El futuro digital es de todos

MinTIC

Spyder

OPERADO POR:

Universidad Tecnológica de Pereira

Mision TIC 2022

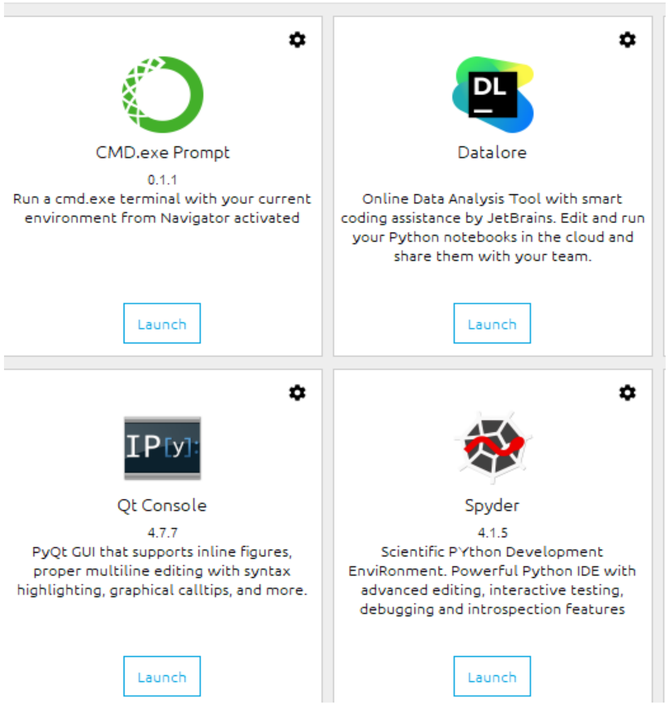
RUTA DE APRENDIZAJE 1

Spyder y dibujar en Python (turtle)

Programación en Spyder.

Spyder es un poderoso entorno científico escrito en Python, para Python, y diseñado por y para científicos, ingenieros y analistas de datos. Presenta una combinación única de la funcionalidad avanzada de edición, análisis, depuración y creación de perfiles de una herramienta de desarrollo integral con la exploración de datos, ejecución interactiva, inspección profunda y hermosas capacidades de visualización de un paquete científico.

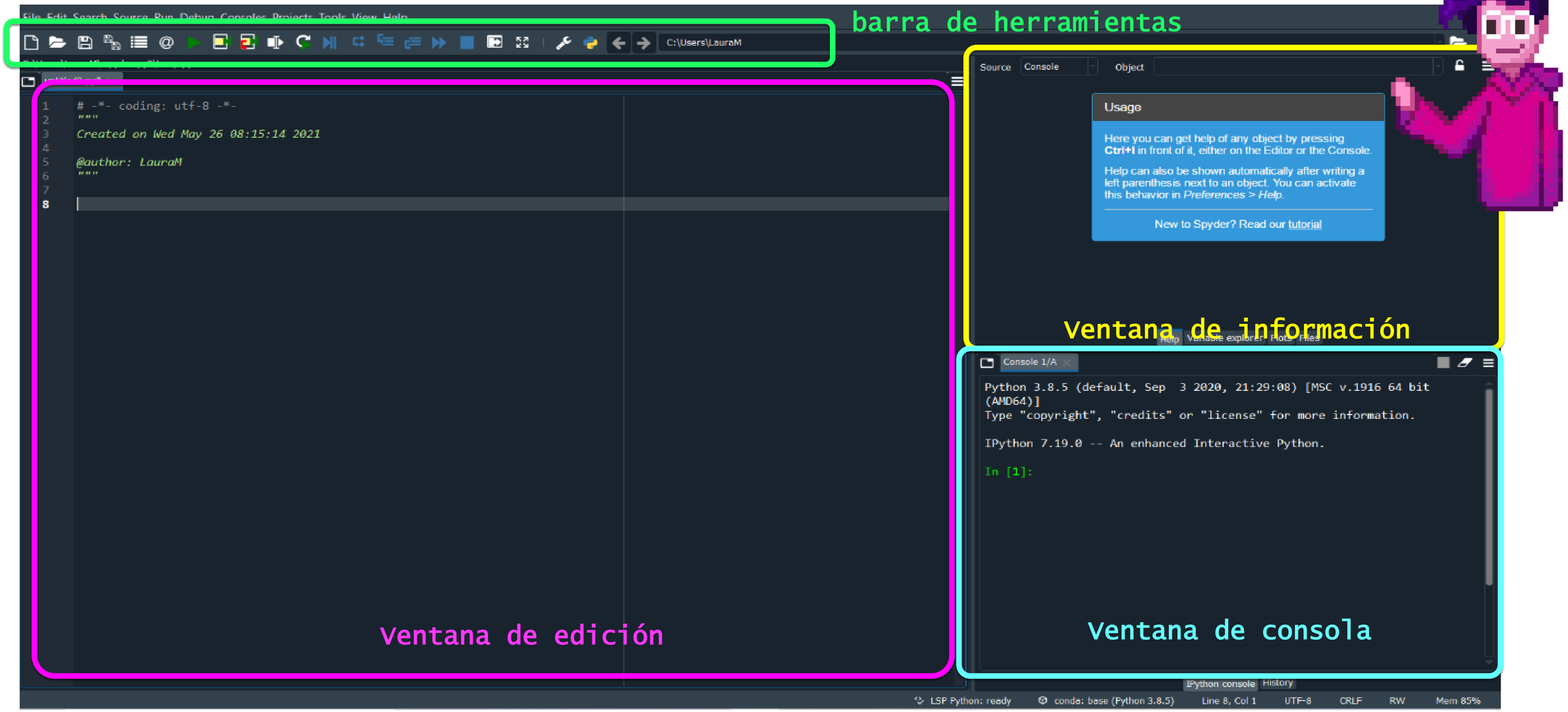
Podemos ejecutar el programa, desde el navegador de anaconda, de igual manera como ejecutamos **Jupiter**. O podemos simplemente buscar **Spyder** en nuestro computador si presenta dificultad para abrir la aplicación.



