Programming Lab - II

Functions

Kaushik Pal

14th Feb, 2017

National Institute Of Technology, Rourkela

Repetitive work

```
a = 2
b = 33
c = 1
d = 99

sum1 = a + b
print '{0} added to {1} is {2}'.format( a, b, sum1 )

sum2 = c + d
print '{0} added to {1} is {2}'.format( c, d, sum2 )
```

Repetitive work

```
a = 2
b = 33
c = 1
d = 99

sum1 = a + b
print '{0} added to {1} is {2}'.format( a, b, sum1 )

sum2 = c + d
print '{0} added to {1} is {2}'.format( c, d, sum2 )
```

Output:

```
2 added to 33 is 35

1 added to 99 is 100
```

Repetitive work - NEW proposal

```
a = 2
b = 33
c = 1
d = 99
```

```
printSum( a, b )
```

```
printSum( c, d )
```

Repetitive work – NEW proposal

printSum(a, b)

```
sum = a + b

print '{0} added to {1} is {2}'.format( a, b, sum )
```

FUNCTION – DEFINITION & SYNTAX

Definition:

Functions are self contained "modules" of code that take inputs, do a computation, and produce outputs.

Syntax:

```
def <function name> ( <param1>, <param2> ) :
    < line 1 >
    < line 2 >
    < line 3 >
```

Function – Definition & Syntax

Definition:

Functions are self contained "modules" of code that take inputs, do a computation, and produce outputs.

Syntax:

```
def <function name> ( <param1>, <param2> ) :

    < line 1 >
    < line 2 >
    < line 3 >
```

FUNCTION – DEFINITION & SYNTAX

```
def printSum ( a, b ) :
     sum = a + b
     print '{0} added to {1} is {2}'.format(a,b,sum)




a = 2
b = 33
c = 1
d = 99


printSum( a, b )

printSum( c, d )
```

Function – Definition & Syntax

```
def printSum ( a, b ) :
    sum = a + b
    print '{0} added to {1} is {2}'.format(a,b,sum)



a = 2
b = 33
c = 1
d = 99


printSum( a, b )

printSum( c, d )
```

Output:

```
2 added to 33 is 35
1 added to 99 is 100
```

FUNCTION – DEFINITION & SYNTAX

```
def printSum ( a, b ) :
     sum = a + b
     print '{0} added to {1} is {2}'.format(a,b,sum)
```

Function definition

(not executed unless called)

