Universidad Católica
Silva Henríquez

# Fundamentos de Programación
# Ayudantía 8

Alexander Inostroza
Felipe Zambrano

# Funciones

```
1  SubAlgoritmo resultado ← Suma(a,b)
2      resultado ← a+b
3  FinSubAlgoritmo
4
5  Algoritmo Principal
6      Escribir Suma(7,5)
7  FinAlgoritmo
8
```

PSeInt - Ejecutando proceso PRINCIPAL

```
*** Ejecución Iniciada. ***
12
*** Ejecución Finalizada. ***
```

```
def suma(a,b):
    resultado = a+b
    return resultado

print(suma(7,5))
```

```
>>>
= RESTART: C:/Users/A
e/fundamentos_ayudant
12
>>>
```

# Ejercicio 1

```
import random

def generar_numero_secreto(limite_inferior,
limite_superior):
    return random.randint(limite_inferior, limite_superior)
```

Utilizando lo anterior,

Generar un número secreto: Utiliza la función **generar_numero_secreto(limite_inferior, limite_superior)** proporcionada (ver código arriba) para generar el número secreto. Esta función recibe dos argumentos: el límite inferior y el límite superior del rango, ambos inclusive.

Pedir al usuario que adivine: En un bucle, solicita al usuario que ingrese su intento de adivinar el número.

Dar retroalimentación:

Si el número ingresado es menor que el número secreto, imprime "Demasiado bajo".

Si el número ingresado es mayor que el número secreto, imprime "Demasiado alto".

o si acertó.

# Ejercicio 2

Primero deberán escribir esto en su editor de python.

```
texto_ejemplo = "Hola, Mundo!"
numero_vocales = contar_vocales(texto_ejemplo)
print(f"El texto '{texto_ejemplo}' tiene {numero_vocales} vocales.")

texto_ejemplo2 = "Este es OTRO ejemplo."
numero_vocales2 = contar_vocales(texto_ejemplo2)
print(f"El texto '{texto_ejemplo2}' tiene {numero_vocales2} vocales.")

texto_ejemplo3 = "PYTHON es genial"
numero_vocales3 = contar_vocales(texto_ejemplo3)
print(f"El texto '{texto_ejemplo3}' tiene {numero_vocales3} vocales.")
```

Ahora deben definir la función **contar_vocales** para poder hacer funcionar correctamente lo anterior.

# Ejercicio 3

Defina la función **mayor_y_menor**, que recibe un string con números enteros, separados por espacios y, retorna un string con el número mayor y el número menor de la secuencia, separados por un espacio.

```
mayor_y_menor("1 2 3 4 5")  => retorna "5 1"

mayor_y_menor("1 2 -3 4 5") => retorna "5 -3"

mayor_y_menor("1 9 3 4 -5") => retorna "9 -5"
```

# Ejercicio 4

El string `"X - O\nO X -\n- - X"` representa el estado de una partida de gato (tic tac toe, tres en linea) donde los símbolos `X` y `O` representan las jugadas de los distintos jugadores y `-` representa una casilla en la que aún no se realiza ninguna jugada. Defina la función **jugar**, que reciba como parámetros el string que representa el estado actual del juego, un string(`"X"`,`"O"`) indicativo del turno del jugador correspondiente y un número para la casilla en la que se desea realizar la jugada, como se aprecia en la imágen de más abajo; y retorne el tablero tras realizar la jugada deseada. En caso de que la casilla ya haya sido utilizada previamente, debe retornar el tablero original sin cambios.

| | | |
|---|---|---|
| 1 | 2 | 3 |
| 4 | 5 | 6 |
| 7 | 8 | 9 |

Propuesto: defina la función ganador, que reciba un tablero como parámetro y entregue un string como respuesta: `"X"`, `"O"` si alguno de los jugadores ganó, `"-"` si aún ningún jugador gana la partida.