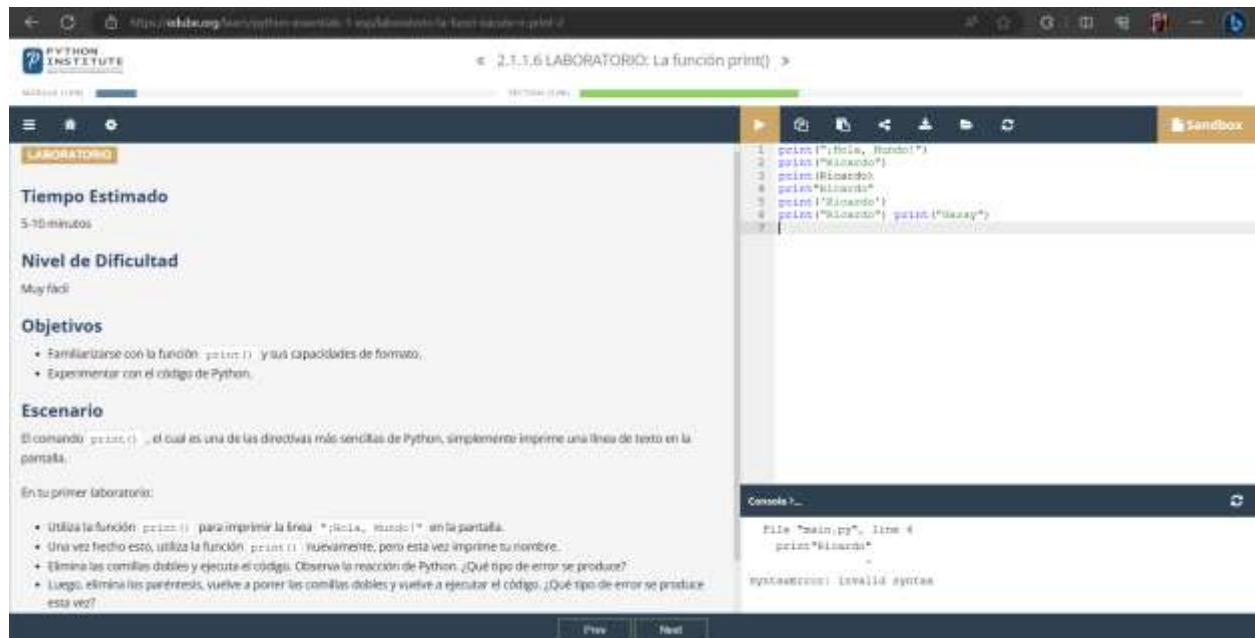


Laboratorio 1



The screenshot shows the Python Institute Lab 1 interface. The left sidebar contains the following information:

- LABORATORIO**
- Tiempo Estimado**: 5-10 minutos
- Nivel de Dificultad**: Muy fácil
- Objetivos**
 - Familiarizarse con la función `print()` y sus capacidades de formato.
 - Experimentar con el código de Python.
- Escenario**

El comando `print()`, el cual es una de las directivas más sencillas de Python, simplemente imprime una línea de texto en la pantalla.

En tu primer laboratorio:

 - Utiliza la función `print()` para imprimir la línea `*¡Hola, mundo!*` en la pantalla.
 - Una vez hecho esto, utiliza la función `print()` nuevamente, pero esta vez imprime tu nombre.
 - Elimina las comillas dobles y ejecuta el código. Observa la reacción de Python. ¿Qué tipo de error se produce?
 - Luego, elimina los paréntesis, vuelve a poner las comillas dobles y vuelve a ejecutar el código. ¿Qué tipo de error se produce esta vez?

