= i
    i += 1

print("El factorial de " + str(numero) + " es " + str(factorial))
```

```
>>>
Calcular factorial de un número
Dime un número: 5
El factorial de 5 es 120
```

Coeficiente binomial

$$C(m,n) = \frac{m!}{(m-n)!\ n!}$$

```
print("Calcular coeficiente binominal")

m = int(input("Dime el valor de m: "))
n = int(input("Dime el valor de n: "))

numero = m

factorial = 1
i = 1
while i <= numero:
    factorial *= i
    i += 1

componente1 = factorial

numero = m-n

factorial = 1
i = 1
while i <= numero:
    factorial *= i
    i += 1

componente2 = factorial
```

$m!$

$(m-n)!$

Coeficiente binomial

$$C(m,n) = \frac{m!}{(m-n)!\ n!}$$

```
numero = n

factorial = 1
i = 1
while i <= numero:
    factorial *= i
    i += 1

componente3 = factorial
```

$n!$

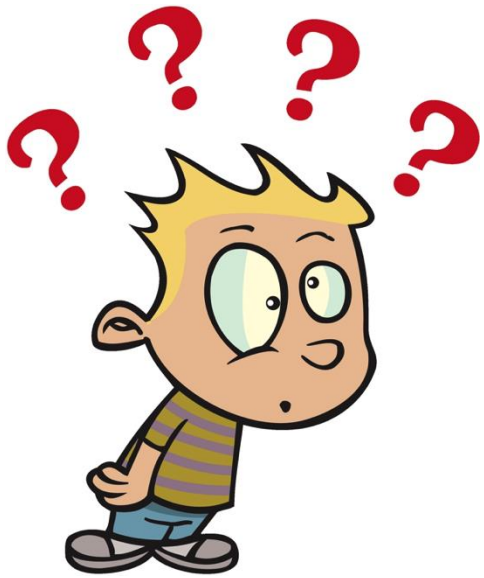
```
combinatoria = componente1 // (componente2 * componente3)

