



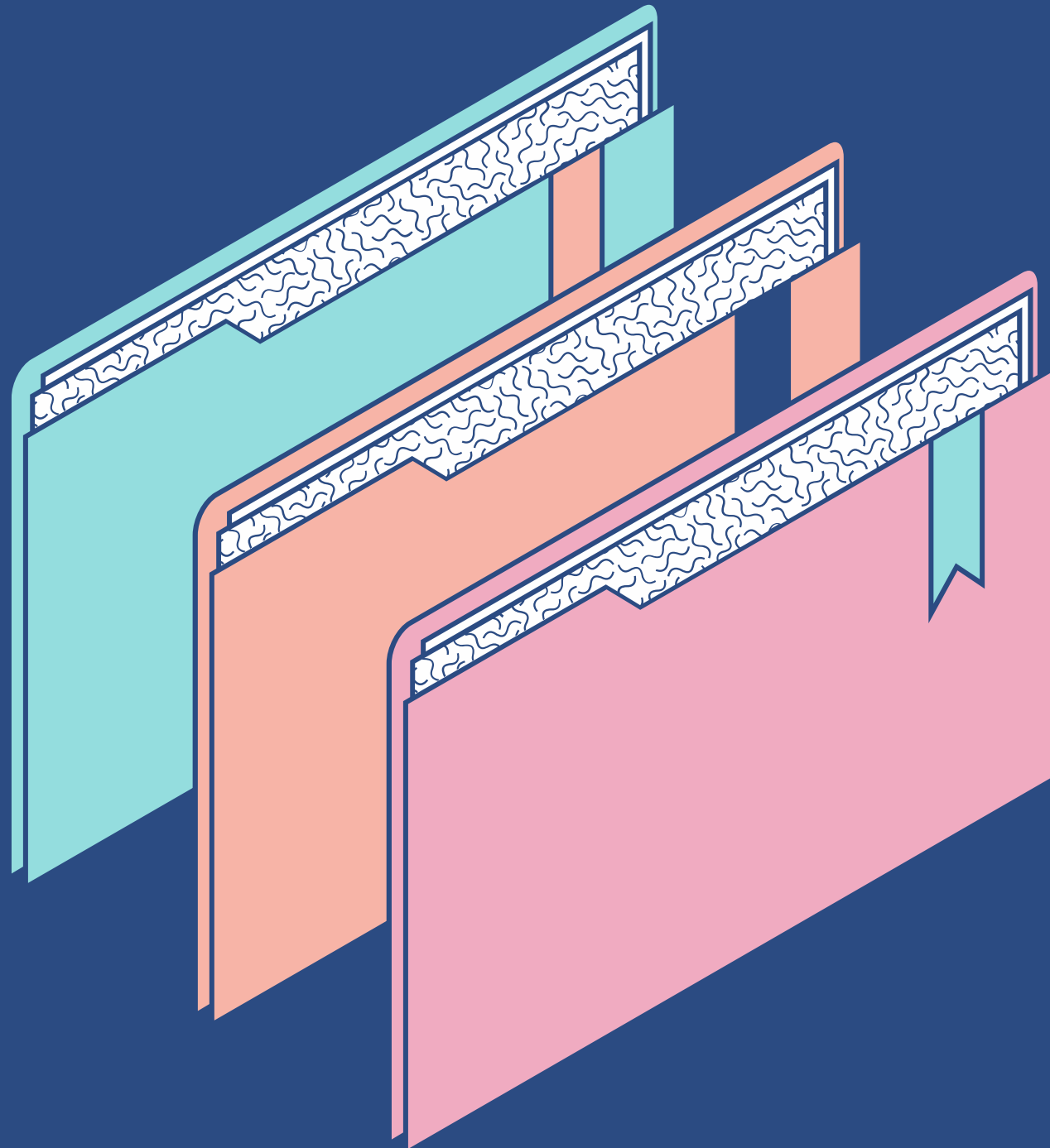
CLASE 17 DE JUNIO

# Ciclos y Funciones

Python desde Cero - NineHub

# Contenido

- Introducción a Ciclos
- While y For
- Uso de Break y Continue
- Definición y llamada de funciones
- Parámetros de una función y valores de retorno



**Tip:** Use links to go to a different page inside your presentation.

**How:** Highlight text, click on the link symbol on the toolbar, and select the page in your presentation you want to connect.

