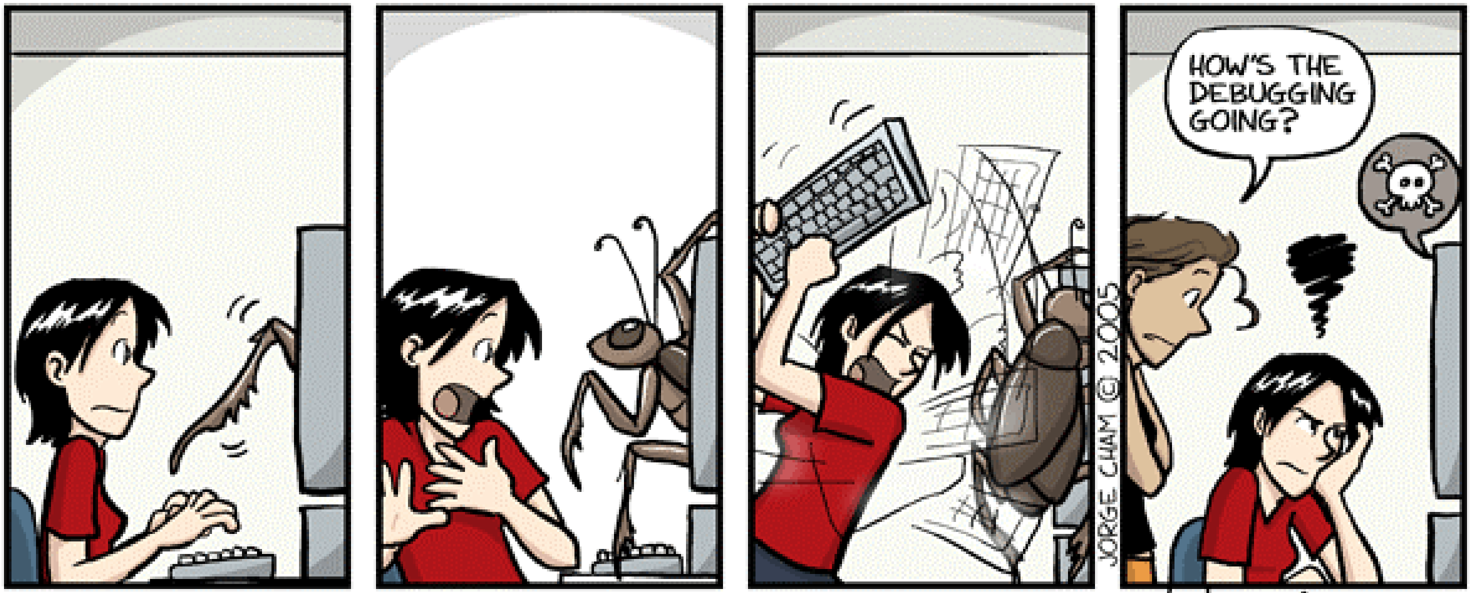
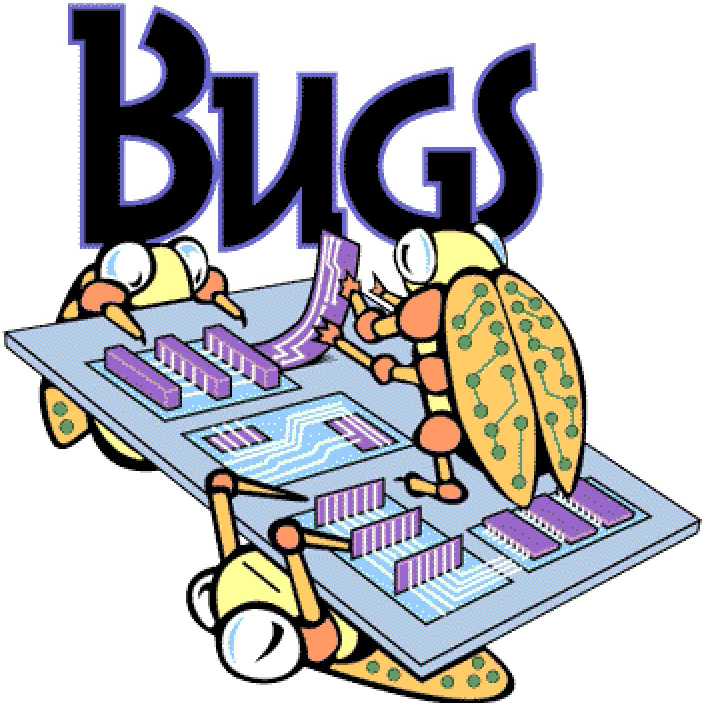
BUGS AND DEBUGGING



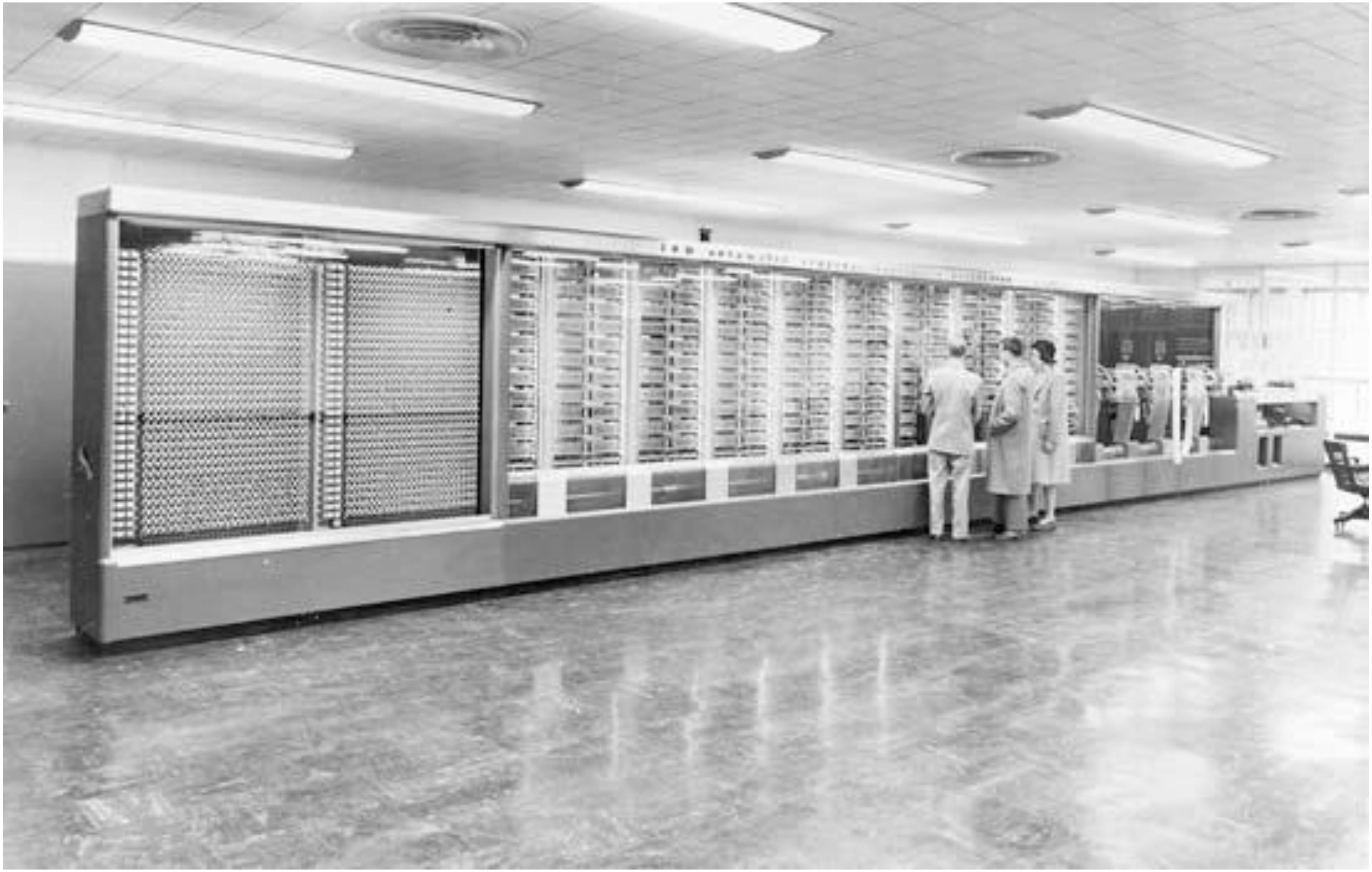
Bugs?

- In 1947, Grace Murray Hopper was working on the Harvard University Mark II Aiken Relay Calculator (a primitive computer).

- On the 9th of September, 1947, when the machine was experiencing problems, an investigation showed that there was a moth trapped between the points of Relay #70, in Panel F.