print("C(" + str(m) + "," + str(n) + ") = " + str(combinatoria))
```

```
>>>
Calcular coeficiente binominal
Dime el valor de m: 5
Dime el valor de n: 2
C(5,2) = 10
```

Problemas del código anterior

- Escribir el código anterior nos puede llevar a cometer errores.
- Código potencialmente difícil de entender.
- No se puede reutilizar.



¿Cómo podemos mejorar este código?

¡ Utilizando funciones !

¿Cómo podemos mejorar este código?

- Podemos utilizar una función llamada **Factorial** que nos permita realizar el cálculo de:

$$C(m,n) = \frac{m!}{(m-n)!\ n!}$$

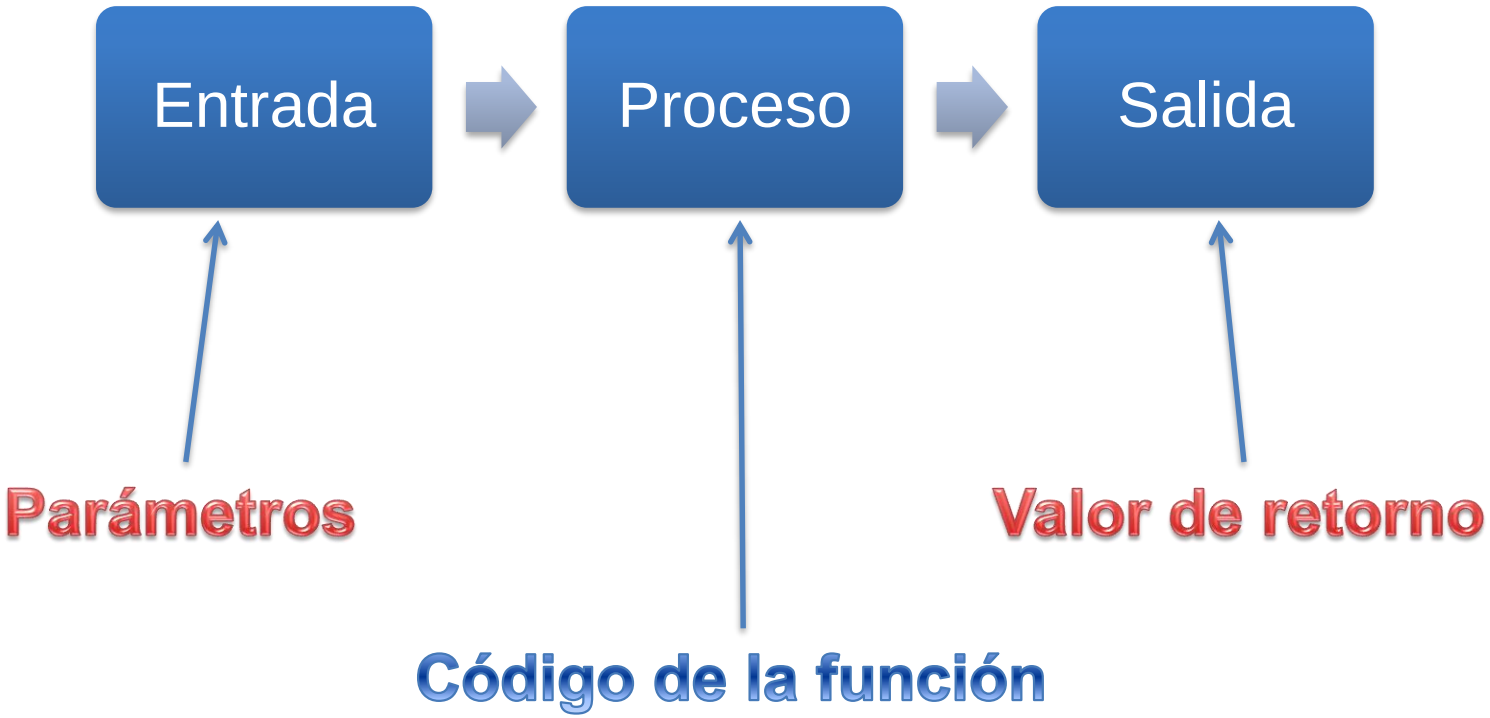
- ¿Cómo?

