PYTHON FOR COMPUTATIONAL PROBLEM SOLVING

Variables, type and id

Prof. Sindhu R Pai
PCPS Theory Anchor - 2024
Department of Computer Science and Engineering

Identifier

•An **identifier** is a sequence of one or more characters used to provide a name for a given program element. Examples: name , srn_number, ph_no, marks1, marks2

•It is used to identify the program element

• It may contain letters and digits and underscore characters

Naming Convention:

1.Can begin with alphabets a-z or A-Z.

2.Cannot begin with a digit 0-9 or a special character and Quotes are not allowed.

3.Spaces are not allowed as part of an identifier.

4.underscore character, _ is also allowed to aid in the readability of long identifier names. The variables that begin with underscore has a special meaning in object oriented programming. So we do not prefer to use _ as the first character.

Valid and invalid Identifiers

Valid Identifiers	Invalid Identifiers	Reason Invalid
totalSales	'totalSales'	quotes not allowed
totalsales	total sales	spaces not allowed
salesFor2010	2010Sales	cannot begin with a digit

Keywords

- Keywords are **reserved** words that have a predefined meaning.
- To know the keywords, type help() in the python prompt and in the help prompt, type keywords

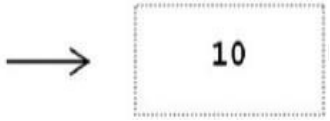
```
help> keywords

Here is a list of the Python keywords.  Enter any keyword to get more help.

False               class               from                or
None                continue            global              pass
True                def                 if                  raise
and                 del                 import              return
as                  elif                in                  try
assert              else                is                  while
async               except              lambda              with
await               finally             nonlocal            yield
break               for                 not
```

Variables

- A **variable** is a name (identifier) that is associated with a value and it is **always reference type**

- A variable can be assigned diffe 

program's execution—hence, the name "variable."

variable `num` is assigned the value 10

- Whenever variable **num** appears in a calculation, it is the current value of **num** that is used `num + 20 *` (10+ 20)

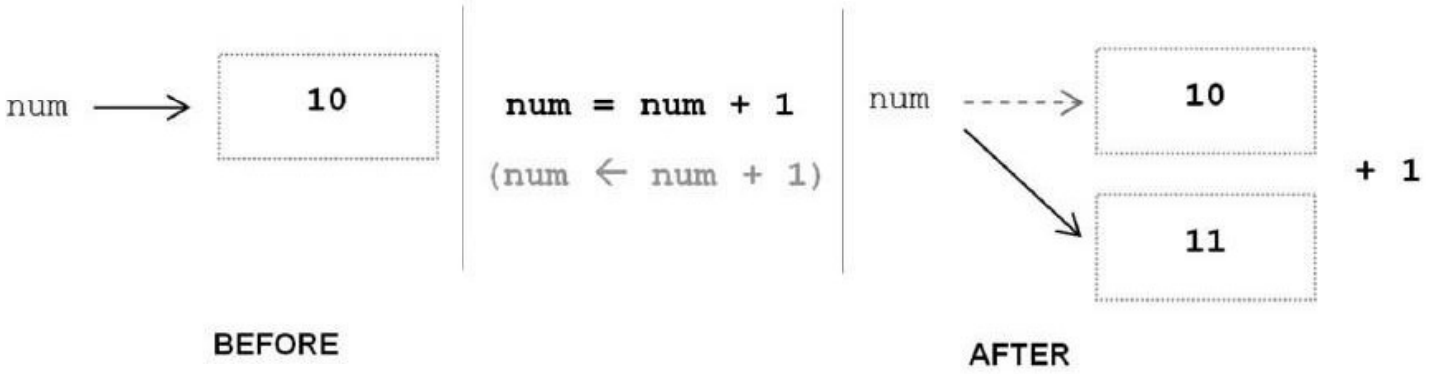
- If variable **num** is assigned a new value, then the same expression will produce a different result.

```
num = 5
num + 20
```
(5 + 20)

