

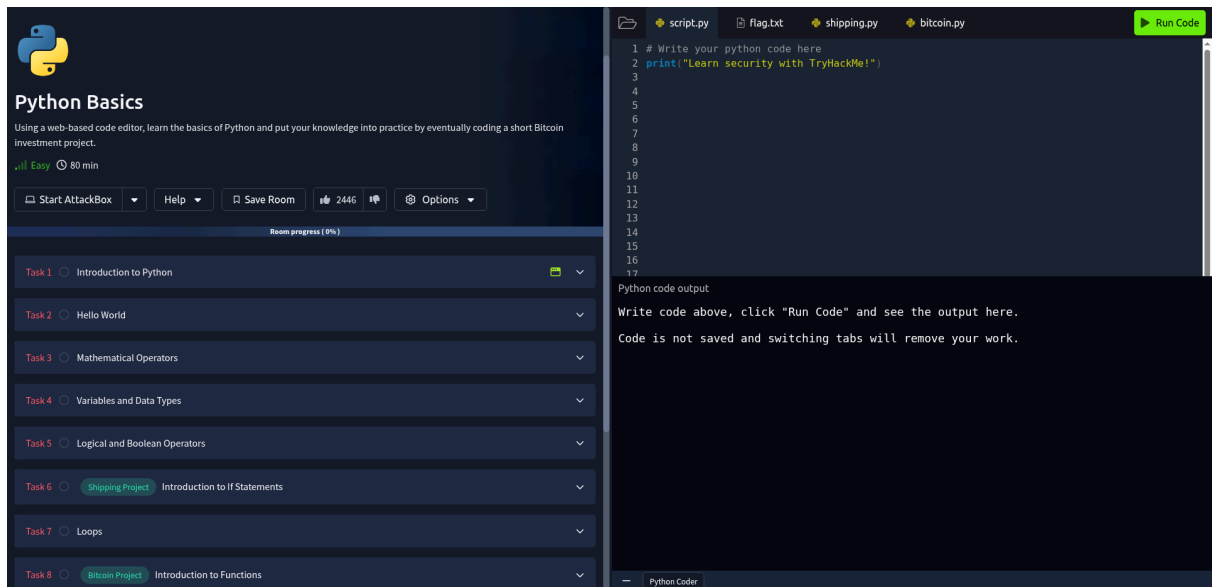
Conceptos básicos de Python

Utilizando un editor de código basado en la web, aprenda los conceptos básicos de Python y ponga sus conocimientos en práctica codificando eventualmente un proyecto corto de inversión en Bitcoin.

Tarea 1: Introducción a Python.

En este apartado, aprenderemos a utilizar el lenguaje de programación de Python, si bien no es necesaria para tener éxito en seguridad, es una gran habilidad.

- Hacemos Click en “View Site” para iniciar la máquina virtual que utilizaremos en este módulo.