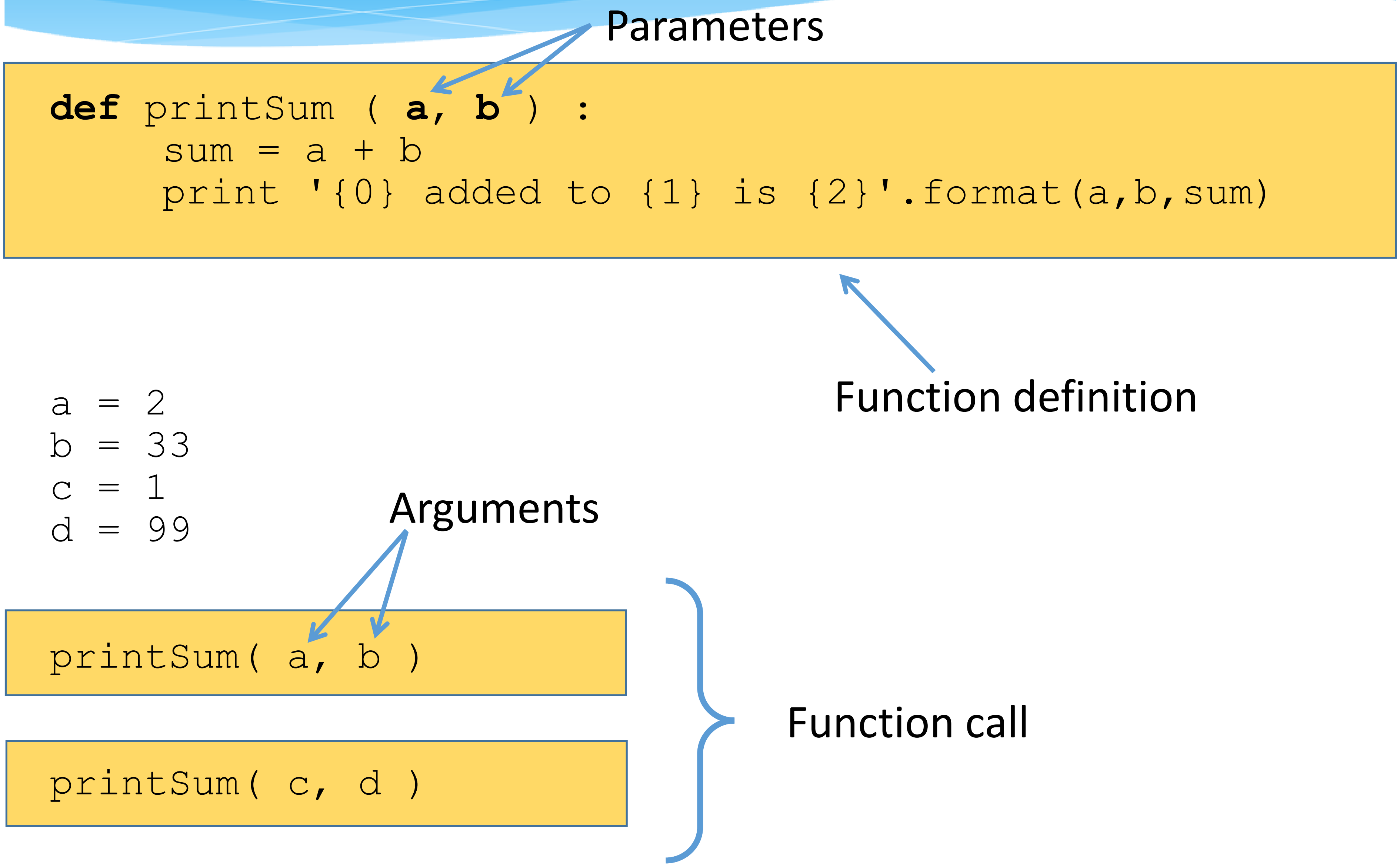
```
a = 2
b = 33
c = 1
d = 99
```

```
printSum( a, b )
```

```
printSum( c, d )
```

Function call

Function – Definition & Syntax



Parameters

```
def printSum ( a, b ) :
     sum = a + b
     print '{0} added to {1} is {2}'.format(a,b,sum)
```