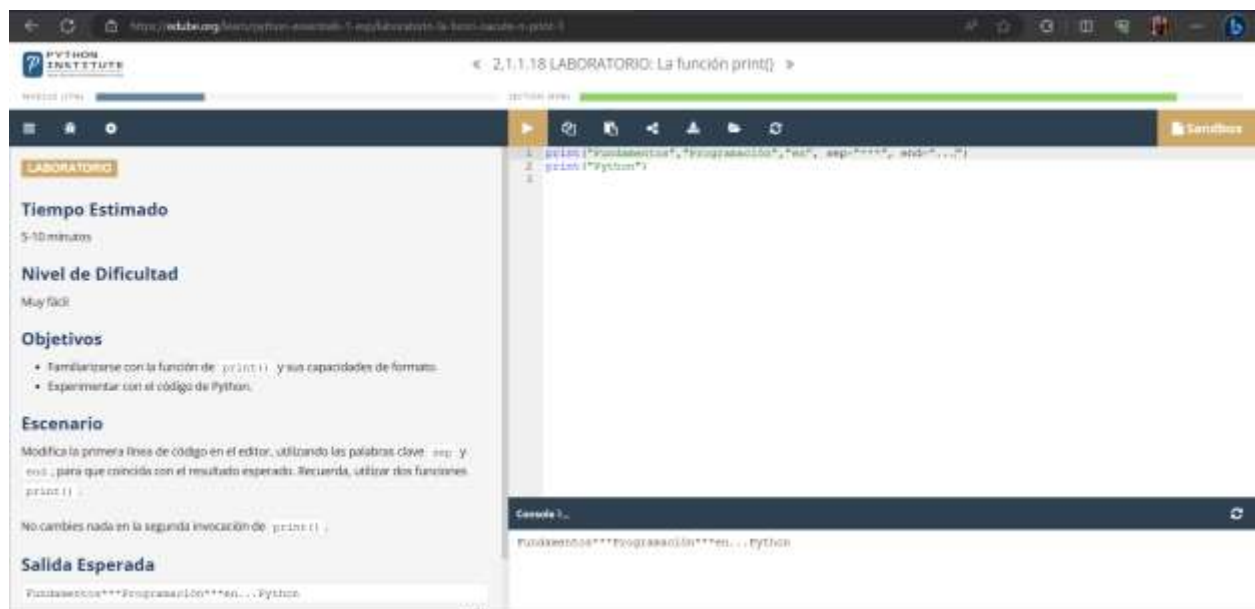
The main area shows a code editor with the following code:

```
1 print("Hola, Mundo!")
2 print("Ricardo")
3 print(Ricardo)
4 print"Ricardo"
5 print('Ricardo')
6 print("Ricardo") print("Guzay")
7
```

The console output shows the following error:

```
File "main.py", line 4
print"Ricardo"
      ^
SyntaxError: invalid syntax
```

Laboratorio 2



The screenshot shows the Python Institute Lab 2 interface. The left sidebar contains the following information:

- LABORATORIO**
- Tiempo Estimado**: 5-10 minutos
- Nivel de Dificultad**: Muy fácil
- Objetivos**
 - Familiarizarse con la función de `print()` y sus capacidades de formato.
 - Experimentar con el código de Python.
- Escenario**

Modifica la primera línea de código en el editor, utilizando las palabras clave `sep` y `end`, para que coincida con el resultado esperado. Recuerda, utilizar dos funciones `print()`.

No cambies nada en la segunda invocación de `print()`.
- Salida Esperada**

```
Fundamentos***Programación***en...Python
```

The main area shows a code editor with the following code:

```
1 print("Fundamentos","Programación","en", sep="***", end="...")
2 print("Python")
```

The console output shows the following result:

```
Fundamentos***Programación***en...Python
```

Laboratorio 3

Python Institute

2.1.1.19 LABORATORIO: Dando formato a la salida

5-15 minutos

Nivel de Dificultad: Fácil

Objetivos

- Experimentar con el código Python existente.
- Descubrir y solucionar errores básicos de sintaxis.
- Familiarizarse con la función `print()` y sus capacidades de formato.

Escenario

Recomendamos que **juegues con el código** que hemos escrito para ti y que realices algunas correcciones (quizás incluso destructivas). Siéntete libre de modificar cualquier parte del código, pero hay una condición: aprende de tus errores y saca tus propias conclusiones.

Intenta

- Modificar el número de invocaciones de la función `print()` insertando la secuencia `\n` en las cadenas.

```

1 print("Invocación original.")
2
3 print("  **")
4 print("  **")
5 print("  **")
6 print("  **")
7 print("**** **")
8 print("  **")
9 print("  **")
10 print("  ****")
11
12 print("Con menos invocaciones de 'print()':")
13 print("*****")
14 print("  **\n  **\n  **\n  **\n  **** **")
15 print("  **\n  **\n  ****")
16
17 print("Más alto:")
18
19 print("  **")
20 print("  **")
21 print("  **")
22 print("  **")
23 print("  **")
24 print("  **")
25 print("  **")
26 print("  **")
27 print("*****")

```

Consola 1..

```

Con menos invocaciones de "print()":
*****
  **
  **
  **
  **
  **** **

```

Laboratorio 4

Python Institute

2.2.1.11 LABORATORIO: Literales de Python - Cadenas

5-10 minutos

Nivel de Dificultad: Fácil

Objetivos

- Familiarizarse con la función `print()` y sus capacidades de formato.
- Practicar el codificar cadenas.
- Experimentar con el código de Python.