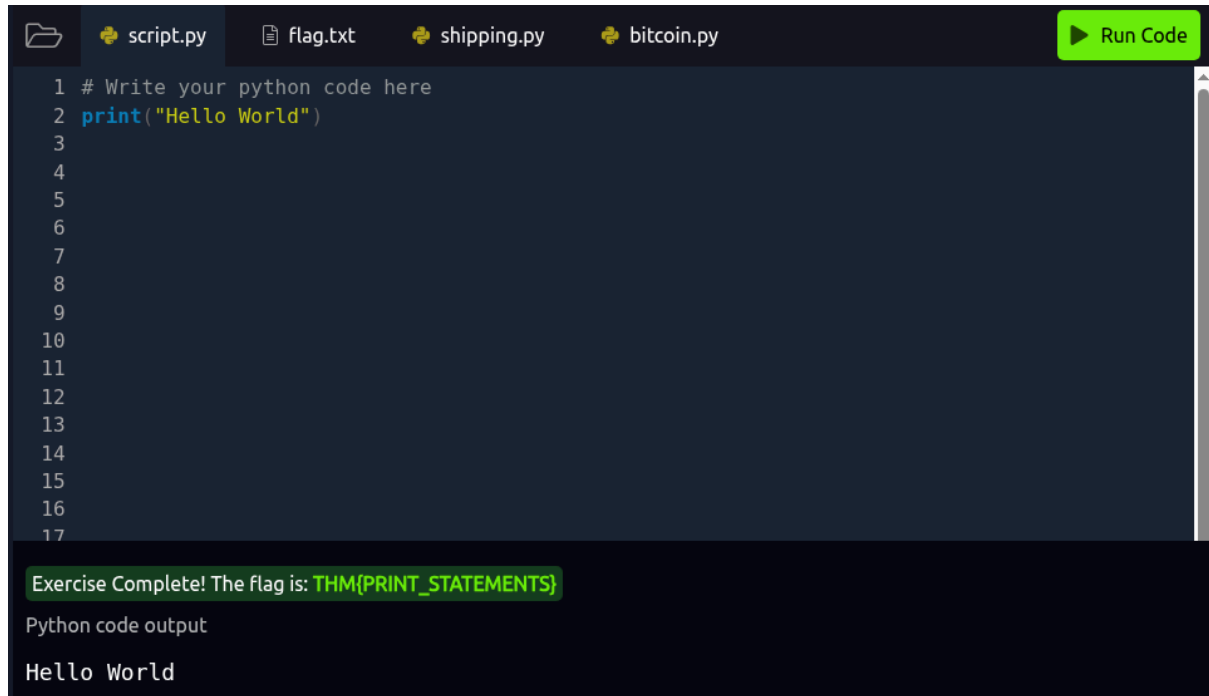


- ❖ **Respuesta:** Simplemente hacer clic para Enviar (no se necesita respuesta).

Tarea 2: Hello World.

Comenzaremos con el armado de un sistema sencillo, creemos un programa simple que genere algo de texto.

Utilizaremos “print()” como declaración, y todo lo que esté dentro de “()” será lo que va a imprimir, teniendo en cuenta que el mensaje debe estar en medio de “”.



The screenshot shows a code editor interface with a dark theme. At the top, there is a tab bar with four tabs: 'script.py' (active), 'flag.txt', 'shipping.py', and 'bitcoin.py'. To the right of the tabs is a green 'Run Code' button. The main editor area contains the following Python code:

```
1 # Write your python code here
2 print("Hello World")
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
```

Below the code editor, there is a green notification bar that says: "Exercise Complete! The flag is: THM{PRINT_STATEMENTS}". Below that, there is a section labeled "Python code output" which displays the text "Hello World".

- **Ejemplo:** print("Hello World") = Al ejecutarlo imprimirá el mensaje dentro.
- ❖ **Pregunta:** On the code editor, print "Hello World". What is the flag?
 - **Nota:** En la consola escribimos el comando “print(“Hello World”)” y nos dará la respuesta.
 - **Respuesta:** THM{PRINT_STATEMENTS}

Tarea 3: Operadores Matemáticos.

Los operadores matemáticos en Python permiten sumar, restar, multiplicar y dividir, funcionando como una calculadora. Los operadores de comparación se usan para evaluar condiciones dentro del programa y son esenciales para estructuras como bucles e instrucciones if.

- ❖ **Pregunta:** In the code editor, print the result of $21 + 43$. What is the flag?
 - **Nota:** Colocamos en la consola `"print(21+43)"`
 - **Respuesta:** THM{ADDITION}
- ❖ **Pregunta:** Print the result of $142 - 52$. What is the flag?
 - **Nota:** Colocamos en la consola `"print(142-52)"`
 - **Respuesta:** THM{SUBTRACT}
- ❖ **Pregunta:** Print the result of $10 * 342$. What is the flag?
 - **Nota:** Colocamos en la consola `"print(10*342)"`
 - **Respuesta:** THM{MULTIPLICATION_PYTHON}
- ❖ **Pregunta:** Print the result of 5 squared. What is the flag?
 - **Nota:** Colocamos en la consola `"print(5**2)"`
 - **Respuesta:** THM{EXPONENT_POWER}

Tarea 4: Variables y tipos de datos.

Las variables permiten almacenar y actualizar datos en un programa. Puedes asignar un valor a una variable y cambiarlo luego en el código, facilitando el manejo de la información durante la ejecución del programa.

- ❖ **Pregunta:** In the code editor, create a variable called height and set its initial value to 200.

```
height = 200
```

- **No answer needed**

- ❖ **Pregunta:** On a new line, add 50 to the height variable.

```
height = height + 50
```

- **No answer needed**

- ❖ **Pregunta:** On another new line, print out the value of height. What is the flag that appears?

```
height = 200
height = height + 50
print(height)
```