Solo tenemos que hacer click en el botón lunch en la opción de spyder. Y con eso se ejecutara en nuestro computador el editor.

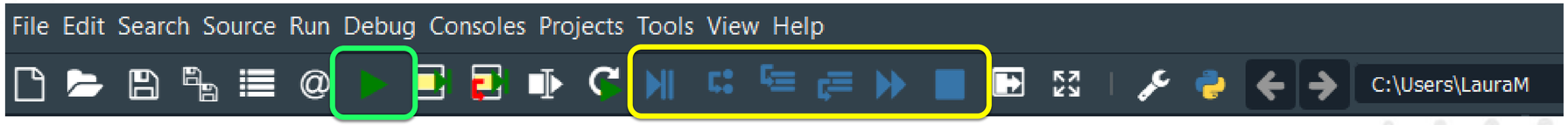
Spyder nos ayuda a programar de una forma mas fácil en Python, es un editor de código que también nos permite ejecutarlo al igual que cuando hacíamos nuestros códigos con con jupiter!

Una vez abrimos Spyder vamos a tener una pantalla de inicio con 4 componentes principales:



La barra de herramientas permite acceder rápidamente a algunos de los comandos más comunes en Spyder, como **ejecutar**, **guardar** y **depurar** archivos.

Todas estas acciones las podemos encontrar en las pestañas de Python. Sin embargo es posible acceder a muchos de ellos, usando los comando que se encuentra en la barra de herramientas.



Run - ejecutar

Elementos para deputar el código

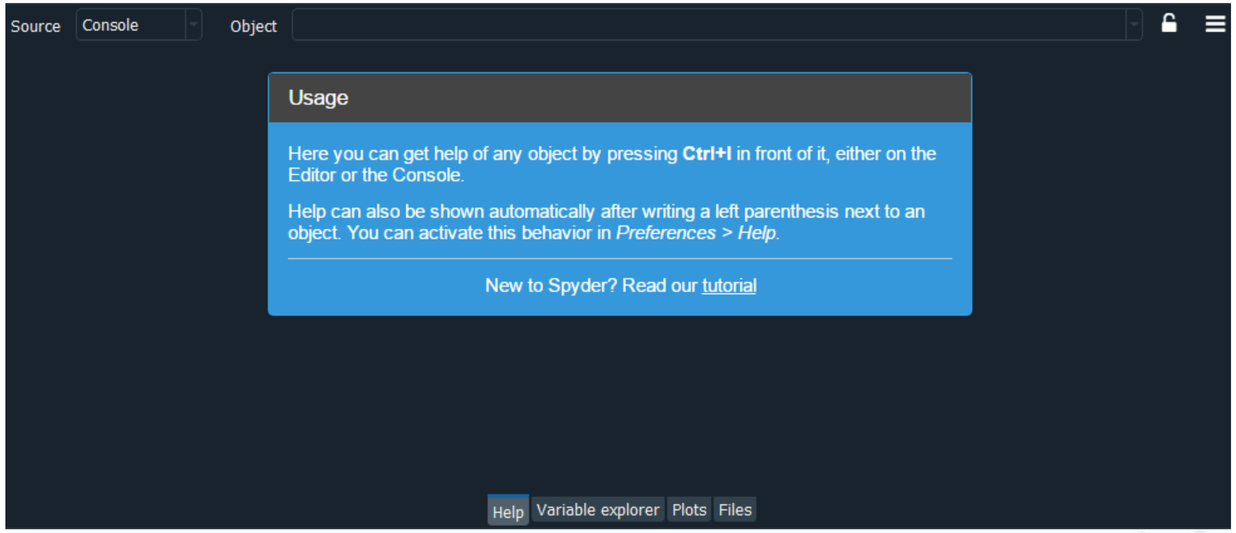
Ppodemos ejecutar el código solo haciendo click sobre la flecha verde, y podemos ver el proceso paso por paso usando las herramientas de debug que veremos mas adelante.

```
1  # -*- coding: utf-8 -*-
2  """
3  Created on Wed May 26  16:04:03 2021
4
5  @author: LauraM
6  """
7
8
```