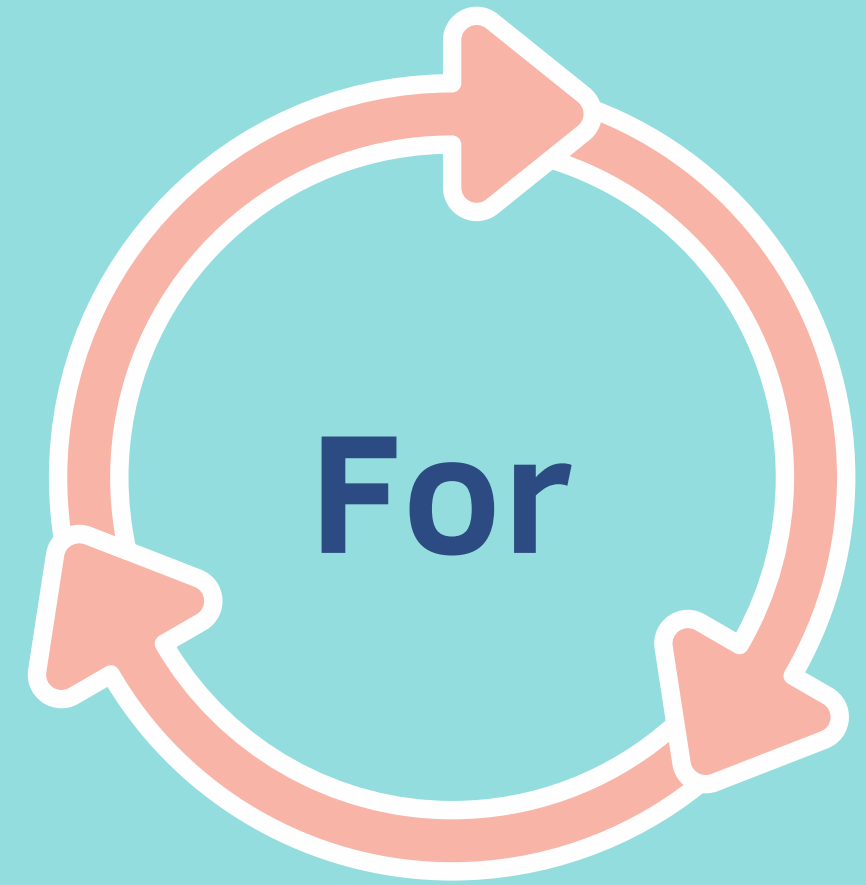
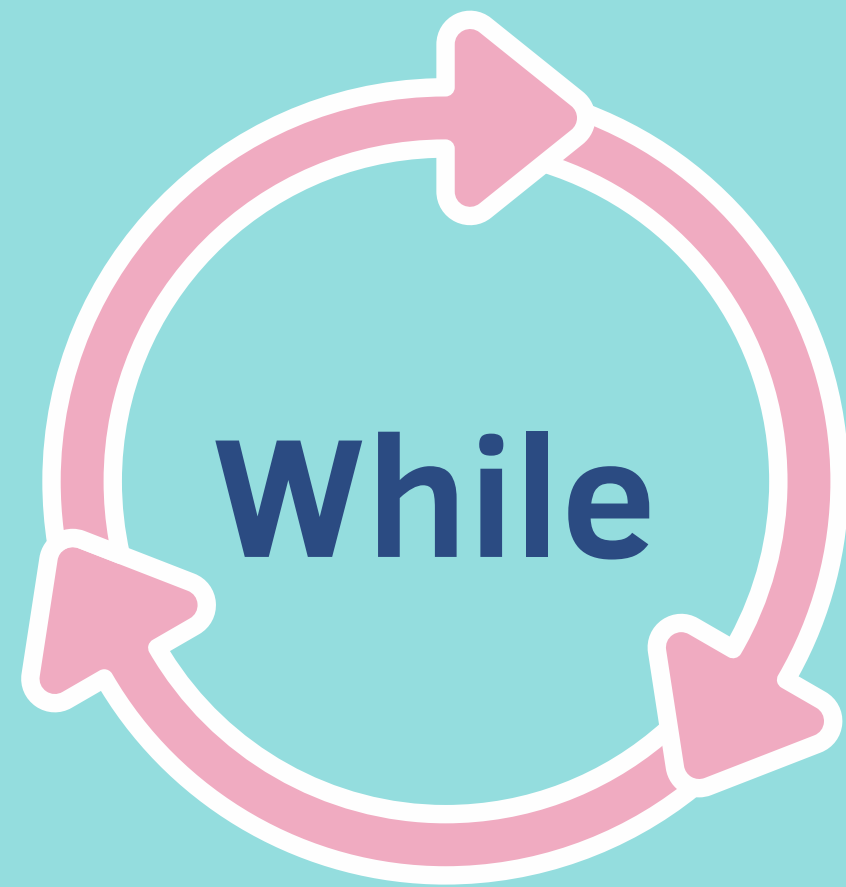


