Lecture 2: Functions

CS 50

January 22, 2026

Announcements

- We have a slack channel! #cs50-spring2026
- Office Hours + Mentor sessions now on the course website
- A1 due on Monday

Review: Expressions

- Values
  - 47
  - "hello, world!\n"
- Variables
  - a_string
  - distance_in_miles
- Types
  - int
  - float
  - str
- Operations on values or variables
  - 1 * 2 * 3
  - "hello" + "world
  - x % 2
- Function calls
  - int("32")
  - print("hello, world")
  - str.isdigit("12345678")

Functions

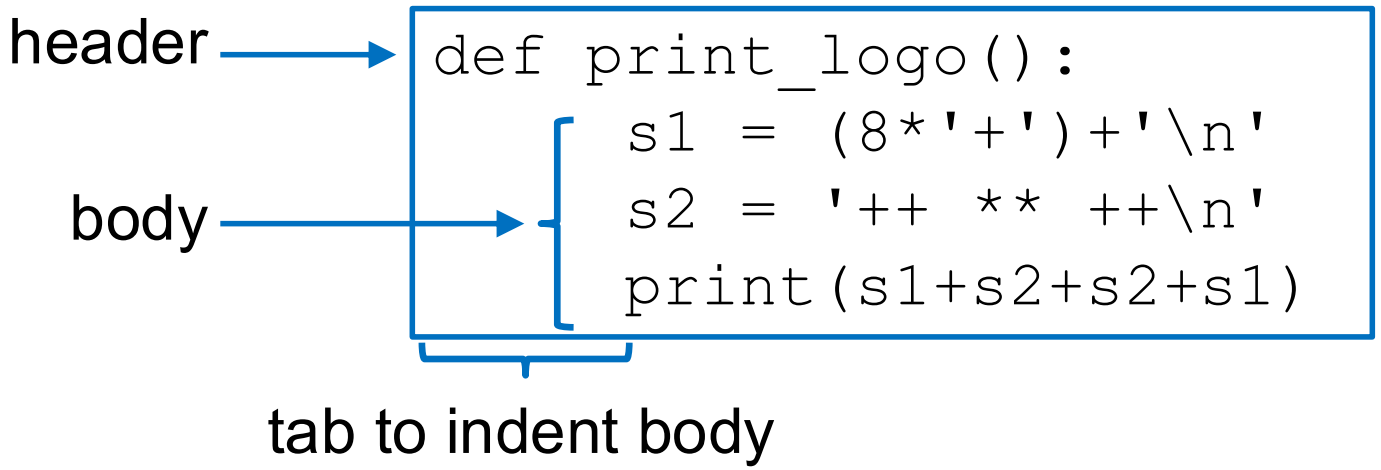
- A function is a named sequence of instructions that performs some useful operation
- When you call a function, the sequence of instructions executes.
- A function call is an expression (it evaluates to a value)
- When should you define a function?
- How can you define your own functions?
- How do you use (call) your own functions?

Defining Functions

- Why?
  - There's some useful operation that you want to do over and over and over
  - Easier to read/understand
  - Easier to modify/change/debug

```
++++++++
++  **  ++
++  **  ++
++++++++
```

- How?



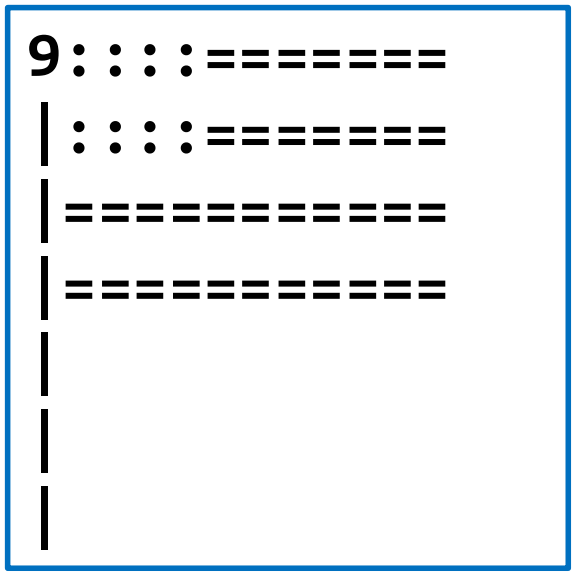
Calling Functions

```
def print_logo():
      s1 = (8*'+')+'\n'
      s2 = '++ ** ++\n'
      print(s1+s2+s2+s1)

print("Here's my company logo:")
print_logo()
print("I can easily print it as many times"
       + "as I need to")
print_logo()
```