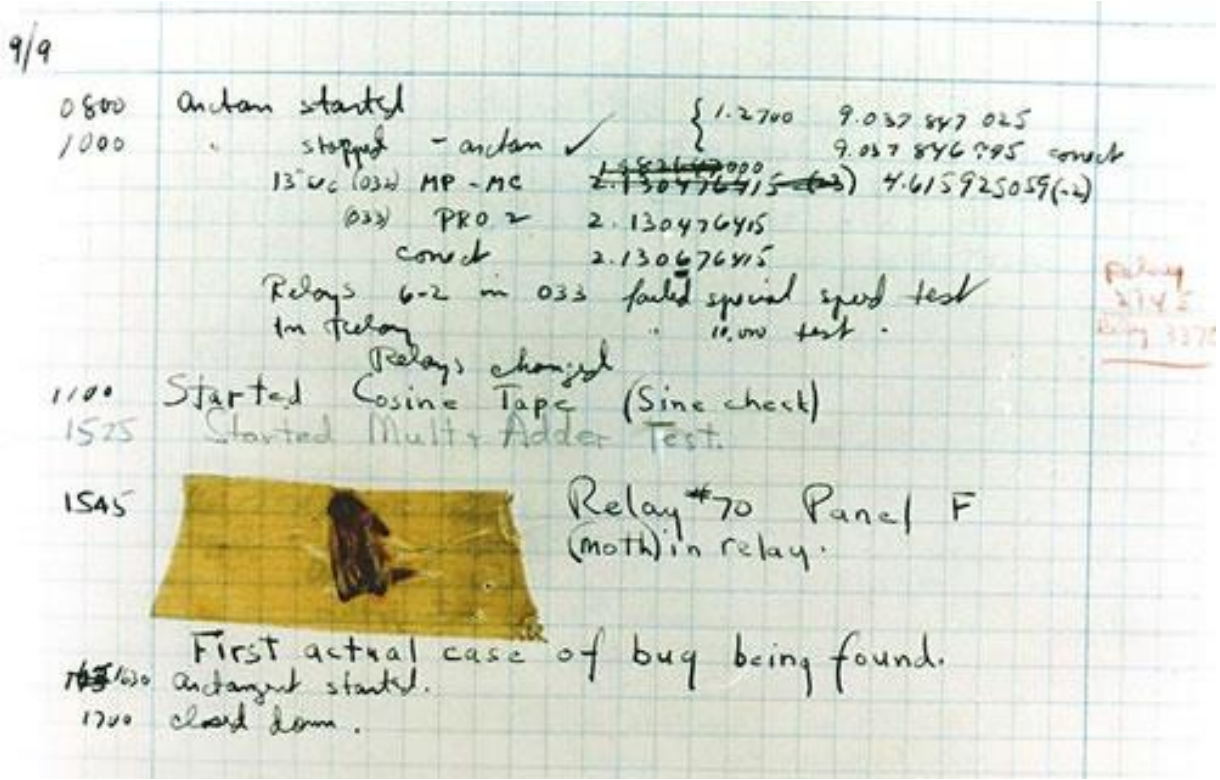


Mark I



Bugs?

- The operators removed the moth and affixed it to the log. (See the picture above.) The entry reads: "First actual case of bug being found."



9/9

0800 Antran started

1000 " stopped - antran ✓ {1.2700 9.037 847 025

9.037 846 995 const

13" UC (032) MP - MC ~~1.982147000~~ ~~2.130476415~~ (-3)) 4.615925059(-2)

(033) PRO 2 2.130476415

const 2.130676415

Relays 6-2 in 033 failed special speed test

In relay " 11,000 test .

Relays changed

Relay 2145

Relay 3370

1100 Started Cosine Tape (Sine check)

1525 Started Mult + Adder Test.

1545 Relay #70 Panel F

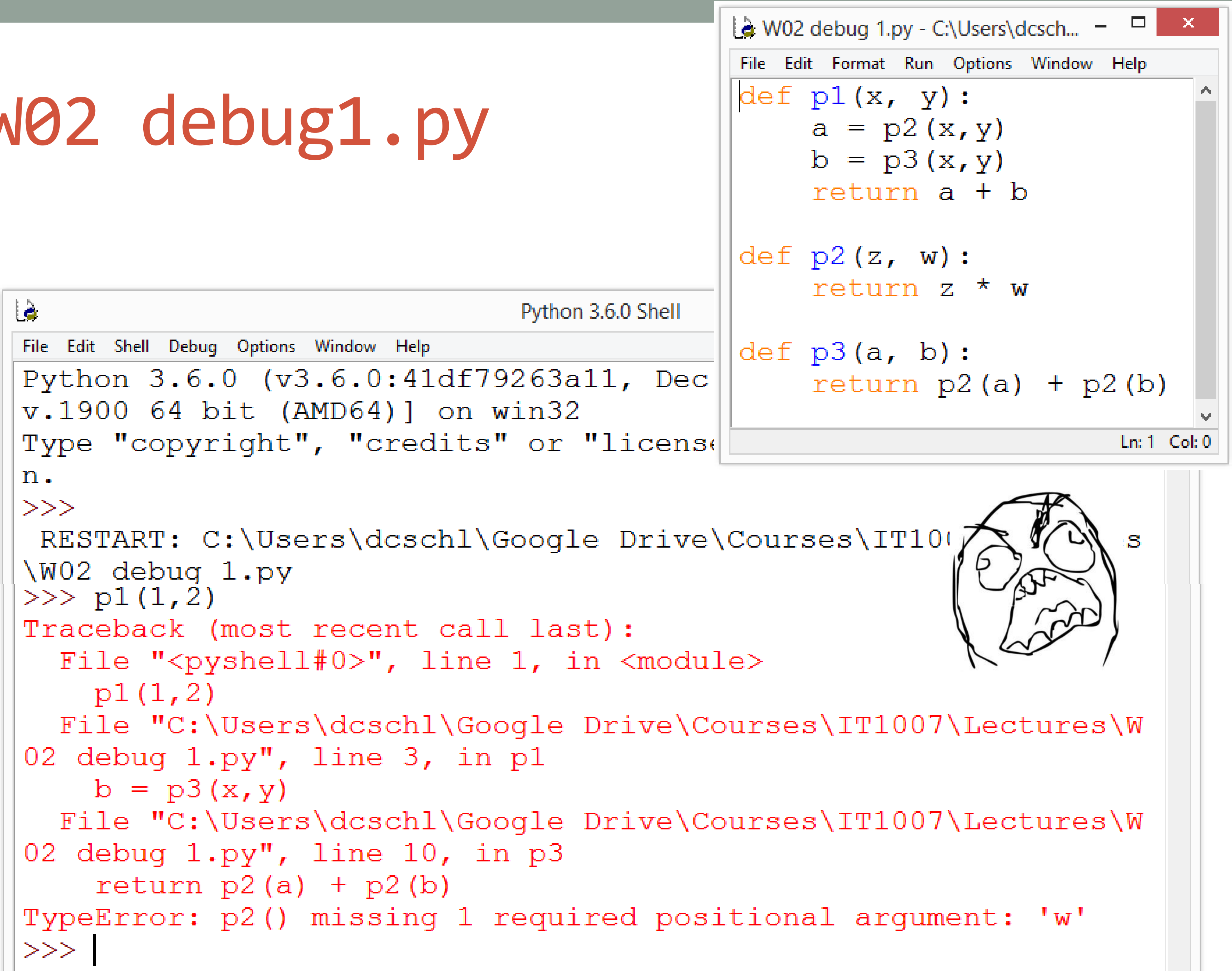
(moth) in relay.

First actual case of bug being found.

1630 Antrangent started.

1700 closed down.

