

# Named Parameters

Thomas Schwarz, SJ

# Named Parameters

- Function parameters can be passed (given) by position

- Example:

- `func` sets `a=s` and `b=r`

- Returns `(s+r)**s`

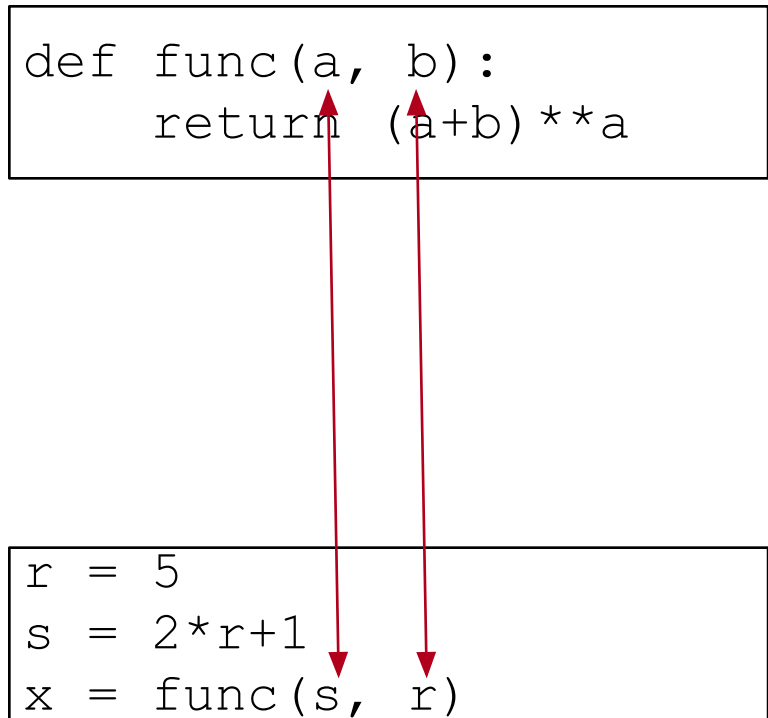
- So, `x` becomes

- $(s + r)^s = (11 + 5)^{11}$

- $= 16^{11}$

- $= 17592186044416$

```
def func(a, b):  
    return (a+b)**a
```

A diagram illustrating the flow of data in a function call. It consists of two rectangular boxes. The top box contains the function definition: `def func(a, b):` followed by `return (a+b)**a`. The bottom box contains the function call: `r = 5`, `s = 2*r+1`, and `x = func(s, r)`. Two red arrows originate from the arguments `s` and `r` in the function call. The arrow from `s` points upwards to the parameter `a` in the function definition. The arrow from `r` points upwards to the parameter `b` in the function definition.

```
r = 5  
s = 2*r+1  
x = func(s, r)
```

# Named Parameters

- This is difficult with functions with very long parameter lists
  - Or if the function is defined in another module and you need to look it up
- Two solutions that should both be used:
  - Use good documentation
  - Make the use of the function more intuitive

# Doc Strings

- To create a doc string, put an explanation after the function definition in triple quotes.

```
def sum_of_div(n):  
    """ A function that calculates the sum of divisors of n  
        by trying out all numbers smaller than n/2 and adds up  
        those that do."""
```

- You can access the doc string by using the help function of the editor
- Or by typing help

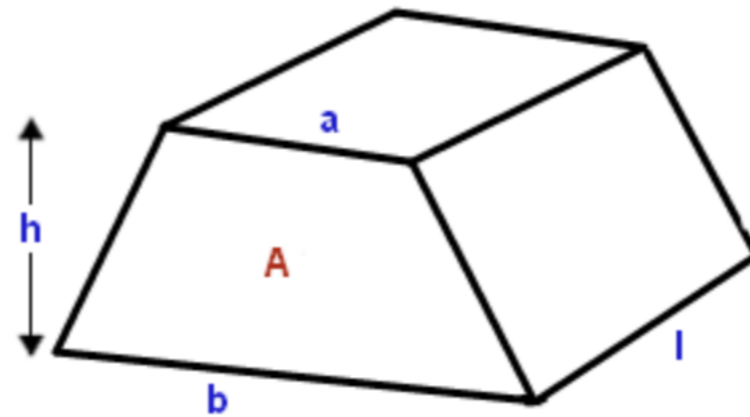
```
print(sum_of_div(12285), sum_of_div(|
```

```
(n)  
A function that calculates the sum of divisors of n  
by trying out all numbers smaller than n/2 and adds up  
those that do.
```

```
>>> help(sum_of_div)  
Help on function sum_of_div in module __main__:  
  
sum_of_div(n)  
    A function that calculates the sum of divisors of n  
    by trying out all numbers smaller than n/2 and adds up  
    those that do.
```

# Named Parameters

- Use explanatory variable names
  - Example: a trapezoidal prism
    - Given by height  $h$ , base lengths  $b$  and  $a$  and a length  $l$



- The formula for the volume is  $\frac{1}{2}(a + b) \cdot h \cdot l$

# Named Parameters

- We define the function

```
def trap_vol(baselength1, baselength2, length, height):  
    """ return the volume of the trapezoid with  
        baselengths, length, and height.  
    """  
    return 0.5*(baselength1+baselength2)*length*height
```

# Named Parameters

- When we call it, we do not use position to indicate which value is what, but we *name* the arguments
  - Now we can call in any order
  - ```
trap_vol (height=3,  
          length=10,  
          baselength1=14,  
          baselength2=8)
```
- And the call itself is self-documenting