Function definition

```
a = 2
b = 33
c = 1
d = 99
```

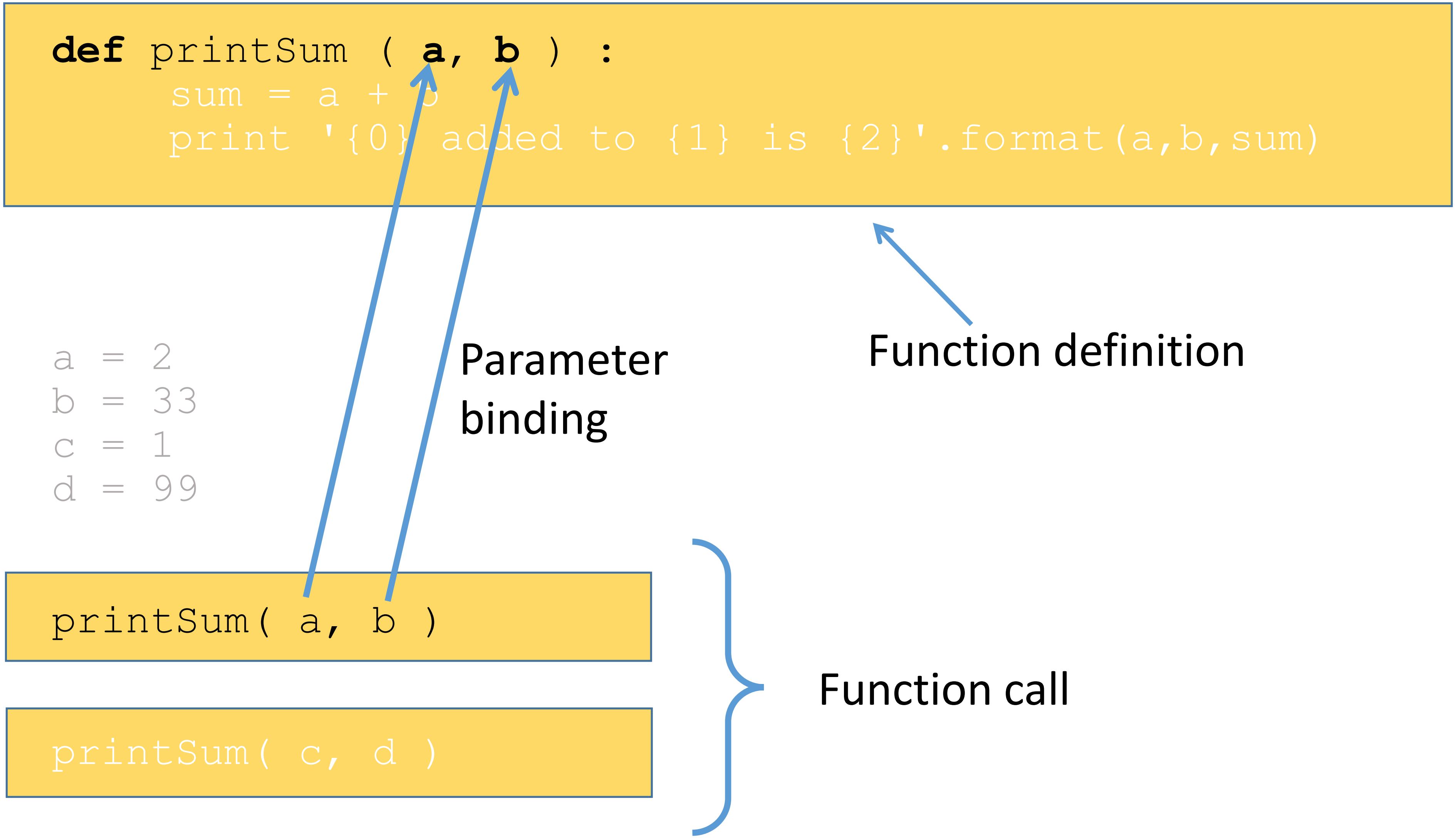
Arguments

```
printSum( a, b )
```

```
printSum( c, d )
```

Function call

Function – Definition & Syntax



```
def printSum ( a, b ) :
        sum = a + b
        print '{0} added to {1} is {2}'.format(a,b,sum)
```