W02 debug1.py



W02 debug 1.py - C:\Users\dcsch...
File Edit Format Run Options Window Help
Ln: 1 Col: 0

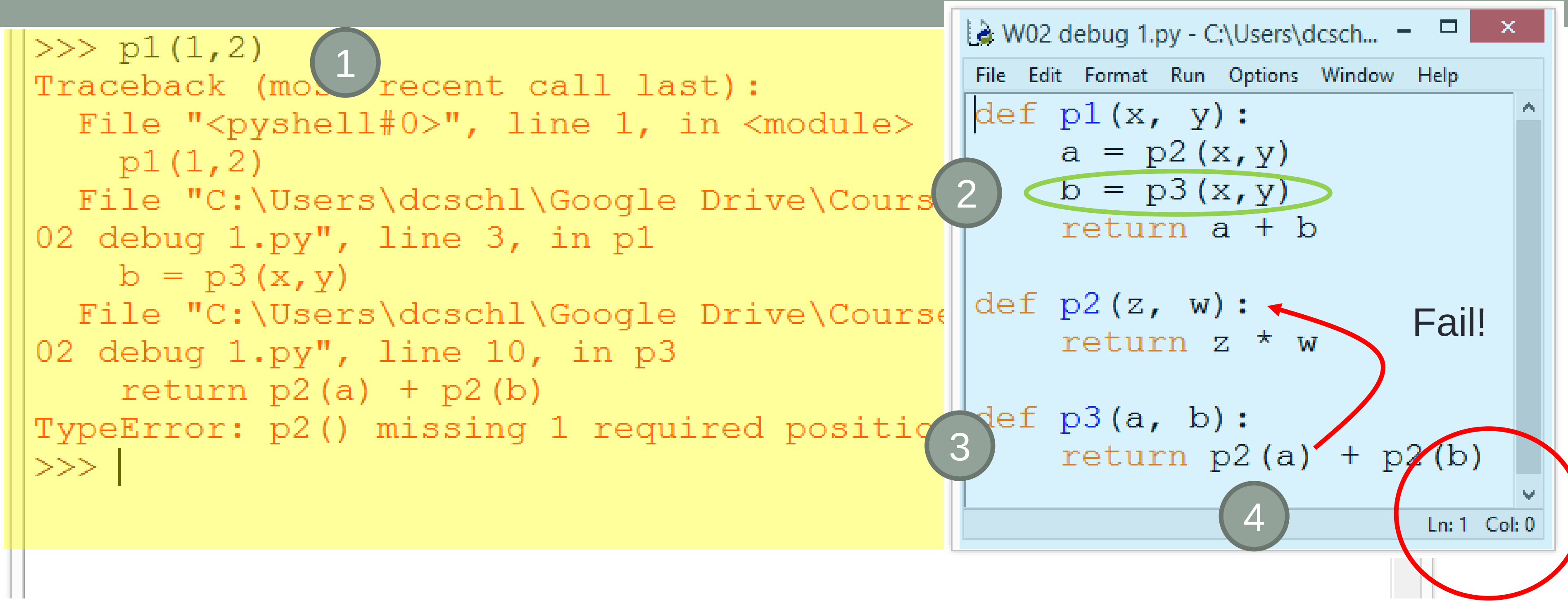
```
def p1(x, y):
    a = p2(x,y)
    b = p3(x,y)
    return a + b

def p2(z, w):
    return z * w

def p3(a, b):
    return p2(a) + p2(b)
```

Python 3.6.0 Shell
File Edit Shell Debug Options Window Help

```
Python 3.6.0 (v3.6.0:41df79263a11, Dec
v.1900 64 bit (AMD64)] on win32
Type "copyright", "credits" or "license
n.
>>>
 RESTART: C:\Users\dcschl\Google Drive\Courses\IT10                s
\W02 debug 1.py
>>> p1(1,2)
Traceback (most recent call last):
  File "<pyshell#0>", line 1, in <module>
    p1(1,2)
  File "C:\Users\dcschl\Google Drive\Courses\IT1007\Lectures\W
02 debug 1.py", line 3, in p1
    b = p3(x,y)
  File "C:\Users\dcschl\Google Drive\Courses\IT1007\Lectures\W
02 debug 1.py", line 10, in p3
    return p2(a) + p2(b)
TypeError: p2() missing 1 required positional argument: 'w'
>>> 
```

OKO

KEEP
CALM
AND
DON'T
PANIC
KeepCalmAndPosters.com



```
>>> p1(1,2)
Traceback (mo   recent call last):
  File "<pyshell#0>", line 1, in <module>
    p1(1,2)
  File "C:\Users\dcschl\Google Drive\Cours
02 debug 1.py", line 3, in p1
    b = p3(x,y)
  File "C:\Users\dcschl\Google Drive\Course
02 debug 1.py", line 10, in p3
    return p2(a) + p2(b)
TypeError: p2() missing 1 required positio
>>> |
```

W02 debug 1.py - C:\Users\dcsch...
File Edit Format Run Options Window Help
1
2
3
4
Fail!
Ln: 1 Col: 0

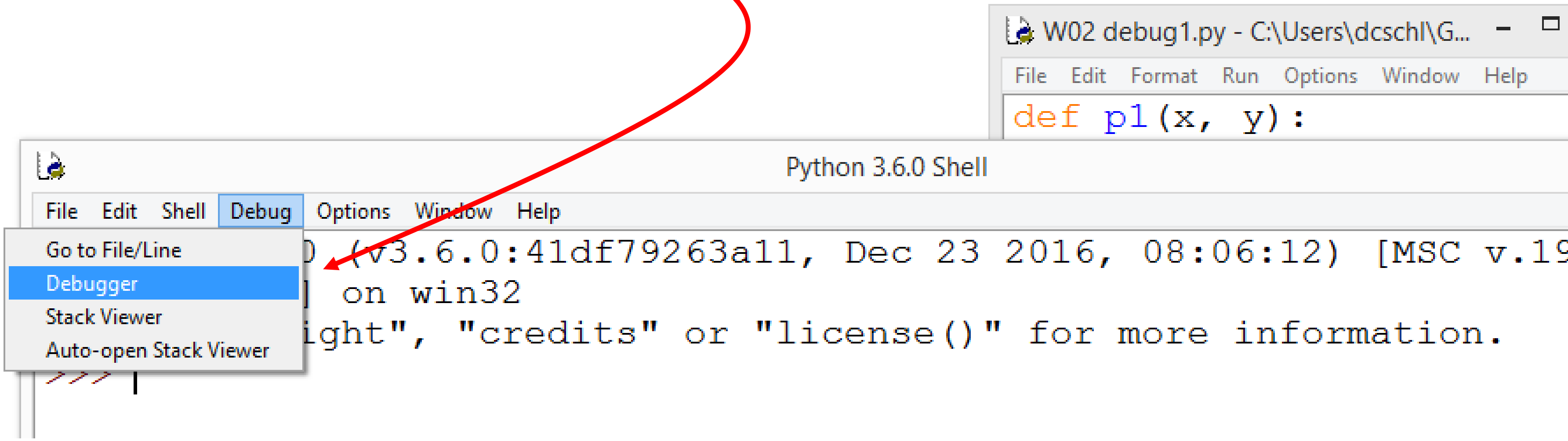
```
def p1(x, y):
    a = p2(x,y)
    b = p3(x,y)
    return a + b

def p2(z, w):
    return z * w

def p3(a, b):
    return p2(a) + p2(b)
```

```
Traceback (most recent call last):
1  File "<pyshell#0>", line 1, in <module>
    p1(1,2)
2  File "C:\Users\dcschl\Google Drive\Courses\IT1007\Lectures\W02 debug 1.py",
line 3, in p1
    b = p3(x,y)
3  File "C:\Users\dcschl\Google Drive\Courses\IT1007\Lectures\W02 debug 1.py",
line 10, in p3
    return p2(a) + p2(b)
4 TypeError: p2() missing 1 required positional argument: 'w'
```

THE IDLE DEBUGGER

Using the IDLE Debugger

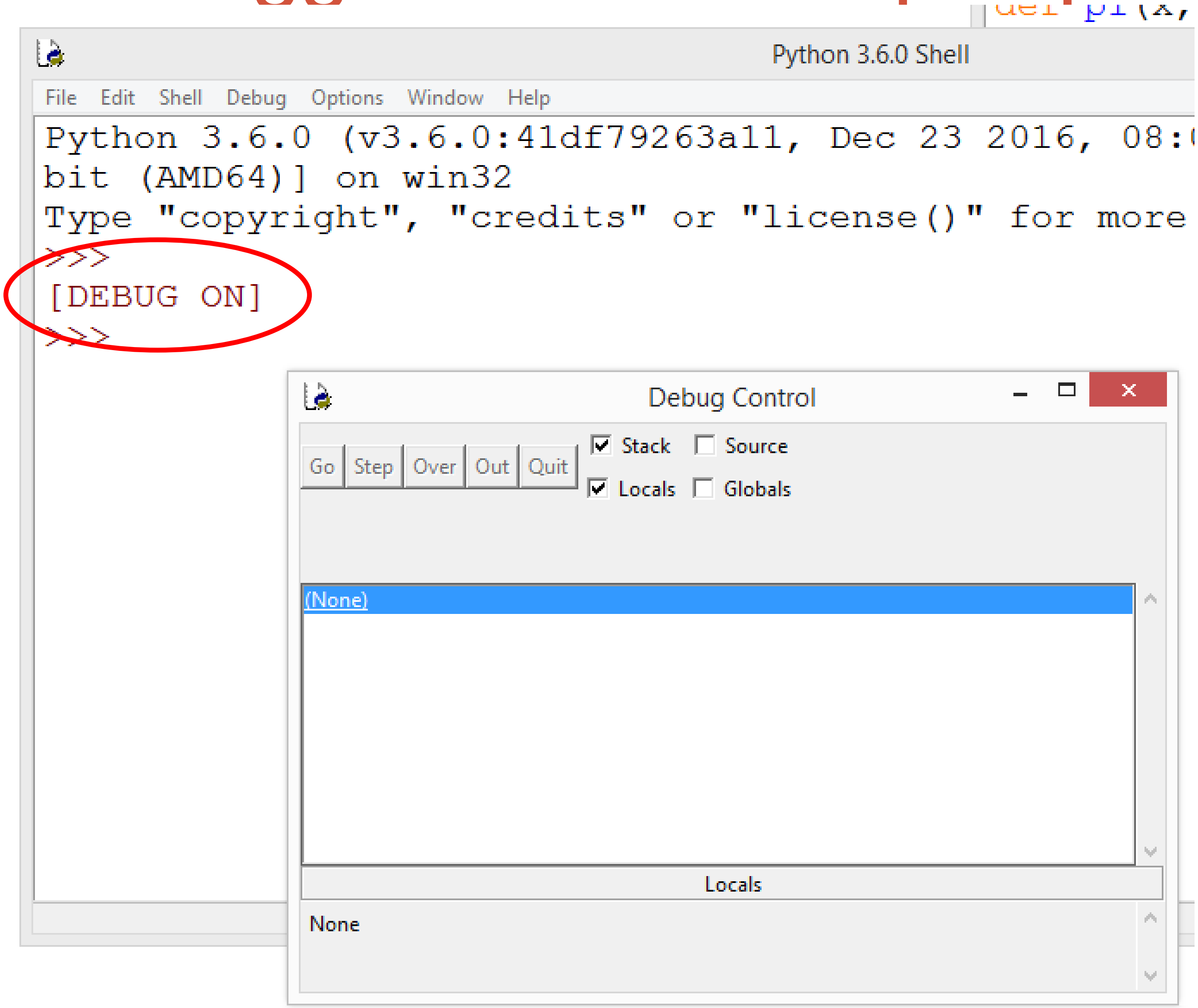
- Load in your source code
- Turn on the debugger by



```
def p1(x, y):
```

```
0 (v3.6.0:41df79263a11, Dec 23 2016, 08:06:12) [MSC v.19
] on win32
ight", "credits" or "license()" for more information.
>>> 
```