- **Respuesta:** THM{VARIABLES}

Tarea 5: Operadores lógicos y booleanos

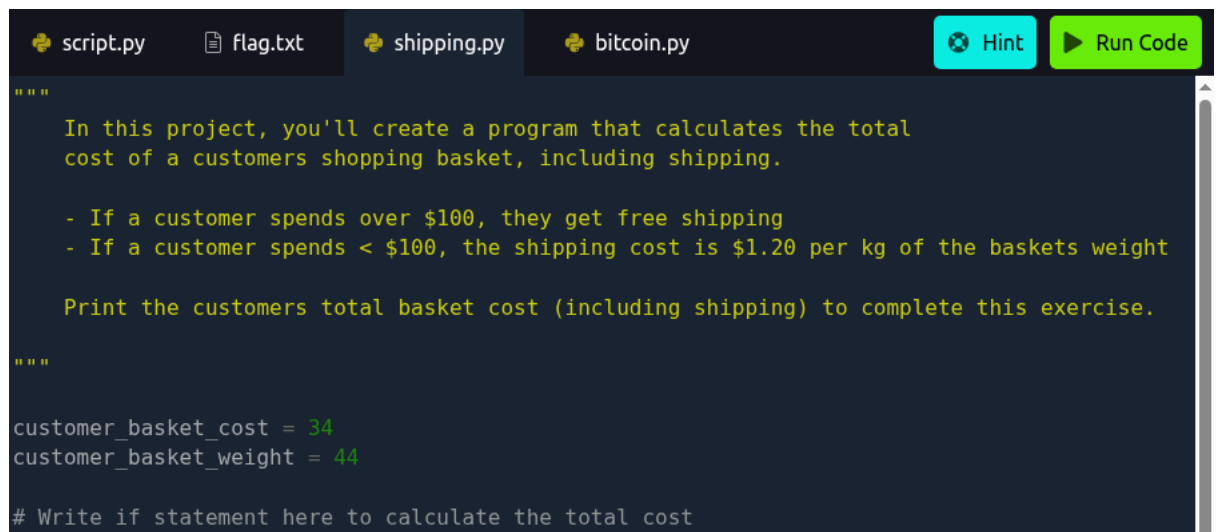
Los operadores lógicos permiten realizar comparaciones y se usan en condiciones como las instrucciones if. Los operadores booleanos conectan y comparan declaraciones, evaluando si las condiciones son verdaderas o falsas.

- ❖ **Respuesta:** Simplemente hacer clic para Enviar (no se necesita respuesta).

Tarea 6: Introducción a las declaraciones If.

El uso de "declaraciones if" permite a los programas tomar decisiones. Permiten que un programa tome una decisión basándose en una condición.

- ❖ **No answer needed**



The screenshot shows a code editor with four tabs: script.py, flag.txt, shipping.py, and bitcoin.py. The shipping.py tab is active. The code in the editor is as follows:

```
"""
In this project, you'll create a program that calculates the total
cost of a customers shopping basket, including shipping.

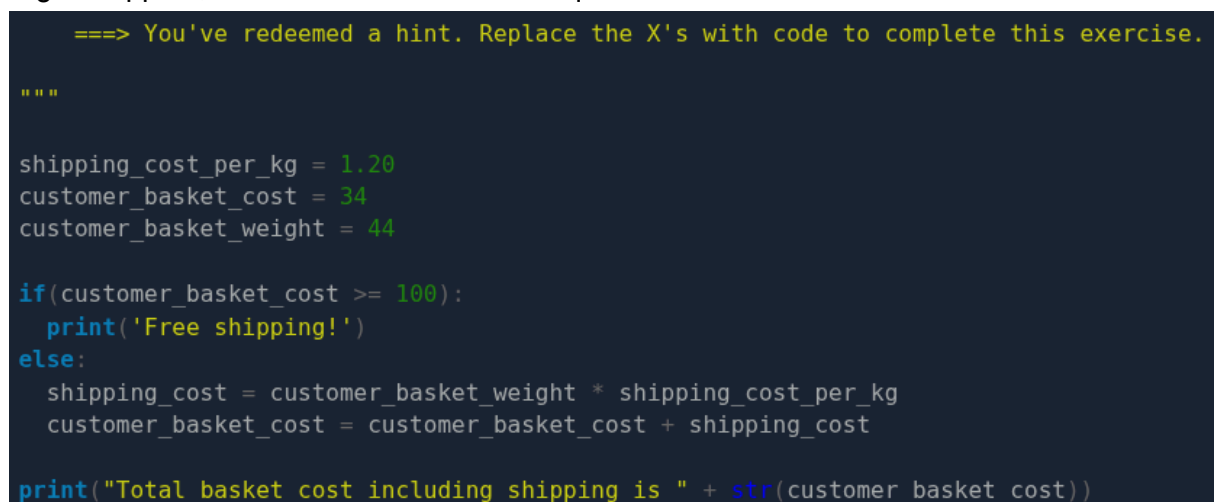
- If a customer spends over $100, they get free shipping
- If a customer spends < $100, the shipping cost is $1.20 per kg of the baskets weight

Print the customers total basket cost (including shipping) to complete this exercise.
"""

customer_basket_cost = 34
customer_basket_weight = 44

# Write if statement here to calculate the total cost
```

- ❖ **Pregunta:** Once you've written the application in the code editor's shipping.py tab, a flag will appear, which is the answer to this question.