```
combinatorial = Factorial(m) / (Factorial(m-n) * Factorial(n))
```

Funciones

- **Función**: es una sección de un programa que realiza una tarea específica de manera independiente al resto del programa

- Tienen 3 componentes importantes:



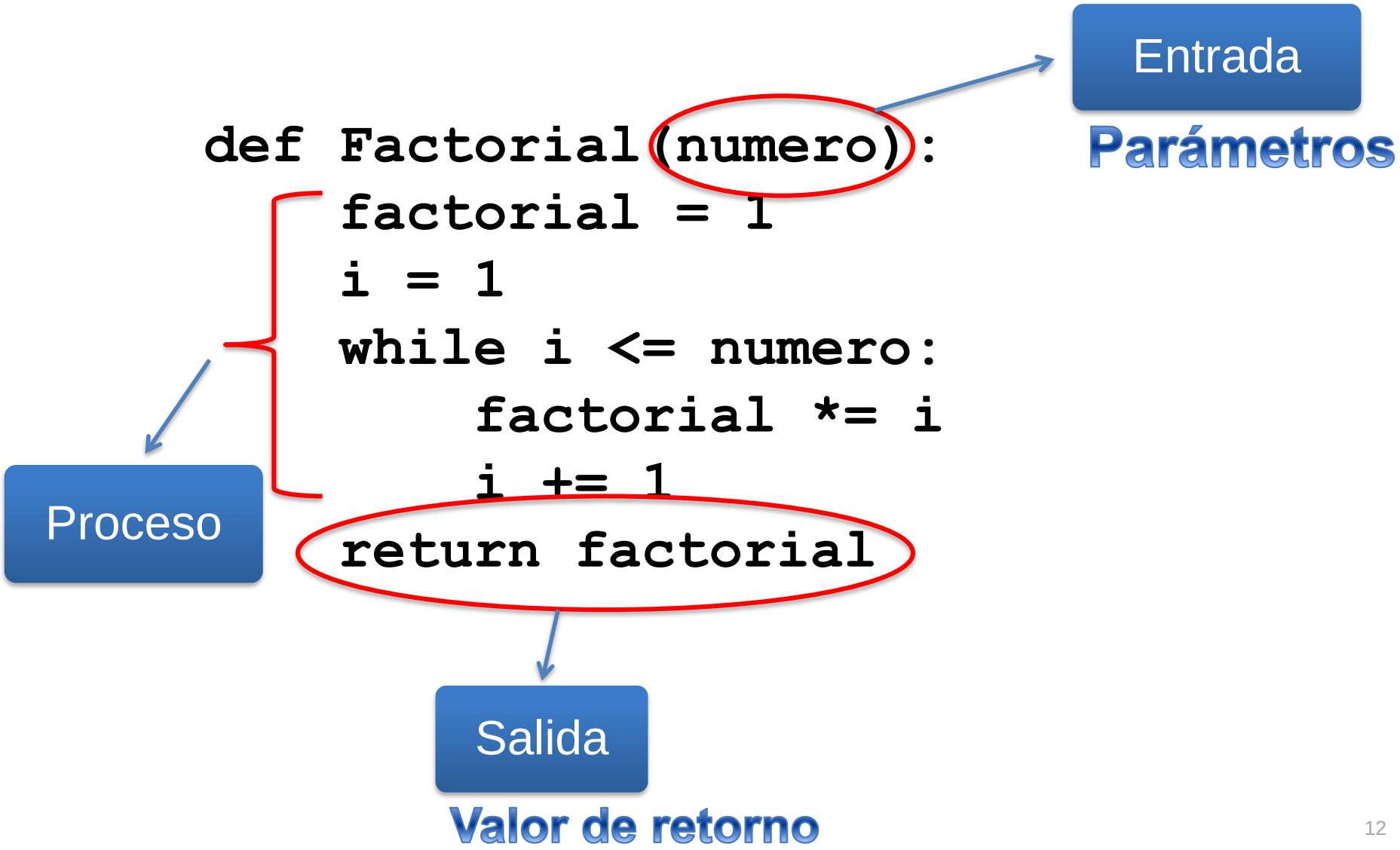
Creación de funciones

- Las funciones en Python son creadas mediante la sentencia `def`:

```
def nombre(parametros):
    #....
    #codigo de la funcion
    #....
    return valor_retorno
```

Función Factorial

$n!$



Función Factorial

$n!$

```
def Factorial(numero):
    factorial = 1
    i = 1
    while i <= numero:
        factorial *= i
        i += 1
    return factorial

print("Calcular factorial de un número")

numero = int(input("Dime un número: "))

factorial = Factorial(numero)

print("El factorial de " + str(numero) + " es " + str(factorial))
```

```
>>>
Calcular factorial de un número
Dime un número: 5
El factorial de 5 es 120
```

Coeficiente binomial – con funciones

$$C(m,n) = \frac{m!}{(m-n)!\ n!}$$

```
def Factorial(numero):
    factorial = 1
    i = 1
    while i <= numero:
        factorial *= i
        i += 1
    return factorial

print("Calcular coeficiente binominal")

m = int(input("Dime el valor de m: "))
n = int(input("Dime el valor de n: "))

combinatoria = Factorial(m) // (Factorial(m-n) * Factorial(n))

print("C(" + str(m) + "," + str(n) + ") = " + str(combinatoria))
```

Coeficiente binomial – solo funciones

$$C(m,n) = \frac{m!}{(m-n)!\ n!}$$