Function definition

Parameter binding

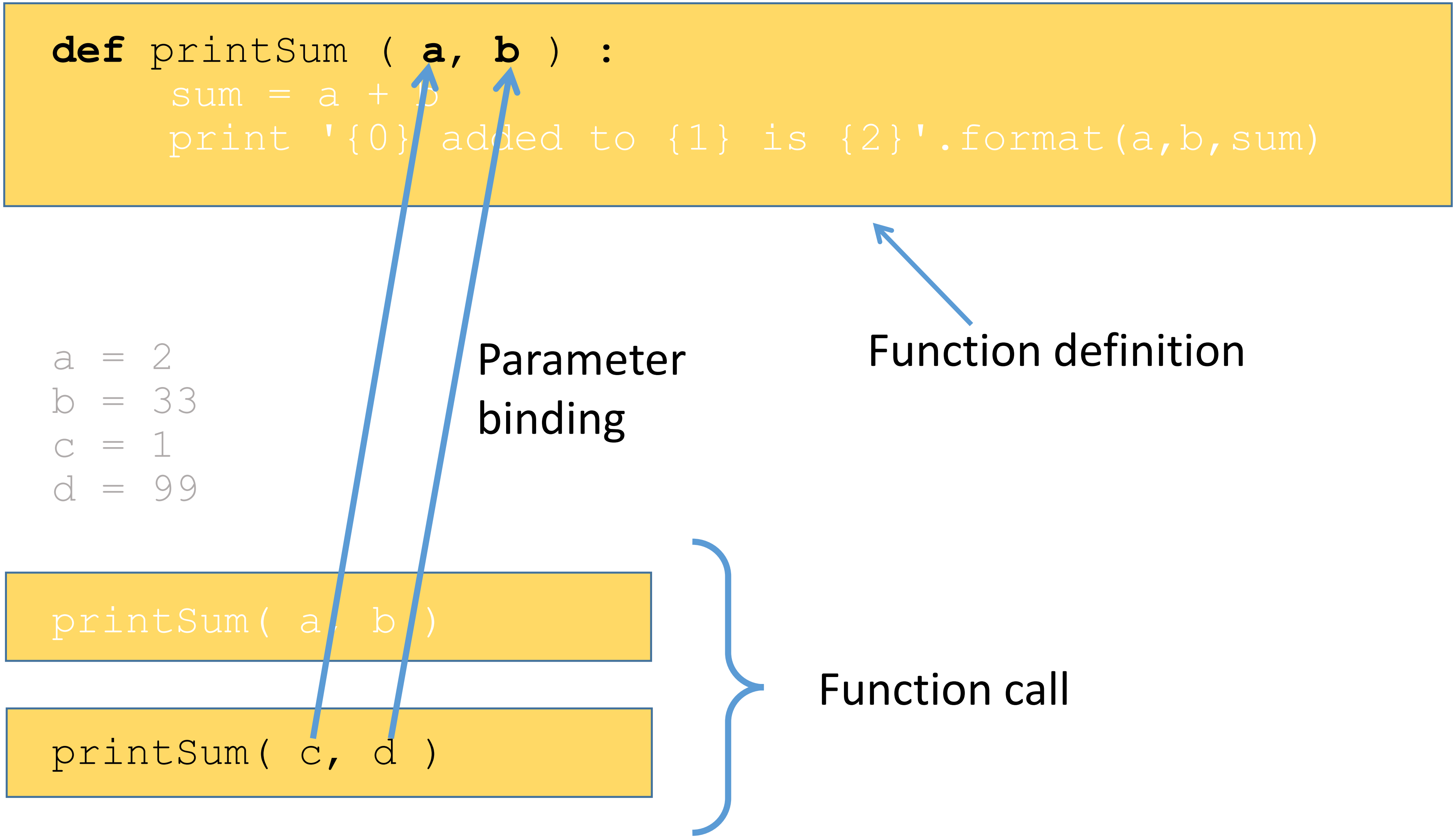
```
a = 2
b = 33
c = 1
d = 99
```

```
printSum( a, b )
```

```
printSum( c, d )
```

Function call

Function – Definition & Syntax



```
def printSum ( a, b ) :
        sum = a + b
        print '{0} added to {1} is {2}'.format(a,b,sum)
```