El archivo trae **información por defecto**, para ayudarnos a organizar los códigos de una mejor manera. Vamos a encontrar la fecha en la que el archivo fue creado, en conjunto con la hora.

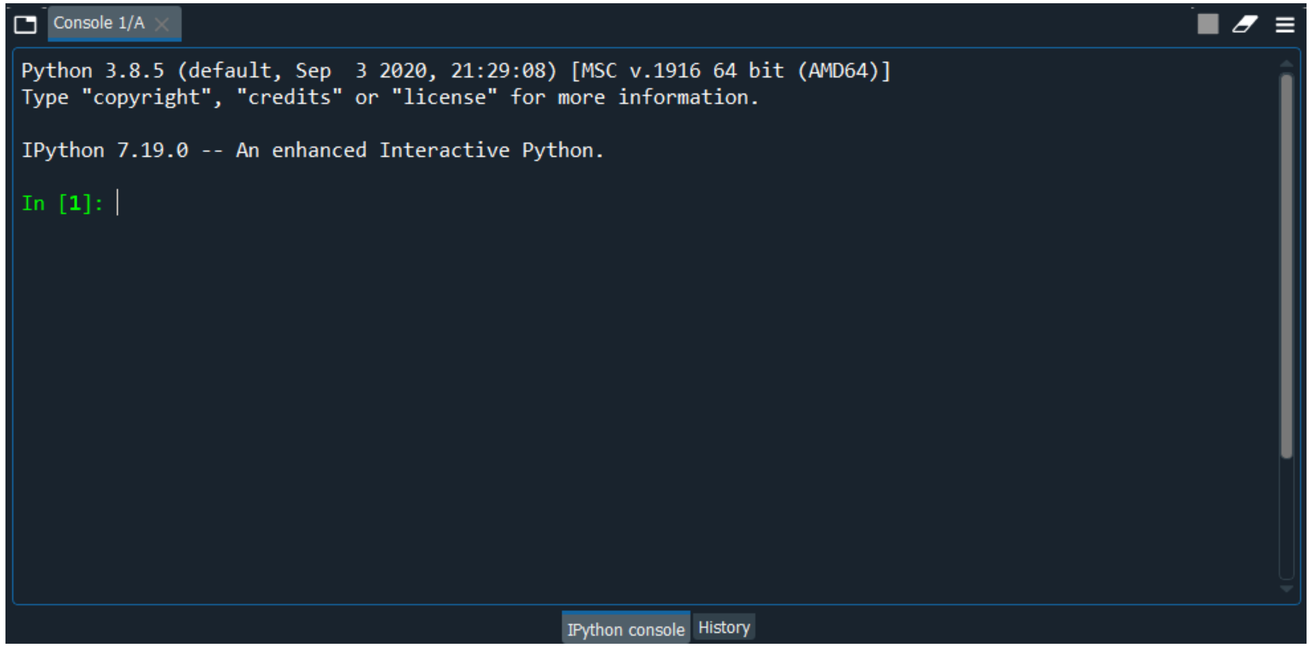
También debajo de esta información, aparecerá el nombre del autor o del computador donde se está implementando el código.

En esta ventana podremos ver mas adelante, al **ejecutar** el código, las **variables** que creamos en nuestro código, y otras características, como su **valor**, su **tamaño**, etc. Adicionalmente, el panel de Ayuda muestra documentación para los objetos que se están utilizando en el Editor.



