

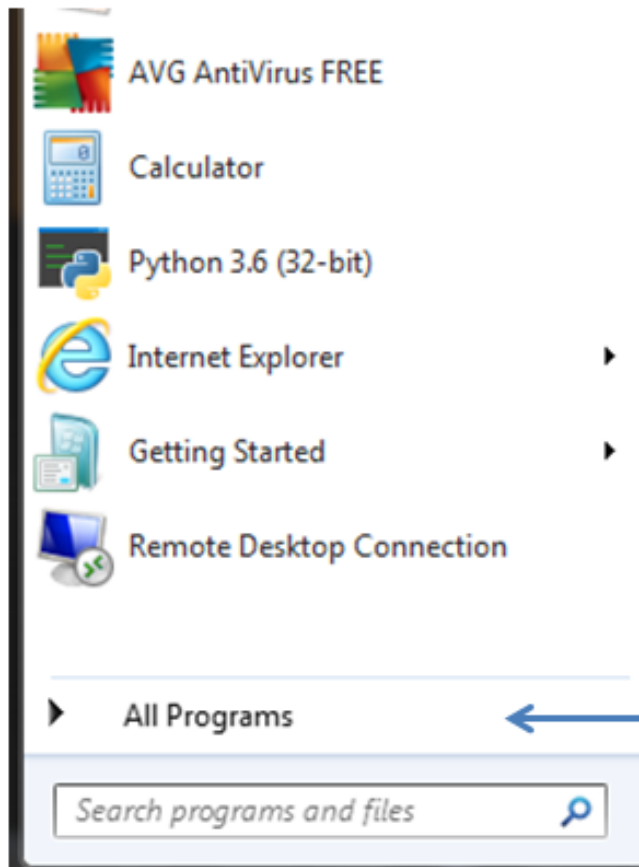
Como crear un programa Python



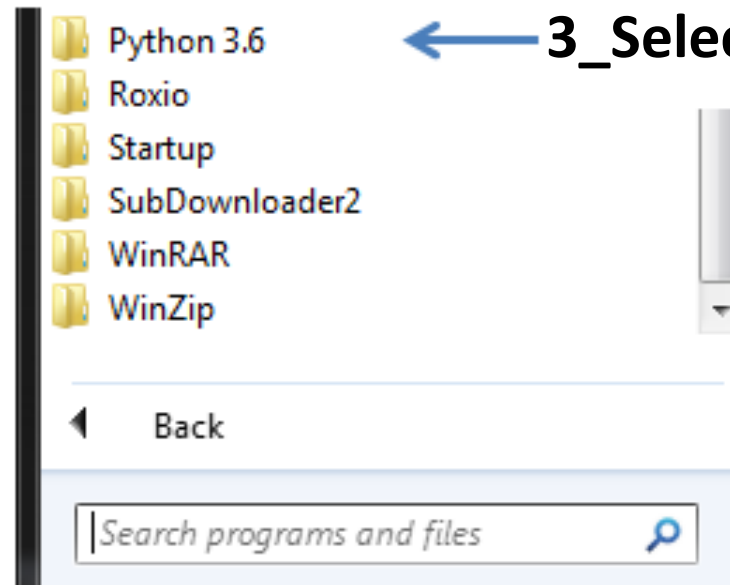
Uso de la interface de desarrollo
(IDE)

Para tener acceso al editor de programas de Python debemos seguir esta secuencia de pasos:

1_Seleccionar botón inicio de Windows

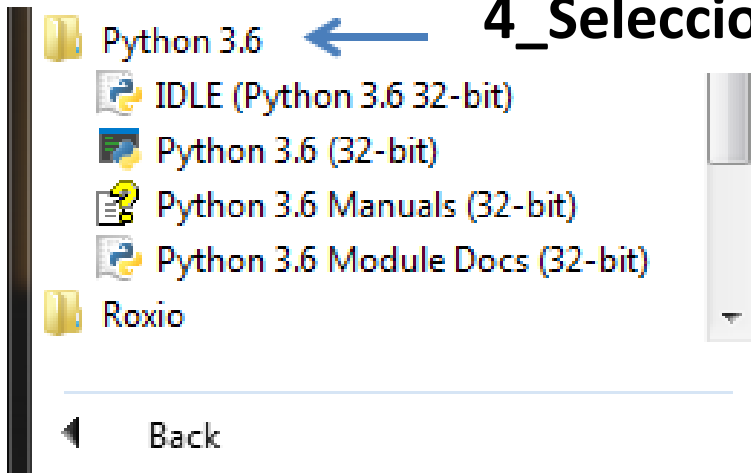


2_Seleccionar

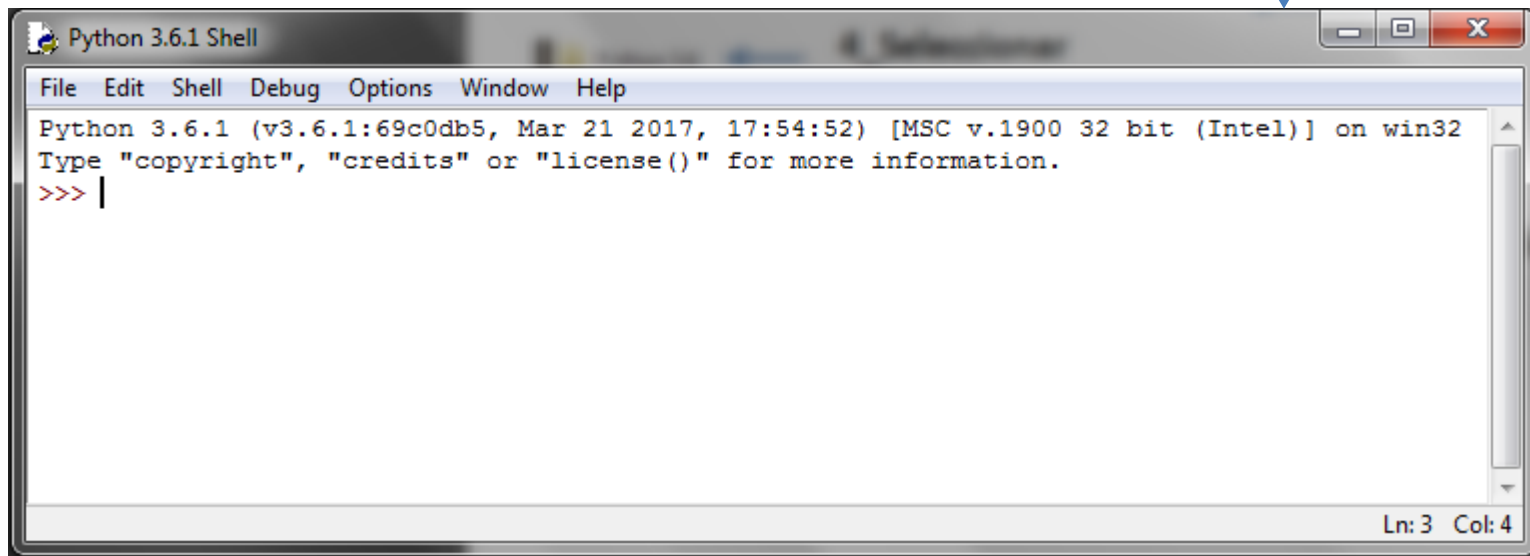


3_Seleccionar

4_Seleccionar

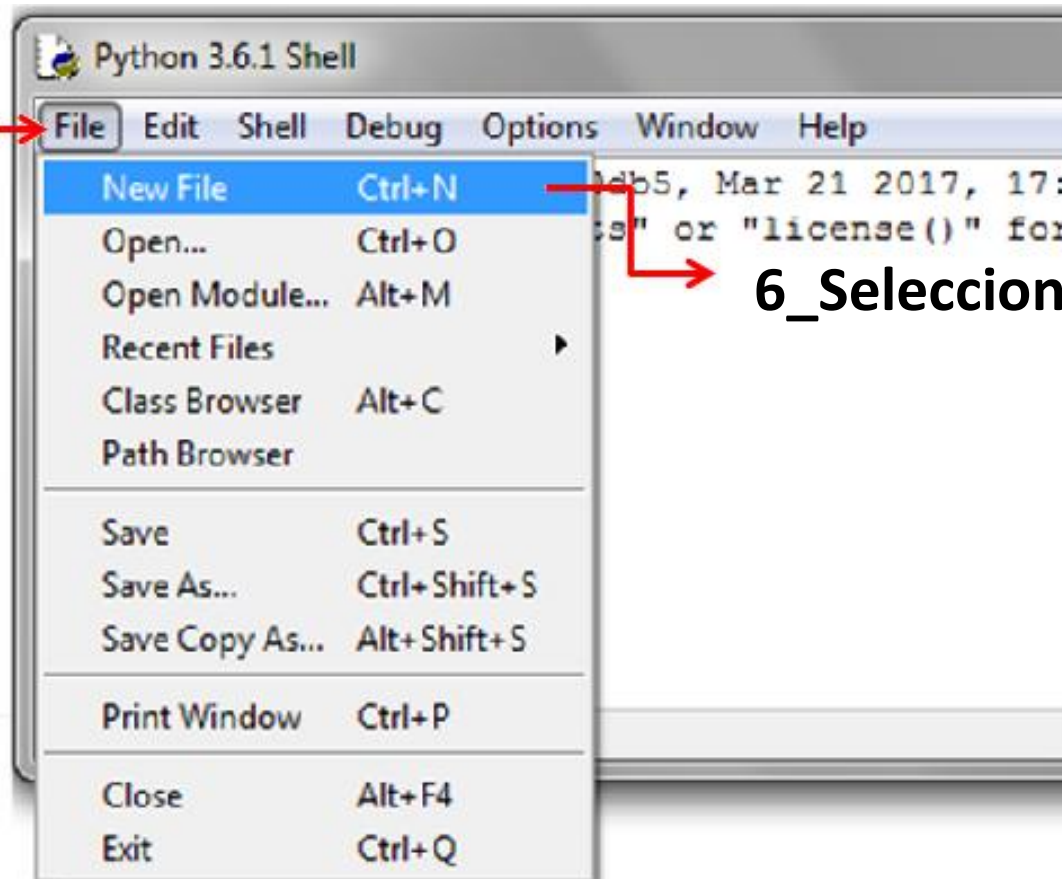


Después del paso 4 se abre la siguiente ventana:



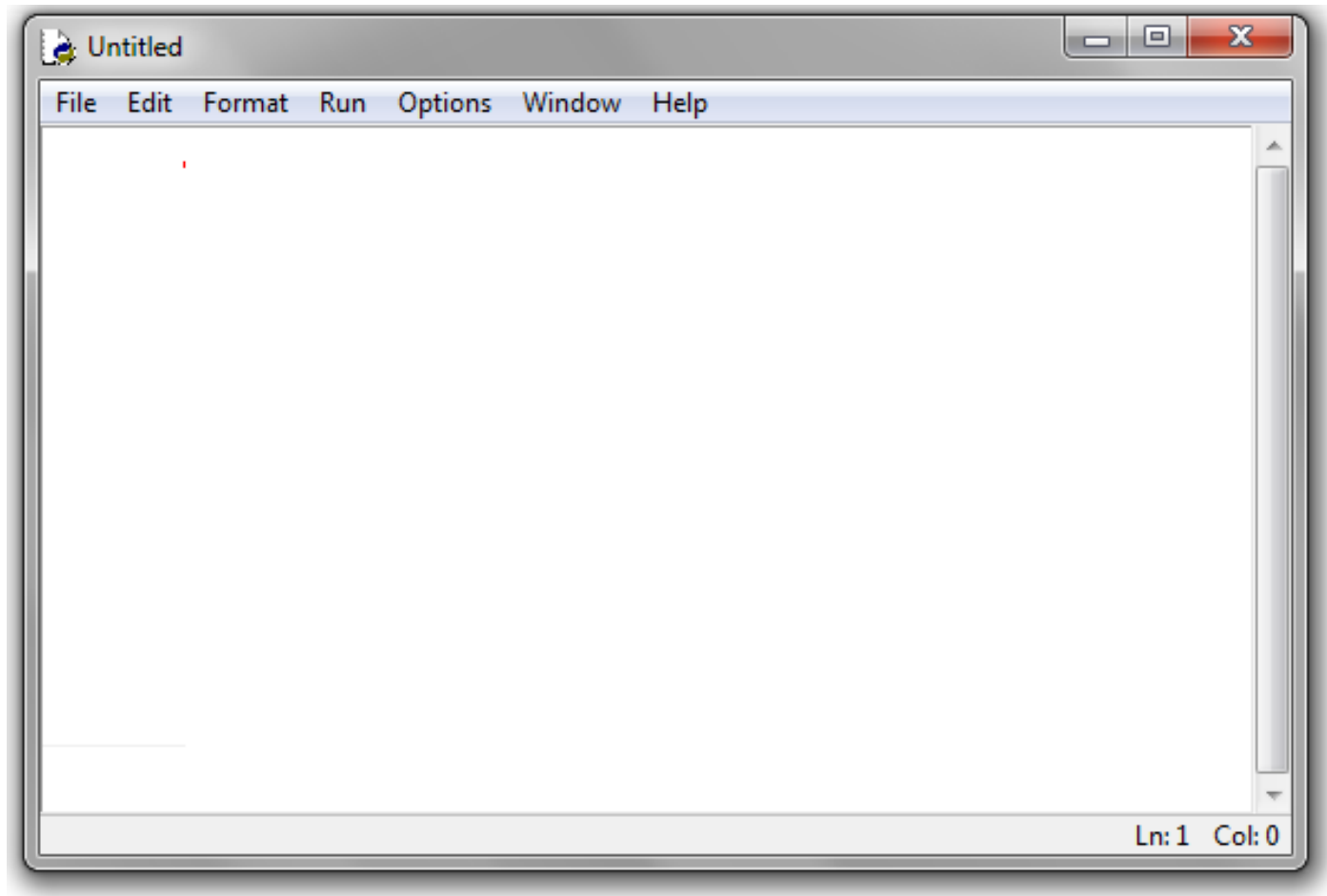
La cual es la calculadora de Python sirva para ejecutar de manera inmediata cálculos o instrucciones que quisiéramos probar pero no para desarrollar un programa. Hacer lo siguiente:

5_Seleccionar

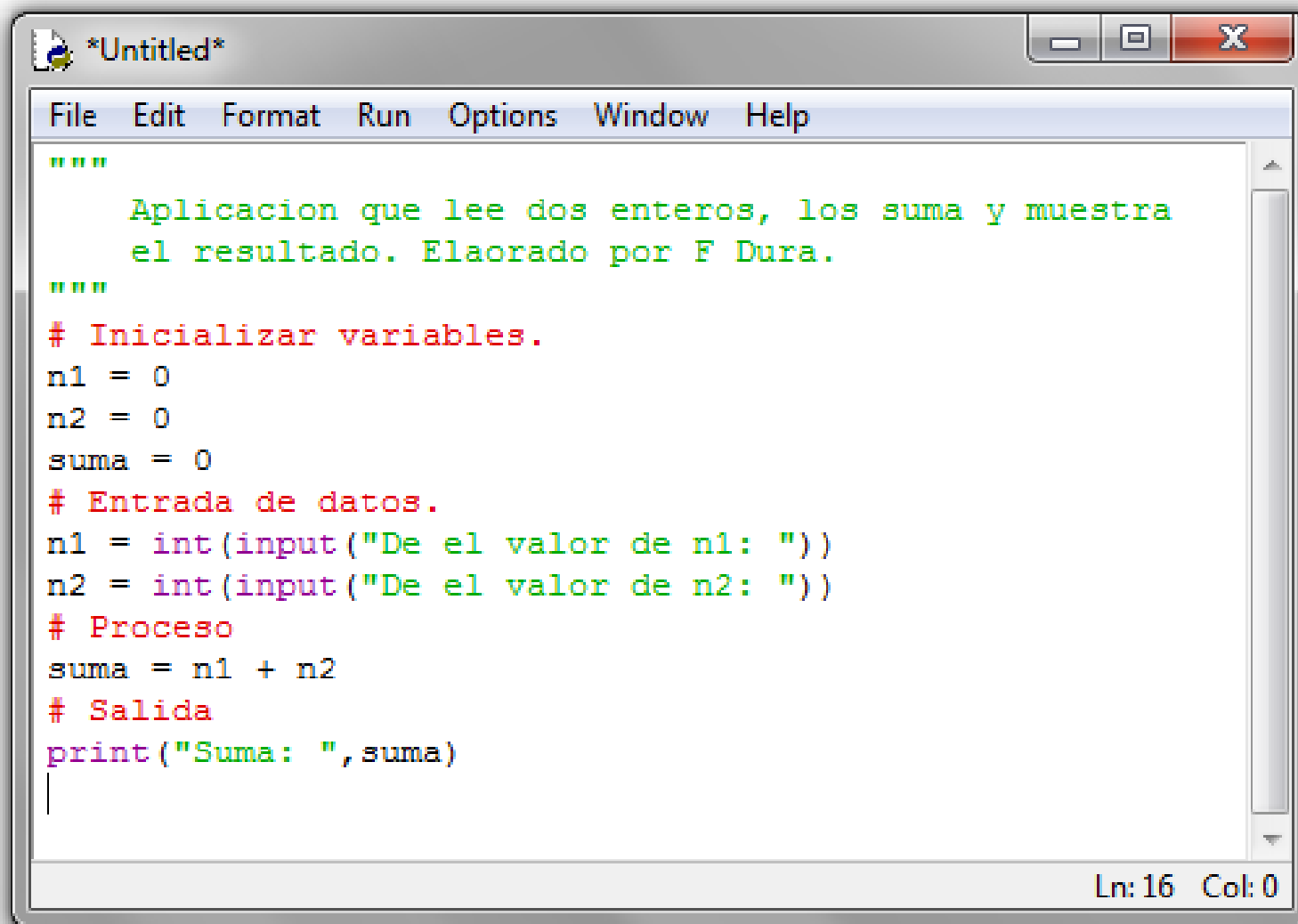


6_Seleccionar

Después del paso **6** se abre el editor de código del Python donde podemos colocar todas las instrucciones que formen parte del programa tal como se muestra continuación:



Vamos a desarrollar una aplicación con lo básico, que leerá dos números, determinara la suma y mostrara el resultado. Usaremos comentarios, instrucciones para la lectura y para mostrar los resultados. Quedando el programa así:



```
****
    Aplicacion que lee dos enteros, los suma y muestra
    el resultado. Elaorado por F Dura.
****
# Inicializar variables.
n1 = 0
n2 = 0
suma = 0
# Entrada de datos.
n1 = int(input("De el valor de n1: "))
n2 = int(input("De el valor de n2: "))
# Proceso
suma = n1 + n2
# Salida
print("Suma: ", suma)
|
```

Ln: 16 Col: 0

Escriba en el editor el contenido de esta pantalla, cuidadosamente, respetando la secuencia.