Escenario

Escribe una sola línea de código, utilizando la función `print()` así como los caracteres de nueva línea y escape, para obtener la salida esperada de tres líneas.

Salida Esperada

```

"Batoy"
**aprendiendo**
***python***

```

```

1 print("Batoy\n","**aprendiendo**\n","***python***")

```

Consola 1..

```

"Batoy"
**aprendiendo**
***python***

```

Laboratorio 5

2.4.1.7 LABORATORIO: Variables

Experimentar con el código en Python.

Escenario

A continuación una historia:

Érase una vez en la Tierra de las Manzanas, Juan tenía tres manzanas, María tenía cinco manzanas, y Adán tenía seis manzanas. Todos eran muy felices y vivieron por muchísimo tiempo. Fin de la Historia.

Tu tarea es:

- Crear las variables: `juan`, `maria`, y `adan`.
- Asignar valores a las variables. El valor debe de ser igual al número de manzanas que cada quien tenía.
- Una vez almacenados los números en las variables, imprimir las variables en una línea, y separar cada una de ellas con una coma.
- Después se debe crear una nueva variable llamada: `total_manzanas` y se debe igualar a la suma de las tres variables anteriores.
- Imprime el valor almacenado en `total_manzanas` en la consola.
- Experimenta con tu código:** crea nuevas variables, asigna diferentes valores a ellas, y realiza varias operaciones aritméticas con ellas (por ejemplo, `+`, `-`, `*`, `/`, `//`, etc.). Intenta poner una cadena con un entero juntos en la misma línea, por ejemplo, "Número Total de Manzanas:" y `total_manzanas`.

```

1 juan = 3
2 maria = 5
3 adan = 6
4 print("Manzanas de Juan: ", juan, "Manzanas de Maria: ", maria, "Manzanas de Adán: ", adan)
5 total_manzanas = juan + maria + adan
6 print("Número Total de Manzanas:", total_manzanas)
7

```

Consola:

```

Manzanas de Juan: 3 Manzanas de Maria: 5 Manzanas de Adán: 6
Número Total de Manzanas 14

```

Laboratorio 6

2.4.1.9 LABORATORIO: Variables, un sencillo convertidor

Hay una cosa interesante más que está ocurriendo. Puedes ver otra función dentro de la función: `round()`. Es la función `round()`. Su trabajo es redondear la salida del resultado al número de decimales especificados en el paréntesis, y regresar un valor flotante (dentro de la función: `round()` se puede encontrar el nombre de la variable, el nombre, una coma, y el número de decimales que se desean mostrar). Se hablará más de esta función muy pronto, no te preocupes si no todo queda muy claro. Sólo se quiere impulsar tu curiosidad.

Después de completar el laboratorio, abre Sandboxes, y experimenta más. Intenta escribir diferentes convertidores, por ejemplo, un convertidor de USD a EUR, un convertidor de temperatura, etc. ¡Ojo que tu imaginación vale! Intenta mostrar los resultados combinando cadenas y variables. Intenta utilizar y experimentar con la función `round()` para redondear tus resultados a uno, dos o tres decimales. Revisa que es lo que sucede si no se provee un dígito al redondear. Recuerda probar tus programas.

Experimenta, saca tus propias conclusiones, y aprende. Sé curioso.

Resultado Esperado

```

7.39 millas son 11.88 Kilometros
12.25 Kilometros son 7.61 millas

```

```

1 kilometros = 0.62
2 miles = 1.61
3
4 miles_to_kilometers = miles * 0.62
5 kilometers_to_miles = kilometers / 0.62
6
7 print(miles, "miles son", round(miles_to_kilometers, 2), "kilometros")
8 print(kilometers, "kilometros son", round(kilometers_to_miles, 2), "miles")
9

```

Consola:

```

7.39 millas son 11.88 Kilometros
12.25 Kilometros son 7.61 millas

```

Play Stop

Laboratorio 7

2.4.1.10 LABORATORIO: Operadores y expresiones

LABORATORIO

Tiempo Estimado
10-15 minutos

Nivel de Dificultad
Fácil

Objetivos

- Familiarizarse con los conceptos de números, operadores y operaciones aritméticas en Python.
- Realizar cálculos básicos.