También tendremos acceso a el panel de Gráficos **(Plots)**, el cual muestra las figuras e imágenes creadas durante la ejecución del código en el caso de que las creáramos dentro de nuestro análisis.

Por último, tendremos la ventana de la **consola**, es en esta sección donde vamos a ejecutar nuestros códigos, o veremos los resultados que normalmente encontramos debajo de las celdas en **Jupiter**.



Ejemplo 1:

Vamos a implementar uno de los códigos que hicimos en **Jupiter**, en Spyder, para que revisemos como es el proceso y como debemos ejecutarlo.

In [ ]:

```
#esta suma retornada un float

numero1 = 10
numero2 = 9.99
numero3 = numero1 + numero2
print(numero3)
type(numero3)
```

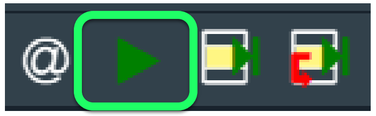
Vamos a usar el ejemplo de la suma de dos números **float**. En Spyder vamos a escribir estas mismas instrucciones en la ventana de edición.

También podemos copiar nuestro código de **Jupiter** y pegarlo en la ventana del editor de Python.

Primero vamos a observar, que el código de colores cambio un poco. Los valores numéricos ahora son de color amarillo, y los comandos de Python son de color naranja.

```
1   # -*- coding: utf-8 -*-
2   """
3   Created on Wed May 26  16:04:03 2021
4
5   @author: LauraM
6   """
7
8   #esta suma retornada un float
9
10  numero1 = 10
11  numero2 = 9.99
12  numero3 = numero1 + numero2
13  print(numero3)
14  type(numero3)
```

Los comentarios ahora aparecen de color gris. Recordemos que en Python los comentarios, o las líneas de código que no queremos que Python "vea" deben llevar un "#" antes.



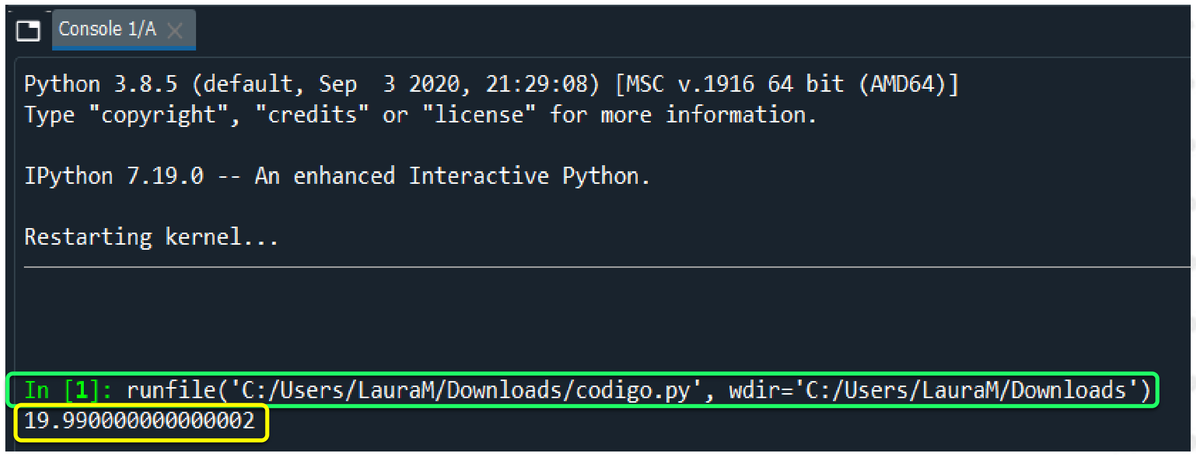
Run - ejecutar

Ahora solo debemos hacer click en la flecha verde que encontramos en la parte superior, para ejecutar el codigo

Si miramos la ventana de consola, vamos a ver que Python indica cual es el archivo que esta ejecutando, mostrando la dirección donde esta guardado en el computador.

Debajo de esta información tenemos la respuesta que nos entrega el programa, si le indicamos que imprima el valor con la función print, entonces la solución aparecerá en consola.

Solución del código



Ejemplo 2:

Vamos a implementar una función que tenga como tarea calcular el área de un triangulo.

