CPE101 Functions

Winter 2019
@ Cal Poly SLO
By
Toshi

Learning Objectives

1. Functions
    a. Simple math style functions.
    b. Simple functions with objects as arguments.
    c. Parameter passing.

What is a Function?

- A function is a named sequence of statements that performs a computation.
- Functions are also called “methods” depending on how they are defined in a program and also called “subroutines” in other programming languages.
- Functions are not necessarily mathematical functions.

```
y = f(x) # f(x)=x**2 + x + 1

print(x)
```

Functional Decomposition

make_pizza()

- make_dough()
- add_toppings()
- bake()

Defining Functions

- A function definition must always start with “def” keyword.
- Function names must only contain alphabets, numerical letters and _, but can not start with numerical letters.
- You can pass values to function as “arguments”. Arguments nee be defined within ().
- You can return values from a function.

```
# define add function
def add(x, y):
    return x + y
```

The indentation (four spaces) indicates that this line is a part of the function body.

```
c = add(1, 2)
```

Using Functions

```
#Functions with arguments with a return value

#a function to multiply two numbers
def mult(x, y):
    Return x * y

#a function with no return value
def greetings(name):
    print “Hello %s!” % (name)

#a function with no argument nor return value
def hello():
    print “Hello World!”
```

```
a = 1
b = 2

#call functions with arguments
c = add(a, b) # c = a + b = 3
d = mult(b, c) # d = b * c = 6

#you can pass an expression as an argument
d = mult(b, add(a, b)) # d = b * (a + b) = 6

name = “John”
greeting(name) # prints “Hello John!”

‘’’you always need () to call functions even
though some functions do not take arguments’’’
hello() # prints “Hello World!”
```

The scope of variables

- Variables defined within a function is only accessible within the function and destroyed when the program execution gets out of the function.
  - Local variables
- Variables defined outside of functions is accessible from anywhere.
  - Global variables

Local Variable Examples

```
def f(x):
    b = 2
    c = 3
    result = b * x + c
    return result

y = f(1)
print y
print x, b, c, result
```

```
5

NameError: name 'x' is
not defined
NameError: name 'b' is
not defined
NameError: name 'c' is
not defined
NameError: name 'result'
is not defined
```

Local Variable Examples

```
my_global = "Global"

def greetings():
    my_global = “Local”
    print "hello", my_global

greetings()
print "hello", my_global
```

```
hello Local
hello Global
```