Consider the following code:

```
1  def main():
2      a = 10
3      b = 55
4      print "in main........dir() = ", dir()
5      result = absval(a,b)
6      print "The absolute value of %d-%d is %d" % (a,b, result)
7
8   def absval(x,y):
9      print "in function...dir() = ", dir()
10     if x > y
11       z = x - y
12     else:
13       z = y - x
14     #
15     #Draw stack frame here
16     #
17     return z
18
19  main()
```

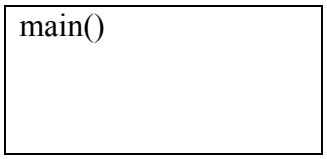
`dir()` is a Python function that lists all of the variables currently **in** scope.
The print statement uses print formatting, which we will cover in class.

---

For tracing through a program using a stack diagram, follow these steps:

1. Draw strack frame for the called function
    a. Allocate parameters in this frame
    b. Store local variables in this frame
2. Assign **parameters** the value of **arguments** used in calling the function
3. Move to the function definition, and execute function step-by-step
4. Send return value back to calling function
5. Remove function from the stack
6. Continue executing program/function remaining at top of stack

For the above example, we begin at line 1. Our stack frame:

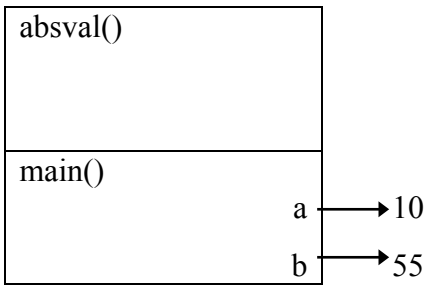



After lines 2 and 3:

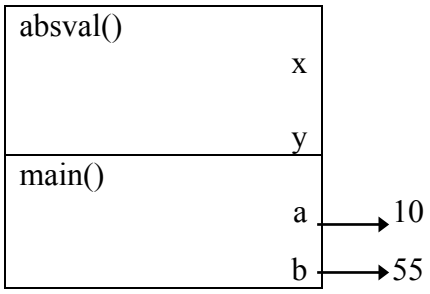

Line 4 prints to the screen

```
in main.......dir() =  ['a', 'b']
```

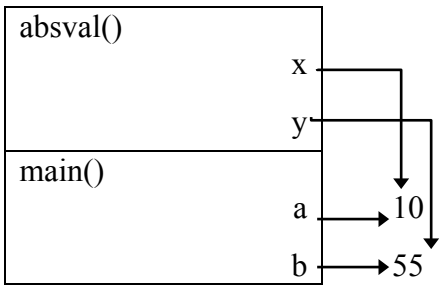
Line 5 calls absval(). We take a look at our procedure and allocate space on the stack first:




Then allocate parameters:




Assign the parameters the value of the arguments from the call in main() – the values `a` and `b` respectively:

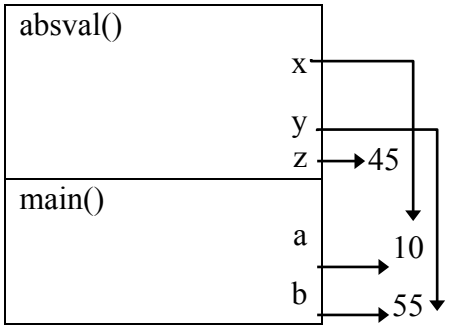



Now we start going through the function line-by-line. At line 9 we print:

```
in function...dir() =  ['x', 'y']
```

line 10 is false since `x` (i.e., 10) is less than `y` (55)

we then jump to the else and execute line 13 yielding…..

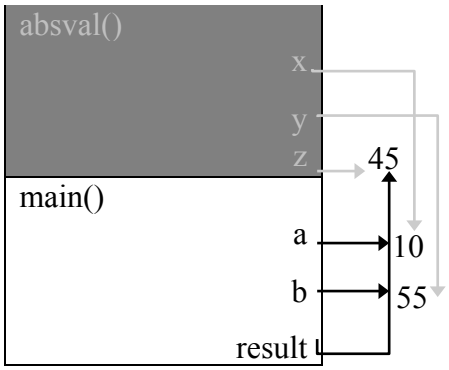



## THIS IS THE SOLUTION!

Note, z is a local variable and should be in the stack
Also, we have not yet hit `return` therefore you can have `result` on the stack, but it **should not** have a value yet.

If you are curious, to finish up…

The next line is 17 where we return `z`. We allocate `result` on the stack, assign it the return value, and then remove `absval` from the stack (shown here as grayed out).




This also returns execution to line 5, which is now done. We move on to line 6 which prints to the screen:

```
The absolute value of 10-55 is 45
```

main() is now complete, so technically we return from this function, erase main() from the stack and are left with nothing, so python exits.

So, in total, the output was:

```
in main.......dir() =  ['a', 'b']
in function...dir() =  ['x', 'y']
The absolute value of 10-55 is 45
```