An Introduction to Python (A One-Hour Tour)

by Binh Q. Nguyen
NOTICES

Disclaimers

The findings in this report are not to be construed as an official Department of the Army position unless so designated by other authorized documents.

Citation of manufacturer’s or trade names does not constitute an official endorsement or approval of the use thereof.

Destroy this report when it is no longer needed. Do not return it to the originator.
An Introduction to Python (A One-Hour Tour)

Binh Q. Nguyen
Computational and Information Sciences Directorate, ARL
An Introduction to Python (A One-Hour Tour)

This tutorial highlights and goes over essential features of the Python programming language while it is still evolving, but sufficiently stable and mature for the development of diverse solutions to computational, networking, and visualization problems. Although the technical details are kept to a minimum to fit diverse background and interests of the audience, they can be used as review materials for experienced and occasional developers of Python applications.

The tutorial was presented to a team of engineers, scientists, and summer students on Wednesday 18 June 2008 at the U.S. Army Research Laboratory in Adelphi, MD.

The Python programming language, tutorial

<table>
<thead>
<tr>
<th>16. Security Classification of:</th>
<th>17. LIMITATION OF ABSTRACT</th>
<th>18. NUMBER OF PAGES</th>
</tr>
</thead>
<tbody>
<tr>
<td>a. REPORT</td>
<td>U</td>
<td></td>
</tr>
<tr>
<td>b. ABSTRACT</td>
<td>U</td>
<td></td>
</tr>
<tr>
<td>c. THIS PAGE</td>
<td>U</td>
<td></td>
</tr>
<tr>
<td>19a. NAME OF RESPONSIBLE PERSON</td>
<td>Binh Q. Nguyen</td>
<td></td>
</tr>
<tr>
<td>19b. TELEPHONE NUMBER (Include area code)</td>
<td>(301) 394-1781</td>
<td></td>
</tr>
</tbody>
</table>
An Introduction to Python
(A One-Hour Tour)

Beth G. Nguyen
U.S. Army Research Laboratory
Adelphi, Maryland
June 2, 2008

Topics
- dynamic programming language
- interactive programmable calculator
- programming paradigms
- markup language processing
- graphical user interface
- development environments
- system services
- web services
- networking
- cryptography
- scientific computing
- extension and embedding

Objectives
- high-level overview of the language
  - features
  - utilities
  - examples
  - sources
- information and suggestion to
  - decide whether to use Python
  - download and run the Python interpreter
  - use it interactively
interactive calculator ...

dynamic data types

```
>>> a = 12.34  # float
>>> b = 1  # int
>>> c = 1.00  # float
>>> complex(3, 4)  # complex number
>>> int(3.5)  # int
```

**int, float, long, complex, big numbers**

---

built-in types ...

```
>>> a = 0x1f; b = 0x3f; c=3.14; d=2.3e10
>>> oct(a), oct(b), oct(c+d), hex(a<<2), hex(b>>1), hex(c-o)
20777, 17777, '20777.117', '0x1f', '0x3f', '0x767f'
>>> abs(c)
3.14
>>> c.conjugate()
(3.14, 0)
```

---

... built-in types ...

```
>>> x=True; y=False; z=None
>>> x and y
False
>>> a = 10
>>> b = 10.0
>>> a < b
False
>>> a > b
False
>>> a is b
False
>>> a is not b
True
>>> a < b
False
>>> a <= b
True
>>> a >= b
False
>>> a == b
False
>>> a != b
True
```

**Built-in Constants**

- None: absence of a value
- False: None, False, zero, empty
- True: All other values

**Logical Operations**

- and, or, not

**Comparison Operations**

- <, <=, ==, !=, >, >=, is, is not
### Built-in Types

- **Sequence types**
  - `str` — ordered collection
  - `list` — `[(1, 2, 3, 'x', 'y', 'z')]` — mutable
  - `tuple` — `((1, 2, 3, 'x', 'y', 'z', 'c'),)` — immutable
  - `len()` — from `0` to `len(s)` or from `-len(s)` to `-1`
  - `s[1:-1]` — `i = 1 ... len(s)`

- **Mapping type**
  - `dict` — unordered collection
  - keys & values (name-value pairs)
  - unorderable — use keys to retrieve values

- **Set**
  - `set` — unordered collection
  - `frozenset` — immutable

### Other String Types

- `cStringIO` — read and write strings as files
- `struct` — string as a packed binary data
- `pack()`, `unpack()`, `calcsize()` ...
- `re` — regular expression

### Control-Flow Statements

- `for i in <sequence>`:
  - `statement(s)`
- `if cond:`
  - `if cond:`
  - `elif cond:`
  - `else:`
- `continue` # start next iteration
- `break` # stop iteration
- `return` # to caller
- `while cond:`
  - `statement(s)`
  - `while x < 0:`
  - `x = x - 1`
### Conditional expression

\[ x = \begin{cases} A & \text{if } \text{condition} == \text{True} \geq 1 \\ B & \text{else} \end{cases} \]

- Equivalent form:
  - \( \text{if A} \) \( \text{condition} == \text{True} \):
  - \( x = A \)
  - \( x = B \)

- \( \text{MAXRANGE = 200} \)
- \( \text{for d in [166, 239, 192, 241, 207]} \)
  - print d, \( \text{166: within range} \)
  - \( 239: \text{out of range} \)
  - \( 192: \text{within range} \)
  - \( 241: \text{out of range} \)
  - \( 207: \text{out of range} \)

