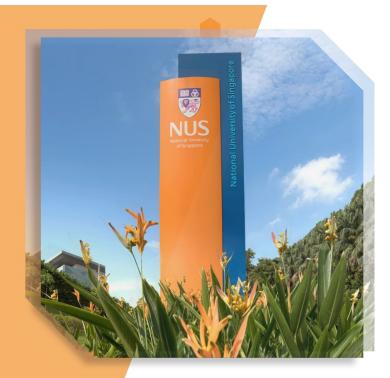
Full Stack Development With Al

Raghavendran V



Saturday, August 16, 2025 11:00 AM SGT





Variables, Data Types, and Operators in Python

Variables

- Created by assignment, dynamically typed.
- Naming: letters, digits, _, not starting with a digit. Prefer snake_case.
- Multiple assignment and swap are easy.

```
python
# assignment
name = "Raghav"
age = 30
# multiple assignment
a, b, c = 1, 2, 3
# swap
a, b = b, a
```

Common data types

- Numbers: int, float
- Text: str
- Boolean: bool (True, False)
- None: None (null-like)
- Sequences/collections: list, tuple, set, dict

python

x = 10 # int
y = 3.14 # float
s = "hello" # str
flag = True # bool
nothing = None # NoneType

Mutability

- Immutable: int, float, str, tuple
- Mutable: list, dict, set

Type checks

```
python
print(type(x))  # <class 'int'>
print(isinstance(s, str))  # True
```

Operators

- Arithmetic: + * / // % **
- Comparison: == != > < >= <=
- Logical: and, or, not
- Assignment: =, augmented +=, -=, *=
- Membership: in, not in
- Identity: is, is not

Example

arithmetic

```
python
```

```
print(7 // 2, 7 % 2, 2 ** 3) # 3 1 8
# logical & comparison
```

```
print(5 > 3 and 2 < 4)

# membership
fruits = ["apple", "banana"]</pre>
```

```
a = [1,2]; b = a
print(a is b) #
```

True

Basic Input and Output with Python

Print

python

```
name = "Raghav"
age = 30
print("Name:", name, "Age:", age)  # space-separated
print(f"{name} is {age} years old.")  # f-string (preferred)
print("{0} is {1}".format(name, age))  # format()
```

Input

python

```
# input returns string
name = input("Enter name: ")  # type: str
age = int(input("Enter age: "))  # convert to int
print(f"Next year you'll be {age+1}")
```

File I/O (context manager)

python # write with open("notes.txt", "w", encoding="utf-8") as f: f.write("Hello\nLine2\n") # read with open("notes.txt", "r", encoding="utf-8") as f: text = f.read() print(text)

Conditional Control Flow with Python

if / elif / else

```
python
score = 78
if score >= 90:
    grade = "A"
elif score >= 75:
    grade = "B"
else:
    grade = "C"
print(grade)
```

Truthy / falsy

• Falsy: 0, 0.0, "", [], {}, set(), None, False

```
python
x = []
if not x:
    print("empty")
```

Ternary expression

```
python
status = "pass" if score >= 50 else "fail"
```

Iterative Control Flow with Python

for loops

```
python
# iterate over list
for i in [1,2,3]:
    print(i)
# range
                   # 0..4
for i in range(5):
    print(i)
# enumerate, zip
names = ["a", "b"]
for idx, n in enumerate(names, start=1):
    print(idx, n)
```

```
python
python
                   for i in range(5):
n = 3
                       if i == 2:
while n > 0:
                           continue
                                         # skip
    print(n)
                       if i == 4:
    n -= 1
                           break
                                       # stop loop
                   else:
                       print("done") # executed if loop not broken
 Comprehensions (compact and fast)
```

break / continue / else

while

squares = [x*x for x in range(5)] # list comp evens = {x for x in range(10) if x % 2 == 0} # set comp mapping = {x: x*x for x in range(5)} # dict comp gen = (x*x for x in range(5)) # generator expr

Python Functions

Define and call

```
python
def greet(name):
    """Return greeting for name (docstring)."""
    return f"Hello, {name}"
print(greet("Alice"))
Default args, positional, keyword
python
                             # n has default
def power(x, n=2):
    return x ** n
print(power(3)) # 9
print(power(3, 3)) # 27
```

Variable args and kwargs	Higher-order functions
python	python
<pre>def mixed(a, *args, **kwargs): print("a:", a) print("args:", args) print("kwargs:", kwargs)</pre>	<pre>nums = [1,2,3] doubled = list(map(lambda x: x*2, nums)) filtered = list(filter(lambda x: x%2==1, nums))</pre>
mixed(1, 2, 3, name="Raghav")	
Lambda (anonymous)	Docstrings & typing (optional)
	python
<pre>add = lambda x, y: x + y print(add(2,3))</pre>	<pre>def add(a: int, b: int) -> int: """Add two integers.""" return a + b</pre>

Basic Data Structures in Python

```
List — ordered, mutable
```

```
L = [1,2,3]
L.append(4)
L[0] = 10
print(L[1:3]) # slicing
```

Tuple — ordered, immutable

python

python

```
t = (1, 2, 3)
a, b, c = t # unpacking
```

```
Set — unordered unique elements
                                             Useful conversions
python
                                             python
s = \{1, 2, 3, 2\}
                                             lst = list(range(5))
s.add(4)
                                             [0,1,2,3,4]
print(s) # duplicates removed
                                             tup = tuple(lst)
# set ops
                                             st = set(1st)
print(\{1,2\} \mid \{2,3\}) # union
Dictionary — key-value store
python
d = {"name": "Raghav", "age": 30}
print(d["name"])
d["city"] = "Mysore"
for k, v in d.items():
    print(k, v)
# safe get
print(d.get("salary", 0))
```

LIVE SESSION EXPERIENCE SURVEY

Before we proceed to Q&A

Take 2 minutes to share your feedback with Us!