Function definition

Parameter binding

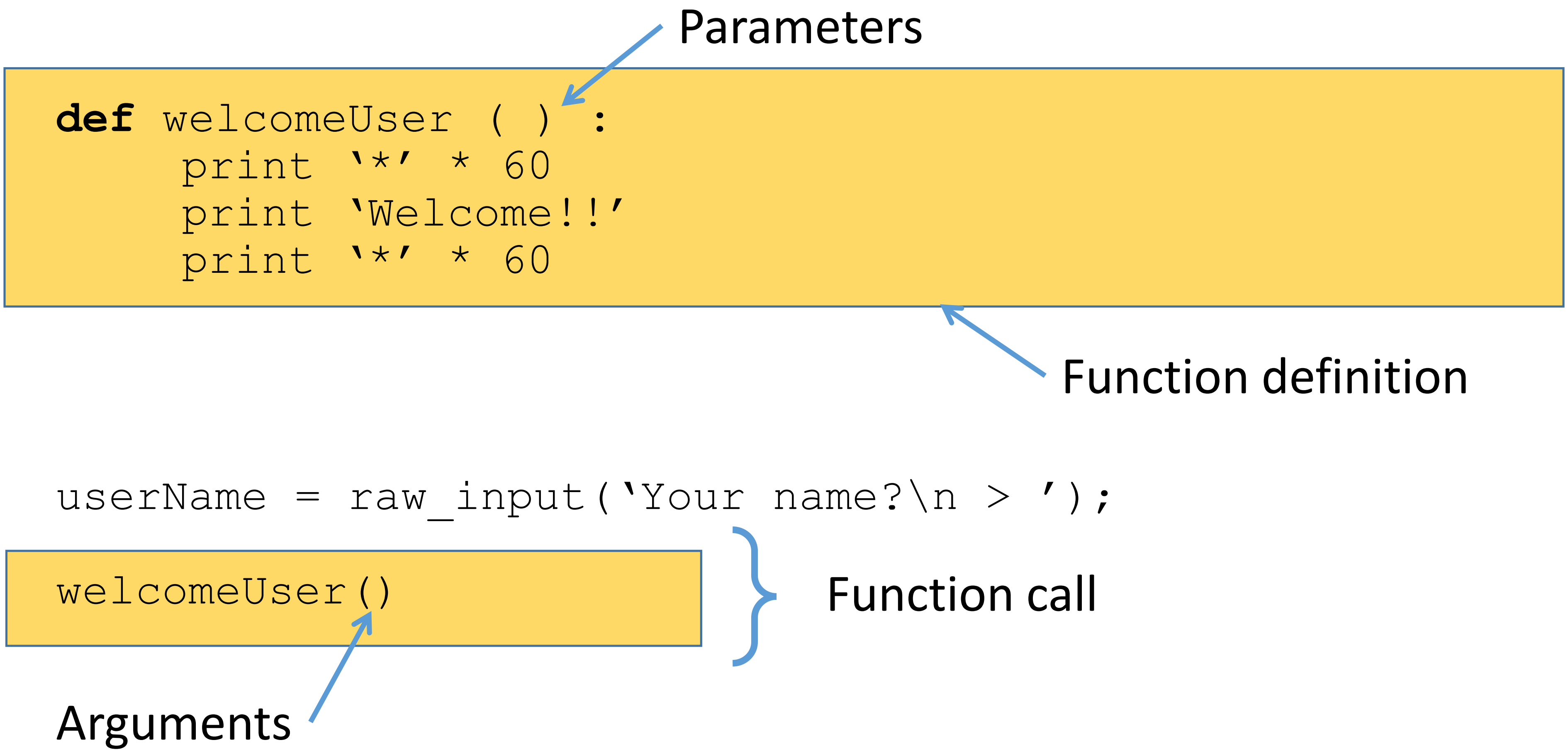
```
a = 2
b = 33
c = 1
d = 99
```

```
printSum( a, b )
```

```
printSum( c, d )
```

Function call

Function – Definition & Syntax



Parameters

```
def welcomeUser ( ) :
      print ‘*’ * 60
      print ‘Welcome!!’
      print ‘*’ * 60
```

Function definition

```
userName = raw_input(‘Your name?\n > ’);
```

```
welcomeUser()
```

Function call

Arguments

Function – Definition & Syntax

```
def welcomeUser ( ) :
      print ‘*’ * 60
      print ‘Welcome!!’
      print ‘*’ * 60




userName = raw_input(‘Your name?\n > ’);

welcomeUser()
```

Output:

```
Your name?
 >
```

Function – Definition & Syntax

```
def welcomeUser ( ) :
     print '*' * 60
     print 'Welcome!!'
     print '*' * 60



userName = raw_input('Your name?\n > ');

welcomeUser()
```

Output:

```
Your name?
 > Cupid
*********************************************************
Welcome!!
*********************************************************
```

Function – Definition & Syntax

What will be the program for the o/p below ?

Output:

```
Your name?
 > Cupid
********************************************************
Welcome Cupid ... !!
********************************************************
```

Function – Definition & Syntax

```
def welcomeUser ( name ) :
     print '*' * 60
     print 'Welcome ', name, ' ... !!'
     print '*' * 60



userName = raw_input('Your name?\n > ');

welcomeUser( userName )
```

Output:

```
Your name?
 > Cupid
************************************************************
Welcome Cupid ... !!
************************************************************
```

Functions – Important points

- *The number of **arguments** "passed" into a function must exactly match the number of **parameters** required for the function.*
- *The type of each argument must exactly match the intended type of each parameter.*
- There can be many functions written in a single file.
- Functions can be called from other functions.
- *All the functions need to be **<u>defined</u>** before they are **<u>called</u>**.*

