

COMP
110

CL05: Memory Diagrams

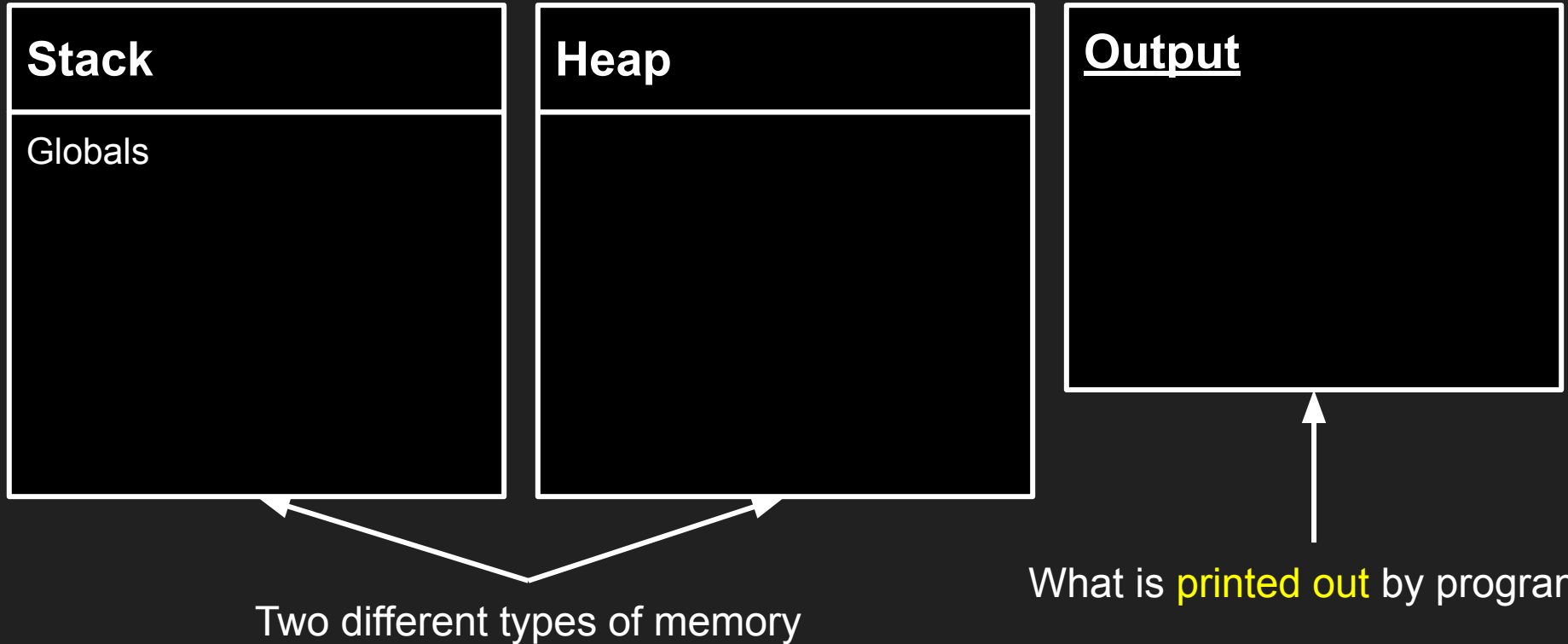
NOTE: No Office Hours Sun.–Tues. (Sep 1-3)

Enjoy the long weekend!

Motivation

- Memory diagrams allow us to **trace code in memory**
- Helps us to understand **what** our code is doing and **why**

Memory Diagram Components



Stack vs. Heap

- Stack: variables, primitive types
- Heap: definitions, certain mutable types (more on this later)

Function Call Steps

- Prepare for call:
 - Has function been defined?
 - Are arguments fully evaluated?
 - Do parameters and arguments agree?
- Establish frame for function call:
 - Frame on stack labeled with function name
 - Return address
 - Copy over arguments

```
1 def sum(num1: int, num2: int) -> int:
2     |     """Add two numbers together."""
3     |     return num1 + num2
4
5
6 print(sum(num1=4, num2=5))
```

Stack

Globals

Heap

Output

```
1 def get_tax(price: int, tax_rate: float) -> float:
2 |     return price * tax_rate
3
4 def total_cost(cost: int, tax: float) -> float:
5 |     return cost + get_tax(price=cost, tax_rate=tax)
6
7 print(total_cost(cost=100, tax=0.07))
```

Stack

Globals

Heap

Output