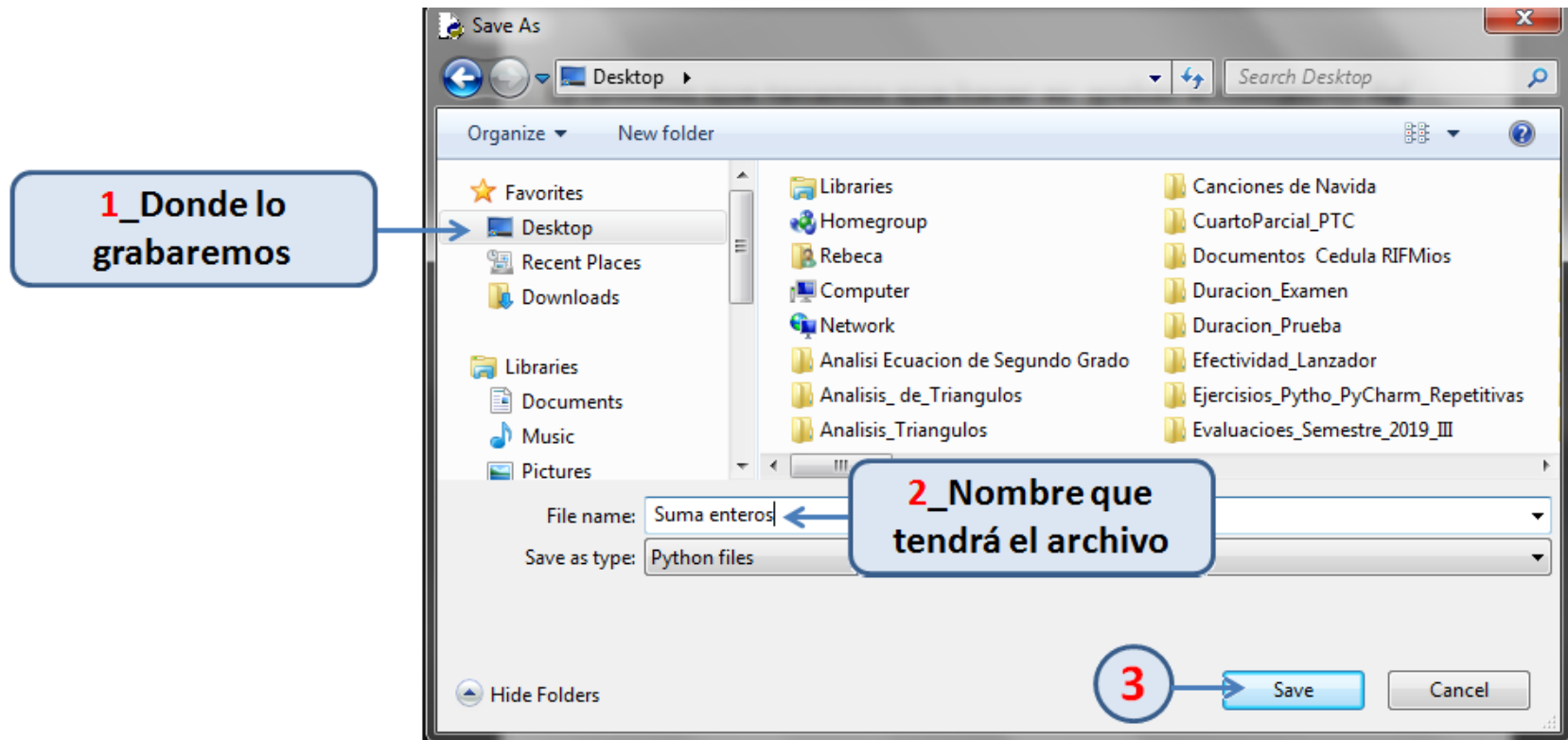
Lo primero que tenemos que hacer es grabar el contenido del editor en un archivo, esto lo esta indicando “Untitled” (“Sin Titulo”) en la barra de titulo del editor.



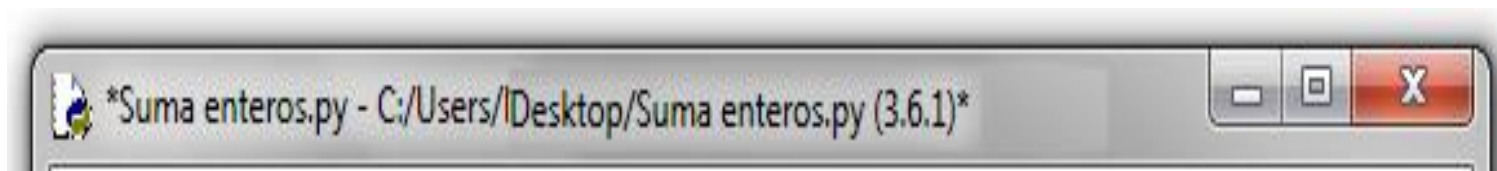
Procederemos ejecutando la siguiente secuencia de pasos:

Menu File → Save as..... (Grabar como)

se abre la siguiente ventana donde indicar:



Dando como resultado que ahora observamos en la barra de titulo:



Explicaremos a continuación el contenido del archivo

“Suma números”:

Comentarios: hay dos tipos de comentarios que podemos usar en Python, de varias líneas y de una sola línea.

De varias líneas:

```
"""  
    Aplicacion que lee dos enteros, los suma y muestra  
    el resultado. Elaorado por F Dura.  
"""
```

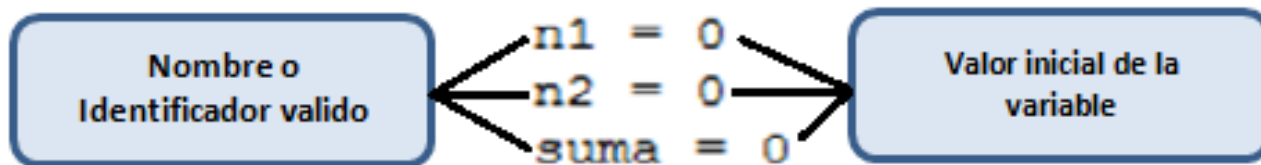
Comienza y termina con un grupo de 3 ("""") comillas dobles, todo el texto colocado en este bloque definido por las triples comillas dobles se cosedera un comentario de varias líneas. De una sola línea:

```
# Inicializar variables.
```

Comienza con numeral (#) y se coloca texto del comentario, nada mas. Un cometario son líneas de texto colocadas en el código a titulo de información, son ignoradas al ejecutar el programa.