Function – Multiple functions

```
def printSum ( a, b ) :
     sum = a + b
     print '{0} added to {1} is {2}'.format(a,b,sum)


def printDiv ( a, b ) :
     div = a / b
     print '{0} divded by {1} is {2}'.format(a,b,div)


#-----------------------------------------------------

a = 25
b = 5

printSum( a, b );

printDiv( a, b );
```

Output:

```
25 added to 5 is 30
25 divided by 5 is 5
```

Function – Call from other function

```
def printSum ( a, b ) :
     sum = a + b
     print '{0} added to {1} is {2}'.format(a,b,sum)

def printDiv ( a, b ) :
     div = a / b
     print '{0} divded by {1} is {2}'.format(a,b,div)

def printBoth ( a, b ) :
     printSum( a, b )
     printDiv( a, b )
#-----------------------------------------------------

a = 25
b = 5

printBoth(a,b);
```

Output:

```
25 added to 5 is 30
25 divided by 5 is 5
```

Function - Ordering

```
def printBoth ( a, b ) :
     printSum( a, b )
     printDiv( a, b )
```

```
def printSum ( a, b ) :
     sum = a + b
     print '{0} added to {1} is {2}'.format(a,b,sum)

def printDiv ( a, b ) :
     div = a / b
     print '{0} divded by {1} is {2}'.format(a,b,div)

#-----------------------------------------------------

a = 25
b = 5

printBoth(a,b);
```

Output:

```
25 added to 5 is 30
25 divided by 5 is 5
```

Function - ordering

```
printSum( a, b )
printDiv( a, b )
```

```
def printSum ( a, b ) :
     sum = a + b
     print '{0} added to {1} is {2}'.format(a,b,sum)

def printDiv ( a, b ) :
     div = a / b
     print '{0} divded by {1} is {2}'.format(a,b,div)

#-----------------------------------------------------

a = 25
b = 5

printBoth(a,b);
```

Output:

```
NameError: name 'printSum' is not defined
```

Functions - Advantages

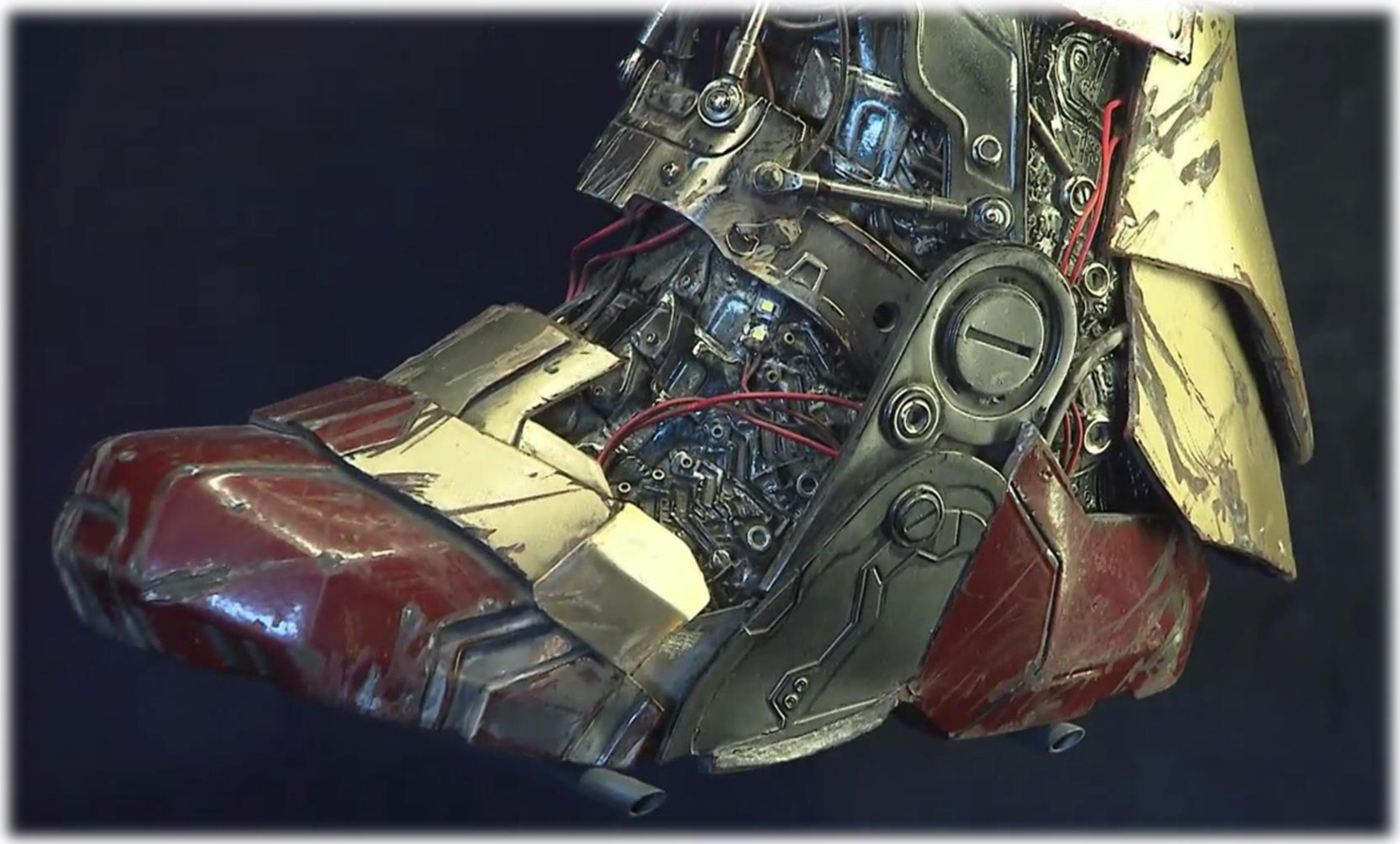
This is an entire program !!

