Join App Team Carolina!

No experience?

Join Bootcamp!

- UI/UX Design
- iOS Development

Got experience?

- iOS Developer
- Backend Developer
- UI/UX Designer
- Product Manager
- Product Marketing Manager

Applications due FRIDAY! (1/16)
Link to applications can be found on our Instagram **@appteamcarolina or** on our website **appteamcarolina.com**

Functions and Memory Diagrams

Please be ready to write on a piece of paper or tablet!

Announcements

- Reminder: MLK Jr. Day on Monday; no class!
  - Office Hours and Tutoring canceled on Monday
- EX01: Tea Party released today, due Wednesday (Jan 21st)
- Quiz 00 next Friday
  - Ways to prepare:
    - Quiz expectations on course site
    - LS questions
    - Practice problems in the slides
    - EX00 and EX01
    - Additional practice problems will be added to the site tomorrow
    - **Hybrid review session:** 6-7pm on Wednesday (Jan 21st) in Sitterson 014 and online (link will be on the site)
      - Can't make it? Recording will be added to the site!

Warm-up: write down at least one line number for each:

line numbers

```
1  """A simple program with a function call."""
2
3
4  def perimeter(length: float, width: float) -> float:
5      """Calculates the perimeter of a rectangle."""
6      return 2 * length + 2 * width
7
8
9  print(perimeter(length=10.0, width=8.0))
```

1. Docstring Lines 1, 5
2. Function call(s) line 9
3. Return statement line 6
4. Function definition lines 4-6
5. Arguments line 9
6. Use of a parameter's name in an *expression* line 6

The `return` statement vs. calls to `print`

- **The return statement is *for your computer*** to send a result back to the function call's "bookmark" *within your program*
  - A bookmark is dropped when you *call* a function with a return type. When that function's body reaches a *return statement*, the returned value replaces the function call and the program continues on
- **Printing is *for humans* to see.** To share some data with the user of the program, you must output it in some way
- If you have a function, `my_func`, that returns some value, you can print the value it returns by:
  1. Printing its return value directly with `print(my_func())` or
  2. (Later in the course) by storing the returned value in a variable and *later* printing the variable

Tracing programs by hand: Intro to memory diagrams!

- The evaluation of a program depends on many interrelated values
- As any non-trivial program is evaluated, what needs to be kept track of includes:
  1. The current line of code, or expression within a line, being evaluated
  2. The trail of function calls that led to the current line and “frame of execution”
  3. The names of parameters/variables and a map of the values they are bound to
  4. More!
- As humans, this quickly becomes more information than we can mentally keep track of.
  - Good news: Memory diagrams will help you keep track of it all on paper!

Memory diagrams

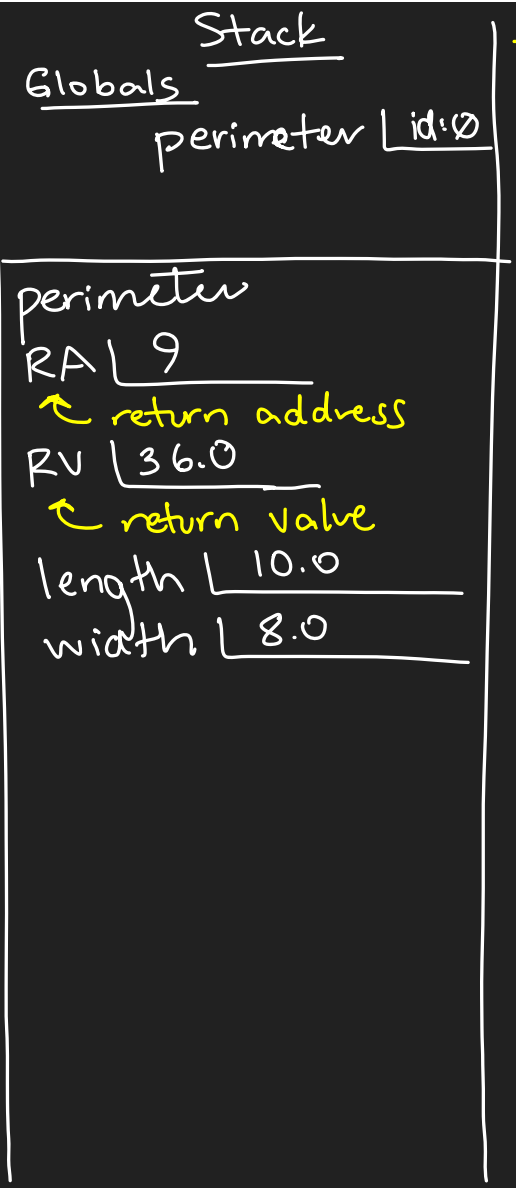
- A program's runtime *environment* is the mapping of *names* in your program to their *locations* in memory
- A program's *state* is made up of the *values stored* in those locations
- You can use memory diagrams to visually keep track of both the environment and its state
- Memory diagrams will help you keep track of how function calls are processed.
  - Where was the function called?
  - What was the return value, and where was it returned to?
  - (and more!)

