2.0 Introduction to Python 3

input('Prompt for user')
\n
Identifiers:

- sequence of letters (a-z, A-Z)
- *underscores* (_)
- digits (0–9)
- Must start with a letter or an underscore.

Properties of objects

1. **Value**: A value such as "20", "abcdef", or 55.
2. **Type**: The type of the object, such as integer or string.
3. **Identity**: A unique identifier that describes the object.
    a. **id()**

Numeric types: Floating-point

Floating-point number: A floating-point number is a real number, like 98.6, 0.0001, or -666.667.
Floating-point literal: A floating-point literal is written with the fractional part even if that fraction is 0, as in 1.0, 0.0, or 99.0.
Scientific notation: A floating-point literal using scientific notation is written using an e preceding the power-of-10 exponent, as in 6.02e23 to represent 6.02x1023.
Overflow: Overflow occurs when a value is too large to be stored in the memory allocated by the interpreter.

- **OverflowError**: Assigning a floating point value outside of this range generates an

3.8 Module basics

Script: Programmers typically write Python program code in a file called a script.
Module: A module is a file containing Python code that can be used by other modules or scripts.
Import: A module is made available for use via the import statement.
dot notation: Once a module is imported, any object defined in that module can be accessed using dot notation.
__name__ :Python programs often use the built-in special name __name__ to determine if the file was executed as a script by the programmer, or if the file was imported by another module.
Math modules: ceil(), floor(), sqrt(), exp(), pow(), and factorial()

Arithmetic operators.

Arithmetic operator	Description
+	The addition operator is +, as in x + y.
-	The subtraction operator is -, as in x - y. Also, the - operator is for negation, as in -x + y, or x + -y.
*	The multiplication operator is *, as in x * y.
/	The division operator is /, as in x / y.
**	The exponent operator is **, as in x ** y (x to the power of y).

Precedence rules for arithmetic operators.

Operator/Convention	Description	Explanation
( )	Items within parentheses are evaluated first.	In 2 * (x + 1), the x + 1 is evaluated first, with the result then multiplied by 2.
unary -	- used for negation (unary minus) is next.	In 2 * -x, the -x is computed first, with the result then multiplied by 2.
* / %	Next to be evaluated are *, /, and %, having equal precedence.	(% is discussed elsewhere.)
+ -	Finally come + and - with equal precedence.	In y = 3 + 2 * x, the 2 * x is evaluated first, with the result then added to 3, because * has higher precedence than +. Spacing doesn't matter: y = 3+2 * x would still evaluate 2 * x first.
left-to-right	If more than one operator of equal precedence could be evaluated, evaluation occurs left to right.	In y = x * 2 / 3, the x * 2 is first evaluated, with the result then divided by 3.

Operator/Convention	Description	Explanation
( )	Items within parentheses are evaluated first	In (a * (b + c)) - d, the + is evaluated first, then *, then -.
* / % + -	Arithmetic operators (using their precedence rules; see earlier section)	z - 45 * y < 53 evaluates * first, then -, then <.
< <= > >= == !=	Relational, (in)equality, and membership operators	x < 2 or x >= 10 is evaluated as (x < 2) or (x >= 10) because < and >= have precedence over or.
not	not (logical NOT)	not x or y is evaluated as (not x) or y
and	Logical AND	x == 5 or y == 10 and z != 10 is evaluated as (x == 5) or ((y == 10) and (z != 10)) because and has precedence over or.
or	Logical OR	x == 7 or x < 2 is evaluated as (x == 7) or (x < 2) because < and == have precedence over or

Common escape sequences.

Escape Sequence	Explanation	Example code	Output
\\	Backslash (\)	`print('\\home\\users\\')`	`\home\users\`
\'	Single quote (')	`print('Name: John O\'Donald')`	`Name: John O'Donald`

\"	Double quote (")	`print("He said, \"Hello friend!\".")`	`He said, "Hello friend!".`

\n	Newline	`print('My name...\nIs John...')`	`My name...`
			`Is John...`

\t	Tab (indent)	`print('1. Bake cookies\n\t1.1. Preheat oven')`	`1. Bake cookies`
			`    1.1. Preheat oven`

Common data types.

Type	Notes
int()	Numeric type: Used for variable-width integers.
float()	Numeric type: Used for floating-point numbers.

Containers: sequence and mapping types.

Type	Notes
string()	Sequence type: Used for text.
list[]	Sequence type: A mutable container with ordered elements. - Can be indexed
tuple()	Sequence type: An immutable container with ordered elements. - Can be indexed
dict{}	Mapping type: A container with key-values associated elements. - Cannot be indexed

Choosing a container type

List: when data has an order, such as lines of text on a page
Tuple: if the contained data should not change
Dictionary: If order is not important, a programmer might use a dictionary to capture relationships between elements, such as student names and grades.

List basics

Adding elements to a list:

- list.append(value): Adds value to the end of list. Ex: `my_list.append('abc')`

Removing elements from a list:

- list.pop(i): Removes the element at index i from list. Ex: `my_list.pop(1)`
- list.remove(v): Removes the first element whose value is v. Ex: `my_list.remove('abc')`

Sequence-type methods and functions

Operation	Description
len(list)	Find the length of the list.
list1 + list2	Produce a new list by concatenating list2 to the end of list1.
min(list)	Find the element in list with the smallest value.
max(list)	Find the element in list with the largest value.
sum(list)	Find the sum of all elements of a list (numbers only).
list.index(val)	Find the index of the first element in list whose value matches val.
list.count(val)	Count the number of occurrences of the value val in list.

Tuples

Common error types.

Error type	Description	Examples
SyntaxError	The program contains invalid code that cannot be understood.	print('Today is Monday")
IndentationError	The lines of the program are not properly indented.	print("Friday, Friday")
ValueError	An invalid value is used – can occur if giving letters to int().	int("Thursday")
NameError	The program tries to use a variable that	day_of_the_week = Friday

	does not exist.	
TypeError	An operation uses incorrect types – can occur if adding an integer to a string.	lyric = 99 + " bottles of pop on the wall"