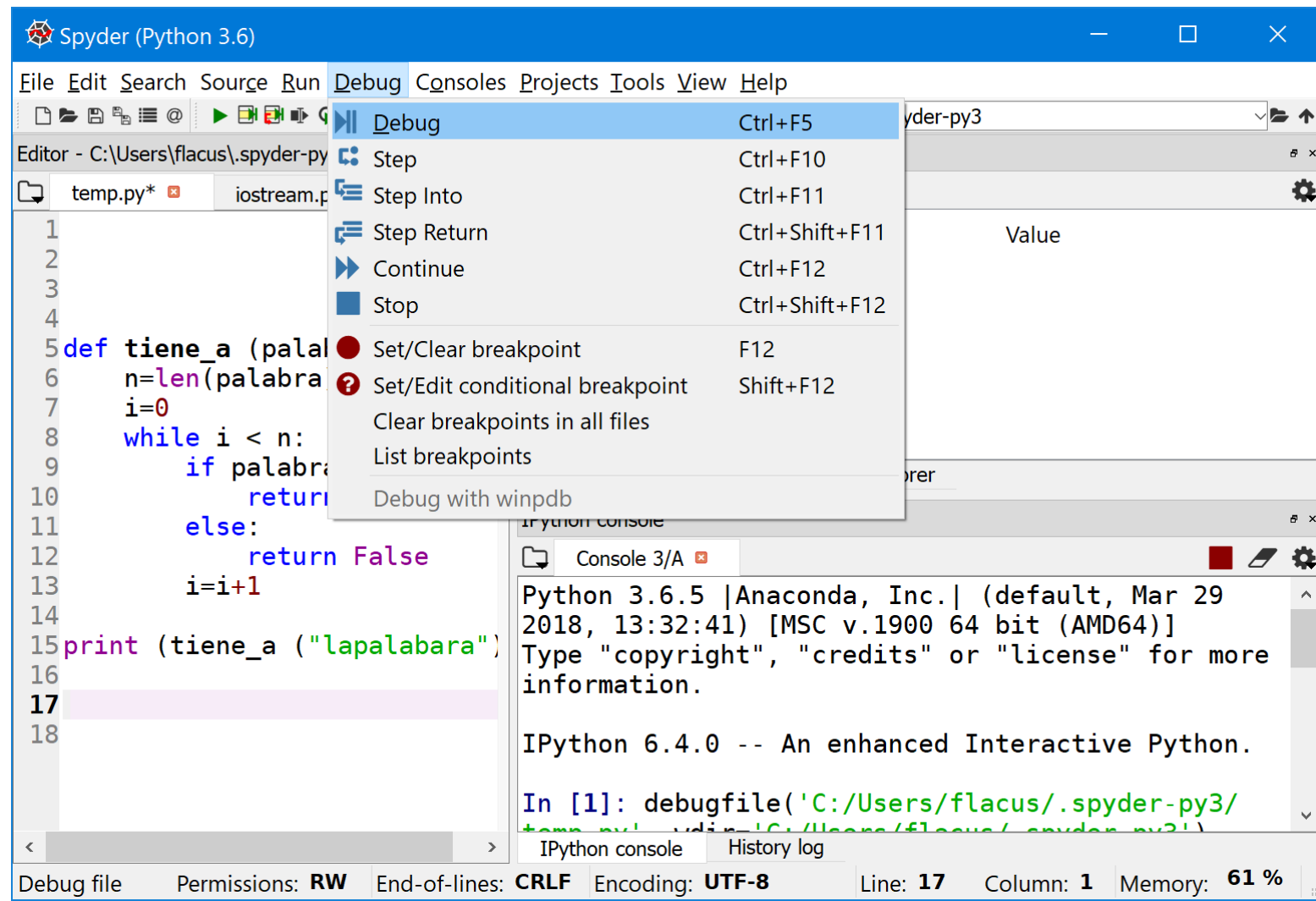


Este es un breve "manual" sobre cómo usar Spyder para hacer debugging de un programa.

