Data Types

Data Types

integers	(whole numbers)
floats	(decimals)
strings	(letters)
booleans	(True/False)
None	(placeholder objects)

Integer/Float Operations

Add	+
Subtract	-
Multiply	*
Divide	/
Modulo	% (for integer only - returns remainder of division)

Integer/Float Order of Operations

PEMDAS - Parentheses, Exponents, Multiplication & Division, Addition & Subtraction
or “Please Excuse My Dear Aunt Sally”

Mixing Integers and Floats

int (op) int → int
int (op) float → float
float (op) int → float
float (op) float → float

Booleans (True or False)

is equal to	==
is NOT equal to	!=

Strings (written characters)

Single Quotes	‘ ‘
Double Quotes	“ “
Concatenation	+
Repeat a String	*

None (placeholder)
check if something is None via “is”

Casting
int() convert to integer
float() convert to float
str() convert to string

Print
print()

Escape Characters (for strings)
' \n ' new line
' \t ' tab
' \' ' single quote (for comma, apostrophe, etc.)
‘ \" ‘ double quote
' \\ ' backslash

Variables

Variable Assignment

x = 1

Naming Variables

Can only contain letters (a–z, A–Z), digits (0–9), and the underscore _

CANNOT begin with numbers

CANNOT contain spaces or other symbols

CANNOT use Python keywords (ie. True, False, None, is, not, and, or, for, if, elif, else, def)

Naming Conventions

camelCase

snake_case

Reassignment

x = x + 1

Assignment Operations

+= for example: **x += 1** → **x = x + 1**

-= for example: **x -= 2** → **x = x - 2**

*= for example: **x *= 3** → **x = x * 3**

/= for example: **x /= 4** → **x = x / 4**

%= for example: **x += 5** → **x = x + 5**

Get user input

raw_input()

Conditionals

Comparing Numbers (integers and floats)

< less than

<= less than or equal to

> greater than

>= greater than or equal to

== is equal to

!= is not equal to

is is identical to

is not is not identical to

Comparing Booleans

== is equal to

!= is not equal to

is is identical to

is not is not identical to

Comparing Strings

== is equal to

!= is not equal to

is is identical to

is not is not identical to

Boolean Logical Operators

not

- not True → False
- not False → True

and

- True and True → True
- True and False → False
- False and True → False
- False and False → False

or

- True or True → True
- True or False → True
- False or True → True
- False or False → False

Boolean Order of Operations

1. ()
2. not
3. <, <=, >, >=, ==, !=, is, is not, math operations
4. and
5. or

Conditional Statements

if, elif, else

Example 1:

```
if <condition>:
print( … )
```

Example 2:

```
if <condition>:
print( … )
else:
print( … )
```

Example 3:

```
if <condition>:
print( … )
elif:
print( … )
```

Example 4:

```
if <condition>:
print( … )
elif:
print( … )
else:
print( … )
```

Functions

Defining a Function Calling a Function

def function_name() function_name() → ‘‘Hello World!”

print(“Hello World!”)

Return Statement Saving the Return

def get_name() class = get_name()

return “UNIST” print(class) → ‘‘UNIST”

Arguments

def greet(name)

print(“Hello “ + name)

greet(“STEM”) → ‘‘Hello STEM”

greet(“there’’)→ ‘‘Hello there”

Multiple Arguments

def greet(name1, name2)

print(“Hello ” + name1 + “ and ” + name2)

greet(“Daniel”, “Dan”) → ‘‘Hello Daniel and Dan”

greet(“Alex”, “Wendy”) → ‘‘Hello Alex and Wendy”

greet(“Nishanth”, “Sabina”) → ‘‘Hello Nishanth and Sabina”

Calling a Function from a Function

def square(num):

return num*num

def sum_of_squares(x, y, z):

return square(x) + square(y) + square(z)

print(sum_of_squares(1, 2, 3)) → 10

print(sum_of_squares(2, 4, 6)) → 56

Data Structures

String Indexing

String indexing starts at 0 (not 1)!!!

R	A	C	E	C	A	R
0	1	2	3	4	5	6

Example

my_str = “RACECAR”

my_str[3] → “E”

Negative Indexing

String indexing starts backwards at -1 (not 0)!!!

R	A	C	E	C	A	R
-7	-6	-5	-4	-3	-2	-1

Example

my_str = “RACECAR”

my_str[-2] → “A”

Slicing

<string>[start : stop : step]

start (inclusive): start index

end (exclusive): end index + 1

Example

s = “UNIST STEM Camp”

s[6:11:1] → “STEM ”

s[0:5:2] → “UIT”

Negative Slicing

<string>[start : stop : step]

start (inclusive): start index

end (exclusive): end index + 1

Example

s = “UNIST STEM Camp”

s[-1:-5:-1] → “pmaC”

s[-4:-12:-2] → “CMT”

Slicing Default Values

default start: 0

defaut end: string length

default step: 1

Example

s = “UNIST STEM Camp”

s[:] → “UNIST STEM Camp”

s[: : 2] → “UTISE ap”

Membership Operators

in	for example: “r” in “racecar”	→ returns True
not in	for example: “b” not in “racecar”	→ returns False

String Length

len(“Hello World”) → 11

Lists

fruits = [“apple”, “banana”, “pear”]

indexing:	fruits[0]	→ ‘‘apple”
slicing:	fruits[0:2]	→ ‘‘apple”, “banana”
in, not in:	‘‘apple” in [“apple”, “banana”]	→ True
length:	len(fruits)	→ 3

List append (add item to the end of a list)

fruits = [“apple”, “banana”, “pear”]

fruits.append(“orange”)

print(fruits) → [“apple”, “banana”, “pear”, “orange”]

List pop (remove item at specified index)

fruits = [“apple”, “banana”, “pear”]

fruits.pop(2)

print(fruits) → [“apple”, “banana”]

List Functions

sum(<list_name>)

max(<list_name>)

min(<list_name>)

sorted(<list_name>)

Adding Lists

[0] + [1] → [0, 1]

Multiplying Lists

[0]*4 → [0, 0, 0, 0]

Lists Within Lists

misc = [“apple”, 3.14, [1, 2, 3, 4]]

print(misc[2]) → [1, 2, 3, 4]

print(misc[2][1]) → 2

Loops

Iterables (objects that can be indexed, or looped over)

strings

loops

while loops

while <condition>:

    ...

Examples

apples = 0 → Prints 1, 2, 3

while apples < 3:

    apples += 1

    print(apples)

for loops

for var in <iterable>:

    …

Examples

for var in [1, 2, 3, 4]: → Prints 1, 2, 3, 4

    print(var)

for char in “banana”: → Prints ‘b’, ‘a’, ‘n’,

    print(char) ‘a’, ‘n’, ‘a’

Infinite Loops

while True: → Prints “hi” forever :(

    print(“hi”)

Create lists with range()

range(start, stop, step)

range(0, 5, 1) → [0, 1, 2, 3, 4]

range(6, 10, 2) → [6, 8]

More Python Resources

Online Python 3 → Want to practice coding at home? Do it here!
**** Make sure to select "Python 3" in "Language" ****
https://www.onlinegdb.com/

Python Like You Mean It → Want to learn more about python? Here's an AMAZING Python Tutorial!
https://www.pythonlikeyoumeanit.com/

Coding Bat → Practice your newfound python skills here!
https://codingbat.com/python

Stack Overflow → Got coding questions? Other people do too! Ask them here :)
**** Type your question in the "Search" box above! ****
https://stackoverflow.com/