Exercise 1: Defining a Function

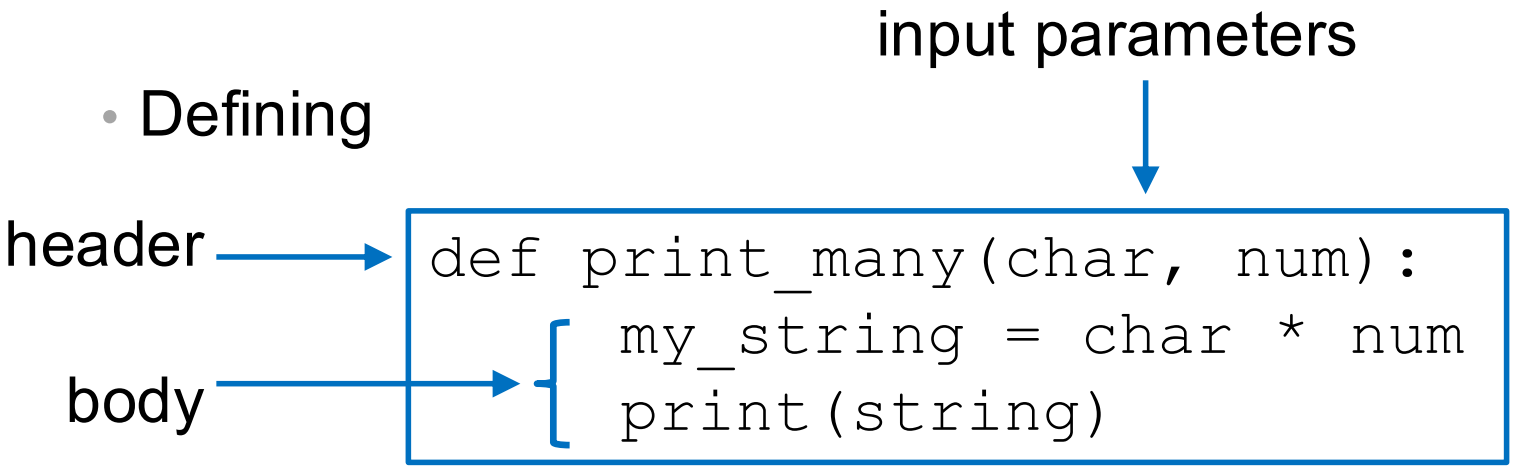
- Define a function `print_flag()` that prints the following image:



- Write a program that prints three flags

Parameterized Functions

- Defining

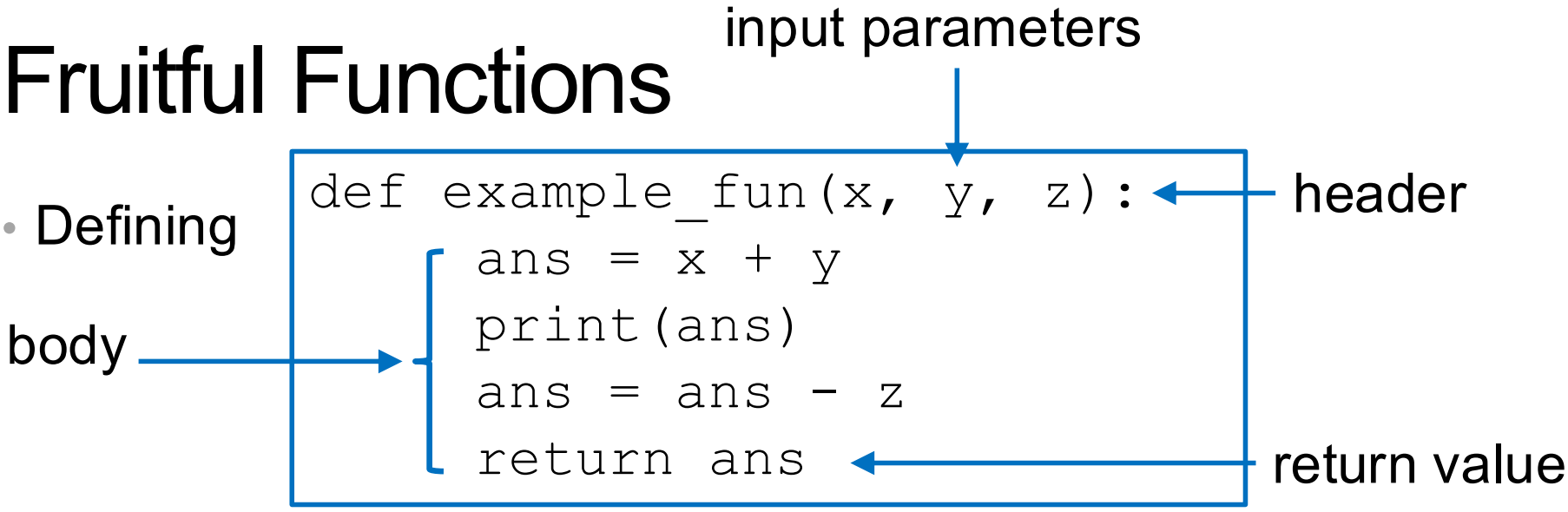


- Calling

char = '+' **num = 8**

```
print_many('+', 8)
s = '++ ** ++\n'
print(s)
print_many('*', 8)
print_many('@', 2)
```

Fruitful Functions



- Defining

- Calling

```
num = 8
total = example_fun(num, 2, 6)
print(total)
print_many('*', total-2)
print_many('@', example_fun(1,5,2))
```

Exercise 2: Code Reading

- What gets printed when you run the following program?

```
def function1(a, b):
     print(a+b)
     return a*b

def function2(x):
     print(x+2)
     y = function1(x, x-2)
     return y+1

num = function1(3, 6)
num2 = function2(num) + 1
print(num2 + 10)
```