Functions - Advantages

These are all functions !!

Functions - Advantages

A function does a single task (complex or simple) ... And does it well !!



Functions - Advantages

1. Reduces code duplicacy

- There should be minimal overlap between function definitions.

2. Modularize the program

*- **Divide** the entire program into smaller modules.*

*- **Group** lines of code meant to do a particular task.*

*3. Functions once created can be **reused** in other Python programs.*

4. Helps in writing clean code

*– Makes C programs more **readable**.*

Function – Return value

WAP to add three numbers.

```
def addThree ( num1 , num2 , num3 ) :
      result = num1 + num2 + num3;
      str = ‘{0} + {1} + {2} = {3}’
      print str.format( num1, num2, num3, result );
#-------------------------------------
addThree( 12, 34, 44 );
```

Output:

```
12 + 34 + 44 = 90
```

Function – Return value

WAP to add three numbers.

```
def addThree ( num1 , num2 , num3 ) :
    result = num1 + num2 + num3;
    return result
#-------------------------------------------------
n1=12 ; n2=34 ; n3=44
sumVal = addThree( n1, n2, n3 );
str = ‘{0} + {1} + {2} = {3}’
print str.format( n1, n2, n3, sumVal );
```

Output:

```
12 + 34 + 44 = 90
```

FUNCTION – RETURN VALUE

WAP to add three numbers.

```
def addThree ( num1 , num2 , num3 ) :
        return ( num1 + num2 + num3 );


#-----------------------------------------

n1=12 ; n2=34 ; n3=44
sumVal = addThree( n1, n2, n3 );
str = '{0} + {1} + {2} = {3}'
print str.format( n1, n2, n3, sumVal );
```

Output:

```
12 + 34 + 44 = 90
```

Function – Return value

What is the return value of functions not returning anything ??

```
def welcomeUser ( ) :
     print '*' * 60
     print 'Welcome!!'
     print '*' * 60


userName = raw_input('Your name?\n > ');

print welcomeUser()
```

Output:

```
Your name?
 > Cupid
*********************************************************
Welcome!!
*********************************************************
None
```

Function – Assignment - I

WAP to get the o/p as shown.

