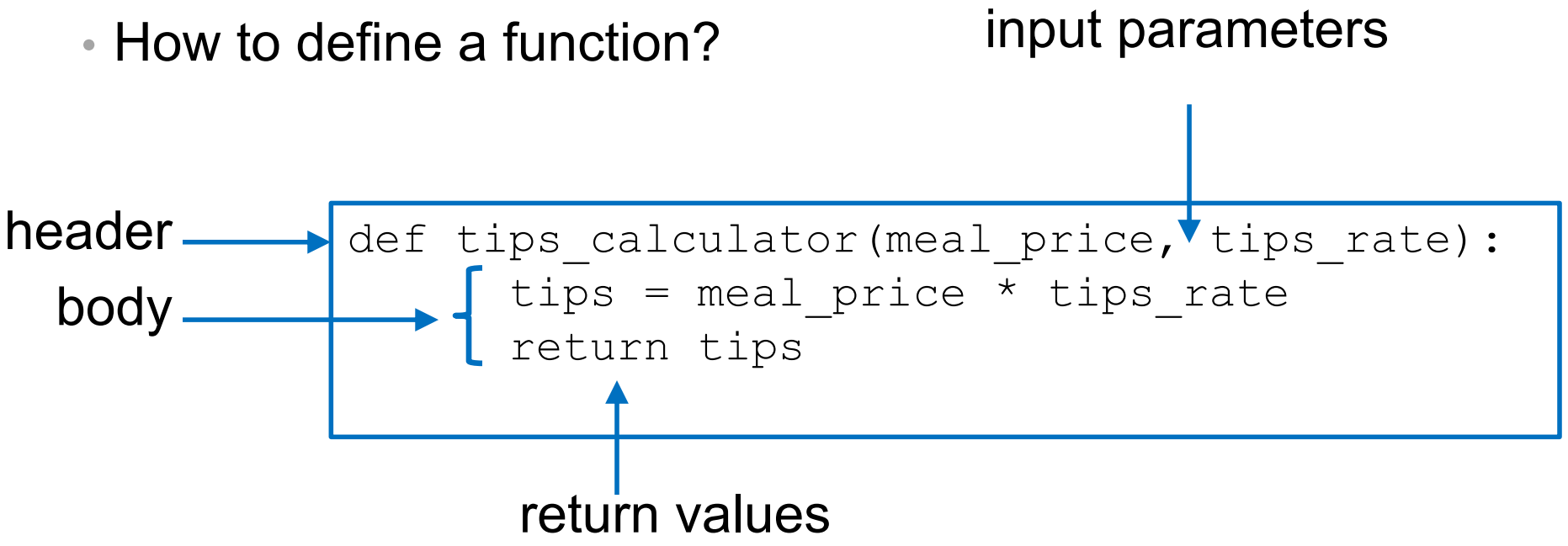
# Lecture 7: More Functions

CS 51P

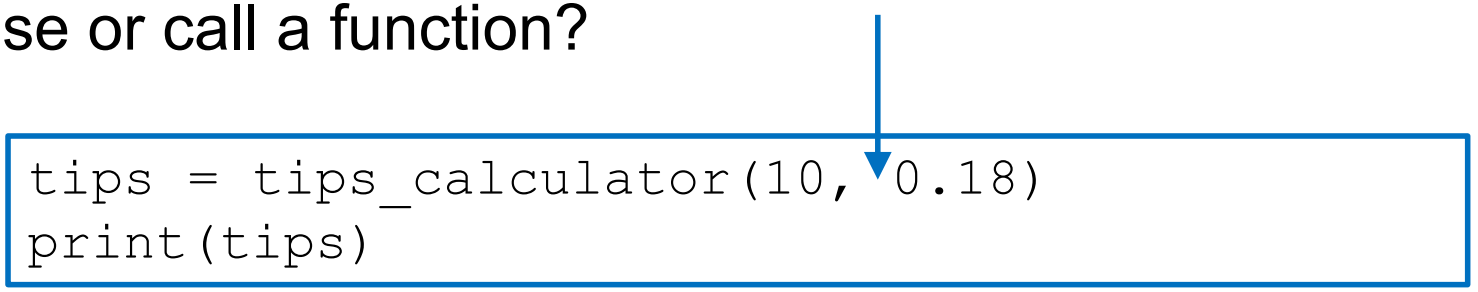
September 25, 2023

# Review: Functions

- How to define a function?

