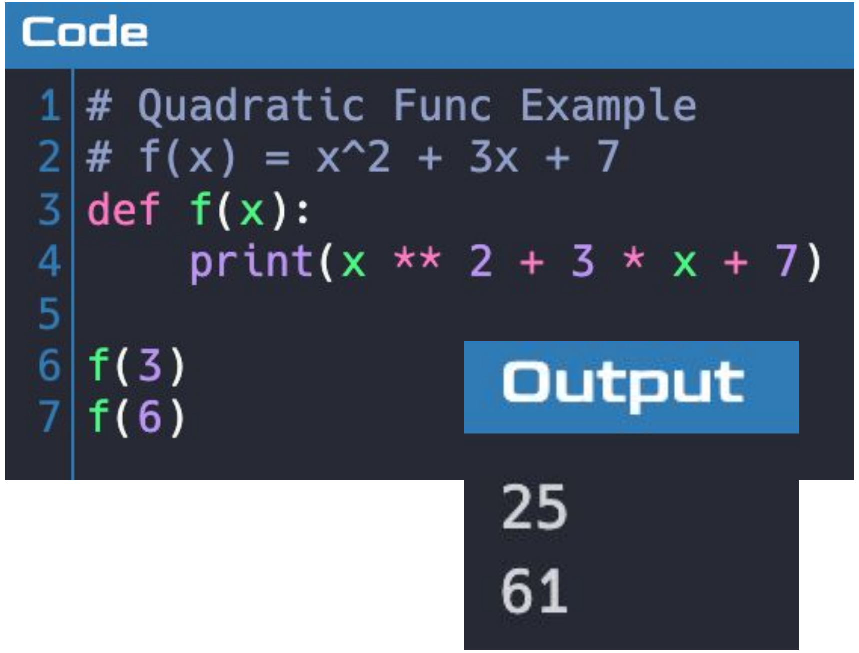
Programming Club Meeting
6 Slides

Prime Number Formula

Functions

Basics

- Works just like a function in math
- Gives lines of code a name to easily call with different inputs
- Vegas Rule
- Parts:
    - Define
    - Name (with parentheses)
    - Parameters / arguments in parentheses
    - Call (must be after the definition)



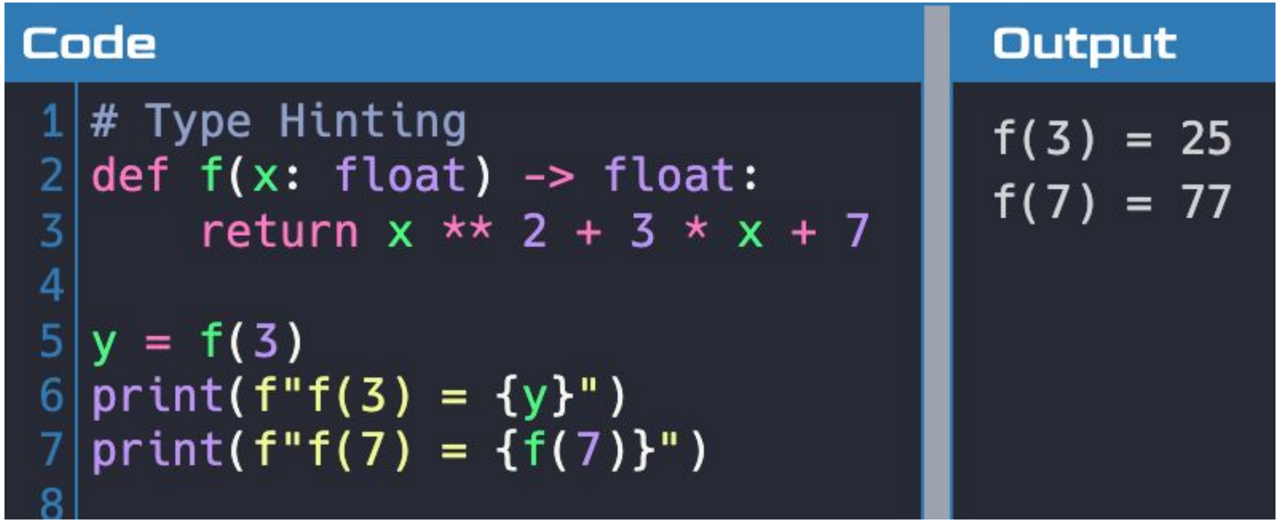
```python
# Quadratic Func Example
# f(x) = x^2 + 3x + 7
def f(x):
    print(x ** 2 + 3 * x + 7)

f(3)
f(6)
```

```
25
61
```

