CS521 O2
Information Structures with Python

Lecture 2

Guanglan Zhang

guanglan@bu.edu

Some slides adapted from Prof. Eugene Pinsky

Table of Content

Namespace and Scope

- An assignment statement creates a symbolic name that references an object
- A namespace is a collection of currently defined symbolic names along with information about the object that each name references. It is a relation between names and objects.
- LEGB rule defines the sequence of namespaces examined when looking for a name
  - ✓ Local
  - ✓ Enclosing
  - ✓ Global
  - ✓ Built-in

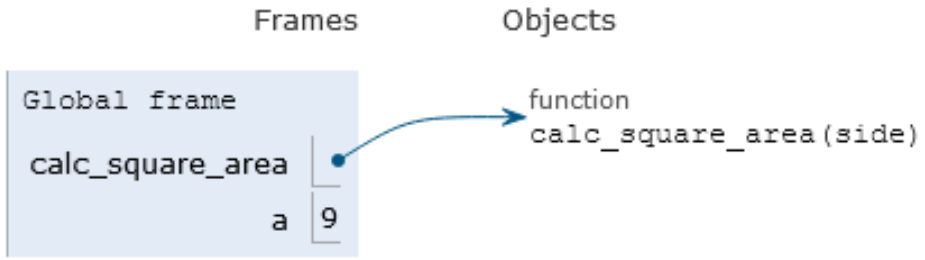
Local namespace & Global namespace

- The scope of a name is the region of a program in which that name has meaning.
- The interpreter creates a new namespace whenever a function executes. That namespace is **local** to the function and remains in existence until the function terminates. Use *locals()* to display a dictionary of the local namespace
- The **global** namespace contains all names defined at the level of the main program. Use *globals()* to display a dictionary of the global namespace
- Local assignment rule: an assignment is assumed to create a name only in the presently active namespace
- Use *global* statement to declare a variable to be a global variable

Namespace and Scope (cont.)

```
def calc_square_area(side):
    area = side*side
    return area

a = calc_square_area(3)
print("Area of the square is", a)
```



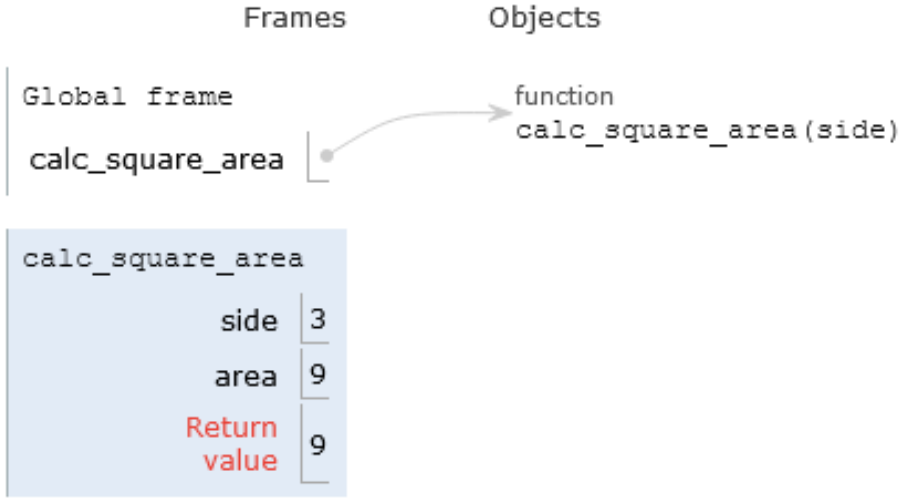
Python 3.6
(known limitations)

```
1  def calc_square_area(side):
2      area = side*side
3      return area
4
5
6  a = calc_square_area(3)
7  print("Area of the square is", a)
```

Edit this code

st executed
execute

Print output (drag lower right corner to resize)



Modules

- A module is stored as a file. Large programs can be split into modules
- A Python script can import a whole module. To reference an object in a module, we precede the associated name with the module name using dot notation.

```
In [54]: import math
    ...: print(math.pi)
3.141592653589793
```

- A Python script can import a whole module use shorter name

```
In [55]: import math as mh
    ...: print(mh.pi)
3.141592653589793
```

- A Python script can import an object from a module

```
In [56]: from math import pi
    ...: print(pi)
3.141592653589793
```

Avoid ambiguity

- To avoid ambiguity, below is the preferred way of importing modules

```
import module1 as m1
Import module2 as m2

x = m1.function_name()
y = m2.function_name()
```

Built-Ins and Enclosed

- When Python starts, it automatically loads two default modules, __main__ and __builtins__, without requiring an import.

- The Built-in namespace provides a link to all the regular Python programs and data types that are provided by default

- The enclosed scope rule applies when a function defines a function – a new function is defined within a function

Python types

- Python has two groups of types
  - ✓ Primitive types
    - ✓ int - integer numbers. e.g. -1, -2, -3
    - ✓ float - Floating-point numbers. e.g. 1.2, 42.42
    - ✓ bool – Boolean: True, False
    - ✓ complex - complex numbers. e.g. 1.0 + 2.0j, 1.5 + 2.5j

  - ✓ Collections
    - ✓ str – strings are used to represent text data, the text is given under quote marks. e.g. "ABCD“, ‘ABCD’
    - ✓ list – a sequence type, e.g. [3, 5.5, ‘abc’]
    - ✓ tuple - a collection which is ordered and unchangeable, e.g. (1, 2, 3)
    - ✓ dict – a map type consists of key-value pairs, e.g. {‘Alice’: 100, ‘Bob’: 88}
    - ✓ set – a set of unique elements, e.g. {1, 3, 6}

- Python has two special types
  - ✓ None type
  - ✓ range type

Objects types, not variable types

- In Python, everything in the system is considered to be an object
- A variable can refer to any object and can change its association over time
- An object in Python has the following
  - ✓ An identity - Whenever an object is created, it receives an identification number. View the number using id()
  - ✓ Some attributes – information about the object, such as the object's type, type()
  - ✓ Zero or more names – used by programmers to make the code more readable
- Each type has an associated constructor – its name is the name of the type

Two types of comparisons

- (x == y) - Whether x and y have the same value
- (x is y) - Do x and y refer to the same object

Exercise 1

write a program that converts a patient's weight in pounds and height in inch to Kg and cm.

1 lb = 0.4536 kg
1 in = 2.54 cm

Exercise 2

x = [7, 8, 13, 14, 15]
Print out odd numbers from the above list. No hard coding. The program should work for a list of different size.

Key takeaways

- There are primitive data types and collections
- In Python, everything in the system is considered to be an object
- An object has an identity, some attributes, and zero or more names.
- LEGB rule defines the sequence of namespaces examined when looking for a name
- There are two types of comparisons