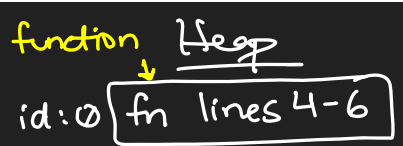
```
1  """A simple program with a function call."""
2
3
4  def perimeter(length: float, width: float) -> float:
5      """Calculates the perimeter of a rectangle."""
6      return 2 * length + 2 * width
7
8
9  print(perimeter(length=10.0, width=8.0))
```

10.0 → length; 8.0 → width; 2 * length = 20.0; 2 * width = 16.0; 20.0 + 16.0 = 36.0

Output

36.0





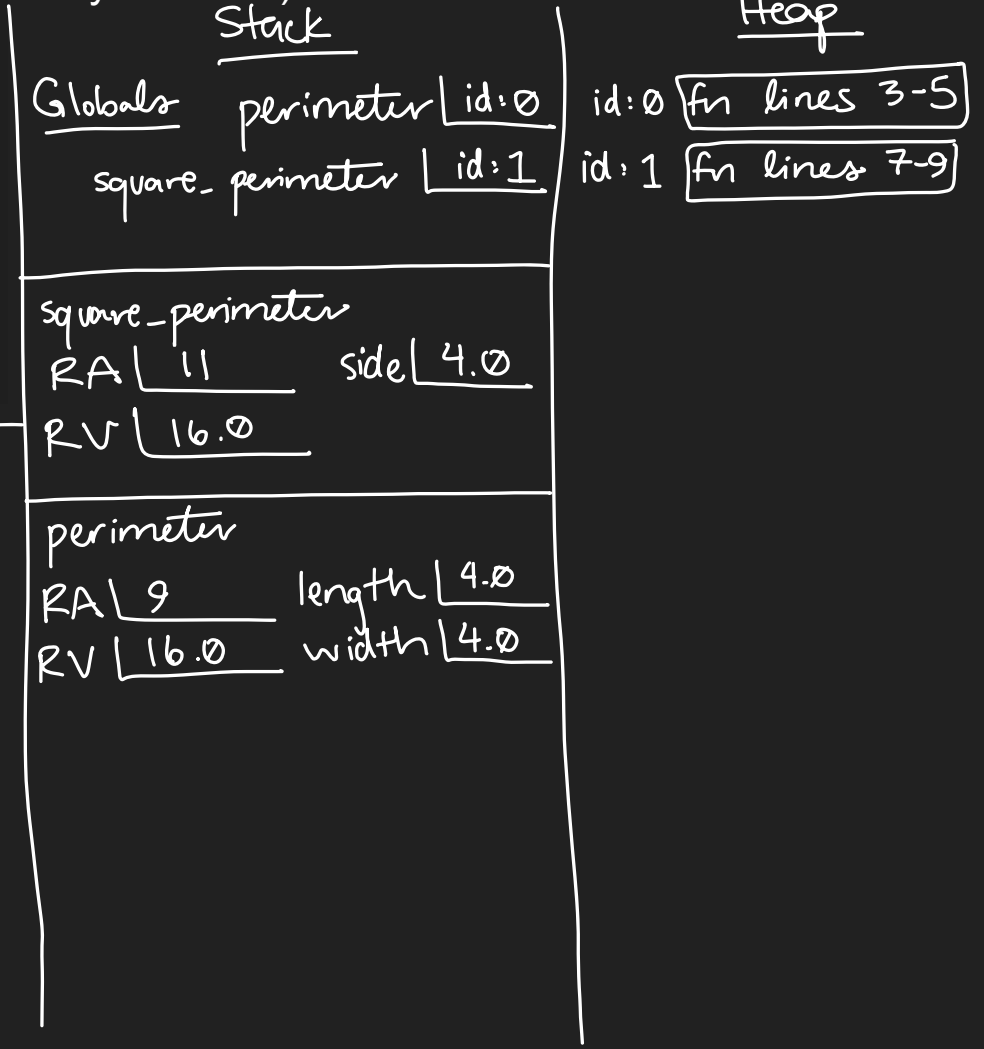
```
1  """A program with *two* function calls."""
2
3  def perimeter(length: float, width: float) -> float:
4      """Calculates the perimeter of a rectangle."""
5      return 2 * length + 2 * width
6
7  def square_perimeter(side: float) -> float:
8      """Calculates the perimeter of a square."""
9      return perimeter(length=side, width=side)
10
11 print(square_perimeter(side=4.0))
```

Output

16.0

Note: this was printed as a result of executing line 11, NOT line 9!

(a second example, in case you want to try this on your own!)



Weekly Tutoring + Office Hours

Office Hours (Sitterson Hall (SN) 008):

- Mondays–Fridays: 11am-5pm
- Sundays: 1-5pm

Tutoring (Fred Brooks (FB) 007):

- Mondays, Wednesdays, Thursdays: 5-7pm

(Reminder: No Office Hours/Tutoring on Monday, Jan 19th!)