Escenario

Observa el código en el editor: lee un valor, `floatante`, lo coloca en una variable llamada `x`, e imprime el valor de la variable llamada `y`. Tu tarea es completar el código para evaluar la siguiente expresión:

$$3x^3 - 2x^2 + 3x + 1$$

El resultado debe ser asignado a `y`.

```

1 x = float(input())
2 x = float(x)
3 y = 3 * x**3 - 2 * x**2 + 3 * x + 1
4 print("y =", y)
5
6 x = 1
7 x = float(x)
8 y = 3 * x**3 - 2 * x**2 + 3 * x + 1
9 print("y =", y)
10
11 x = -1
12 x = float(x)
13 y = 3 * x**3 - 2 * x**2 + 3 * x + 1
14 print("y =", y)
15

```

Console 1...

```

y = -1.0
y = 3.0
y = -3.0

```

Prev Next

Laboratorio 8

2.5.1.2 LABORATORIO: Comentarios

LABORATORIO

Tiempo Estimado
5 minutos

Nivel de Dificultad
Muy Fácil

Objetivos

- Familiarizarse con el concepto de comentario en Python.
- Utilizar y no utilizar los comentarios.
- Reemplazar los comentarios con código.
- Experimentar con el código de Python.

Escenario

El código en el editor contiene comentarios. Intenta mejorarlos: agrega o quite comentarios donde consideres que sea apropiado (en ocasiones el remover un comentario lo hace más legible); además, cambia el nombre de las variables donde consideres que esto mejorará la comprensión del código.

NOTA

Los comentarios son muy importantes. No solo hacen que el programa sea más fácil de

```

1 # Este programa calcula los segundos en cierto número de horas determinadas
2
3 # 2 = número de horas
4 seconds = 2400 # número de segundos en una hora
5
6 print("Horas: ", h) # imprime el número de horas
7 print("Segundos en Horas: ", s) # se imprime el número de segundos en determinado número de h
8
9 Esto es el fin del programa que calcula el número de segundos en 2 horas
10

```

Console 1...

```

Horas: 2
Segundos en Horas: 7200

```

Prev Next

Laboratorio 9

PYTHON INSTITUTE

2.6.1.11 LABORATORIO: Operadores y expresiones

Python 3.9.1

Datos de Prueba

Entrada de muestra:

```
13
17
29
```

Salida esperada: 13:18

Entrada de muestra:

```
23
30
642
```

Salida esperada: 10:40

Entrada de muestra:

```
0
1
2039
```

Salida esperada: 1:0

```

1 hour = int(input("Hora de inicio (horas): "))
2 mins = int(input("Minuto de inicio (minutos): "))
3 dur = int(input("Duración del evento (minutos): "))
4 mins = mins + dur # convierte el número total de minutos
5 hour = hour + mins // 60
6 mins = mins % 60
7 hour = hour % 24
8 print(hour, ":", mins, sep="")
9

```

Console:

```

Hora de inicio (horas): 13
Minuto de inicio (minutos): 17
Duración del evento (minutos): 29
13:18
Hora de inicio (horas): 23
Minuto de inicio (minutos): 30
Duración del evento (minutos): 642
10:40
Hora de inicio (horas): 0
Minuto de inicio (minutos): 1
Duración del evento (minutos): 2039
1:0

```

Laboratorio 12

PYTHON INSTITUTE

3.1.1.4 LABORATORIO: Preguntas y respuestas

Python 3.9.1

Resultado esperado: False

Ejemplo de entrada: 99

Resultado esperado: False

Ejemplo de entrada: 100

Resultado esperado: True

Ejemplo de entrada: 101

Resultado esperado: True

Ejemplo de entrada: -5

Resultado esperado: False

Ejemplo de entrada: +123

Resultado esperado: True

```

1 n = int(input("Ingresa el valor de n: "))
2 print(n >= 100)
3

```

Console:

```

Ingresa el valor de n: 99
False
Ingresa el valor de n: 99
False
Ingresa el valor de n: 100
True
Ingresa el valor de n: 101
True
Ingresa el valor de n: -5
False
Ingresa el valor de n: +123
True

```

Laboratorio 13

3.1.1.10 LABORATORIO: Operadores de comparación y ejecución condicional

Prueba tu código con los datos que te proporcionamos. (y fíjate de un ESPATIFILLO también)

Datos de Prueba

Entrada de muestra: `espatifilo`

Resultado esperado: `no, ¡quiero un gran ESPATIFILLO!`

Entrada de ejemplo: `palmarfino`

Resultado esperado: `"ESPATIFILLO", ¡No palmarfino!`

Entrada de muestra: `ESPATIFILLO`

Resultado esperado: `si, ¡el ESPATIFILLO es la mejor planta de todos los tiempos!`

```

1 entrada = input("Ingresa la palabra: ")
2
3 if entrada == "ESPATIFILLO":
4     print("¡El ESPATIFILLO es la mejor planta de todos los tiempos!")
5 elif entrada == "espatifilo":
6     print("No, ¡quiero un gran ESPATIFILLO!")
7 else:
8     print("ESPATIFILLO", (No " + entrada + ")")
9

```

Consola:

```

Ingresa la palabra: espatifilo
No, ¡quiero un gran ESPATIFILLO!
Ingresa la palabra: palmarfino
(ESPATIFILLO), ¡No palmarfino!
Ingresa la palabra: ESPATIFILLO
Si, ¡el ESPATIFILLO es la mejor planta de todos los tiempos!

