CS21: INTRODUCTION TO COMPUTER SCIENCE

Prof. Mathieson

Fall 2017

Swarthmore College

Outline Sept 25:

- Functions
  - pretty_function.py
  - factorial.py
  - lettercount.py

- In operator and random library practice

Notes

- **Lab 3** due **Saturday** night
- **Quiz 1** returned on **Wednesday**
- Let me know if you will not be in class

Recap string formatting

String formatting

- **%d** – decimal (same behavior as **%i** it turns out!) use for integers, in newer python formatting **%i** is sometimes not supported

- **%f** – float, use **%8.2f** (for example) to make each float be 8 characters total (spaces used as padding on the left), rounded to 2 decimal places

- **%s** – string (same notation applies, but **%8.2s** will pad with spaces as well as truncate to 2 characters)

Functions

```python
def add(x, y):
    """
    This function takes two numerical arguments and returns their sum.
    """
    s = x + y
    return s

def multiply(x, y):
    """
    This function takes two numerical arguments (x,y) and returns x*y
    """
    m = x * y
    return m

def main():
    """
    In the main function, test our functions
    """

    num1 = 3
    num2 = 5
    sum_result = add(num1, num2)
    mult_result = multiply(num1, num2)
    print(sum_result, mult_result)

main()
```

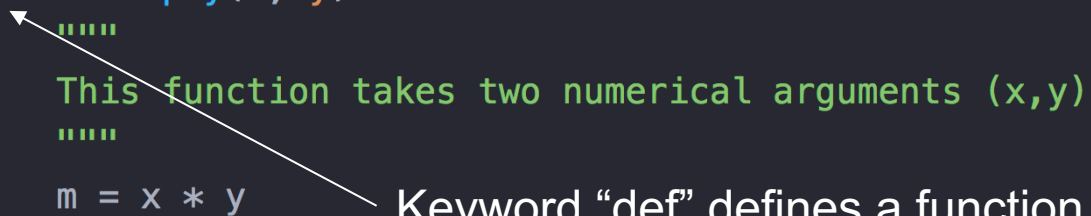
```
def add(x, y):
    """
    This function takes two numerical arguments and returns their sum.
    """
    s = x + y
    return s

def multiply(x, y):
    """
    This function takes two numerical arguments (x,y) and returns x*y
    """
    m = x * y
    return m

def main():
    """
    In the main function, test our functions
    """

    num1 = 3
    num2 = 5
    sum_result = add(num1, num2)
    mult_result = multiply(num1, num2)
    print(sum_result, mult_result)

main()
```



```
def add(x, y):
    """
    This function takes two numerical arguments and returns their sum.
    """
    s = x + y
    return s

def multiply(x, y):
    """
    This function takes two numerical arguments (x,y) and returns x*y
    """
    m = x * y
    return m

def main():
    """
    In the main function, test our functions
    """

    num1 = 3
    num2 = 5
    sum_result = add(num1, num2)
    mult_result = multiply(num1, num2)
    print(sum_result, mult_result)

main()
```

Arguments or *parameters* (input)

Keyword “def” defines a function

```
def add(x, y):
    """
    This function takes two numerical arguments and returns their sum.
    """
    s = x + y
    return s

def multiply(x, y):
    """
    This function takes two numerical arguments (x,y) and returns x*y
    """
    m = x * y
    return m

def main():
    """
    In the main function, test our functions
    """

    num1 = 3
    num2 = 5
    sum_result = add(num1, num2)
    mult_result = multiply(num1, num2)
    print(sum_result, mult_result)

main()
```

Description of function in triple quotes

Arguments or *parameters* (input)

Keyword “def” defines a function

```
def add(x, y):
    """
    This function takes two numerical arguments and returns their sum.
    """
    s = x + y
    return s

def multiply(x, y):
    """
    This function takes two numerical arguments (x,y) and returns x*y
    """
    m = x * y
    return m

def main():
    """
    In the main function, test our functions
    """

    num1 = 3
    num2 = 5
    sum_result = add(num1, num2)
    mult_result = multiply(num1, num2)
    print(sum_result, mult_result)

main()
```

Description of function in triple quotes

Arguments or *parameters* (input)

Indented: *body* of function

Keyword “def” defines a function

```
def add(x, y):
    """
    This function takes two numerical arguments and returns their sum.
    """
    s = x + y
    return s

def multiply(x, y):
    """
    This function takes two numerical arguments (x,y) and returns x*y
    """
    m = x * y
    return m

def main():
    """
    In the main function, test our functions
    """

    num1 = 3
    num2 = 5
    sum_result = add(num1, num2)
    mult_result = multiply(num1, num2)
    print(sum_result, mult_result)

main()
```

Description of function in triple quotes

Arguments or *parameters* (input)

Indented: *body* of function

Keyword “def” defines a function

main() is also a function (no arguments!)

```
def add(x, y):
    """
    This function takes two numerical arguments and returns their sum.
    """
    s = x + y
    return s

def multiply(x, y):
    """
    This function takes two numerical arguments (x,y) and returns x*y
    """
    m = x * y
    return m

def main():
    """
    In the main function, test our functions
    """

    num1 = 3
    num2 = 5
    sum_result = add(num1, num2)
    mult_result = multiply(num1, num2)
    print(sum_result, mult_result)

main()
```

Description of function in triple quotes

Arguments or *parameters* (input)

Indented: *body* of function

Keyword “def” defines a function

main() is also a function (no arguments!)

Calling or *invoking* a function

```
def add(x, y):
    """
    This function takes two numerical arguments and returns their sum.
    """
    s = x + y
    return s

def multiply(x, y):
    """
    This function takes two numerical arguments (x,y) and returns x*y
    """
    m = x * y
    return m

def main():
    """
    In the main function, test our functions
    """

    num1 = 3
    num2 = 5
    sum_result = add(num1, num2)
    mult_result = multiply(num1, num2)
    print(sum_result, mult_result)

main()
```

Description of function in triple quotes

Arguments or *parameters* (input)

Indented: *body* of function

Keyword “def” defines a function

main() is also a function (no arguments!)

Calling or *invoking* a function

Passing in arguments

Problems to try now with a partner

- cs21/week04/factorial.py
- cs21/week04/lettercount.py

Function practice problems

- cs21/practice/sum_list.py
- cs21/practice/series_function.py
- cs21/practice/fib_function.py