SCOPE

An Introduction to Computer Science

Scope

- "Lifetime"
- "Visibility"
- "Availability"

"How long the variable is available"

Global Scope

grade is now available

```
print("Starting program")
grade = 64
grade = grade + 5
print("Grade:", grade)
```

Local Scope

```
def calculate_grade(grade:int, weight:float)->float:
    curved = 100 * grade ** .5
    final = curved * weight
    return final

calculate_grade(90, .1)
```

The local variables are `grade`, `weight`, `curved`, and `final`

Returning Values

Functions return values, not variables!

```
def get_grade(points:int, possible:int)->float:
    grade = points / possible
    return grade


my_grade = get_grade(70, 100)
print(my_grade)
```

The local variables `grade`, `points`, and `possible` all die after the return statement.

Same Named Variables

```
def add1(number:int)->int:
    total = number + 1
    return total

total = 3
total = add1(total)
answer = 5
answer = add1(answer)
```

The local variables of `add1` are `number` and `total`

The global variables are `total` and `answer`

The local `total` and `global` total are different variables

Global Variables Are Bad

```python
from cisc108 import assert_equal

my_title = "Lord "
def add_title(name: str) -> str:
    titled_name = my_title + name
    return titled_name

assert_equal(add_title("Bart"), "Lord Bart")
my_title = "Dr. "
assert_equal(add_title("Bart"), "Dr. Bart")
```

Complicated!

Scope Rule of Thumb

- Variables INSIDE a local scope should not be used OUTSIDE that scope

- Variables OUTSIDE a local scope should not be used INSIDE that scope