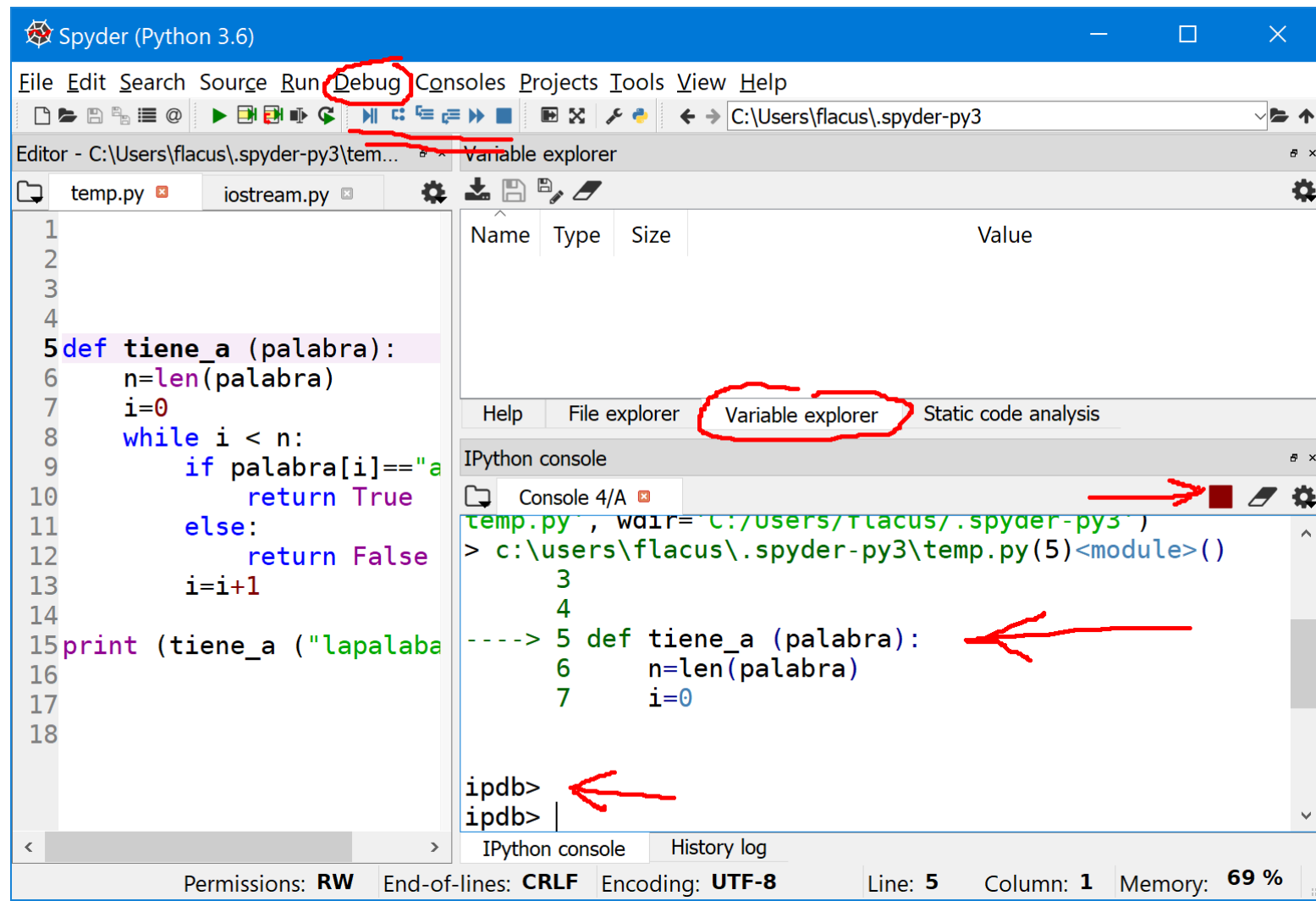
Primero entramos al modo DEBUG: (Ctrl+F5) El programa queda pausado antes de comenzar. Luego pedimos Step (ejecutar línea) / Step into (entrar) / Step return (ejecutar hasta salir), comparando el estado esperado de las variables con su estado real, PREDICIENDO lo que ocurrirá.



Queremos ver la solapa Variable Explorer (arriba derecho). Sabemos que el programa está en ejecución pero pausado por el "Stop" rojo de la derecha. Sabemos que estamos en modo DEBUG por el prompt "ipdb" abajo

Step ejecuta una línea. Step Into entra en la función referida. Step return ejecuta hasta salir de la función en la que está.

Vamos a debuggear el Código del ejercicio 11 de la clase 1 (el primero de los tres).



Pedimos algunos STEP Into hasta llegar a la línea 9. Vemos que todas las variable internas de la función están definidas y con sus valores.