```

Laboratorio 14

3.1.1.11 LABORATORIO: Fundamentos de la sentencia if-else

Prueba tu código con los datos que hemos proporcionado.

Datos de Prueba

Entrada de muestra: `10000`

Resultado esperado: `El impuesto es: 1248.0 pesos`

Entrada de muestra: `100000`

Resultado esperado: `El impuesto es: 19470.0 pesos`

Entrada de muestra: `1000`

Resultado esperado: `El impuesto es: 0.0 pesos`

Entrada de muestra: `-100`

Resultado esperado: `El impuesto es: 0.0 pesos`

```

1 ingreso = float(input("Introduce el ingreso anual: "))
2
3 if ingreso <= 85528:
4     impuesto = ingreso * 0.11 - 256.02
5 else:
6     impuesto = (ingreso - 85528) * 0.32 + 14839.02
7
8 if impuesto < 0:
9     impuesto = 0.0
10
11 impuesto = round(impuesto)
12 print("El impuesto es:", impuesto, "pesos")
13

```

Consola:

```

Introduce el ingreso anual: 10000
El impuesto es: 1248 pesos
Introduce el ingreso anual: 100000
El impuesto es: 19470 pesos
Introduce el ingreso anual: 1000
El impuesto es: 0 pesos
Introduce el ingreso anual: -100
El impuesto es: 0 pesos

```

Laboratorio 15

« 3.1.1.12 LABORATORIO: Fundamentos de la sentencia if-elif-else »

Datos de Prueba

Entrada de muestra: 2000

Resultado esperado: Año Bisiesto

Entrada de muestra: 2013

Resultado esperado: Año Común

Entrada de muestra: 1999

Resultado esperado: Año Común

Entrada de muestra: 1994

Resultado esperado: Año Bisiesto

Entrada de muestra: 1900

Resultado esperado: No está dentro del periodo del calendario Gregoriano

```

1 año = int(input("Introduce un año: "))
2
3 if año < 1582:
4     print("No está dentro del periodo del calendario Gregoriano")
5 else:
6     if año % 4 != 0:
7         print("Año Común")
8     elif año % 100 != 0:
9         print("Año Bisiesto")
10    elif año % 400 != 0:
11        print("Año Común")
12    else:
13        print("Año Bisiesto")

```

Introduce un año: 2000

Año Bisiesto

Introduce un año: 2013

Año Común

Introduce un año: 1999

Año Común

Introduce un año: 1994

Año Bisiesto

Introduce un año: 1900

No está dentro del periodo del calendario Gregoriano

Laboratorio 16

« 3.2.1.3 LABORATORIO: Lo esencial del bucle while - Adivina el número secreto »

¡Bienvenido a mi juego, muggle!

Desafortunadamente, él no sabe cómo completar el código.

Tu tarea es ayudar al mago a completar el código en el editor de tal manera que el código:

- Pedirá al usuario que ingrese un número entero.
- Utilizará un bucle `while`.
- Comprobará si el número ingresado por el usuario es el mismo que el número escogido por el mago. Si el número elegido por el usuario es diferente al número secreto del mago, el usuario deberá ver el mensaje "¡Ja, ja! ¡Estás atrapado en mi magia!" y se le solicitará que ingrese un número nuevamente. Si el número ingresado por el usuario coincide con el número escogido por el mago, el número debe imprimirse en la pantalla, y el mago debe decir las siguientes palabras: "¡Bien hecho, muggle! Eres libre ahora".

El mago está contando contigo! No lo decepciones.

Nota importante:

Por cierto, observa la función `print()`. La forma en que lo hemos utilizado aquí se llama impresión múltiple. Puede utilizar **comillas triples** para imprimir líneas en varias líneas para facilitar la lectura del texto o crear un diseño especial basado en texto. Experimenta con ello.

```

9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99

```

Introduce el número: 100

¡Ja, ja! ¡Estás atrapado en mi magia!

Introduce el número: 200

¡Ja, ja! ¡Estás atrapado en mi magia!

Introduce el número: 2540

¡Ja, ja! ¡Estás atrapado en mi magia!

Introduce el número: 777

¡Bien hecho, muggle! Eres libre ahora

Laboratorio 17