Ciclos

# ¿Qué es un ciclo?

Los ciclos en Python (y en cualquier lenguaje de programación) son usados para ejecutar las mismas instrucciones de repetidas veces.

# Tipos de ciclo



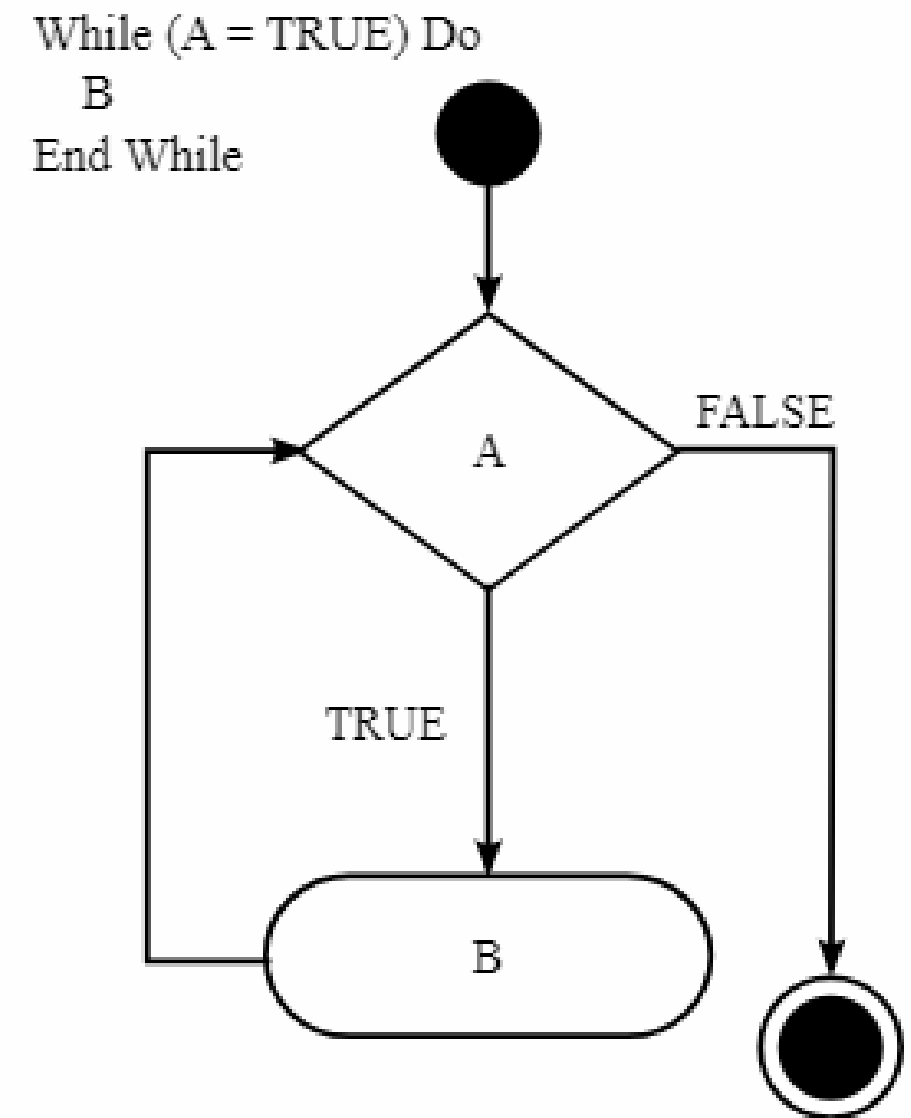
# Ciclo While

Un ciclo while en Python se utiliza para ejecutar repetidamente un bloque de código **mientras** una condición especificada sea verdadera.