Write functions to :

*1. Take a **single input** from user and **return** its integer value.*

*2. **Print** the results of the arithmetic operations as shown*

Output:

```
Enter the value of  a
 > 25
Enter the value of  b
 > 5

Adding  25  with  5
Ans:  30

Subtracting  5  from  25
Ans:  20

Multiplying  25  with  5
Ans:  125

Dividing  25  by  5
Ans:  5
```

Standard Functions

- abs()
- all()
- any()
- basestring()
- bin()
- bool()
- bytearray()
- callable()
- chr()
- classmethod()
- cmp()
- compile()
- complex()
- delattr()
- dict()
- divmod()
- enumerate()
- eval()
- execfile()
- file()
- filter()
- float()
- format()
- frozenset()
- getattr()
- globals()
- hasattr()
- hash()
- help()
- hex()
- input()
- int()
- isinstance()
- issubclass()
- iter()
- len()
- list()
- locals()
- long()
- map()
- max()
- memoryview()
- min()
- next()
- object()
- open()
- ord()
- pow()
- print()
- property()
- range()
- raw_input()
- reduce()
- reload()
- repr()
- reversed()
- round()
- set()
- setattr()
- slice()
- staticmethod()
- str()
- sum()
- super()
- tuple()
- type()
- unichr()
- unicode()
- vars()
- xrange()
- zip()
- __import__()

Member Functions

String:

```
‘Hello {0} !!’.format( ‘world’ )
‘N I T’.replace( ‘ ’ , ‘ . ’)
‘abcababbab’.count( ‘ab’ )
```

Float:

```
x = 1.0
y = 2.5e-3
x.is_integer()
y.as_integer_ratio()
```

Functions – Variable Scope

Variable Scope: 1. Local

2. Global

```
def short():
     q = “NIT”
     print q


s = “National Institute of Technology"
short()
print s
```

Output:

```
NIT
National Institute of Technology
```

Functions – Variable Scope

```
def short():
     q = “NIT”
     print q


s = “National Institute of Technology"
short()
print s

print q
```

Output:

```
NIT
National Institute of Technology

NameError: name 'q' is not defined
```

Functions – Variable Scope

```
def short():
     q = “NIT”
     print q
     print s


s = “National Institute of Technology"
short()
print s
```

Output:

```
NIT
National Institute of Technology
National Institute of Technology
```

Functions – Variable Scope

```
def short():
     q = “NIT”
     print q
     print s


s = “National Institute of Technology"
short()
print s
```

s : *Global* variable

q : Variable *local* to short()

Output:

```
NIT
National Institute of Technology
National Institute of Technology
```

Functions – Variable Scope

```
def short():
     q = “NIT”
     print q
     print ‘inside before: ’, s
     s = “NIT Rourkela”
     print ‘after before: ’, s


s = “National Institute of Technology“
print ‘before: ’, s
short()
print ‘after: ’, s
```

Output:

Functions – Variable Scope

```
def short():
     q = “NIT”
     print q
     print ‘inside before: ’, s
     s = “NIT Rourkela”
     print ‘after before: ’, s


s = “National Institute of Technology“
print ‘before: ’, s
short()
print ‘after: ’, s
```

Output:

```
before:  National Institute of Technology
NIT
inside before:
UnboundLocalError: local variable 's' referenced before
assignment
```

Functions – Variable Scope

```
def short():
     global s
     q = “NIT”
     print q
     print ‘inside before: ’, s
     s = “NIT Rourkela”
     print ‘after before: ’, s


s = “National Institute of Technology“
print ‘before: ’, s
short()
print ‘after: ’, s
```

Output:

```
before:  National Institute of Technology
NIT
inside before:  National Institute of Technology
inside after:  NIT - Rourkela
after:  NIT - Rourkela
```

Function – Assignment - II

```
Your name?
 > Cupid
Hello  Cupid ...!!

Enter the value of a
 > 30
Enter the value of b
 > 5

----------------------------------
Calculator MENU:

1. Add   2. Subtract   3. Multiply
4. Divide     5. Power
6. Enter a    7. Enter b
0. Exit

 > 1

Adding  30  with  5
Ans:  35
```

Use global variables for a & b

```
----------------------------------
Calculator MENU:

1. Add   2. Subtract   3. Multiply
4. Divide     5. Power
6. Enter a    7. Enter b
0. Exit

 > 0

Thank you!
----------------------------------
```

Exit:

```
import sys


sys.exit()
```

First statement of the file

Inside ***your*** exit function

Thank You !