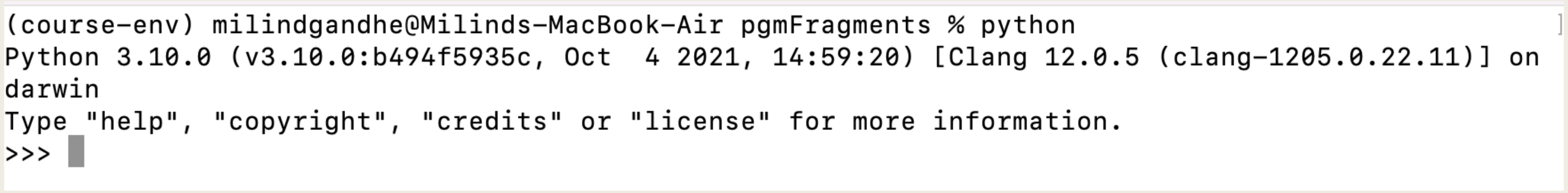
INTRODUCTION TO PROGRAMMING USING PYTHON

ES 112

A First Glance at Python

- First run an interpreter: also called a shell



```
(course-env) milindgandhe@Milinds-MacBook-Air pgmFragments % python
Python 3.10.0 (v3.10.0:b494f5935c, Oct  4 2021, 14:59:20) [Clang 12.0.5 (clang-1205.0.22.11)] on
darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> 
```

- The Python Interpreter is in a REPL loop
  - *Read-Evaluate-Print-Loop*
- Type a python program (script) at the prompt >>>
  - *Expression*
  - *Statement*
  - *Definitions*

Data

- All data in Python is stored as an “object”
  - *Address or location in memory (also called id)*
  - *Type*
  - *Value*
- Example
  - `5`
  - `3.1416`
  - `True`
  - `None`

Basic Constructs in Python

- Objects
  - *Scalar : indivisible like atoms*
  - *Non-scalar : internal structure*
- Scalar Objects
  - `int`
  - `float`
  - `bool`
  - `None`

Expressions in Python

- Objects and operators are combined into expression
- Operators
  - *Operators on types* `int` *and* `float`, *result is either* `int` *or* `float`*:* `+, -, *, //, %, /,**`
  - *Operators on type* `bool`, *result is* `bool` *:* `and, or, not`
  - *Operators on type* `int` *and* `float`*, result is* `bool`*:* `==, !=, >, <, >=, <=`
- An expression evaluates to an object
  - *Every object has a type*
- Arithmetic operators have the standard precedence

Examples of Expressions

- `int` expressions
  - `2 + 3`
  - `3 * 2`
  - `3 // 2` *but not* `3 / 2`
- `float` expressions
  - `2.0 + 3.5`
  - `2 + 3.5`
  - `3.4 * 2`
  - `3 / 2`
  - `3.6 / 1.8`
  - `3.6 // 1.8`
- `bool` expressions
  - `2 > 3`
  - `3.5 < 5.6`
- `Bool` expressions (contd)
  - `3.5 < (2.6 + 3)`
  - `True and 2 > 3`
  - `False or 3.5 < 5.6`
  - `not 2 > 3`
- `None` expressions
  - `None`
  - `None or False`
  - *what about*
    - `False or None`
    - `3 + None`
    - `None + 3`

Examples of Operator Precedence

- `3 + 5 * 6`
- `(3 + 5) * 6`

- `3.5 < (2.6 + 3)`
- `3.5 < 2.6 + 3`
- `(3.5 < 2.6) + 3`

- `not (3 == 2) or (4.0 >= 3)`
- `(not (3 == 2)) or (4.0 >= 3)`
- `not ((3 == 2) or (4.0 >= 3))`

Variables and Assignment

- Variables associate names with objects
  - *Variable is just a name*
- Assignment
  - *Remember each object has an address in memory*
  - *Assignment "**binds**" a name to that address*

```
pi = 3.14
radius = 11
diameter = radius * 2
```

More About Assignments

- Assignments and Types
  - *Variables take on the type of the expression on the right*

```
variable1 = 42

variable2 = 42.0

type (variable1)

type(variable2)
```

- Assignments are **not** math equations!

```
pi = 3

radius = 11

area = pi * (radius ** 2)

pi = 3.14
```

- Will the value of `area` change?
  - *What will make the value of* `area` *change?*
- What happens to the type of `pi`?

More About Bool Operators

A	B	A and B	A or B
False	False	False	False
False	True	False	True
True	False	False	True
True	True	True	True

A	not A
False	True
True	False

- `True` and `False` have first letter capital
- `true` and `false` are not `Bool` values
- What happens if we use non `Bool` values in `and` or `or`

Understanding Programming Languages

- Programming languages are like English!
  - *English uses*
    - words: “cat”, ”dog”, ”green”, “cheese”
    - phrases: “green cheese”, “red cat”, “jumped over the fence”
    - sentences: “the moon is made of green cheese”
  - *Python uses*
    - Literals: `36, 3.1416, ’the cat’`
    - Operators: `+, -, <, >=`
    - Variables`: nameOfCat, materialofMoon`
    - Expressions`: 3.1416 * (radius ** 2)`
    - Statements`: materialOfMoon = ‘green cheese’`

What Can Go Wrong?

- English
  - *Grammatically incorrect:*
    - The moon green cheese
  - *Grammatically inconsistent*
    - The moon are made of green cheese
  - *No meaning in real life*
    - The moon is made of green cheese

- Python
  - *Syntax error*
    - `materialOfMoon 3.1416`
  - *Static (semantic) error*
    - `‘green cheese’ / 3.1416`
  - *Dynamic Error*
    - `circumference = pi * (diameter ** 2)`
    - `materialOfMoon = ‘green cheese’ materialOfMoon / 3.1416`

Would a Rose by any other Name Smell as Sweet?

```
x = 3.1416
y = 11
z = x * (y**2)
```

```
pi = 3.1416
radius = 11
circumference = pi * (radius **2)
```

```
pi = 3.1416
diameter = 11
circumference = pi * (diameter **2)
```

Variable names matter!!!

Augmented Assignments

- You can assign multiple variables simultaneously!!
- All expressions on the right are evaluated before any bindings are made

```
x = 1
y = 2
x, y = y, x
```

- Combining operators and assignment

```
x = x + 1
x += 1
```

- = can be combined with several operators
  - *experiment yourself*
- Warning: += can sometimes have strange outcomes!! More on this later

Reserved Words in Python

- and
- as
- assert
- break
- class
- continue
- def
- del
- elif
- else
- except
- False
- finally
- for
- from
- global
- if
- import
- in
- is
- lambda
- nonlocal
- None
- not
- or
- pass
- raise
- return
- True
- try
- while
- with
- yield

Do not use these words as variable names!