CSC 280 Introduction to Computer Science: Programming with Python

Lecture 4 : Functions

September 10, 2015

Prof. Bei Xiao

Fall, 2015

American University

Outline

- Function Calls
- Built-in function
- Modules, Math Module
- from keyword
- Function definitions
- Parameters and arguments

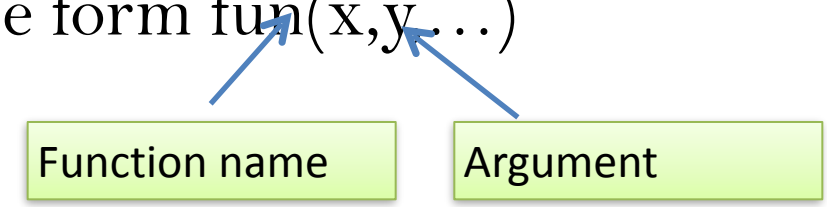
Take home reading

- Read Chapter 3.1-6.

- Python Math Functions:
- https://docs.python.org/2/library/math.html

Function calls

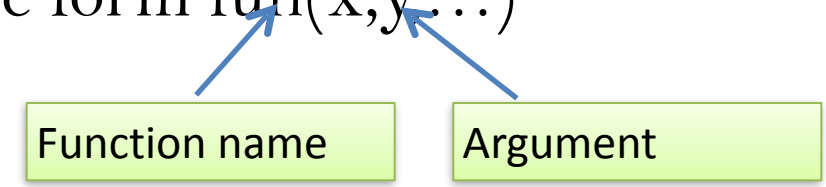
- Python supports expressions with math-like functions
  - A function in an expression is called function call
- Function expressions have the form fun(x,y…)



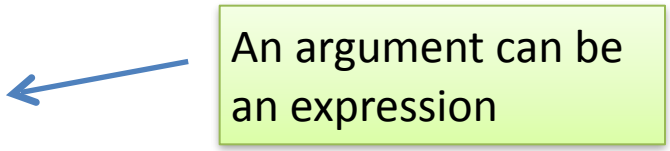
- Examples (math functions that work with Python)
  - round(2.34)
  - max(a+3,24)

Function calls

- Python supports expressions with math-like functions
  - A function in an expression is called function call
- Function expressions have the form fun(x,y…)



- Examples (math functions that work with Python)
  - round(2.34)
  - max(a+3,24)



Built-In Functions

- You have seen many functions already
  - Type casting functions: **int()**, **float()**, **bool()**
  - Dynamically type an expression: **type()**
  - Help function: **help()**
- Getting user input: **raw_input()**
- print<string> is not a functional call.
  - It is simply a statement (like assignment)
  - But it is in Python *3*

Argument goes in (), name() refers to function in general

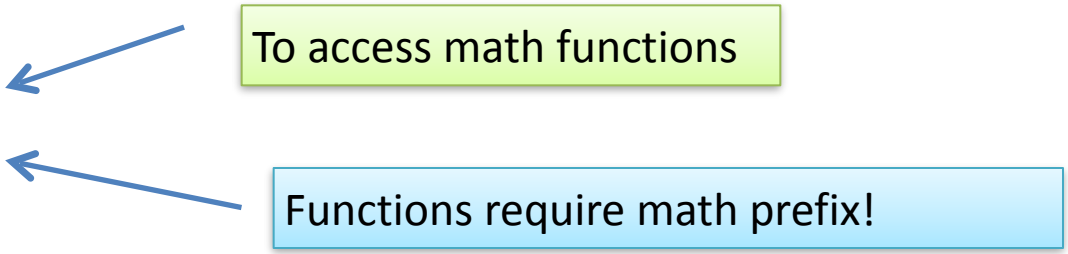
Built-In Functions vs. Module

- The number of built-in functions is small
  - https://docs.python.org/2/library/functions.html
- Missing a lot of functions you would expect
  - Example: cos(), sqrt()
- **Module**: file that contains Python code
  - A way for Python to provide optional functions
  - To access a module, the import command
  - Access the functions using module as *prefix*