Diagrammatic Representation

- Variables are assigned values by use of the **assignment operator , =**

num = 10

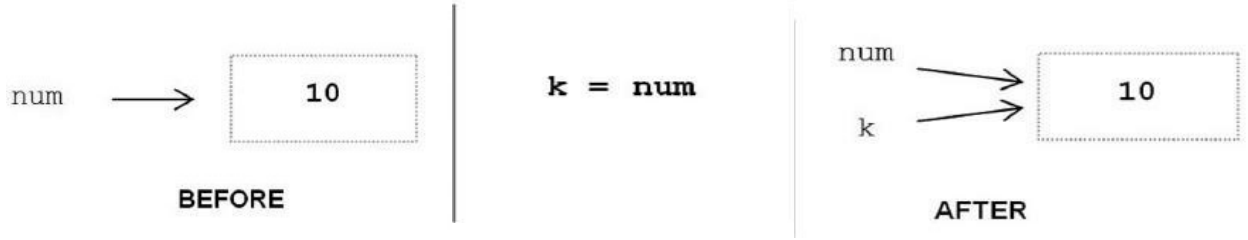


Note: The right side of an assignment is evaluated first, then the result is assigned to the variable on the left.

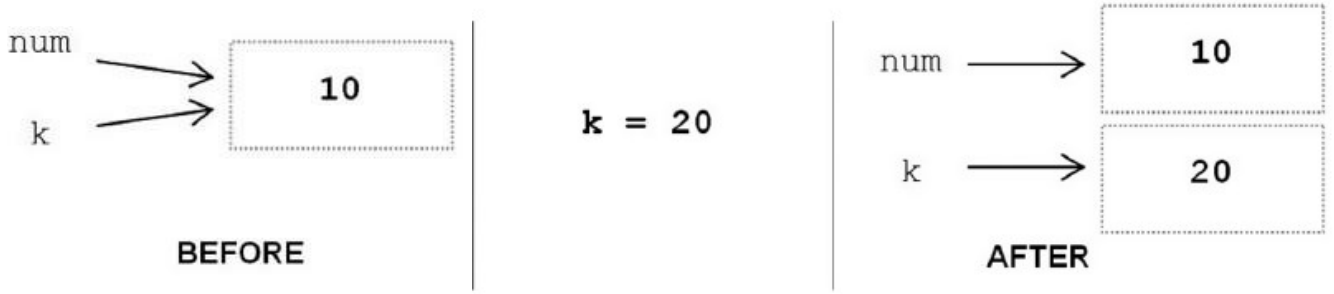
Diagrammatic Representation

- Variables may also be assigned to the value of another variable.

num = 10



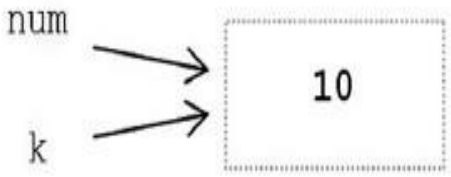
Note: Variables `num` and `k` are both associated with the same literal value 10 in memory. You can check with id()



Note: If no other variable references the memory location of the original value, the memory location is deallocated (that is, it is made available for reuse).

Diagrammatic Representation

- If the value of `num` changed, would variable `k` change along with it?



- Here variables refer to integer values, and integer values are *immutable*.
- An **immutable value** is a value that cannot be changed.
- Thus, both will continue to refer to the same value until one (or both) of them is reassigned

id function – id()

- This built-in function that returns the identity of an object.

- Commonly used to check if two variables or objects refer to the same memory location.

- The **is** keyword is used to test whether two variables belong to the same object. The test will return **True** if the two objects are the same else it will return **False.**

```
>>> num=10
>>> k=10
>>> id(num)
2863970058768
>>> id(k)
2863970058768
>>> num is k
True
>>> k=20
>>> id(num)
2863970058768
>>> id(k)
2863970059088
>>> num is k
False
```

Data Types

- Datatype refers to the type of value a variable refers to.
- Significance of data type:
  - Memory associated with it
  - Operations that can be performed on it.
  - Range of values allowed in it
- Types:
  - Scalar - Integers, floats, boolean, complex
  - Reference - List, tuple, set, dict

Type function – type()

- A built-in function, that returns the type of the object

  type(object)

- Type of a variable depends on the value assigned to it

```
a = 10
print(type(a))  #int
        a = 10.0
        print(type(a)) # float
```

THANK YOU

Introduction to Computer Science Using Python – Dierbach

Department of Computer Science and Engineering

Dr. Shylaja S S, Director, CCBD & CDSAML, PESU
Prof. Sindhu R Pai – sindhurpai@pes.edu
Prof. Chitra G M
Prof. Gayatri S