



OBJECTS AND VARIABLES

PYTHON BASICS

prepared by:

Gyro A. Madrona

Electronics Engineer

TOPIC OUTLINE

Objects

Variables

Binding Objects and Variables



OBJECTS



OBJECTS

In Python, everything is an object. Objects have a type that defines the kinds of operations programs can perform on it.



SCALAR OBJECTS

Scalar objects represent a single, indivisible value and cannot be modified after creation (immutable).

int: represents integers (e.g., 5)

float: represents real numbers (e.g., 3.5)

bool: represents Boolean values **True** or **False**

str: represents a sequence of characters

NoneType: special and has one value, **None**

The **type()** function returns the type (class) of an object.

type(5) returns **int**

type(3.5) returns **float**

Type conversion (**casting**) converts an object of one type to another.

float(3) converts integer 3 to float 3.0

int(3.9) truncates float 3.9 to integer 3



EXERCISE

Find the type of the following:

25

2.83

10.0

True

False

None

'Hello World!'

`float(25)`

`int(2.83)`

`round(2.83)`

`float(round(2.83))`



EXPRESSIONS

Expression is a combination of objects and operators that evaluates to a single value and has a type.

Operators on `ints` and `floats`

`i + j` → the sum

`i - j` → the difference

`i * j` → the product

`i/j` → the division

`i%j` → the remainder when `i` is divided by `j`

`i**j` → `i` to the power of `j`

Operator precedence

(1) Power (`**`)

(2) Multiplication (`*`)

Division (`/`)

Floor Division (`//`)

Modulo (`%`)

(3) Addition (`+`)

Subtraction (`-`)

note : parentheses override precedence



EXERCISE

Evaluate the following expressions:

$1+2**2$

$(1+2)**2$

$5*4/5$

$5/4*5$

$5+4-5$

$5-4+5$

Operator precedence

(1) Power ($**$)

(2) Multiplication ($*$)

Division ($/$)

Floor Division ($//$)

Modulo ($\%$)

(3) Addition ($+$)

Subtraction ($-$)

note : parentheses override precedence



VARIABLES



FRUIT CONTAINER ANALOGY

A variable is like a storage container that holds a specific type of fruit.

location	quantity	label
1		apple
2	400	
3		
4	200	orange
5		
.		
.		
.		
.		
1000	100	grape



VARIABLES

A variable is a name (identifier) that refers to an object. Unlike some other languages, Python variables **do not store objects directly** – they **bind** (point) to objects.

address	value	name
0001h		apple
0002h	400	
0003h		orange
0004h	200	
0005h		
.		
.		grape
.		
.		
.		
FFFFh	100	



BINDING VARIABLES AND VALUES

Assignment operator (=) binds the variable to a value from right-to-left.

syntax

name = value

example

apple = 400

address	value	name
0001h		
0002h	400	← apple
0003h		
0004h		
0005h		
.		
.		
.		
.		
.		
FFFFh		



BINDING VARIABLES AND VALUES

Assignment operator (=) binds the variable to a value from right-to-left.

syntax

name = value

example

apple = 400

apple = 20

address	value	name
0001h		
0002h	400	
0003h		
0004h		
0005h		
.		
.		
.		
.		
.		
FFFFh		



BINDING VARIABLES AND VALUES

Assignment operator (=) binds the variable to a value from right-to-left.

syntax

name = value

example

apple = 400

apple = 20

address	value	name
0001h		
0002h	400	
0003h		
0004h		
0005h		
.	20	
.		
.		
.		
.		
FFFFh		



BINDING VARIABLES AND VALUES

Assignment operator (=) binds the variable to a value from right-to-left.

syntax

name = value

example

apple = 400

apple = 20

Python dynamically infers variable types based on assigned values without requiring explicit declarations.

address	value	name
0001h		
0002h	400	
0003h		
0004h		
0005h		
.	20	← apple
.		
.		
.		
.		
FFFFh		



IDENTIFIER

Variable is identified by a unique name, called an identifier.

- It can contain letters, digits, and underscores.
- It cannot have “space”.
- It cannot start with a digit.
- It cannot be a reserved keyword (**int**, **return**, **class**).
- It is case-sensitive (e.g., **age** and **Age** are different variables)

valid identifier

contains only letters

age

starts with an underscore

_salary

contains letters and a digit

grade1

uses an underscore instead of space

total_price



IDENTIFIER

Variable is identified by a unique name, called an identifier.

- It can contain letters, digits, and underscores.
- It cannot have “space”.
- It cannot start with a digit.
- It cannot be a reserved keyword (**int**, **return**, **class**).
- It is case-sensitive (e.g., **age** and **Age** are different variables)

invalid identifier

starts with a digit

1stRank

uses a reserved keyword

class

contains a space

total price



EXERCISE

Which of these are allowed in Python?

`x = 6`

`6 = x`

`x*y = 3+4`

`xy = 3+4`

`try = 'me'`

`_try = 'me'`



LABORATORY

