Python as a Calculator

Table of contents

See csse1001[1] for course logistics — this note covers Lecture 1B's technical content.

Today's outline

- The Python REPL
- Syntax vs. semantics
- Arithmetic expressions, order of operations, and associativity
- Variables
- Worked exam questions

Learning objectives

1. Python expressions have syntax and semantics.
2. Python can evaluate arithmetic expressions, but this requires an order of operations and rules for associativity.

[1] courses/csse1001/csse1001.pdf

The Python REPL

A REPL (read-evaluate-print loop) is an interactive environment: it *reads* input, *evaluates* it, *prints* the result, then *loops*.

```
>>> 2*3
6
>>>
```

The `>>>` prompt is only shown in the interactive REPL — it isn't included when writing Python instructions into files.

Syntax vs. semantics

A programming language like Python has both:

- **Syntax** — defines what it means to be a *valid program*.
- **Semantics** — defines what a valid program *actually does*.

`2 + 3` is syntactically correct Python and evaluates to `5`. Prefix notation like `+ 2 3` (valid in Lisp) is *not* valid Python syntax:

```
>>> + 2 3
    + 2 3
      ^
SyntaxError: invalid syntax
```

Python's language rules must be followed *precisely* for a program to run.

Arithmetic expressions

See python-operator-precedence-and-associativity[2] for the full syntax/semantics rules for brackets, negation/affirmation, and the arithmetic operators (`+ - * / // % **`), the order-of-operations table, associativity (including why `**` is right-associative), integer division, and the `^` xor gotcha.

[2] courses/csse1001/python-operator-precedence-and-associativity.pdf

Worked exam questions

Q: What does `2 ** 3 % 5 - 1` evaluate to? `((2 ** 3) % 5) - 1 = (8 % 5) - 1 = 3 - 1 = 2`

Q: What value gets assigned to `x` in `x = 35 * 2 % 5 ** 2`? `(35 * 2) % (5 ** 2) = 70 % 25 = 20` — 60% of the class answered correctly.

Variables

Values can be named — these names are called **variables**. Assigning a value to a variable doesn't print anything in the REPL:

```
>>> width = 2
>>> height = 3
>>> area = width * height
>>> 4 * area
24
```

`min` and `max`

```
>>> max(1, 2)
2
>>> min(1, 2)
1
>>> max(5, 1, -2, 3, 2*7)
14
```

Exercise: second largest of three numbers

Given three distinct numbers `a`, `b`, `c`, write an expression for the *second largest*:

```
>>> a, b, c = -9, 19, 3
>>> a + b + c - max(a, b, c) - min(a, b, c)
3
>>> min( max(a, b), max(a, c), max(b, c) )
3
>>> max( min(a, b), min(a, c), min(b, c) )
3
```

Next lecture

1. 2025-08-04-python-memory-model[3] — Python memory model.
2. 2025-08-04-primitive-data[4] — Primitive data types.

[3] courses/csse1001/2025-08-04-python-memory-model.pdf
[4] courses/csse1001/2025-08-04-primitive-data.pdf