Return and Type Hinting

- Function can return value instead of printing it
- Can tell Python data types of everything



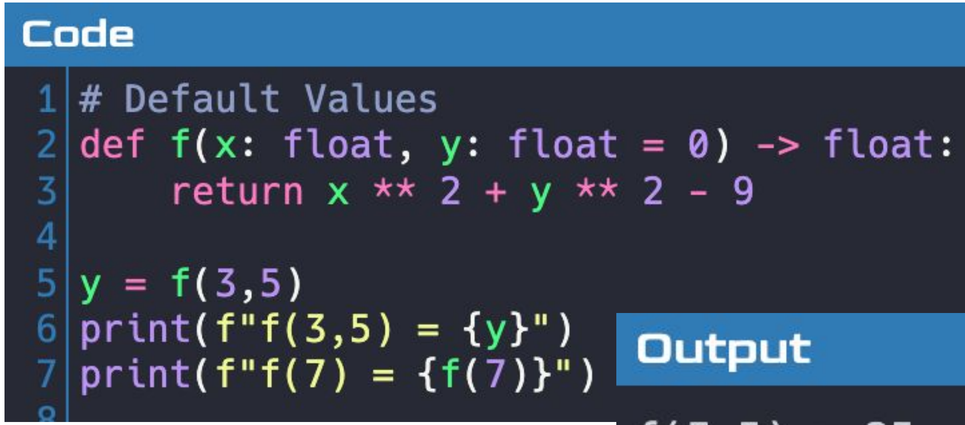
```python
# Type Hinting
def f(x: float) -> float:
    return x ** 2 + 3 * x + 7

y = f(3)
print(f"f(3) = {y}")
print(f"f(7) = {f(7)}")
```

```
f(3) = 25
f(7) = 77
```

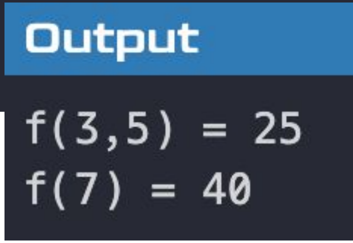
More Parameters and Default Values

- Add more parameters with a comma followed by the new parameter
- Do the same thing for arguments in the call
- To give a parameter a default value, add '= val'

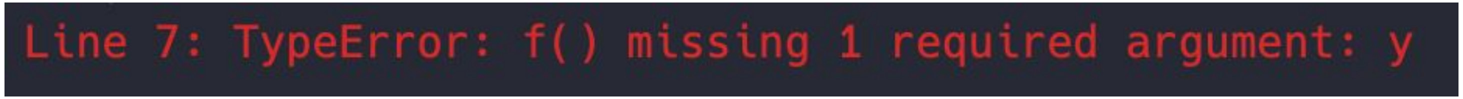

Code

```python
# Default Values
def f(x: float, y: float = 0) -> float:
    return x ** 2 + y ** 2 - 9

y = f(3,5)
print(f"f(3,5) = {y}")
print(f"f(7) = {f(7)}")
```