## Sintaxis:

**while** condición:

# Bloque de código que se ejecuta mientras la condición sea verdadera



# EJEMPLO Ciclo While

```
contador = 1

while contador <= 5:
    print(contador)
    contador = contador + 1
```

```
>>> %Run
```

```
1
2
3
4
5
```

# EJERCICIO:

- Imprimir los números del 1 al 100
- Imprimir únicamente los números impares del 1 al 50
- Imprimir los siguientes números:
  - 1, 4, 9, 16, 25, 36, 49, 64, 81, 100



# Ciclo For

El ciclo *for* en Python se utiliza para iterar sobre una lista, una tupla, un diccionario, un conjunto, un string o cualquier otro **objeto iterable** (como los generadores). Es útil para ejecutar un bloque de código **un número determinado de veces** o para iterar **sobre elementos de una colección**.

## Sintaxis:

```
for elemento in secuencia :  
    # Bloque de código que se ejecuta para cada  
    elemento en la secuencia
```

# EJEMPLO Ciclo For

```
1 numeros = [1, 2, 3, 4, 5]
2
3 for numero in numeros:
4     print(numero)
5
```

```
>>> %Run
1
2
3
4
5
```

```
1 texto = "Python es un lenguaje de programación fantástico para aprender"
2 contador_a = 0
3
4 for i in range(len(texto)):
5     if texto[i] == 'a':
6         contador_a += 1
7
8 print(f"El número de letras 'a' en el texto es: {contador_a}")
```

El número de letras 'a' en el texto es: 7

# range()

La función **range** devuelve una secuencia de números, comenzando por 0 (por defecto) y con incrementos de 1 (por defecto), y se detiene antes de el número especificado.

## EJEMPLO

```
1 for i in range(5):  
2     print(i)
```

```
>>> %Run  
0  
1  
2  
3  
4
```

# Break

La instrucción **break** se utiliza para **salir de un bucle prematuramente**. Cuando se ejecuta **break**, el control del programa se transfiere a la primera línea de código que sigue inmediatamente después del bucle.

# Continue

La instrucción **continue** se utiliza para **saltar la iteración actual del bucle** y pasar a la siguiente iteración. Cuando se ejecuta **continue**, el bucle no ejecuta las instrucciones restantes en la iteración actual y pasa inmediatamente a la siguiente iteración del bucle.

# EJEMPLO Break

```
numeros = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

# Queremos encontrar el número 5
for numero in numeros:
    if numero == 5:
        print("Número encontrado:", numero)
        break # Salimos del bucle cuando encontramos el número
    print("Revisando número:", numero)
```

```
>>> %Run -c $EDITOR_CO

Revisando número: 1
Revisando número: 2
Revisando número: 3
Revisando número: 4
Número encontrado: 5
```

# EJEMPLO Continue

```
1 numeros = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
2
3 # Queremos imprimir solo los números impares
4 for numero in numeros:
5     if numero % 2 == 0: # Si el número es par
6         continue # Saltamos a la siguiente iteración
7     print("Número impar:", numero)
```

```
>>> %Run -c $EDIT

Número impar: 1
Número impar: 3
Número impar: 5
Número impar: 7
Número impar: 9
```

# EJERCICIO:

- Imprimir los números del 1 al 100 **USANDO CICLO FOR**
- Imprimir únicamente los números impares del 1 al 50 **USANDO CICLO FOR**
- Escribe un programa que calcule el factorial de un número.



# Funciones

# ¿Qué es una función?

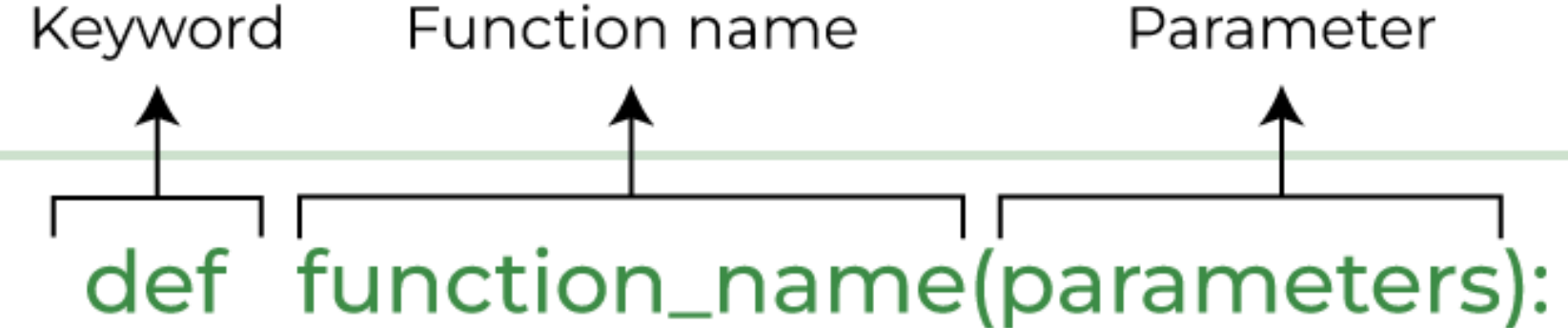
Una función es una serie de instrucciones que tienen asociado un nombre para realizar una tarea y puede devolver un resultado.

```
def MiFuncion(a,b):  
    c = a + b  
    return c
```

```
def MiFuncion2(a,b):  
    c = a*b  
    return c
```