Variables: valores que cambian durante la ejecución de un programa, desde el punto de vista del computador se puede decir que son áreas de memoria que se reservan para almacenar datos de un determinado tipo cuyo contenido puede variar durante la ejecución de un programa, identificadas con un nombre o identificador valido. En Python se coloca el nombre de las variables se iguala a un valor, la variable asumirá el tipo en función del valor asignado de acuerdo a la siguiente sintaxis:

Nombre_Variable = Valor_de_la_variable



Las variables `n1`, `n2` y `suma` se usaran para guardar valores **enteros**, para el caso de valores **reales** se igualarían a **0.0**.

Entrada de información:

Otra acción imprescindible que realiza un programa es la de tomar información. Si un programa no toma información los resultados siempre serán los mismos, por lo que resultará de poca utilidad.

La función `input()`:

Permite al usuario introducir información por dispositivo estándar de entrada (Teclado).

Sintaxis:

`Variable = input("Mensaje")`

Al ejecutarse esta instrucción el programa se detiene, despliega el mensaje y espera por el usuario para que introduzca la información, al presionar **Enter** la data se guardara en la variable. Es de hacer notar que la función solo captura texto, esto implica que si queremos leer valores numéricos el texto leído debe ser convertido al valor numérico correspondiente, entero o real, para esto usaremos las funciones **int** y **float**, la primera convierte un texto a un entero y la segunda a un real.

Los textos en Python son secuencias de caracteres que comienzan y terminan con comillas dobles, ejemplos: `"F Duran"`, `"4"`, `"25.2"`, el primero un texto que representa un nombre, el segundo un 4 y el tercero 25.2, los dos últimos no se podrían usar en cálculos numéricos.

Convierte el texto a un
valor numérico entero

```
n1 = int(input("De el valor de n1: "))  
n2 = int(input("De el valor de n2: "))
```

Captura el valor numérico
en forma de texto

Supongamos quiere introducir 3 y 5 el proceso se
desarrollaría así:

1. Captura de input: `n1 = int("3")` y `n2 = int("5")`
2. Conversión a entero: `n1 = 3` y `n2 = 5`

Salida de información

Todo programa hace una serie de acciones básicas. Una de estas acciones es la de mostrar información: texto, números, resultados... es algo imprescindible.

Para mostrar texto en Python usamos la función **print**, cuya sintaxis es:

print("El texto que quieres mostrar")

Ejemplo: **print("Salida de resultados")**, al ejecutarse el print se mostrara por pantalla **Salida de resultados**.

Si queremos mostrar un texto junto con la información que guarda una variable la sintaxis seria así:

print("El texto que quieres mostrar", variable)

Veamos la instrucción de salida de nuestro programa:

```
print("Suma: ", suma)
```