Como  $i=0$  sabemos que es la primera iteración. Corroboramos que  $n=9$  ("lapalabra" tiene 9 letras)

En este punto se evalúa el IF, y saltaremos a alguna de las dos ramas de ejecución.

Pedimos un Step.

The screenshot shows the Spyder Python IDE with a file named `temp.py` open. The code defines a function `tiene_a` that checks if a word contains the letter 'a'. The current execution point is at line 9, where the condition `if palabra[i] == 'a':` is being evaluated. A red arrow points to this line. The Variable explorer on the right shows the current state of the function's variables: `i` is an integer with value 0, `n` is an integer with value 9, and `palabra` is a string with value 'lapalabra'. The IPython console at the bottom shows the stack trace and the current state of the function's execution, indicating that the function is returning True.

```
1# -*- coding: utf-8 -*-
2
3
4
5def tiene_a(palabra):
6    n = len(palabra)
7    i = 0
8    while i<n:
9        if palabra[i] == 'a':
10            return True
11        else:
12            return False
13        i += 1
14
15print (tiene_a ("lapalabra"))
```

| Name    | Type | Size | Value     |
|---------|------|------|-----------|
| i       | int  | 1    | 0         |
| n       | int  | 1    | 9         |
| palabra | str  | 1    | lapalabra |

```
ipdb> > c:\users\flacus\temp.py(9)tiene_a()
7      i = 0
8      while i<n:
----> 9          if palabra[i] == 'a':
10              return True
```

Permissions: RW End-of-lines: CRLF Encoding: UTF-8 Line: 9 Column: 1 Memory: 63 %

El IF result FALSO, lo que concuerda con lo esperado porque `palabra[1]="l" != "a"` (la primera letra) no es "a".

La próxima instrucción será RETURN !!!

Pero si es Return saldremos de la function y aún no evaluamos mas que la primera posición. Esto es lo que deseamos ?

Para no entrar dentro del return (que es complicada) pedimos un Step (NO un Step Into)

The screenshot shows the Spyder Python IDE interface. The main editor displays a Python script named `temp.py` with the following code:

```
1# -*- coding: utf-8 -*-
2
3
4
5def tiene_a(palabra):
6    n = len(palabra)
7    i = 0
8    while i < n:
9        if palabra[i] == 'a':
10            return True
11        else:
12            return False
13        i += 1
14
15print (tiene_a ("lapalabra"))
```

A red arrow points to line 12, `return False`, indicating a step-through operation. The Variable explorer on the right shows the current state of variables:

| Name    | Type | Size | Value     |
|---------|------|------|-----------|
| i       | int  | 1    | 0         |
| n       | int  | 1    | 9         |
| palabra | str  | 1    | lapalabra |

The IPython console at the bottom shows the execution of the code, with the current step being line 12:

```
11 else:
12     return False
13     i += 1
14

ipdb>
ipdb>
```

The status bar at the bottom indicates: Permissions: RW, End-of-lines: CRLF, Encoding: UTF-8, Line: 12, Column: 1, Memory: 65 %.

En este punto acabamos de volver de la function. Las variables internas ya no están visibles (salimos de su "scope"). El programa sigue en ejecución, (flechas)

Salir de la función después de haber analizado solo la primera letra no era lo deseado. Que pasó ? A pensar !

Pedimos un Step ...

The screenshot shows the Spyder Python IDE interface. The editor window displays the following code:

```
1# -*- coding: utf-8 -*-
2
3
4
5def tiene_a(palabra):
6    n = len(palabra)
7    i = 0
8    while i < n:
9        if palabra[i] == 'a':
10            return True
11        else:
12            return False
13        i += 1
14
15print (tiene_a ("lapalabra"))
```

The Variable explorer on the right is empty, with a red circle around it. The IPython console at the bottom shows the execution flow:

```
12         return False
13         i += 1
14
---> 15 print (tiene_a ("lapalabra"))

ipdb>
ipdb>
```

A red arrow points to the 'ipdb>' prompt. The status bar at the bottom indicates: Permissions: RW, End-of-lines: CRLF, Encoding: UTF-8, Line: 15, Column: 1, Memory: 65 %.



El programa terminó. Las flechas indicant el STOP apagado y el prompt normal. Tenemos control del IDE y la tarea ahora es pensar porqué la función terminó antes de lo deseado.