```
def Factorial(numero):
    factorial = 1
    i = 1
    while i <= numero:
        factorial *= i
        i += 1
    return factorial

def Combinatoria(m, n):
    return Factorial(m) // (Factorial(m-n) * Factorial(n))

print("Calcular coeficiente binominal")

m = int(input("Dime el valor de m: "))
n = int(input("Dime el valor de n: "))

combinatoria = Combinatoria(m, n)

print("C(" + str(m) + "," + str(n) + ") = " + str(combinatoria))
```

¿Por qué funciones?

- Crear nuevas funciones nos entrega la oportunidad de nombrar y luego repetir un grupo de sentencias, lo que hace el programa más fácil de leer y de depurar (debugging).

- Las funciones pueden hacer un programa más pequeño, eliminando código repetitivo.

- Además, si más tarde se necesita realizar un cambio, se realiza en un solo lugar.

- Dividir un programa largo en funciones permite debuggear las partes una a una.

- Funciones bien diseñadas a menudo son útiles para escribir otros programas.

Algunas funciones que ya hemos usado

Funciones que convierten:

- `n = int(sys.stdin.readline())`
- `f = float(sys.stdin.readline())`
- `print("El número n vale: " + str(n))`

Algunas funciones que ya hemos usado

Funciones Input/Output:

- `n = int(sys.stdin.readline())`
- `n = int(input("Ingresa un número: "))`
- `print("Hoy estamos aprendiendo funciones")`

Funciones que no retornan nada

- Una función puede realizar acciones sin necesariamente retornar un resultado.

- Ejemplo:

```
def MostrarDatos(nombre, apellido, rut, dia, mes, año):
    print("Nombre completo: ", nombre, apellido)
    print("Rut: ", rut)
    print("Fecha de nacimiento: ", dia, "/", mes, "/", año)
    print("------")

MostrarDatos("Fulano", "de tal", "12345678-9", 1, 1, 1901)
MostrarDatos("Juan", "Pérez", "9867432-1", 31, 12, 2001)
MostrarDatos("Pedro", "González", "22333444-5", 6, 6, 1966)
```

Funciones que retornan múltiples valores

- Una función puede retornar más de un valor y de cualquier tipo.
- Los valores retornados se deben separar por comas.
- Al invocar a la función, se debe asignar sus valores resultantes a un grupo de variables separadas por comas.

- Ejemplo:
  - Función recibe tiempo en segundos, y retorna el equivalente en horas, minutos y segundos:

```
def Convertir_Segundos(tiempo):
    horas = tiempo // (60*60)
    tiempo_restante = tiempo % (60*60)
    minutos = tiempo_restante // 60
    segundos = tiempo_restante % 60
    return horas, minutos, segundos

tQuedaDeClases = int(input("Uff, ¿Cuántos segundos quedan de clases? "))
horas, minutos, segundos = Convertir_Segundos(tQuedaDeClases)

print(tQuedaDeClases, "segundos equivalen a:")
print(horas, minutos, segundos, sep=':')
```

```
>>>
Uff, ¿Cuántos segundos quedan de clases? 3665
3665 segundos equivalen a:
1:1:5
```

Funciones pre-definidas

- Python posee funciones pre-definidas que están siempre disponibles.
- Por ejemplo: ***abs***, ***float***, ***input***, ***int***, ***print***, ***str***
- Ver detalles en:
  - https://docs.python.org/3/library/functions.html

Ejemplo – calcular valor absoluto

- Creando una función

```
def ValorAbsoluto(numero):
    if (numero < 0):
        valor_absoluto = -numero
    else:
        valor_absoluto = numero
    return valor_absoluto

x = float(input("x: "))
print("El valor absoluto de:", x, "es", ValorAbsoluto(x))
```

- Usando ***abs***

```
x = float(input("x: "))
print("El valor absoluto de:", x, "es", abs(x))
```

Ejercicios

1. Crear un programa que determine si un número es palíndromo
2. Crear un programa que transforme un número de base decimal a base binaria
3. Crear un programa que liste los primeros 20 números de la serie de Fibonacci