Output

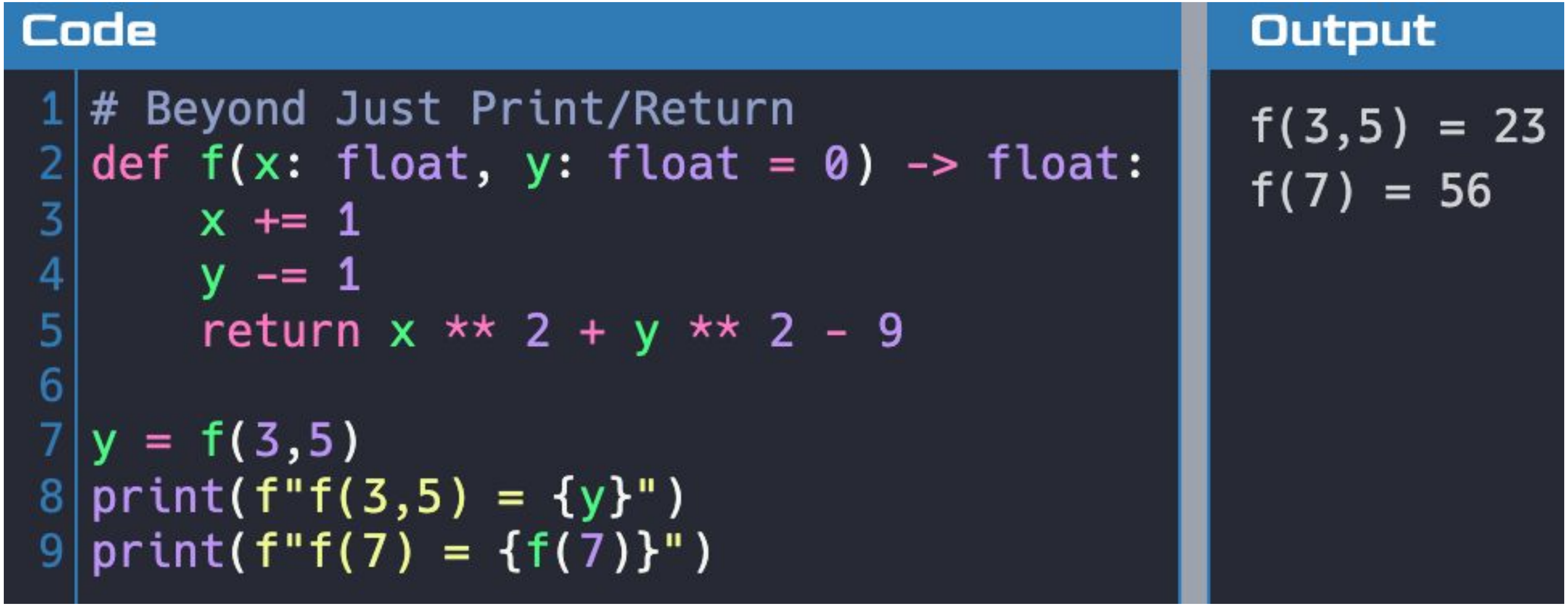
```
f(3,5) = 25
f(7) = 40
```



```
Line 7: TypeError: f() missing 1 required argument: y
```

Error caused by line 7 w/o default

More Than 1 Line

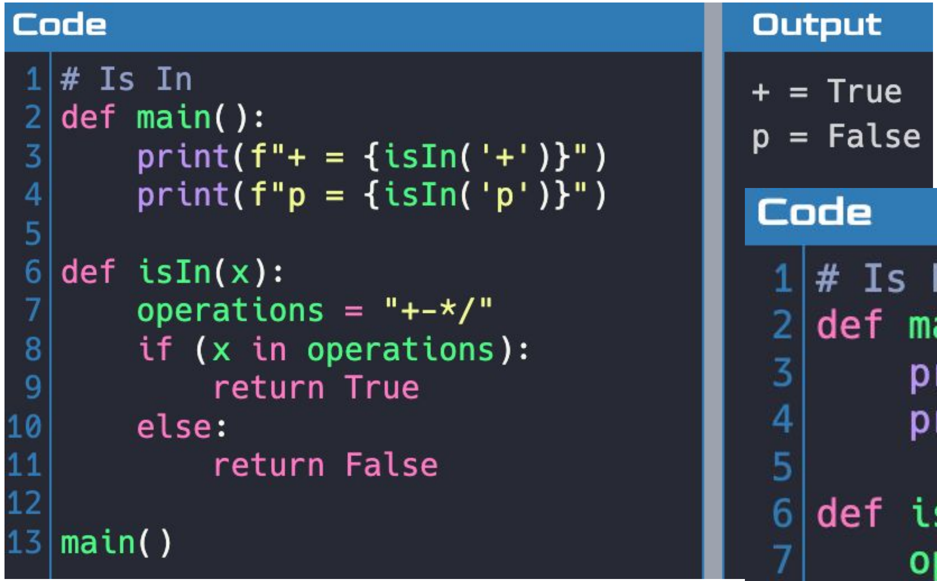


```python
# Beyond Just Print/Return
def f(x: float, y: float = 0) -> float:
    x += 1
    y -= 1
    return x ** 2 + y ** 2 - 9

y = f(3,5)
print(f"f(3,5) = {y}")
print(f"f(7) = {f(7)}")
```

```
f(3,5) = 23
f(7) = 56
```