The Debugger Window Pops up



```
Python 3.6.0 (v3.6.0:41df79263a11, Dec 23 2016, 08:0
bit (AMD64)] on win32
Type "copyright", "credits" or "license()" for more
>>>
[DEBUG ON]
>>>
```

☑ Stack □ Source
☑ Locals □ Globals

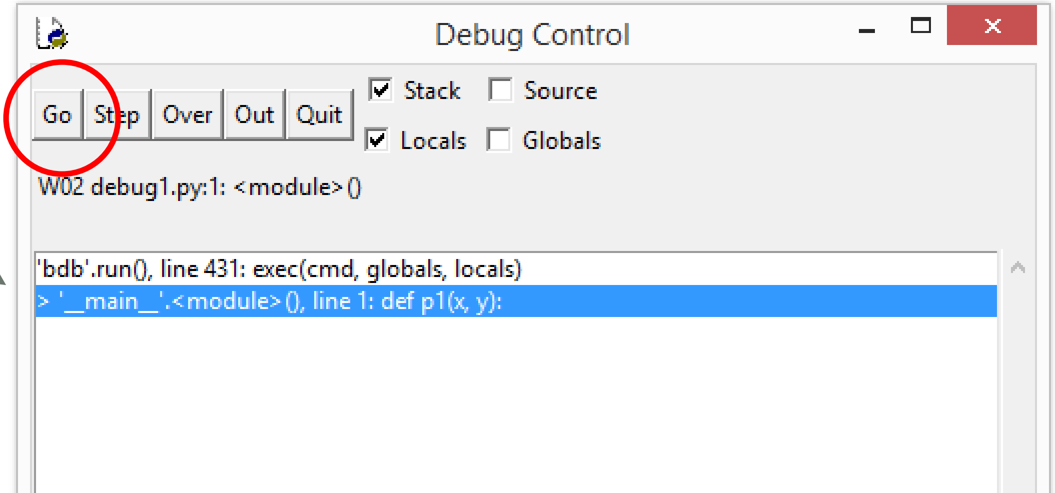
(None)

Locals

None

Using the IDLE Debugger

- Go to your source code window to "run"
- Then the debugger will pause the program at the first line of code and wait for you
- You can click the button "Go"
  - That will make the program run
  - At this point we don't have any error
    - Because by "running" the code, we just define the three functions



W02 debug1.py:1: <module>()

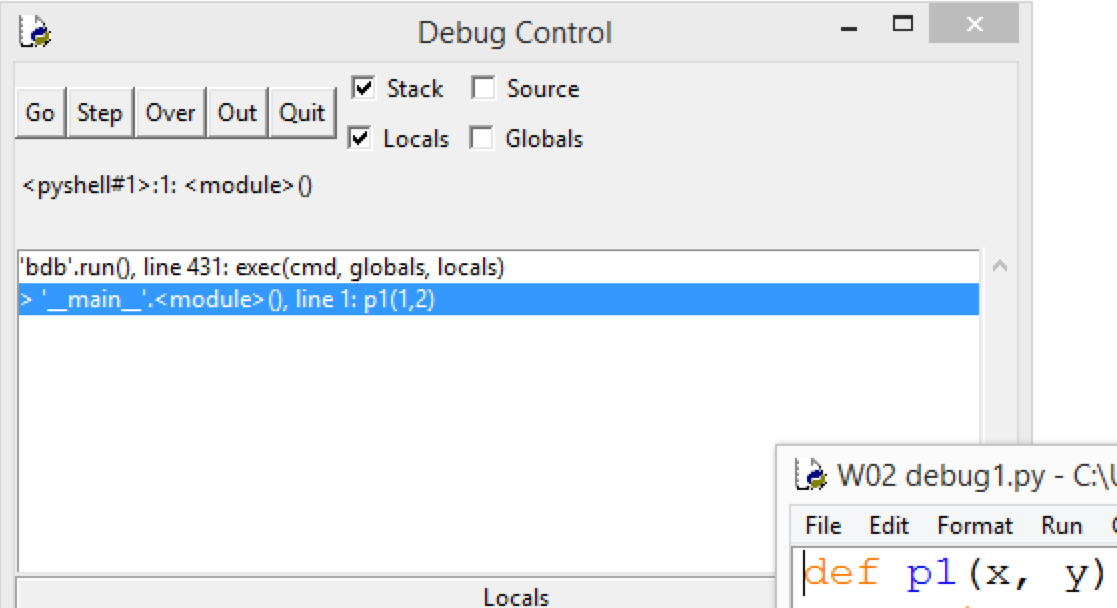
```
'bdb'.run(), line 431: exec(cmd, globals, locals)
> '__main__'.<module>(), line 1: def p1(x, y):
```

Using the IDLE Debugger

- Let’s execute the function in debug mode
- In the shell, type

```
p1(1,2)
```

- Then the debugger will pause at the first line of p1
    - If you type “go” now, you will get an error like the last time



```
<pyshell#1>:1: <module>()
```

```
'bdb'.run(), line 431: exec(cmd, globals, locals)
> '__main__'.<module>(), line 1: p1(1,2)
```