Example: Module math

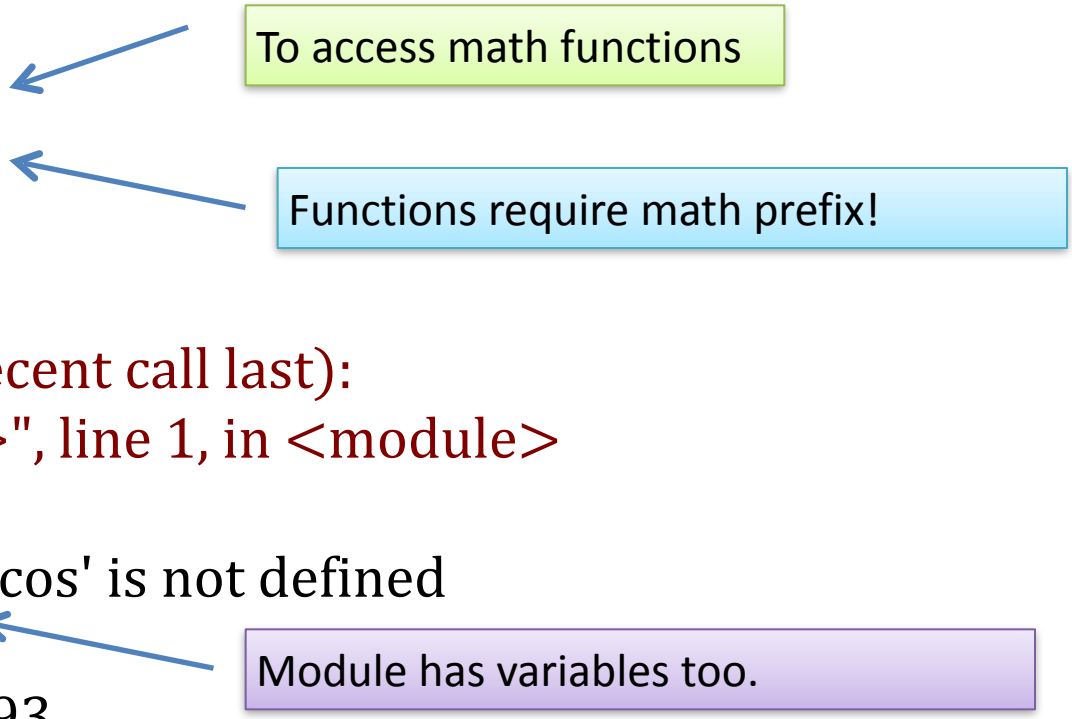
```
>>> import math
>>> math.cos(0)
1.0
>>> cos(0)
Traceback (most recent call last):
  File "<pyshell#2>", line 1, in <module>
    cos(0)
NameError: name 'cos' is not defined
>>> math.pi
3.141592653589793
>>> math.cos(math.pi)
-1.0
```

Example: Module math




```
>>> import math
>>> math.cos(0)
1.0
>>> cos(0)
Traceback (most recent call last):
  File "<pyshell#2>", line 1, in <module>
    cos(0)
NameError: name 'cos' is not defined
>>> math.pi
3.141592653589793
>>> math.cos(math.pi)
-1.0
```

Example: Module math




```
>>> import math
>>> math.cos(0)
1.0
>>> cos(0)
Traceback (most recent call last):
  File "<pyshell#2>", line 1, in <module>
    cos(0)
NameError: name 'cos' is not defined
>>> math.pi
3.141592653589793
>>> math.cos(math.pi)
-1.0
```

Using the **from** keyword

```
>>>import math
>>> math.pi
3.141592653589793
>>> math.cos(math.pi)
-1.0
>>> import math
>>> math.pi
3.141592653589793
>>> from math import pi
>>> pi
3.141592653589793
>>> from math import *
>>> cos(pi)
-1.0
```

Must prefix with module name

No prefix needed for variable pi

No prefix needed for variable pi

- Be careful using from
- Using import is *safer*
  - Modules might conflict
  - (functions w/ same name)

Using the **from** keyword

```
>>>import math
>>> math.pi
3.141592653589793
>>> math.cos(math.pi)
-1.0
>>> import math
>>> math.pi
3.141592653589793
>>> from math import pi
>>> pi
3.141592653589793
>>> from math import *
>>> cos(pi)
-1.0
```