The screenshot shows the same code editor as before, but with the completed code in the shipping.py tab. The code is as follows:

```
====> You've redeemed a hint. Replace the X's with code to complete this exercise.

"""

shipping_cost_per_kg = 1.20
customer_basket_cost = 34
customer_basket_weight = 44

if(customer_basket_cost >= 100):
    print('Free shipping!')
else:
    shipping_cost = customer_basket_weight * shipping_cost_per_kg
    customer_basket_cost = customer_basket_cost + shipping_cost

print("Total basket cost including shipping is " + str(customer_basket_cost))
```

- **Respuesta:** THM{IF_STATEMENT_SHOPPING}

- ❖ **Pregunta:** In shipping.py, on line 15 (when using the Code Editor's Hint), change the customer_basket_cost variable to 101 and re-run your code. You will get a flag (if the total cost is correct based on your code); the flag is the answer to this question.

```
shipping_cost_per_kg = 1.20
customer_basket_cost = 34
customer_basket_weight = 44

if(customer_basket_cost >= 100):
    print('Free shipping!')
else:
    shipping_cost = customer_basket_weight * shipping_cost_per_kg
    customer_basket_cost = 101

print("Total basket cost including shipping is " + str(customer_basket_cost))
```

➤ Respuesta: THM{MY_FIRST_APP}

Tarea 7: Bucles.

En programación, los bucles permiten que los programas iteren y realicen acciones varias veces. Existen dos tipos de bucles, **for** y **while** bucles.

- ❖ **Pregunta:** On the code editor, click back on the "script.py" tab and code a loop that outputs every number from 0 to 50.
- ❖ **Nota:** En programación, 0 suele ser el número inicial, por lo que contar hasta 5 es de 0 a 4 (pero tiene 5 números: 0, 1, 2, 3 y 4)

```
for i in range(51):
    print(i)
```

➤ Respuesta: THM{L00PS_WHILE_FOR}

Tarea 8: Introducción a las funciones.

Las funciones son bloques de código reutilizables que evitan la repetición, permitiendo ejecutar cálculos o tareas específicas en distintas partes de un programa de forma más organizada y eficiente.

- ❖ **Pregunta:** Ejercicio a resolver.

```
investment_in_bitcoin = 1.2
bitcoin_to_usd = 40000

# 1) write a function to calculate bitcoin to usd
def bitcoinToUSD(bitcoin_amount, bitcoin_value_usd):
    usd_value = bitcoin_amount * bitcoin_value_usd
    return usd_value
# 2) use function to calculate if the investment is below $30,000
bitcoin_amount = 1.2
bitcoin_value_usd = 25000

usd_value = bitcoinToUSD(bitcoin_amount, bitcoin_value_usd)

if usd_value < 30000:
    print("Alert! Your Bitcoin value is below $30,000.")
```

➤ **Respuesta:** THM{BITCOIN_INVESTOR}

- ❖ No answer needed
- ❖ **Nota:** Actualizamos el valor de “*bitcoin_value_usd* = 24000” para que salte la Alerta!

```
16 # 1) write a function to calculate bitcoin to usd
17 def bitcoinToUSD(bitcoin_amount, bitcoin_value_usd):
18     usd_value = bitcoin_amount * bitcoin_value_usd
19     return usd_value
20 # 2) use function to calculate if the investment is below $30,000
21 bitcoin_amount = 1.2
22 bitcoin_value_usd = 24000
23
24 usd_value = bitcoinToUSD(bitcoin_amount, bitcoin_value_usd)
25
26 if usd_value < 30000:
27     print("Alert! Your Bitcoin value is below $30,000.")
```

Exercise Complete! The flag is: THM{BITCOIN_INVESTOR}

Python code output

Alert! Your Bitcoin value is below \$30,000.

Tarea 9: Archivos.

En Python, puedes leer y escribir archivos para almacenar salidas de scripts o importar datos, como listas de sitios web para automatizar tareas en ciberseguridad.

- ❖ **Pregunta:** In the code editor, write Python code to read the flag.txt file. What is the flag in this file?

```
f = open("flag.txt", "r")
print(f.read())
```

➤ **Respuesta:** THM{F1LE_R3AD}

Tarea 10: Importaciones.

En Python, importar bibliotecas permite usar funciones ya descritas, facilitando tareas y evitando tener que programar todo desde cero.

```
14 import datetime
15 current_time = datetime.datetime.now()
16 print(current_time)
17
```

Python code output

2025-07-01 20:39:09.243500

Importará (**import**) la Fecha y Hora de este momento, y con **print** lo imprime para mostrar.

Nota: En este ejemplo, imprime la Fecha y Hora del momento que se ejecuta.

- ❖ **No answer needed**