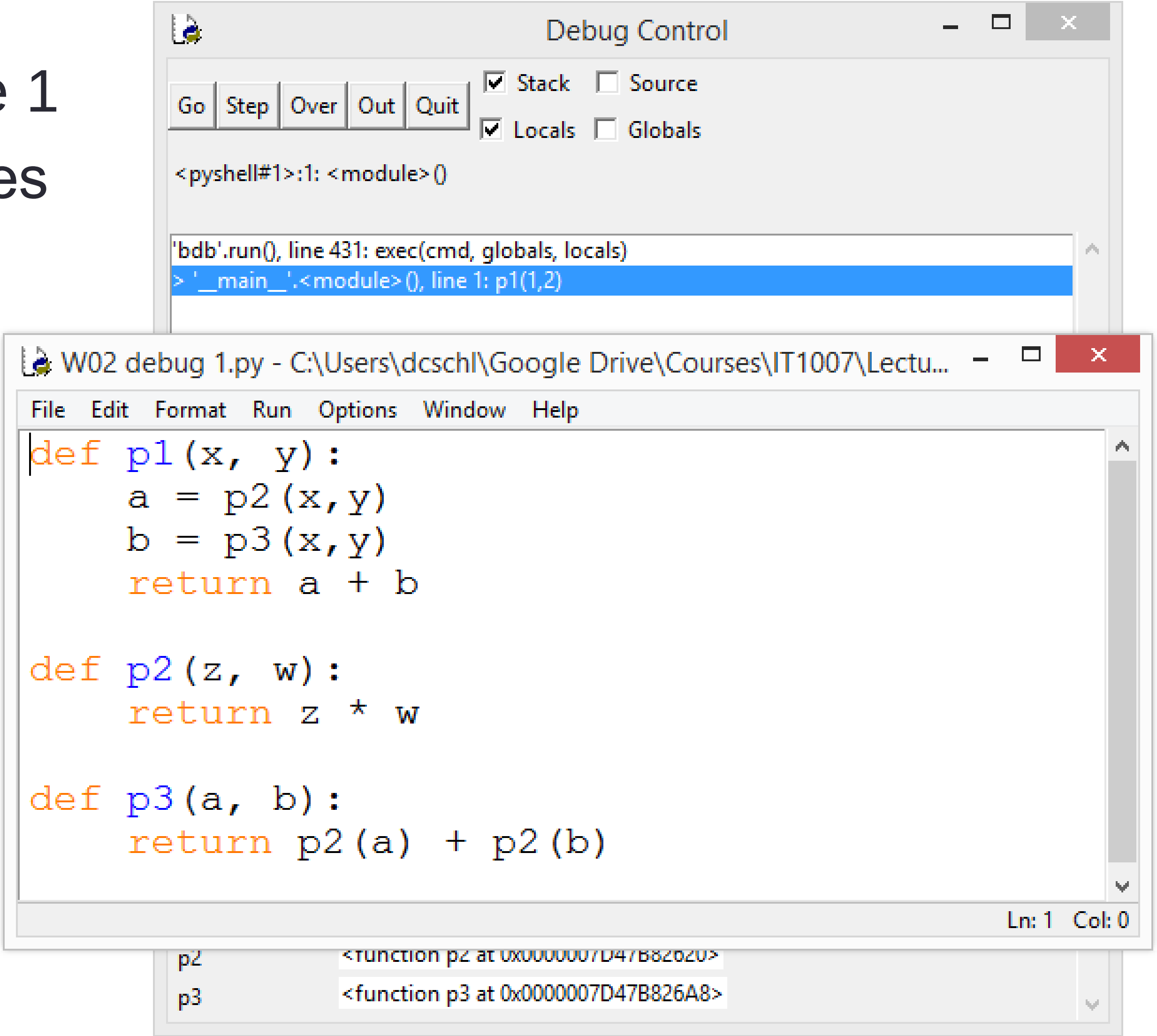
```
def p1(x, y)
```

Using the IDLE Debugger

- Go
  - Clicking this will run the program until the next break point is reached. You can insert break points in your code by right clicking and selecting Set Breakpoint. Lines that have break points set on them will be highlighted in yellow.
- Step
  - This executes the next statement. If the statement is a function call, it will enter the function and stop at the first line.
- Over
  - This executes the next statement just as Step does. But it does not enter into functions. Instead, it finishes executing any function in the statement and stops at the next statement in the same scope.
- Out
  - This exits the current function and stops in the caller of the current function.
  - After using Step to step into a function, you can use Out to quickly execute all the statements in the function and get back out to the outer function.
- Quit: This terminates execution.

Using the IDLE Debugger

- Currently in line 1
- Click “Step” goes to line 2



```
<pyshell#1>:1: <module>()
```

```
'bdb'.run(), line 431: exec(cmd, globals, locals)
> '__main__'.<module>(), line 1: p1(1,2)
```

```python
def p1(x, y):
    a = p2(x,y)
    b = p3(x,y)
    return a + b

def p2(z, w):
    return z * w

def p3(a, b):
    return p2(a) + p2(b)
```

```
p2    <function p2 at 0x0000007D47B82620>
p3    <function p3 at 0x0000007D47B826A8>
```

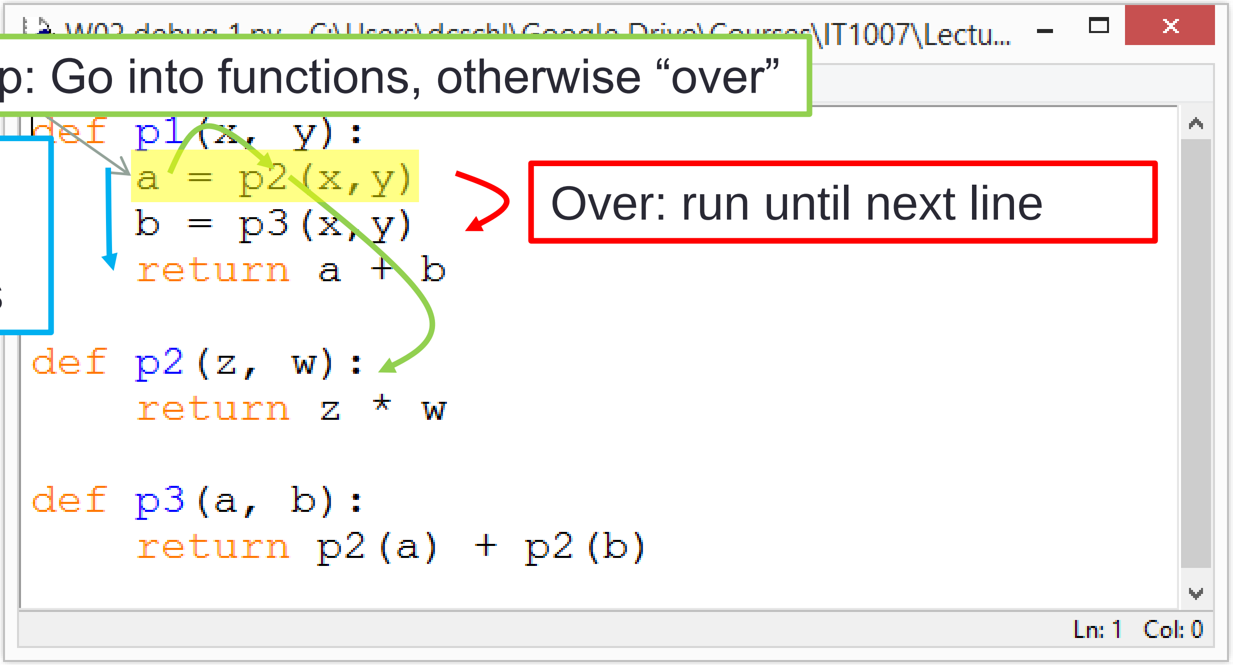
Using the IDLE Debugger

Current position

Step: Go into functions, otherwise “over”

Out: run until the current function ends

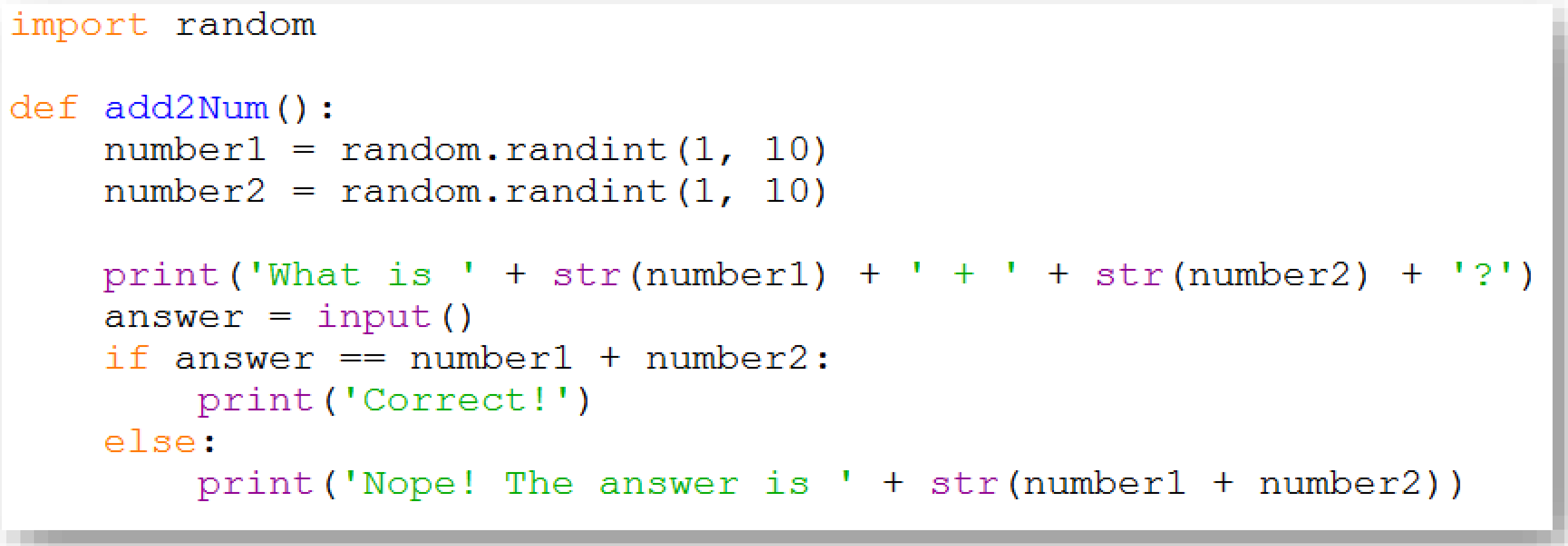
Over: run until next line



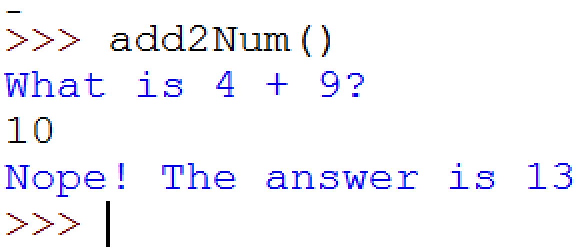
W02_debug_1.py - C:\Users\dcsahl\Google Drive\Courses\IT1007\Lectu...

```
def p1(x, y):
    a = p2(x,y)
    b = p3(x,y)
    return a + b

def p2(z, w):
    return z * w

def p3(a, b):
    return p2(a) + p2(b)
```