Must prefix with module name

No prefix needed for variable pi

No prefix needed for variable pi

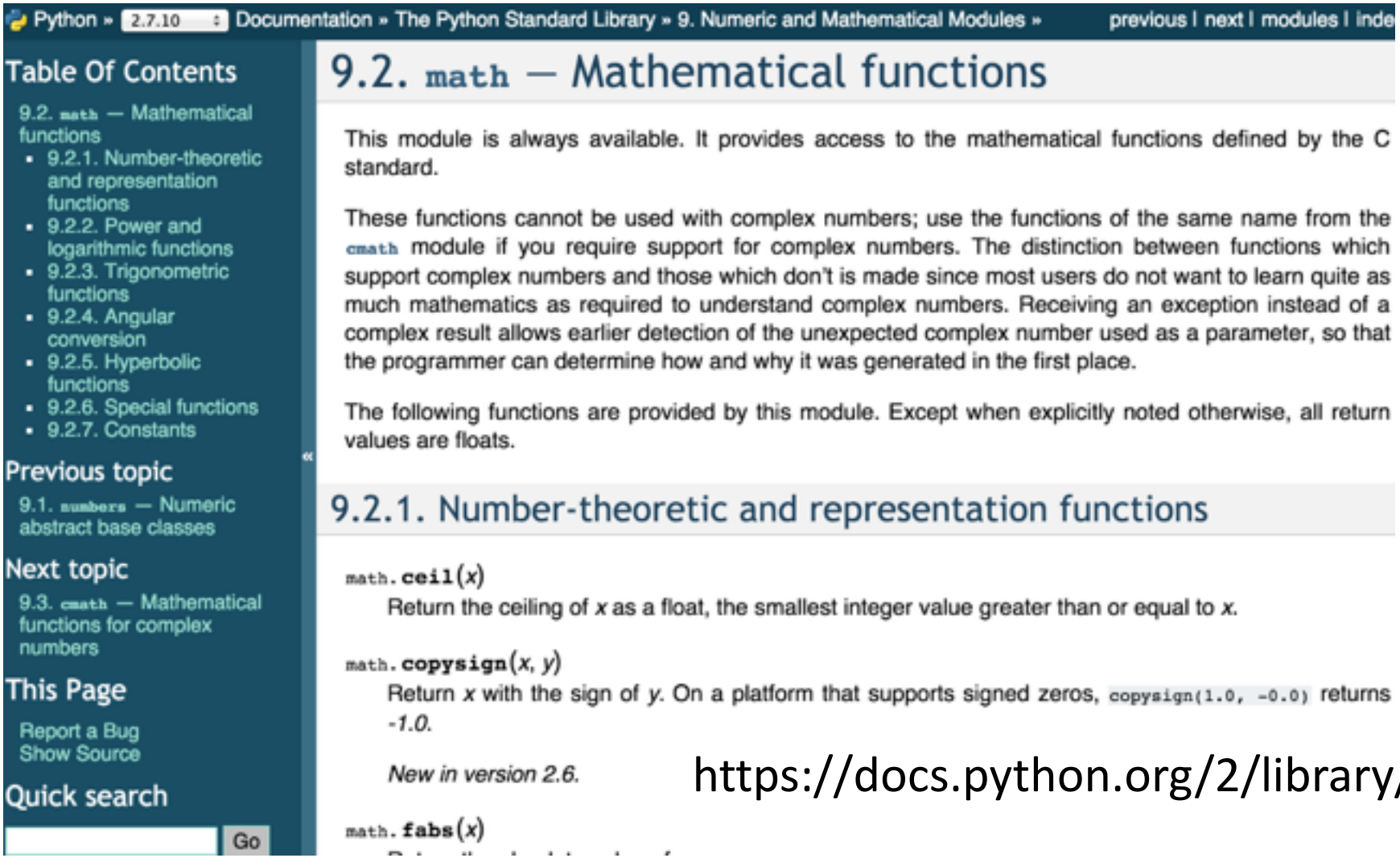
- Be careful using from
- Using import is *safer*
  - Modules might conflict
  - (functions w/ same name)

Composition

- One amazing thing about programming language is that we can **compose** small building blocks to create new ones.
- **Example:** convert degree to radius and compute the sin:

```
>>> x = math.sin((degrees/360.0)*2*math.pi)
>>> x
0.25881904510252076
```

Reading the Python documentation



https://docs.python.org/2/library/

Define a new function

- Function is an named sequence of statements that performs a desired operation.