Primero debemos definir la función, usando el comando def, le damos el nombre a la función, en este caso, área.

```
# Función para calcular el área de un triangulo

def area(parametro1:float, parametro2:float)->float:
```

Entre paréntesis vamos a escribir los argumentos que necesita la función para hacer el calculo. Como es un triangulo solo necesitamos dos valores.

En este ejemplo vamos a hacer algo un poco mas complejo, vamos a definir desde el inicio que tipo de variable va leer, una variable float.

```python
# Función para calcular el área de un triangulo

def area(parametro1:float, parametro2:float)->float:
```

También podemos definir si la salida de la función es float, esto no es algo necesario, pero es una opción que podemos indicar al inicio.

```python
    area= (parametro1 * parametro2)/(2)

    return area

lado1= 10.2
lado2= 30.0


respuesta=area(lado1,lado2)
print("El area del triagngulo es: " +str( respuesta));
```

Ahora solo debemos agregar las instrucciones para calcular un área, creamos la variable área que será igual a multiplicar los parámetros y dividirlos por 2.

Ahora debemos definir con el comando **return** cual es el valor que necesitamos que nos entregue la función

```
    area= (parametro1 * parametro2)/(2)

    return area

lado1= 10.2
lado2= 30.0


respuesta=area(lado1,lado2)
print("El area del triagngulo es: " +str( respuesta));
```

Solo nos queda para este ejemplo crear una variable llamada respuesta que llame la función área y donde le decimos cuales será los valores que necesita

Una vez tengamos lista la variable respuesta que llama la función, podemos escribir el comando print para que imprima un mensaje mas especifico usando un nuevo comando llamado str, el cual convierte un valor numérico a una cadena de string.

```
respuesta=area(lado1,lado2)
print("El area del triagngulo es: " +str( respuesta));
```

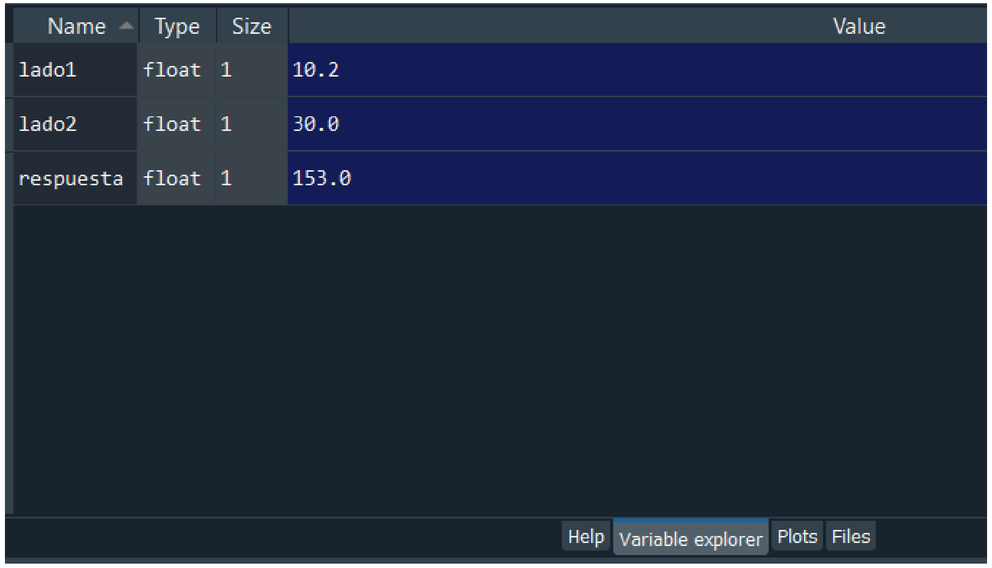
Recordemos que una cadena de string solo tiene caracteres, que guardan información, no puede tener valores numéricos, por lo tanto el comando str nos ayuda a convertir esos valores para que Python pueda imprimirlos.

Por ultimo hacemos click en el botón de ejecutar, o la flecha verde y tenemos la respuesta de nuestro código con las especificaciones que les dimos.

```
In [1]: runfile('C:/Users/LauraM/Downloads/codigo.py', wdir='C:/Users/LauraM/Downloads')
El area del triagngulo es: 153.0
```

Imprime entonces la respuesta de: el área del triangulo es: 153.0

Al ejecutar el código también se genera un conjunto de variables que podemos revisar en la ventana superior, si hacemos click en la sección de variable explorer podemos acceder a la información de las variables, sus valores, tipo y tamaño.



El futuro digital es de todos

MinTIC

GRACIAS

OPERADO POR:

Universidad Tecnológica de Pereira

MisiónTIC2022