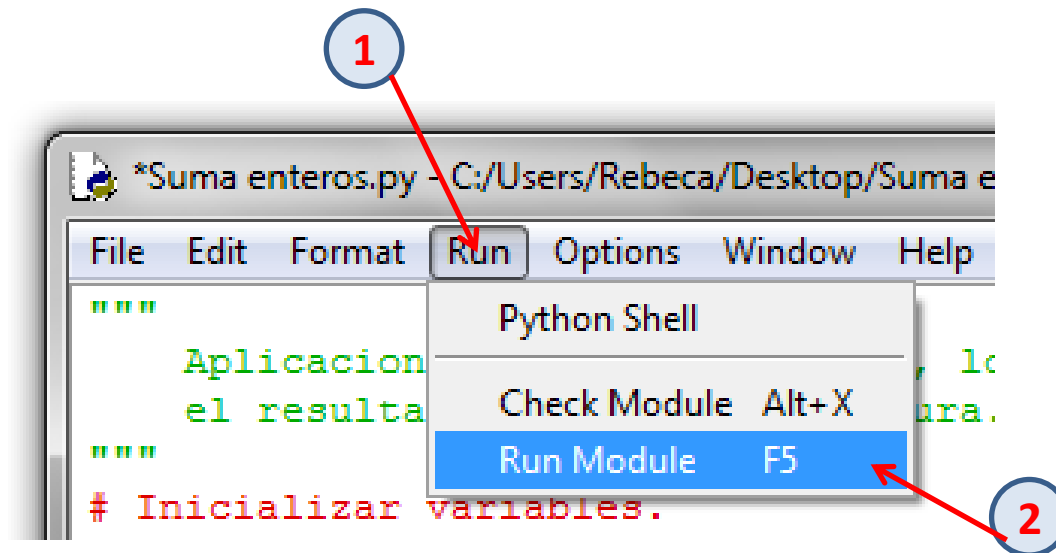
Texto indicando la
información mostrada

Variable cuyo contenido se
quiere mostrar

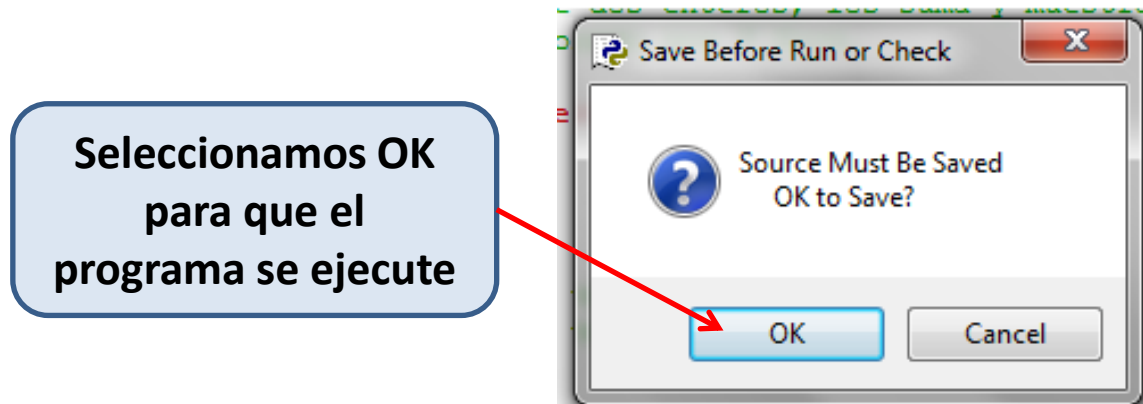
Ejecución de un programa e Python:

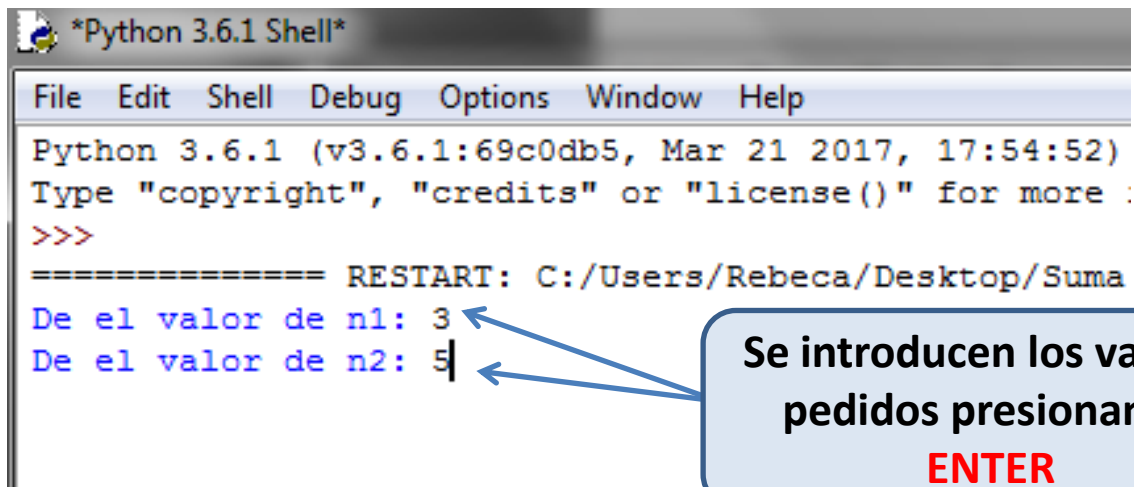
Una vez colocadas en el editor todas las instrucciones que forma parte del programa procederemos a su ejecución siguiendo este procedimiento:

Menu Run → Run module



En respuesta a esto el Python pide que guardemos el programa mostrando la siguiente ventana.

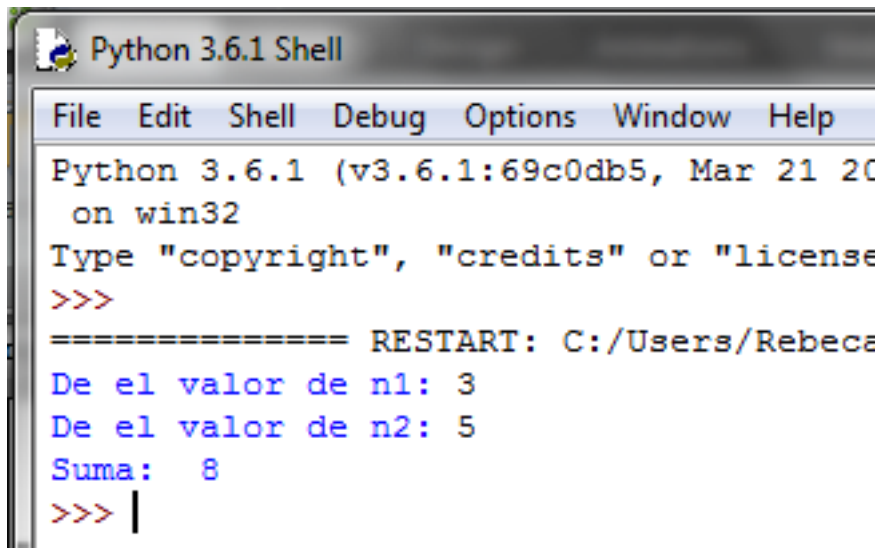




```
*Python 3.6.1 Shell*
File Edit Shell Debug Options Window Help
Python 3.6.1 (v3.6.1:69c0db5, Mar 21 2017, 17:54:52)
Type "copyright", "credits" or "license()" for more :
>>>
===== RESTART: C:/Users/Rebeca/Desktop/Suma
De el valor de n1: 3
De el valor de n2: 5|
```

Se introducen los valores pedidos presionando **ENTER**

Seguido se muestra la salida:



```
Python 3.6.1 Shell
File Edit Shell Debug Options Window Help
Python 3.6.1 (v3.6.1:69c0db5, Mar 21 20
on win32
Type "copyright", "credits" or "license
>>>
===== RESTART: C:/Users/Rebeca
De el valor de n1: 3
De el valor de n2: 5
Suma: 8
>>> |
```

¿Qué es la depuración (debugging)?