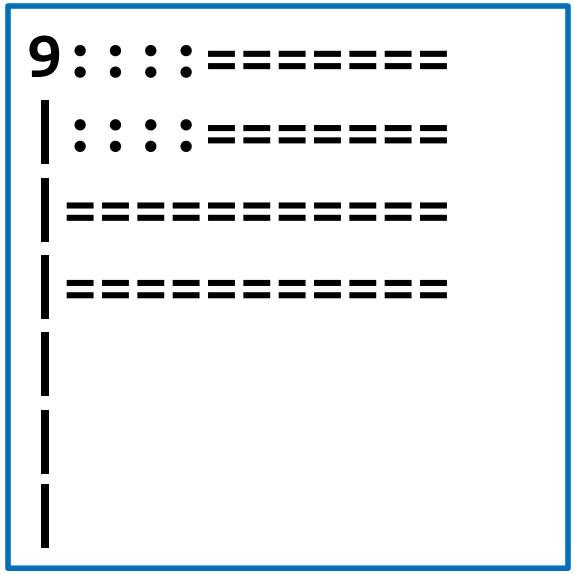



- How to use or call a function?



# Revisit input parameters

- Input parameters can be int, float, Boolean or str, etc.
- Sometimes, there could be no input parameters
- Define a function `print_flag()` that prints the following image:



# Example

- Define a function get_pos_int() that repeatedly asks the user for an input until the user enters a positive integer and then returns that number as an int.

- Write a program that gets a positive integer from the user (using get_pos_int()) and then prints that number of flags (using print_flag())

# Exercise

- Define a function called `exp` that takes a number n (an `int` or `float`) and a number p (an int or float) as parameters and returns the value $n^p$. Please use a for loop to solve it.

# More about functions as helpers

- Function is like a helper or an assistant

- Control flow
  - A function can be called in another function
  - The return value may not be a final product, yet it could be an intermediate result that can be used in other places
  - return will return the value back to where you call the function
  - Local variable, scope

# Example

- Define a function called `sum_powers` that takes a number n (an `int or float`) and a power p (an `int or float`). If n is a positive `int`, it returns the sum of the powers $1^p + 2^p$ ... $+ n^p$. Otherwise it returns 0.

# Exercise

- Define a function called contain_vowels, which takes in a word (a str) as input, and returns True if the word contains vowels; False otherwise.

- Define a function called print_valid_word, which takes in a word (a str) as input. If the given word contains vowels, this function will print out the given word, otherwise this function will print out “bad word”.

# Docstring

```
def contain_vowels(word):
    """
    check if the given word contains vowels
    :param word: (str) a word
    :return: (boolean) True if the given word contains
    vowels; False, otherwise
    """
```

- Function description
- Parameter description, with data type
- Return value description, with data type

# Main functions

- By convention, the only code that goes in the body of a Python file is the two-line program

```
if __name__ == "__main__":
    main()
```

- The rest of the program is defined in a function called main()
- (or in other functions!)

```
def exp(n, p)…
    …

def sum_power(n, p)…
    …

def main():
    n = int(input(“Enter a base”))
    p = int(input(“Enter a power”))
    print(“The sum power of ” + str(n)
    + “ and ” + str(p) + “ is ” +
    str(sum_power(n, p)))


if __name__ == "__main__":
    main()
```

# return v.s. print

- Whether to return a value or print out the value depends on the specification of the requirement

- When design a function by yourself, how to choose return or print?
  - If the result is only an intermediate result that needs to be further processed, then use return rather than print

  - If the result is a final result, then you can use either return or print