Department of Computer Science and Engineering
PES University, Bangalore, India

Lecture Notes
Python for Computational Problem Solving
UE23CS151A

Lecture #13
Variables, Data types and id

By,
Prof. Sindhu R Pai,
Anchor, PCPS - 2023
Assistant Professor
Dept. of CSE, PESU

Verified by,
PCPS Team - 2023

Many Thanks to
Dr. Shylaja S S (Director, CCBD and CDSAML Research Centers, Former Chairperson, CSE, PES University)
Prof. Chitra G M (Asst. Prof, Dept. of CSE, PCPS Anchor – 2022)

Identifiers:

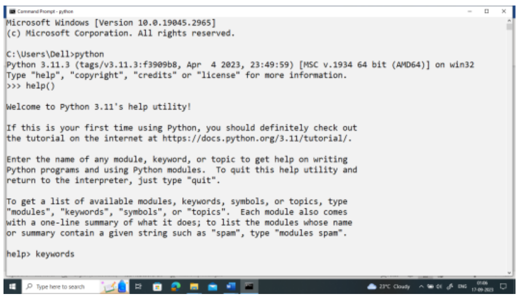
A sequence of one or more characters used to provide a name for a given program element. Python is case sensitive. Example: Line is different from line. Few rules to be followed are here:

- Identifiers may contain letters and digits but cannot begin with a digit.
- The underscore character (_) is also allowed to aid in the readability of long identifier names. However, it should not be the first character, as identifiers beginning with an underscore have special meaning in Python.

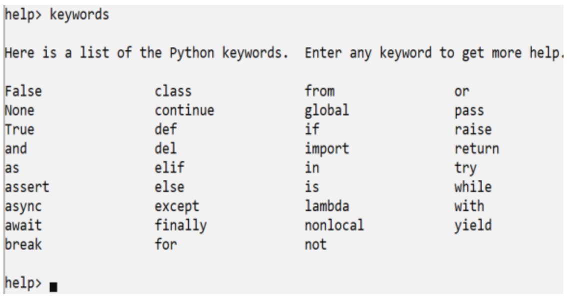
Keywords:

An identifier that has **predefined meaning in a programming language**. Therefore, keywords cannot be used as "regular" identifiers. Doing so will result in a syntax error.

How do you list the set of keywords in python?



```
Microsoft Windows [Version 10.0.19045.2965]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Dell>python
Python 3.11.3 (tags/v3.11.3:f3909b8, Apr  4 2023, 23:49:59) [MSC v.1934 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> help()

Welcome to Python 3.11's help utility!

If this is your first time using Python, you should definitely check out
the tutorial on the internet at https://docs.python.org/3.11/tutorial/.

Enter the name of any module, keyword, or topic to get help on writing
Python programs and using Python modules.  To quit this help utility and
return to the interpreter, just type "quit".

To get a list of available modules, keywords, symbols, or topics, type
"modules", "keywords", "symbols", or "topics".  Each module also comes
with a one-line summary of what it does; to list the modules whose name
or summary contain a given string such as "spam", type "modules spam".

help> 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

help> 
```

Note: Press q to come out of help page and then enter exit() / quit()

Variables:

A variable is a name (identifier) that is **associated with a value**. A variable is **created the moment you first assign a value to it.** It can be assigned different values during a program execution. Hence the name variable.

Note: Every variable in Python is a reference (a pointer) to an object and not the actual value itself. For example, the assignment statement just adds a new reference to the right-hand side.

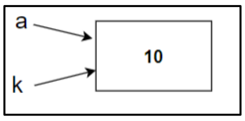
Program 1:

a = 10

k = 10

print(a)

print(k)



Output: 10

10

Program 2:

a = 10

k = a

print(a)

print(k)

Above diagram and output remains the same.

Point to think: If you change the value of k, will the value of a be changed? – Nooo!

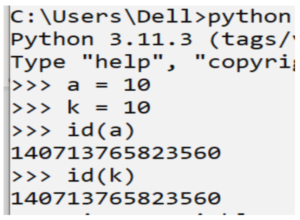
Program 3: This code results in Error as and is a keyword.

and = 10

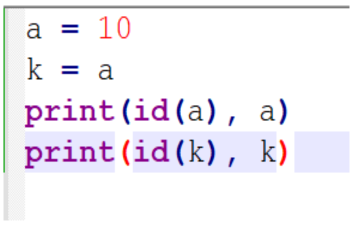
print(and)

Observation: In 1 and 2, Variables **a** and **k** are both associated with the same literal value 10 in memory. One way to prove this is by the use of **built-in function id().**