# Encabezado de una función



```
def function_name(parameters):
```

The diagram illustrates the components of a function header in Python. It shows the code `def function_name(parameters):` with three labels above it: 'Keyword' pointing to `def`, 'Function name' pointing to `function_name`, and 'Parameter' pointing to `parameters`. The entire code line is enclosed in a light green box.

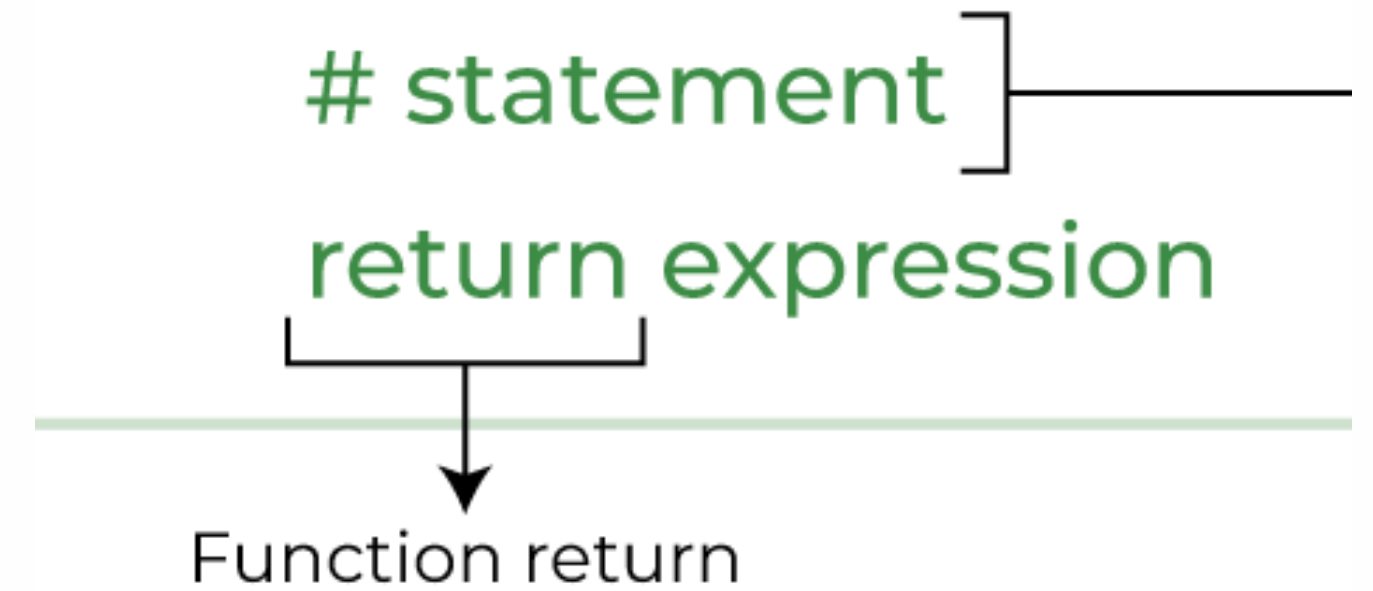
EL ENCABEZADO DE UNA FUNCIÓN ES LA LÍNEA QUE DEFINE LA FUNCIÓN, ESPECIFICANDO SU NOMBRE Y SUS PARÁMETROS, E INCLUYE LA PALABRA CLAVE **DEF**.