Is In and Is Not In



Code

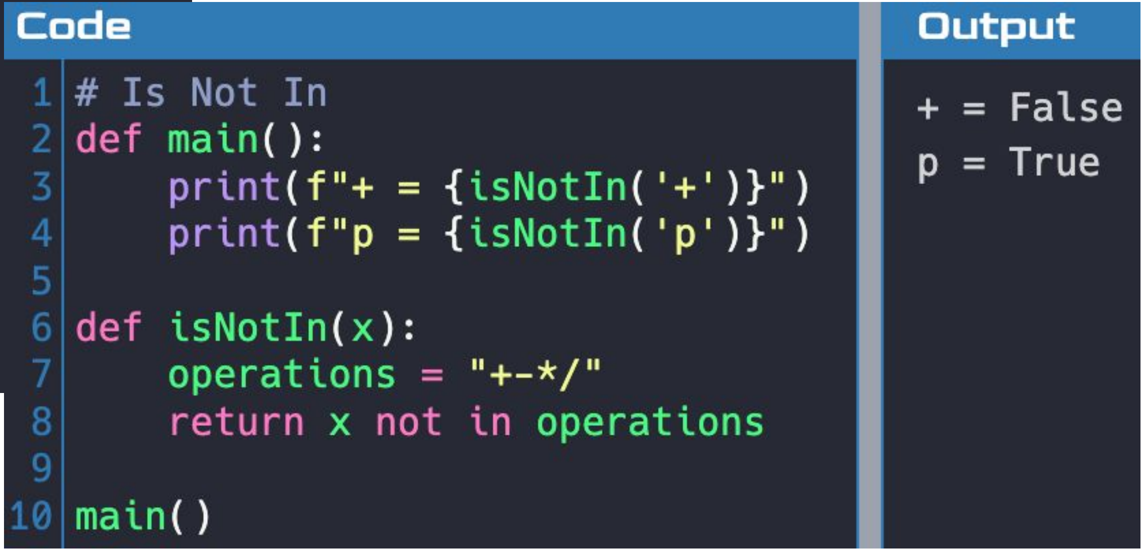
```python
# Is In
def main():
    print(f"+ = {isIn('+')}")
    print(f"p = {isIn('p')}")

def isIn(x):
    operations = "+-*/"
    if (x in operations):
        return True
    else:
        return False

main()
```

Output

```
+ = True
p = False
```



Code

```python
# Is Not In
def main():
    print(f"+ = {isNotIn('+')}")
    print(f"p = {isNotIn('p')}")

def isNotIn(x):
    operations = "+-*/"
    return x not in operations

main()
```

Output

```
+ = False
p = True
```

Practice Problems

Practice Problem 1: Quadratic Formula

- Goal: Write a Python function that will find the roots of a quadratic function by the quadratic formula.
- Also write a program that will test this functionality.
- Relevant Information:
    - Quadratic Formula: (-b ± √(b^2 - 4*a*c) / 2a
    - Quadratic Eq: f(x) = ax^2 + bx + c
    - If b^2 - 4*a*c is negative, roots are imaginary (print an error)
    - To get the square root, include “from math import sqrt” at the top
        - Ex: sqrt(b^2 - 4*a*c)
    - Assume integer or float inputs (ignore fractions)
    - To make things easier, just print both roots

Practice Problem 2: caseCalc Function

- Goal: Write a Python function that will determine the number of upper and lower case letters in a string.
- Also write a program that will test this functionality.
- Relevant Information:
    - You can print the number of upper and lower case letters
    - Or you can use “return (lower, upper)” with the cal being “lower, upper = caseCalc(string)”

Practice Problem 3: Prime Factorization

- Src: https://www.101computing.net/prime-factor-tree-algorithm/
- Goal: Write a Python function that will output the prime factorization of an inputted number.
- Also write a program that will test this functionality.
- Relevant Information:
  - The source page asks for a whole factorization tree, you’re just looking for the prime factors
  - Ex: 140 = 2^2 * 5 * 7
    - Note: you can just output 2 * 2 instead
  - The easiest method is diving by the smallest prime number the current number is divisible by, outputting this prime, and repeating until the current number becomes a prime
  - You should use the isPrime function from last meeting

Next meeting:

Lists