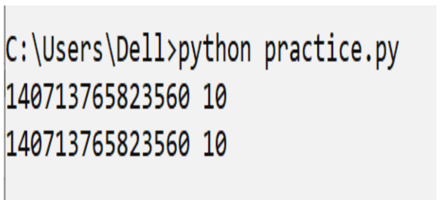
id() : Id function produces a unique number identifying a specific value (object) in memory.



```
C:\Users\Dell>python
Python 3.11.3 (tags/
Type "help", "copyri
>>> a = 10
>>> k = 10
>>> id(a)
140713765823560
>>> id(k)
140713765823560
```



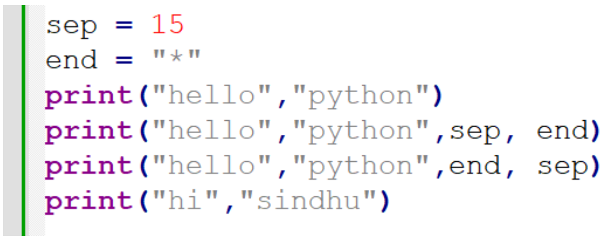
```
a = 10
k = a
print(id(a), a)
print(id(k), k)
```



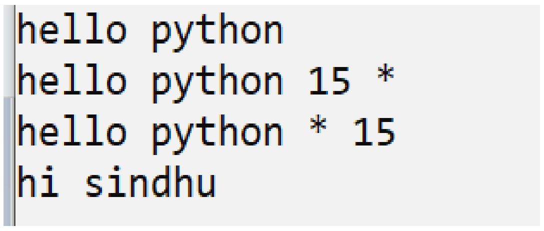
```
C:\Users\Dell>python practice.py
140713765823560 10
140713765823560 10
```

Think about it: Can we use sep and end as a variable?

Program 4:

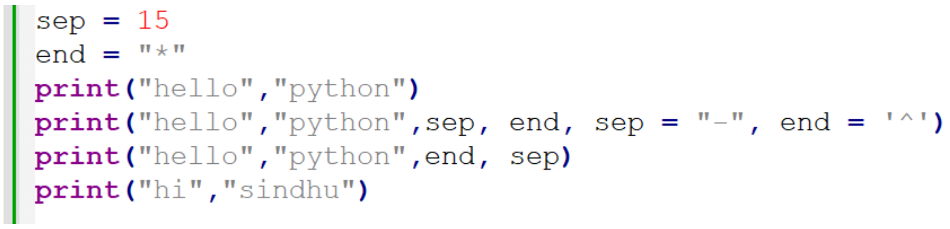


```
sep = 15
end = "*"
print("hello","python")
print("hello","python",sep, end)
print("hello","python",end, sep)
print("hi","sindhu")
```

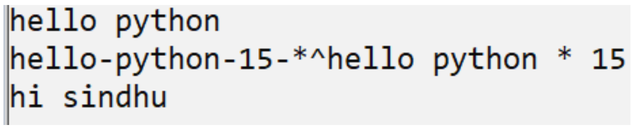


```
hello python
hello python 15 *
hello python * 15
hi sindhu
```

Program 5: usage of variable sep and end along with the keyword parameter for print



```
sep = 15
end = "*"
print("hello","python")
print("hello","python",sep, end, sep = "-", end = '^')
print("hello","python",end, sep)
print("hi","sindhu")
```



```
hello python
hello-python-15-*^hello python * 15
hi sindhu
```

Literals:

In arithmetic, we talk about values which do not change. Even in programming, we talk about entities which do not change. They are considered as they are. They are said to be constants or literals. A literal is a sequence of one or more characters that stands for itself. Literals can be categorized into: **Numeric literals and String Literals**

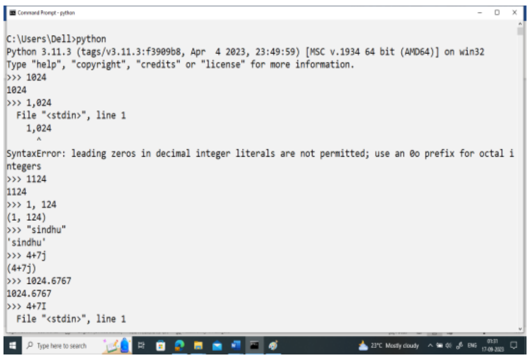
Numeric Literals:

- A literal containing only the digits 0–9, an optional sign character (+ or -), and a possible decimal point.
- If a numeric literal contains a decimal point, then it denotes a floating-point value , or " float " (e.g., 10.24); otherwise, it denotes an integer value (e.g., 10).
- **Commas are never used in numeric literal**.
- Complex numbers have a real and imaginary part.
- There is no limit to the size of an integer that can be represented in Python

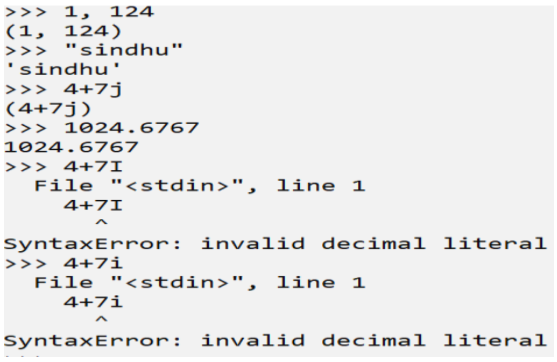
String literals or string:

- A sequence of characters denoted by a pair of matching single or double (and sometimes triple) quotes in Python.

Valid Literal examples are below.



```
C:\Users\Dell>python
Python 3.11.3 (tags/v3.11.3:f3909b8, Apr  4 2023, 23:49:59) [MSC v.1934 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> 1024
1024
>>> 1,024
  File "<stdin>", line 1
    1,024
      ^
SyntaxError: leading zeros in decimal integer literals are not permitted; use an 0o prefix for octal i
ntegers
>>> 1124
1124
>>> 1, 124
(1, 124)
>>> "sindhu"
'sindhu'
>>> 4+7j
(4+7j)
>>> 1024.6767
1024.6767
>>> 4+7I
  File "<stdin>", line 1
```



```
>>> 1, 124
(1, 124)
>>> "sindhu"
'sindhu'
>>> 4+7j
(4+7j)
>>> 1024.6767
1024.6767
>>> 4+7I
  File "<stdin>", line 1
    4+7I
      ^
SyntaxError: invalid decimal literal
>>> 4+7i
  File "<stdin>", line 1
    4+7i
      ^
SyntaxError: invalid decimal literal
```

Data Types:

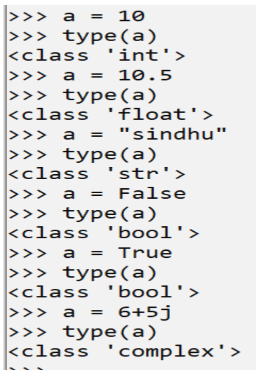
Every variable is associated with a type based on the value assigned to it. Significance of data types are listed below.

- **Range of values** allowed in a variable.
- **Memory allocation** for a variable
- **Allowed set of operations** on the data

type() in python: A built in function, which returns the type of the given object.

Categories of types: Scalar type and Reference Types

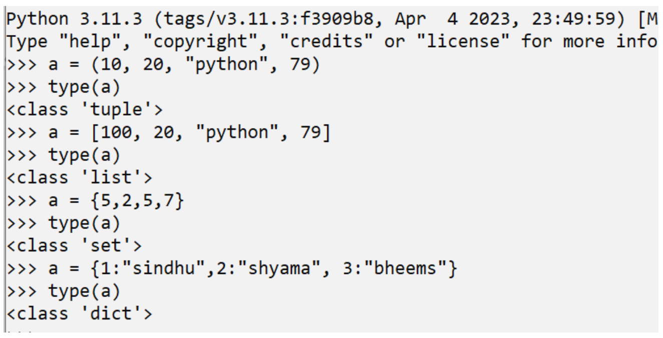
Scalar Types: Simple ones which have a single value.



```
>>> a = 10
>>> type(a)
<class 'int'>
>>> a = 10.5
>>> type(a)
<class 'float'>
>>> a = "sindhu"
>>> type(a)
<class 'str'>
>>> a = False
>>> type(a)
<class 'bool'>
>>> a = True
>>> type(a)
<class 'bool'>
>>> a = 6+5j
>>> type(a)
<class 'complex'>
```

Reference types:

Structured or reference types having more than one value.



```
Python 3.11.3 (tags/v3.11.3:f3909b8, Apr  4 2023, 23:49:59) [M
Type "help", "copyright", "credits" or "license" for more info
>>> a = (10, 20, "python", 79)
>>> type(a)
<class 'tuple'>
>>> a = [100, 20, "python", 79]
>>> type(a)
<class 'list'>
>>> a = {5,2,5,7}
>>> type(a)
<class 'set'>
>>> a = {1:"sindhu",2:"shyama", 3:"bheems"}
>>> type(a)
<class 'dict'>
```

- END -