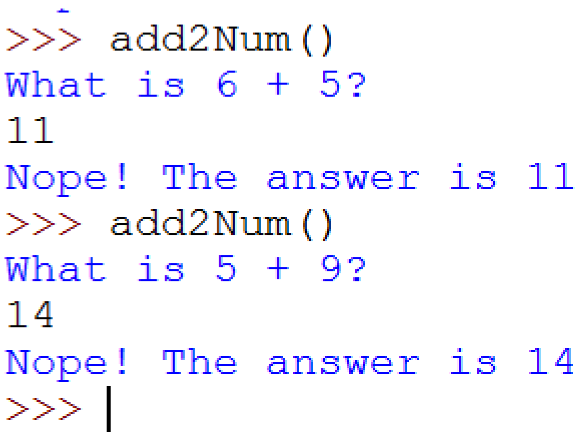
More Debugging (BuggyAddNum)



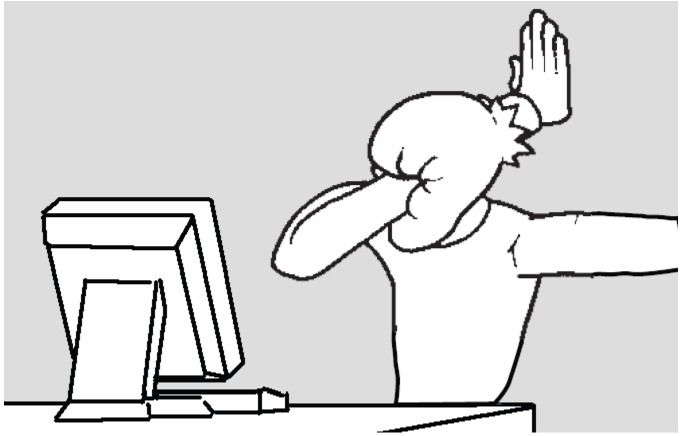
```
import random

def add2Num():
    number1 = random.randint(1, 10)
    number2 = random.randint(1, 10)

    print('What is ' + str(number1) + ' + ' + str(number2) + '?')
    answer = input()
    if answer == number1 + number2:
        print('Correct!')
    else:
        print('Nope! The answer is ' + str(number1 + number2))
```

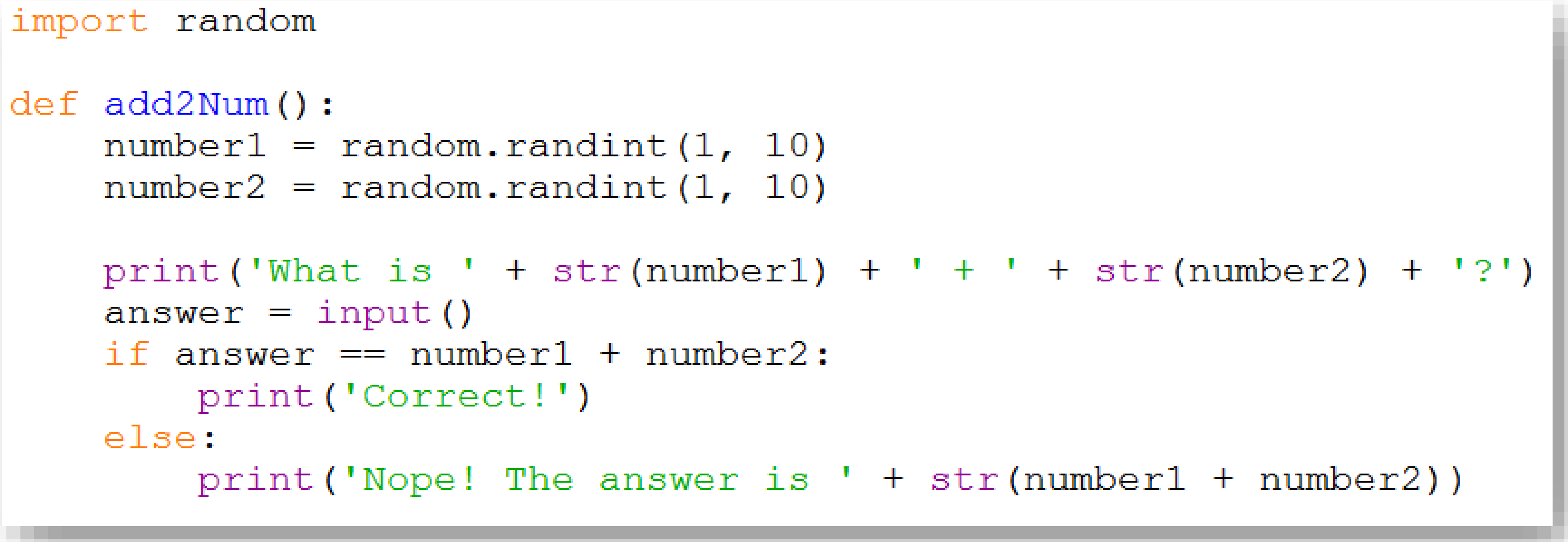


```
-
>>> add2Num()
What is 4 + 9?
10
Nope! The answer is 13
>>> |
```



```
>>> add2Num()
What is 6 + 5?
11
Nope! The answer is 11
>>> add2Num()
What is 5 + 9?
14
Nope! The answer is 14
>>> |
```

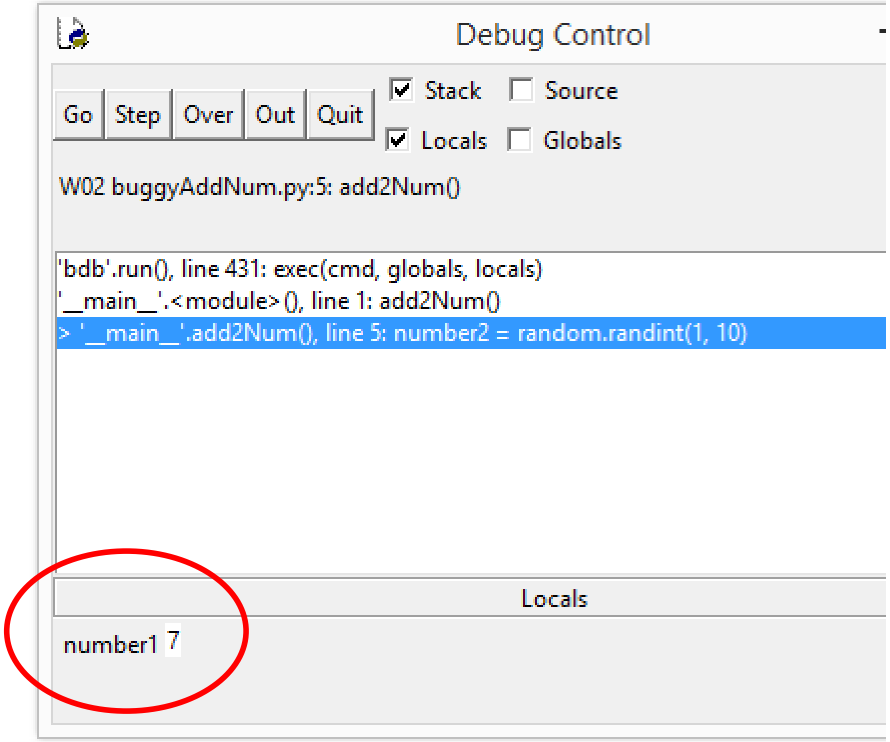




```
import random

def add2Num():
    number1 = random.randint(1, 10)
    number2 = random.randint(1, 10)

    print('What is ' + str(number1) + ' + ' + str(number2) + '?')
    answer = input()
    if answer == number1 + number2:
        print('Correct!')
    else:
        print('Nope! The answer is ' + str(number1 + number2))
```