La programación es un proceso complejo, y a veces este proceso lleva a errores indefinidos, también llamados defectos o errores de programación (en inglés **'bugs'**), y el proceso de buscarlos y corregirlos es llamado depuración (en inglés **'debugging'**). Hay tres tipos de errores que pueden ocurrir en un programa. Es muy útil distinguirlos para encontrarlos más rápido.

Errores de sintaxis:

Python solo puede ejecutar un programa si está bien escrito. Al contrario, si el programa tiene algún error de sintaxis, el proceso falla y devuelve un mensaje de error. La palabra sintaxis se refiere a palabras del lenguaje mal escritas, falta de paréntesis etc. Si hay aunque sea un error de sintaxis en el programa, Python mostrará un mensaje de error y abortará su ejecución.

Errores en tiempo de ejecución

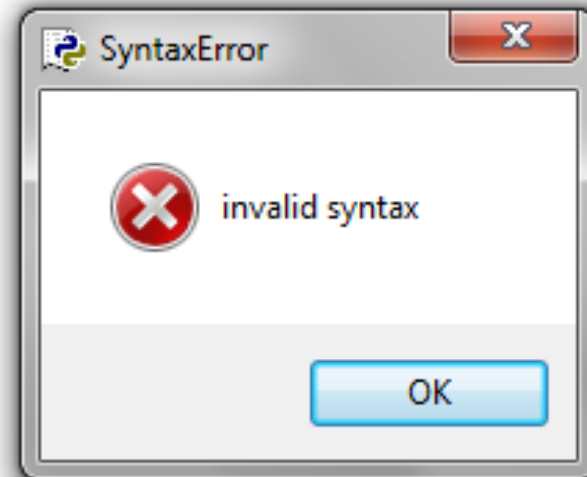
El segundo tipo de error es el de tiempo de ejecución. Este error aparece solo cuando se ejecuta un programa.

Errores de lógica

El tercer tipo de error es de lógica. Si hay un error de lógica en su programa, este será ejecutado sin ningún mensaje de error, pero el resultado no será el deseado. El programa ejecutará la lógica que usted le dijo que ejecutara.

Error sintaxis:

```
"""
    Aplicacion que lee dos enteros, los suma y muestra
    el resultado. Elaorado por F Dura.
"""
# Inicializar variables.
n1 = 0
n2 = 0
suma = 0
# Entrada de datos.
n1 = int(input("De el valor de n1: "))
n2 = int(input("De el valor de n2: "))
# Proceso
suma = n1 + n2
# Salida
print("Suma: ", suma)
```



Aquí ha provocado el error de manera intencional he borrado el ultimo paréntesis de la línea anterior donde se resalta **n2**, al ejecutar el programa se genera esta condición de error, el editor señala a la línea siguiente para indicar que el error esta aquí, pero no es así, si no que esta al final de la línea anterior se coloca el paréntesis y todo funciona bien. **Moraleja hay que revisar la anterior y la que señala con el error**

Errores en tiempo de ejecución:

```
# Inicializar variables.  
n1 = 0  
n2 = 0  
suma = 0  
# Entrada de datos.  
n1 = int(input("De el valor de n1: "))  
n2 = int(input("De el valor de n2: "))  
# Proceso  
suma = n1 + n2 + (n3) ← Error línea 13  
# Salida  
print("Suma: ", suma)
```

La expresion usa una variable a la cual no se le asigno ninguna data dando error al momento de ser evaluada.

```
===== RESTART: C:\Users\Rebeca\Desktop\Suma enteros.py ==  
De el valor de n1: 2  
De el valor de n2: 4  
Traceback (most recent call last):  
  File "C:\Users\Rebeca\Desktop\Suma enteros.py", line 13, in <module>  
    suma = n1 + n2 + (n3) ←  
NameError: name 'n3' is not defined
```

En este caso se ejecuta el programa se piden los valores de **n1** y **n2** pero al evaluar la suma se interrumpe la ejecución, mostrándose el mensaje que indica que **n3** no esta definida, corrección: se elimina, o, se inicializa y se le asigna un valor funcionando bien todo.

Error de lógica: podría ser que al momento de escribir la instrucción $\text{suma} = n1 + n2$, en de colocar el operador **+** colocáramos un **-** dando como resultado una resta en vez de una suma.

Teniendo un programa que funcione bien altérela quitado o cambiando algunas partes del programa, ejecútelo y vea el error que se produce corríjalo, y pruebe de nuevo, es una forma de aprender los errores que se puede generar en diversas situaciones.