- **A function definition** specifies the name of a new function and the sequence of statements that execute when the function is called:

```
def hello():
    print 'Hello!'
```

→ Function **Header**, Has to end with a :

→
- Function body: any number of statement
- Must be indented

Define a new function

- Function is an named sequence of statements that performs a desired operation.

- **A function definition** specifies the name of a new function and the sequence of statements that execute when the function is called:

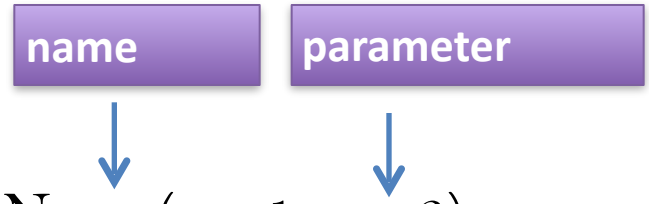
```
def hello():
    print 'Hello!'

hello()
```

Function **Header**, Has to end with a :

- Function body: any number of statement
- Must be indented

Calling the function, must have ()

Calling a function




```
Def  Name(arg1, arg2):
        Statements
```

```
Name(arg1, arg2)
```

Without calling a function, a function output **NOTHING.**

Calling a function

Function can be called using a statement.

Without calling a function, a function output **NOTHING.**

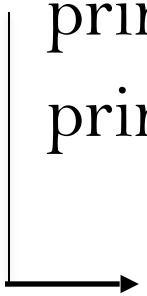
Example:

```
def printNumber(number):
    print str(number)

printNumber(5)

print type(printNumber)
```

A word about indentation

- Python is strict about **indentation**.

```
>>> def  print_lyrics():
    print "I am a lumberjack, and I am okay."
    print "I sleep all night and I work all day."
```



This indentation indicates the start of the **body** of the function . By convention, it is always 4 spaces.

A word about indentation (use tab key)

- Python is strict about **indentation.**

```
>>> def  print_lyrics():
    print "I am a lumberjack, and I am okay."
    print "I sleep all night and I work all day."
```

This indentation indicates the start of the **body** of the function

```
>>> print_lyrics()
I am a lumberjack, and I am okay.
I sleep all night and I work all day.
>>>
```

No indentation means outside the function definition

Function can contains another function

```
def two_pipes():
    print '| |'

def more_pipes():
    two_pipes()
    print ''
    two_pipes()
```

Parameters and arguments

Function Call

Command to do the function

greet('Walker')

Function
Header

Function Definition

Defines what function does

```
def greet(n):
    print "hello " +n + "!"
```

- Parameter: variable that is listed within the parentheses of a method header.
- Arguments: a value to assign to the method parameter when it is called

Parameters and arguments

Function Call

Command to do the function

greet('Walker')

Function Definition

Function **Header**

Defines what function does

```
def greet(n):
    print "hello " +n + "!"
```

Function **Body**

- Parameter: variable that is listed within the parentheses of a method header.
- Arguments: a value to assign to the method parameter when it is called

Parameters and arguments

Function Call

Command to do the function

greet('Walker')

Function Definition

Defines what function does

```
def greet(n):
    print "hello " +n + "!"
```

Function **Header**

Declaration of parameter **n**

Function **Body**

- Parameter: variable that is listed within the parentheses of a method header.
- Arguments: a value to assign to the method parameter when it is called

