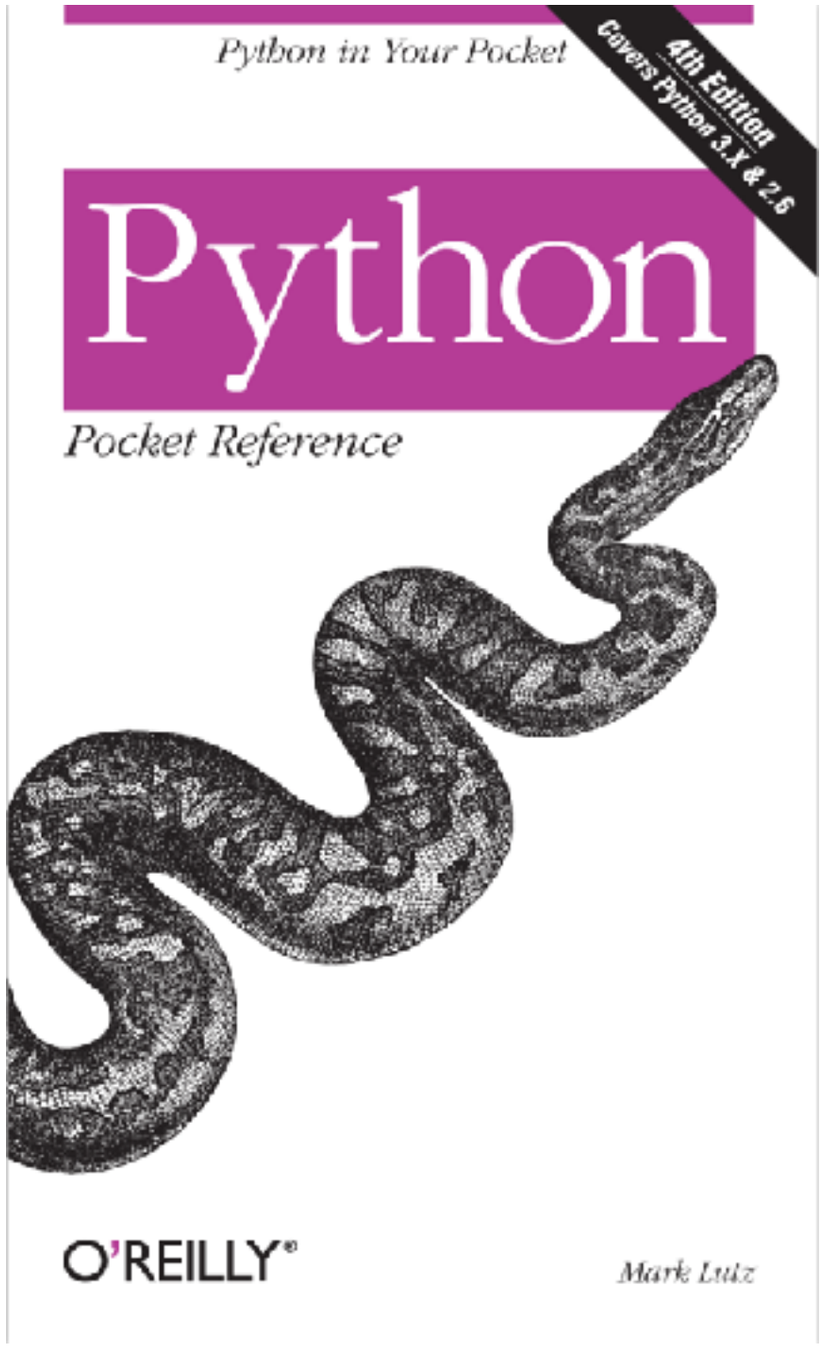
Intro to Python3

Venkatesh-Prasad Ranganath
Kansas State University

Python

- General-purpose
- Object-oriented
- Dynamically Typed
- Strong Typing
- Interpreted



Builtin Data Types

- Boolean
  - `True, False`
- Integer
  - `1234, -24, 0`
- Float
  - `3.14, 314e-2, 1., -.1`

Builtin Data Types

- String
  - Immutable
  - `“Python’s”`
  - `‘she said “Python”’`
  - `“””String with newline character”””`
  - `“This” “is” “one” “string” “literal”`
  - Operators — let’s look at scratch_pad.py
  - Methods — for you to explore :)

Operators

- `X <= Y`
- `X < Y`
- `X >= Y`
- `X > Y`
- `X == Y`
- `X != Y`
- `X is Y`
- `X is not Y`
- `X < Y < Z`
- `not X`
- `X and Y`
- `X or Y`

Control Structure

- `if-else` or `if-elif-else`
- `while-else`
- `for-else`
- `continue`
- `break`
- `try-except-else-finally`
- `pass` (basically a no-operation statement)
- `raise` (similar to `throw` in Java/C#)

- `:` is used to terminate the "header" of control structures
- Indentation is used mark-off code blocks
- Don't mix tabs and spaces in indentation
- Use consistent indentation

Functions

`def` <function-name>**`(`**?<param-list>**`):`**
  <body>

- Indentation is used to mark-off code blocks
- **`return`** ?<value> is used to return from functions
- Call-by-reference (for non-simple data types)
- Supports default arguments

Complex Data Types

- Tuples - `(2, 4)`
- Lists - `[2, 4]`
- Dictionaries - `{2 : 4}`
- Sets - `set('spam')`
- Supports comprehension/generator expressions

Classes

```
class <class-name>(<parent-class>):
    <attribute-def>
    <function-def>
```

- Supports “constructor” method (`__init__`)
- Supports both class and instance attributes (fields)
- Supports both static and instance methods
- Does not support for access specifiers
- Supports inheritance

Putting it all together

- `from x import y`
- `import y`