Turn on Debugger

- After a few steps



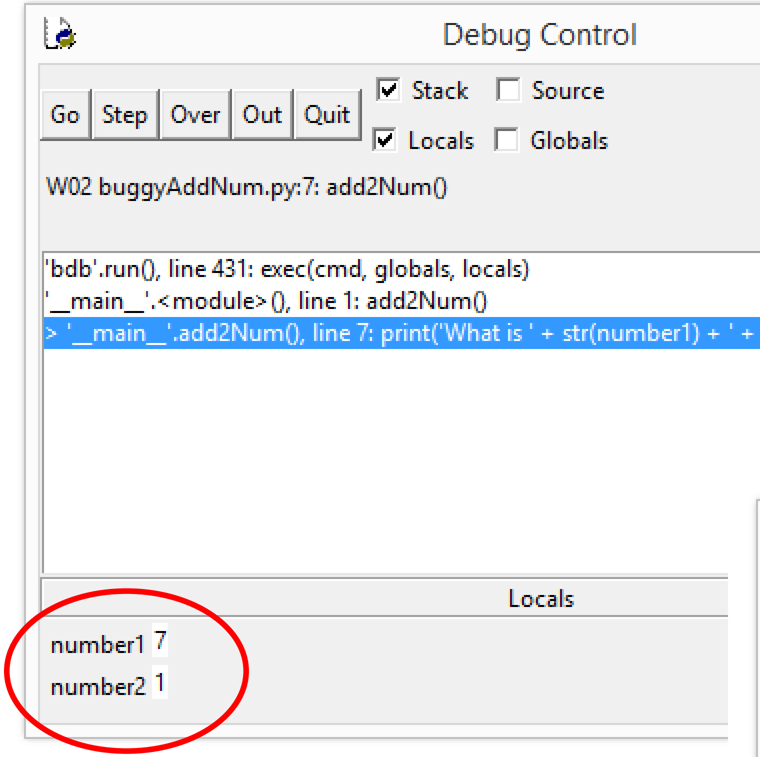
☑ Stack □ Source
☑ Locals □ Globals

W02 buggyAddNum.py:5: add2Num()

```
'bdb'.run(), line 431: exec(cmd, globals, locals)
'__main__'.<module>(), line 1: add2Num()
> '__main__'.add2Num(), line 5: number2 = random.randint(1, 10)
```

Locals

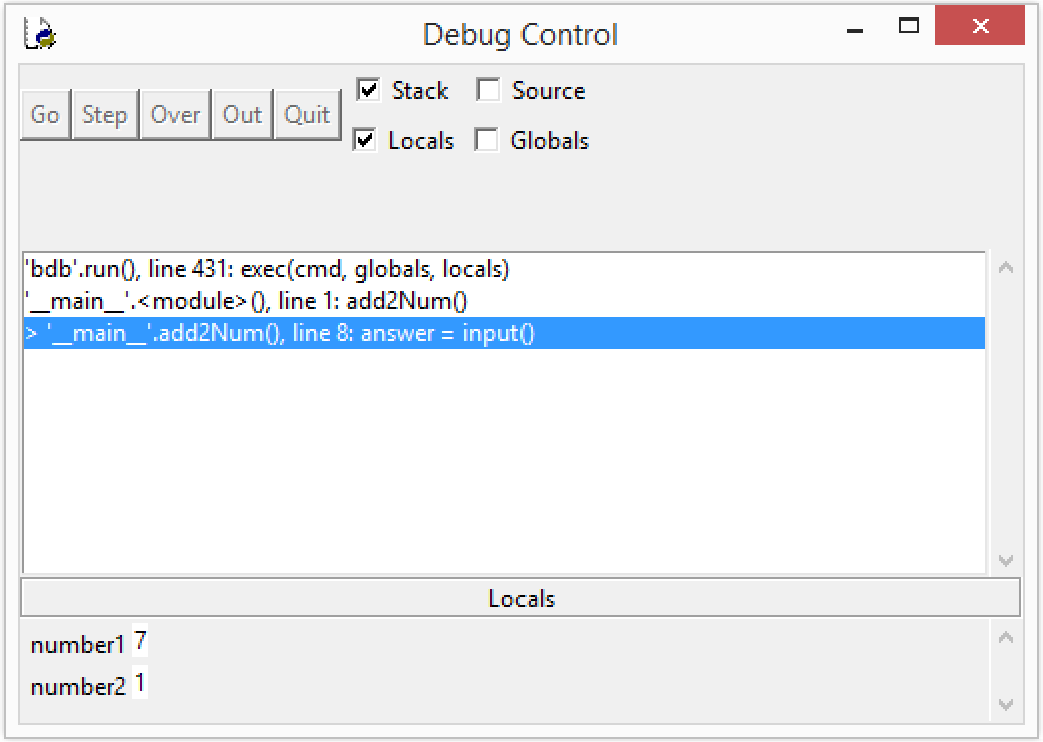
number1 7



W02 buggyAddNum.py:7: add2Num()

```
'bdb'.run(), line 431: exec(cmd, globals, locals)
'__main__'.<module>(), line 1: add2Num()
> '__main__'.add2Num(), line 7: print('What is ' + str(number1) + ' + ' + str(number2) + '?')
```

number1 7
number2 1

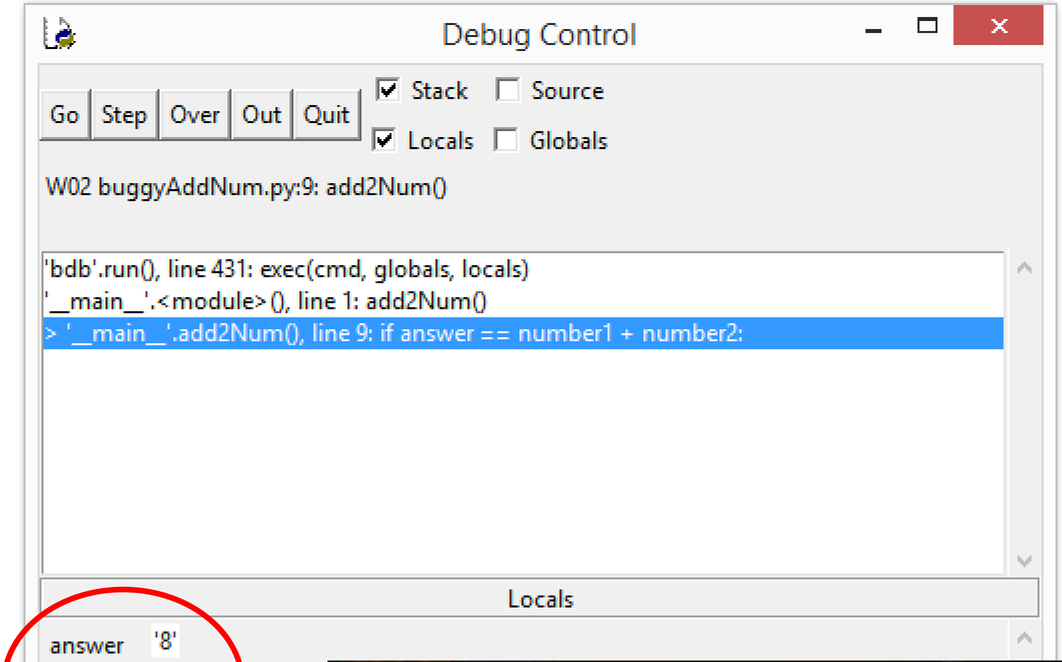


```
'bdb'.run(), line 431: exec(cmd, globals, locals)
'__main__'.<module>(), line 1: add2Num()
> '__main__'.add2Num(), line 8: answer = input()
```