Parameters and arguments

Function Call

Command to do the function

```
greet('Walker')
```

argument to assign to n

Function **Header**

Function Definition

Defines what function does

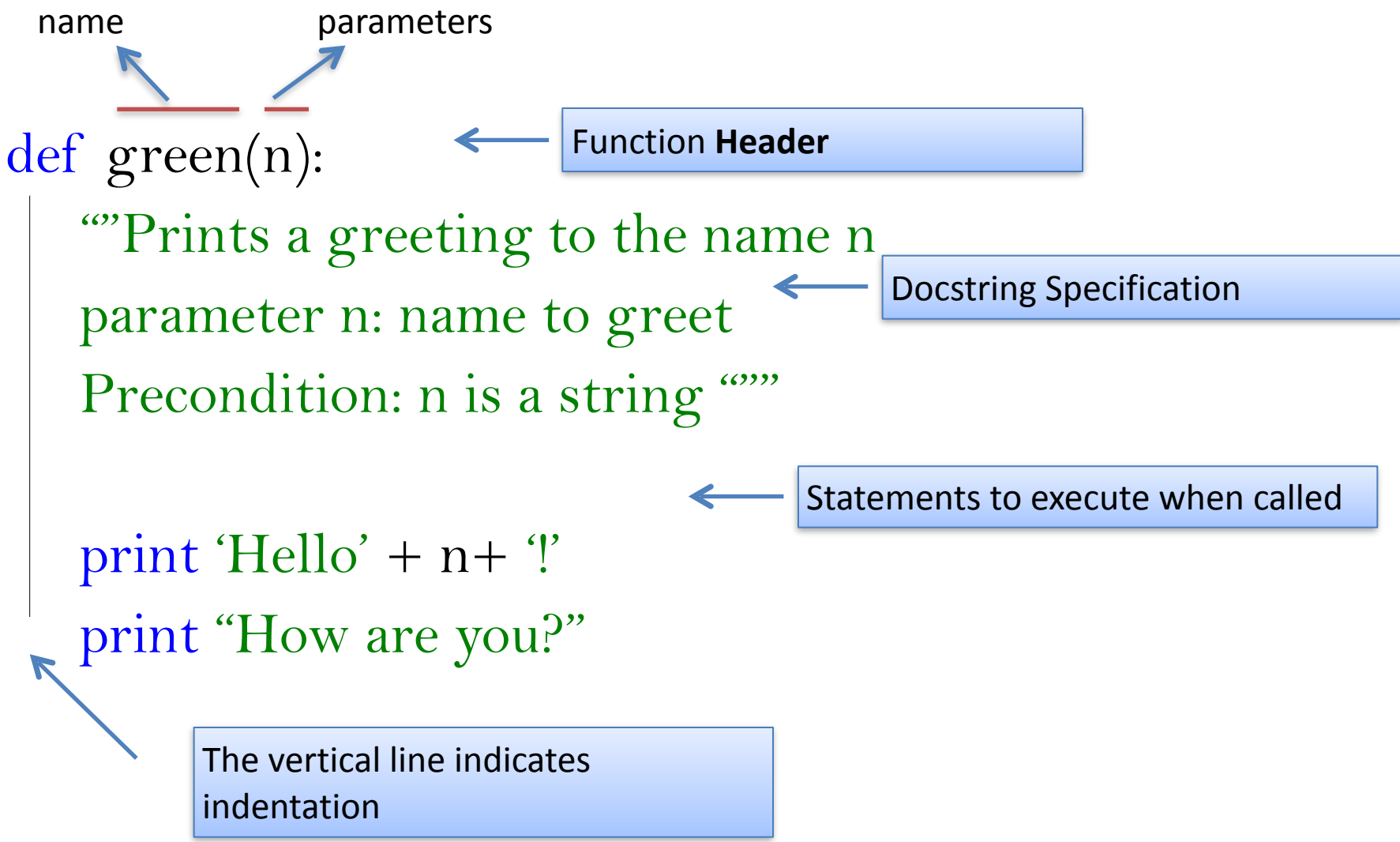
```
def greet(n):
    print "hello " +n + "!"
```

Declaration of parameter **n**

Function **Body**

- Parameter: variable that is listed within the parentheses of a method header.
- Arguments: a value to assign to the method parameter when it is called

Anatomy of a function definition



Exercise 1: say hello

- Define a function to print “hello” to a specific name.
- Call the function by sayHello(“Bob”)
- Which should print: Hello, Bob
- Call the function again to say Hello to Emily
- Using raw_input to ask the user about his or her
- name and all the function sayHello(name)

Exercise 2: write a few arithmetic functions

Write a the following functions that output arithmetic operation on two numbers. Note: please print out the functions.

- def add(num1, num2):
- def sub(num1, num2):
- def mul(num1, num2):
- def div(num1,num2):

Calling the above function by:

add(2,3)

num1 = 2

num2 = 3

num3 = 6.0

sub(num2,num1)

div(num3, num1)

Example: function without argument

Write a function that print the following shape:

Write another function to call the above function to print:

Next week

- Function with returns
- Modules
- Lab 1 is due on Wed before class. No need to hand in the code. Just hand in the sheets.