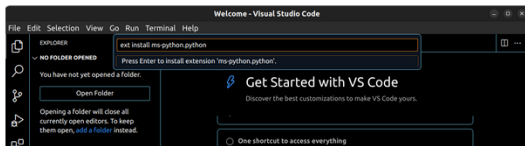


Introducción a la programación

Práctica 6: Introducción al Lenguaje Imperativo

Usamos VS para Python

- ▶ Nos aseguramos de tener Python instalado en la compu: abrimos una terminal y escribimos `Python3`, Enter. Esto abre la consola interactiva de Python. Podemos cerrarla usando `quit()` o matar el proceso usando `CTRL+C`.
- ▶ Abrimos VSCode
- ▶ Instalamos la extensión: `ms-python`. En la ventana de Inicio acceder al buscador (al cual se llega con `Ctrl+P` / `Command+P`): `ext install ms-python.python`



Empezando con Python

Primeros pasos:

- ▶ Por ahora vamos a trabajar similar que con Haskell
 1. Hacemos nuestro archivo con las funciones (`test.py`)
 2. Abrimos el compilador interactivo (`python` o `python3`, ver version)
 3. Importamos el archivo con nuestras funciones (`import test`)
 4. Las usamos interactivamente (`test.<nombre_funcion>`)
 5. Para recargar el archivo hay que cerrar el intérprete (`exit()`) y volver a empezar

Empezando con Python

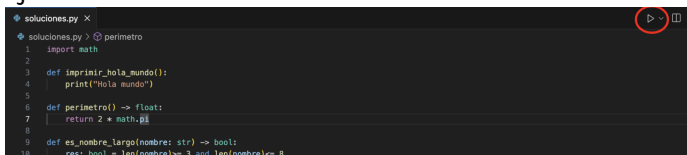
Primeros pasos:

- ▶ También podemos ejecutar todo el script desde consola (`python3 test.py`)
 - ▶ En este caso además de las definiciones de las funciones tiene que haber código “libre” que las ejecute

Empezando con Python

Primeros pasos:

- ▶ También podemos ejecutar todo el script desde consola (`python3 test.py`)
 - ▶ En este caso además de las definiciones de las funciones tiene que haber código “libre” que las ejecute
- ▶ En lugar de ejecutar desde consola, si tenemos instalada la extensión de Python en el Visual Studio Code, podemos ejecutarlo desde ahí:

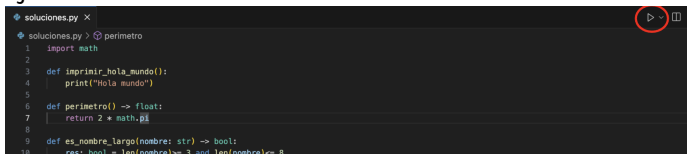


```
soluciones.py x
soluciones.py > perimetro
1 import math
2
3 def imprimir_hola_mundo():
4     print("Hola mundo")
5
6 def perimetro() -> float:
7     return 2 * math.pi
8
9 def es_nombre_largo(nombre: str) -> bool:
10    res: bool = len(nombre) >= 3 and len(nombre) <= 8
```

Empezando con Python

Primeros pasos:

- ▶ También podemos ejecutar todo el script desde consola (`python3 test.py`)
 - ▶ En este caso además de las definiciones de las funciones tiene que haber código “libre” que las ejecute
- ▶ En lugar de ejecutar desde consola, si tenemos instalada la extensión de Python en el Visual Studio Code, podemos ejecutarlo desde ahí:



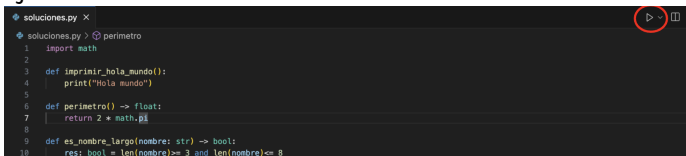
```
soluciones.py x
soluciones.py > perimetro
1 import math
2
3 def imprimir_hola_mundo():
4     print("Hola mundo")
5
6 def perimetro() -> float:
7     return 2 * math.pi
8
9 def es_nombre_largo(nombre: str) -> bool:
10    res: bool = len(nombre) >= 3 and len(nombre) <= 8
```

- ▶ O usar las herramientas de *debugging* que vamos a ver más adelante

Empezando con Python

Primeros pasos:

- ▶ También podemos ejecutar todo el script desde consola (`python3 test.py`)
 - ▶ En este caso además de las definiciones de las funciones tiene que haber código “libre” que las ejecute
- ▶ En lugar de ejecutar desde consola, si tenemos instalada la extensión de Python en el Visual Studio Code, podemos ejecutarlo desde ahí:



```
soluciones.py x
soluciones.py > perimetro
1 import math
2
3 def imprimir_hola_mundo():
4     print("Hola mundo")
5
6 def perimetro() -> float:
7     return 2 * math.pi
8
9 def es_nombre_largo(nombre: str) -> bool:
10    res: bool = len(nombre) >= 3 and len(nombre) <= 8
```

- ▶ O usar las herramientas de *debugging* que vamos a ver más adelante
- ▶ Para comentar una línea de código, la misma debe comenzar con `#` (en Haskell lo hacíamos con `--`)
`# Esto es código comentado`

Ej 1.1

Ejercicio 1. Definir las siguientes funciones y procedimientos

1. problema imprimir_hola_mundo () {
 requiere: { True }
 asegura: { imprime 'Hola Mundo'por consola}
}

Ejercicio 1.

```
5. problema perimetro () {  
    requiere: { True }  
    asegura: {  $res = 2 * \pi$  }  
}
```

Ej 2.5

Ejercicio 2. Definir las siguientes funciones y procedimientos con parámetros

5. problema es_multiplo_de (in $n: \mathbb{Z}$, in $m: \mathbb{Z}$) : Bool {
 requiere: { $m \neq 0$ }
 asegura: { $res = True \leftrightarrow (\text{existe un } k \in \mathbb{Z} \text{ tal que}$
 $n = m \times k)$ }
}

Ej 3.3

Ejercicio 3. Resuelva los siguientes ejercicios utilizando los operadores lógicos `and`, `or`, `not`. Resolverlos sin utilizar alternativa condicional (`if`).

3. problema `es_nombre_largo` (in `nombre: String`) : `Bool` {
 requiere: { `True` }
 asegura: { $res = True \leftrightarrow 3 \leq |nombre| \leq 8$ }
}

Ej 5.1

Ejercicio 5. Implementar los siguientes problemas de alternativa condicional (if/else). Los enunciados pueden no ser del todo claros, especificar los problemas en nuestro lenguaje de especificación sin utilizar `IfThenElseFi` y programar en base a tú propuesta de especificación.

1. `devolver_el_doble_si_es_par(un_numero)`. Que devuelve el doble del número en caso de ser par y el mismo número en caso contrario.

Ej 6.2

Ejercicio 6. Implementar las siguientes funciones usando repetición condicional `while`.

2. Escribir una función que imprima los números pares entre el 10 y el 40.
4. Escribir una función de cuenta regresiva para lanzar un cohete. Dicha función irá imprimiendo desde el número que me pasan por parámetro (que será positivo) hasta el 1, y por último "Despegue".

Ej 6.2

Ejercicio 6. Implementar las siguientes funciones usando repetición condicional `while`.

2. Escribir una función que imprima los números pares entre el 10 y el 40.
4. Escribir una función de cuenta regresiva para lanzar un cohete. Dicha función irá imprimiendo desde el número que me pasan por parámetro (que será positivo) hasta el 1, y por último "Despegue".

Ejercicio 7. Implementar las funciones del ejercicio 6 utilizando `for num in range(i,f,p):`.

Ej similar 9

Analizar el siguiente código:

```
def ejemploArgumento(xArgumento: int):  
    print("En ejemploArgumento: ", xArgumento)  
    xArgumento = xArgumento + 40
```

```
def ejemploVarGlobal():  
    global xGlobal  
    print("En ejemploVarGlobal: ", xGlobal)  
    xGlobal = xGlobal + 30
```

```
xGlobal: int = 1  
ejemploArgumento(xGlobal)  
print("En codigo libre: ", xGlobal)  
ejemploVarGlobal()  
print("En codigo libre: ", xGlobal)  
ejemploArgumento(xGlobal)  
print("En codigo libre: ", xGlobal)  
ejemploVarGlobal()  
print("En codigo libre: ", xGlobal)
```

Ej similar 9

Analizar el siguiente código:

```
def ejemploReturn(xArgumento: int) -> int:
    print("En ejemploArgumento: ", xArgumento)
    xArgumento = xArgumento + 40
    return xArgumento
```

```
def ejemploVarGlobal():
    global xGlobal
    print("En ejemploVarGlobal: ", xGlobal)
    xGlobal = xGlobal + 30
```

```
xGlobal: int = 1
xGlobal = ejemploReturn(xGlobal)
print("En codigo libre: ", xGlobal)
ejemploVarGlobal()
print("En codigo libre: ", xGlobal)
xGlobal = ejemploReturn(xGlobal)
print("En codigo libre: ", xGlobal)
ejemploVarGlobal()
print("En codigo libre: ", xGlobal)
```