number1 7
number2 1

```
>>> add2Num()
What is 7 + 1?
8
```

Using the IDLE Debugger



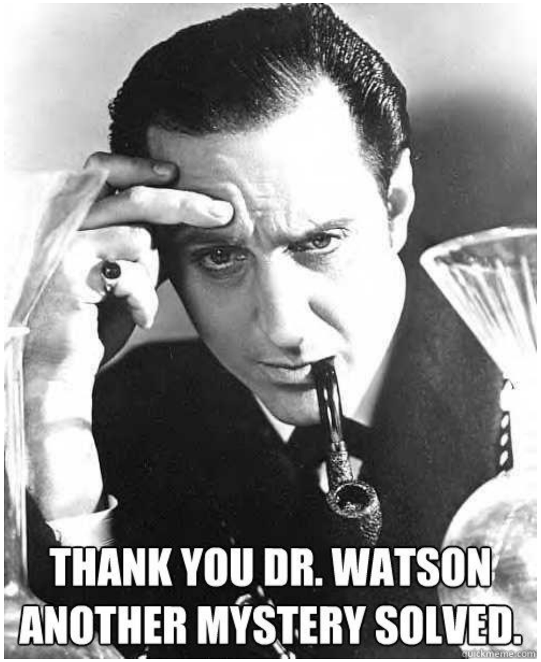
☑ Stack □ Source

☑ Locals □ Globals

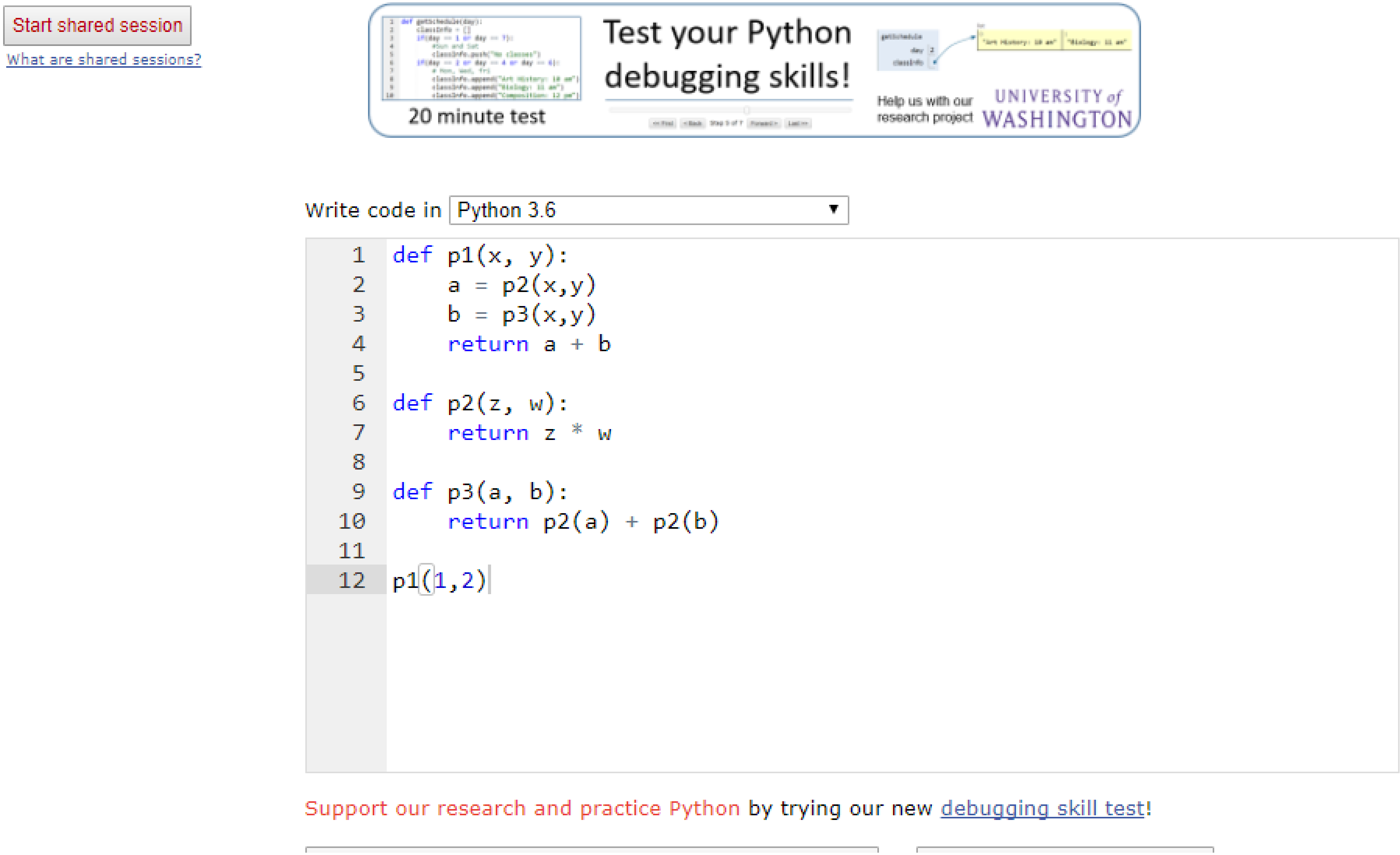
```
W02 buggyAddNum.py:9: add2Num()
```

```
'bdb'.run(), line 431: exec(cmd, globals, locals)
'__main__'.<module> (), line 1: add2Num()
> '__main__'.add2Num(), line 9: if answer == number1 + number2:
```

answer	'8'
number1	7
number2	1



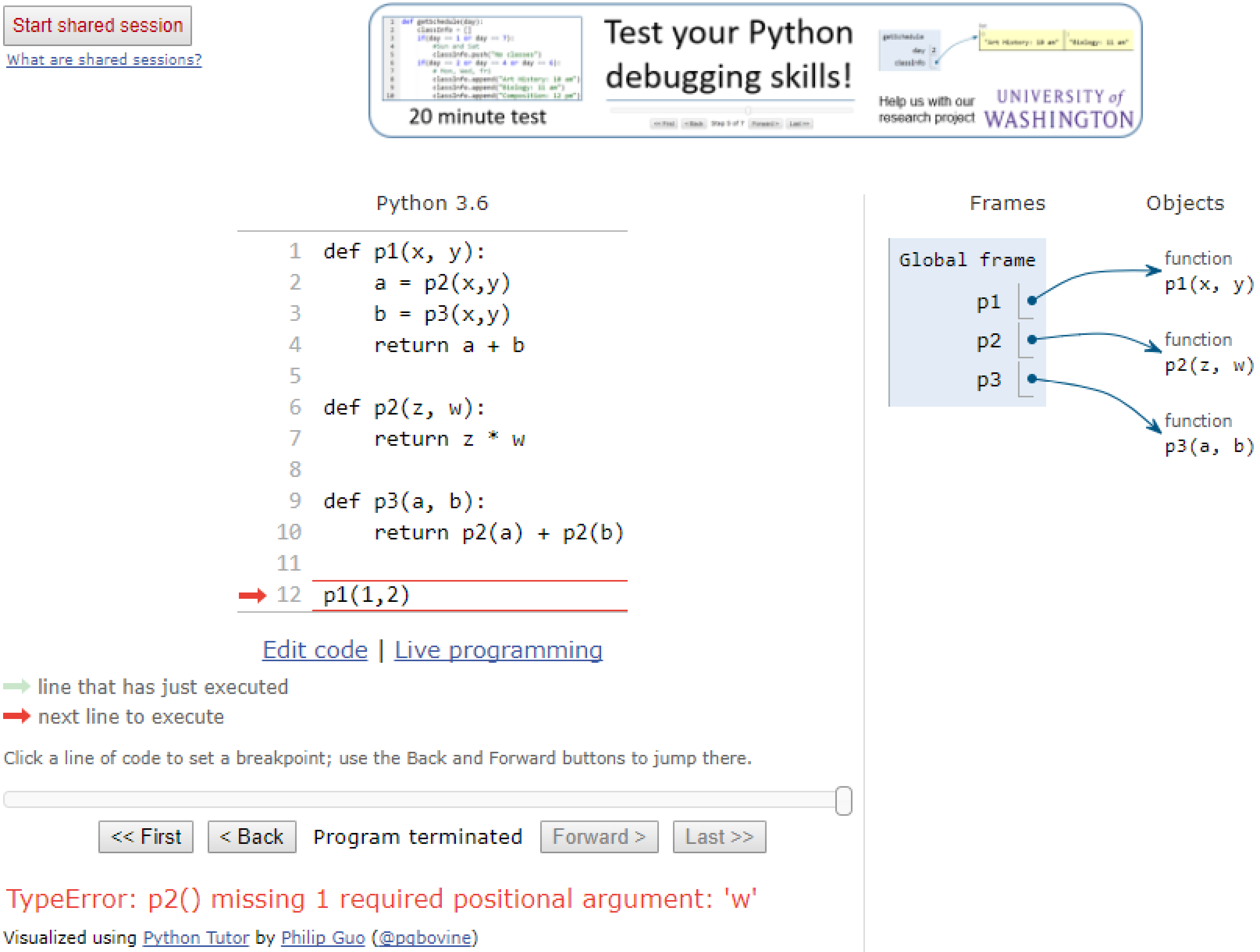
Another Debugger: pythontutor.com


Start shared session
What are shared sessions?
20 minute test
Test your Python debugging skills!
Help us with our research project
UNIVERSITY of WASHINGTON

Write code in Python 3.6

```python
def p1(x, y):
    a = p2(x,y)
    b = p3(x,y)
    return a + b

def p2(z, w):
    return z * w

def p3(a, b):
    return p2(a) + p2(b)

p1(1,2)
```

Support our research and practice Python by trying our new debugging skill test!



```
def p1(x, y):
    a = p2(x,y)
    b = p3(x,y)
    return a + b

def p2(z, w):
    return z * w

def p3(a, b):
    return p2(a) + p2(b)

p1(1,2)
```

Click a line of code to set a breakpoint; use the Back and Forward buttons to jump there.

Program terminated

```
TypeError: p2() missing 1 required positional argument: 'w'
```

Visualized using Python Tutor by Philip Guo (@pgbovine)

Debugging is an Art