Intro to Python (Class 2)

Ben Bettisworth

Section 1

Values

Values

- As we saw in the previous class, programs operate on the state of the computer to produce a new state
- The collections of memory that a program operates on are called "Values".
  - `3` is a value
  - `3.14` is a value
  - `"foo"` is a value
  - $1 + 2i$ is a value
  - `None` is a value

Operations on Values

- Each instruction in a program can be thought of as "take these values, and perform some operation to them"
- Examples
  - `3 + 2` means take the values 3 and 2 and add them together
  - `3.14 * 2` means take the value 3.14 and double it.
  - `"ice" + "cream"` means take the values `ice` and `cream` and concatenate them to make `icecream`
  - `(1 + 2j) * (1 - 2j)` means take the values $1 + 2i$ and $1 - 2i$ and multiply them

Types

- You might have noticed in the previous examples that + is used in both:
  - `3 + 2`, and
  - `"ice" + "cream"`.
- But these are not the same operation!
- How does python know what to do?

Types

- Values in python have a “Type”, which indicates what operations are valid, and what those operations do.
- Examples of types are:
  - Integer (`3`)
  - Float (`3.14`)
  - Complex Number (`1+1j`)
  - String (`"foo"`)
  - Boolean(`True`)
  - Null value (`None`)

Exercise

Try the following operations in python:

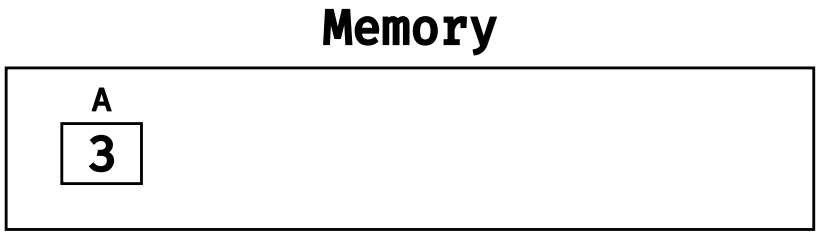
- `"foo" * 2`
- `3 / 2`
- `3 // 2`
- `3 % 2`
- `3 ** 2`

Can you think of other operations that would make sense? Try them!

Section 2

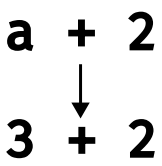
Variables

Variables



- A variable is a “box” that a value is stored in.
- For example, `a = 3` stores the value 3 in a box with the label a

Variables



- When used in place of a value, a variable acts as if it has the value in it's box
- a + 2 is the same as 3 + 2

Variables

- In python, variables are created with the "assignment operator" `=`
- `a = 3` or `foo = "bar"`
- Variables name have some requirements:
  - Must start with a letter;
  - Must not contain punctuation (`.`, `+`, `/`, `%`, `-`, etc.)
    - Exception is `_`, which is allowed;
  - Can't be a keyword (e.g. `for`, `None`, etc.);
  - And no spaces.

Variable Name Examples

- `a`
- `foo`
- `hereIsACleverName`
- `biology_rules`
- `physics_drools`
- `a1`
- `x2x`

Assignment

- Properties of assignment
  - Transitive `a = b = c`
  - non-associative `a = b; b = c` is not the same as `b = c; a = b`
  - non-communicative `a = b` is not the same as `b = a` (same for a=b=c)

Exercise

Demonstrate that the following properties hold for assignment (Write an example): - Transitivity (`a = b = c`) And demonstrate that the following properties do *not* hold: - Associativity (`a = (b = c)`) - But compare (`a = (b := c)`) - Communication (`a = b = c` vs `b = a = c`)

Section 3

Collections

Collections

- In addition to the "primitive" types (`int`, `float`, `str`), there are more complex types.
- Example, a list of values:
  - `[1,2,3]`
- A list is a way to store a collection of values, with an order.