LA PALABRA **DEF** ES LA PALABRA RESERVADA EN PYTHON PARA INICIAR A DECLARAR UNA FUNCIÓN

# Cuerpo y Retorno de una función

EL **CUERPO** DE UNA FUNCIÓN EN PYTHON ES EL BLOQUE DE CÓDIGO QUE SE ENCUENTRA INDENTADO DESPUÉS DEL ENCABEZADO DE LA FUNCIÓN Y CONTIENE LAS INSTRUCCIONES QUE SE EJECUTAN CUANDO SE LLAMA A LA FUNCIÓN (STATEMENT).

EL **VALOR DE RETORNO** ES LO QUE DEVUELVE LA FUNCIÓN UNA VEZ SE EJECUTA EL CÓDIGO.



# Ejemplos de funciones

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

```
>>> MiFuncion()  
  
Hola Mundo
```

```
def Suma(numero1,numero2):  
    suma = numero1 + numero2  
    return suma
```

```
>>> Suma(2,3)  
  
5
```

```
def AreaRectangulo():  
    altura = 5  
    base = 3  
    area = altura * base  
    return area
```

```
>>> print("El Area del Rectangulo es",AreaRectangulo())  
  
El Area del Rectangulo es 15
```

# Ejercicio en Clase de Funciones

1. Programa una función que calcule el área de un triángulo. Esta debe recibir al menos dos parámetros y tener un valor de regreso.
2. Programa una función que sea capaz de determinar si un número es par o impar. Debe tener al menos un parámetro y regresar verdadero si es par, falso si no es par.



# Solución Ejercicio 1

```
def AreaTriangulo(base_triangulo, altura_triangulo):  
    area = (base_triangulo * altura_triangulo) / 2  
    return area
```

```
>>> base = 5  
>>> altura = 3  
>>> print("La base del triangulo es", AreaTriangulo(base, altura))  
La base del triangulo es 7.5
```

# Solución Ejercicio 2

```
def NumeroPar(numero):  
    if numero % 2 == 0:  
        return True  
    else:  
        return False
```

```
>>> NumeroPar(2)  
True  
  
>>> NumeroPar(3)  
False
```

# Ejercicio Ciclos y funciones

EN VIRTUD DEL PROBLEMA DE LA CLASE ANTERIOR, VEAMOS EL CASO PARTICULAR DE TIM.

TIM TIENE ACTUALMENTE 14 AÑOS Y QUIERE SABER CUÁNTOS AÑOS LE FALTAN PARA PODER VOTAR EN LAS PRÓXIMAS ELECCIONES. LAS LEYES ELECTORALES ESTIPULAN QUE UNA PERSONA ES CAPAZ DE VOTAR SI TIENE AL MENOS 18 AÑOS, ES UN CIUDADANO REGISTRADO DE LA COMUNIDAD, Y NO TIENE RESTRICCIONES LEGALES QUE LE IMPIDAN VOTAR.

PARA AYUDAR A TIM Y PERSONAS COMO TIM SE TE HA PEDIDO QUE DESARROLLES UN PROGRAMA EN PYTHON QUE:

1. UTILICE UNA FUNCIÓN PARA VERIFICAR SI TIM CUMPLE CON LOS REQUISITOS PARA VOTAR.
2. UTILICE UN CICLO PARA CALCULAR CUÁNTOS AÑOS FALTAN HASTA QUE TIM SEA ELEGIBLE PARA VOTAR.
3. IMPRIMA UN MENSAJE QUE INDIQUE CUÁNTOS AÑOS LE FALTAN A TIM PARA PODER VOTAR.

ALGUNAS CONSIDERACIONES QUE DEBES TOMAR EN CUENTA SON LAS SIGUIENTES:

- LA FUNCIÓN DEBE RECIBIR LA EDAD DE TIM, SU CIUDADANÍA Y SI TIENE CONDENAS
- EL CICLO DEBE AUMENTAR LA EDAD DE TIM AÑO POR AÑO HASTA QUE CUMPLA CON LA EDAD MÍNIMA PARA VOTAR
- EL PROGRAMA DEBE IMPRIMIR CUÁNTOS AÑOS LE FALTAN A TIM PARA PODER VOTAR.

ADICIONAL AL CONTENIDO QUE HAZ APRENDIDO HOY, PUEDES (Y DE HECHO DEBES) USAR EL CONTENIDO APRENDIDO EN CLASES ANTERIORES ¡ÉXITOS!