### Bitwise ops & control flow

- \( a = 0xff \)
- \( b = 0xf \)
- \( \text{hex}(a ~\&~ b) \)
  - '00f'
  - \( \text{hex}(a | b) \)
  - '5536'
  - \( \text{hex}(a \& b) \)
  - '10f'
  - \( \text{hex}(a >> 4) \)
  - '0x18'
  - \( \text{hex}(a << 8 | 0x0c) \)
  - '124 b16 320 126256'
  - \( \text{hex}(a << 8) \)
  - '124 b16 320 126256'
  - \( \text{hex}(a + a) \)
  - '0x18'

### Built-in functions

- \( x = [3, 1, 2, -1] \)
- \( \text{print} \) 
  - '%5.1f = %d s=%s % (2.12, 4, 'abc'), x
  - f = 2.1 \( \equiv 4 \) s=abc [3, 1, 2, -1]
- \( \text{sorted}(x); \text{len}(x) \)
  - 
- \( \text{max}(x); \text{min}(x); \text{sum}(x) \)
  - 
- \( \text{abs}, \text{ord}, \text{chr}, \text{enumerate}, \text{eval}, \text{hex}, \text{id}, \text{hash}, \text{raw_input}, \text{open}, \text{type}, \text{zip}, \text{...} \)
function

def function_name([arguments]):
    statement(s)
    [return [type]]

def fact(n):
    if n <= 1: return 1
    return n * fact(n-1)
print fact(5)  # 120

def fact(n):
    return 1 if n < 2 else n * fact(n-1)

class

class class_name:
    statement(s)

class Node:
    def __init__(self, name, links):
        self.name = name  # Node ID
        self.links = links  # list of links
    def get_links(self): return self.links
    def set_links(self, links): self.links = links

class

inheretence

class DerivedClass(Base1, Base2,...):
    statement(s)

class PropModel:
    def __init__(self, params):
        self.params = params
    def get_params(self):
        return self.params
    def get_LER(self):
        return LER

class FSL(PropModel):
    def __init__(self, params):
        PropModel.__init__(params)
    def get_LER(self):
        return LER
        # overriding the base class method

    return LER
        # return Link-Error Rate

June 5, 2008
An Introduction to Python
more on `str()` ...

```python
>>> s = 'abc .. xyz'
>>> s.isupper()  # False
>>> s.capitalize()  # 'Abc .. xyz'
>>> s.startswith('y')  # False
>>> s.endswith('y')  # False
>>> s.lower()  # 'abc .. xyz'
>>> s.replace(' .. ', ' . ', 1)  # 'abc . xyz'
>>> s.split()  # ['abc', ' .. ', 'xyz']
>>> len(s)  # 9
>>> s.find('xyz')  # 6
>>> s + '123'  # 'abc .. xyz123'
>>> s + 'ab' * 3  # 'abc .. xyzababab'
>>> help(str)
```

more on `list()` ...

```python
>>> x = [1, 'a', 3, 'x']  # initial values
>>> y = range(6, 11)  # y = [6, 7, 8, 9, 10]
>>> x.pop(1)  # remove 'a' (the 2nd item) from x
>>> x.pop()  # remove 'z' from x, x.pop(-1)
>>> x  # ['1', 'x', '3']
>>> x.append(4)  # add '4' to the end of the list x
>>> x.insert(1, '2')  # insert '2' to the 2nd position (index 1)
>>> x.append(8)  # add '8' to the end of the list x
>>> x  # ['1', '2', '3', '4', '8']
>>> x.extend(y)  # same as x = x + y
>>> x  # ['1', '2', '3', '4', '6', '7', '8', '9', '10']
>>> x[3:7]  # slice x[3:7] = [4, 5, 6, 7]
>>> x[::-1]  # reverse x = ['1', '2', '3', '1', '2', '3']
>>> help(list)  # help(x), help(list)
```

stack & queue using list

- `x = range(5)`
  ```python
  x = [0, 1, 2, 3, 4]
  ```
- `def size(x): return len(x)`
- `def is_empty(x): return size(x) == 0`

- stack - last in first out (LIFO)
  - `push(x): x.append(s)`  # x = [0, 1, 2, 3, 4, 5]
  - `pop(x): x.pop()`  # x = [0, 1, 2, 3, 4]

- queue - first in first out (FIFO)
  - `enqueue(x): x.append(s)`  # x = [0, 1, 2, 3, 4, 5]
  - `dequeue(x): x.pop()`  # x = [0, 1, 2, 3, 4, 5]
creating instances

```python
[a] ==> [b] ==> [c]

class Node:
    def __init__(self, name, links=None):
        self.name = name
        self.links = links

a = Node('a')
b = Node('b')
c = Node('c', [b])
a.set_links([b])
b.set_links([a, c])
```

---

functional programming

```python
filter(), map(), reduce(), lambda, list comprehension

>>> def even(x): return x % 2 == 0
>>> def odd(x): return not even(x)
>>> r = [random.randint(1, 10) for i in range(1000)]
>>> set(filter(even, r))  # { set([8, 2, 4, 10, 6])
>>> set(filter(odd, r))  # { set([1, 3, 5, 9, 7])
>>> map(lambda x: x % 2 == 0, r))  # [{1, 3, 5, 9, 7}]
>>> map(lambda x: x % 2, r)  # [{1, 3, 5, 9, 7}]
>>> reduce(lambda x, y: x * y, range(1, 6))  # 1 * 2 * 3 * 4 * 5 = 120
>>> reduce(lambda x, y: x * y, range(1, 6))  # 120
```

---

file operation

- f = open('<filename>', ['r', 'a', 'w', '...'])
- f.read()
- f.readlines()
- f.write(s)
- f.seek(n)
- f.tell()
- f.close()
- ...
- help(file)
exceptions

- built-in exceptions: Exception, IOError, KeyError, MemoryError, RuntimeError, SystemExit, ...
- user-defined exceptions
- raising exceptions: raise <ExceptionType>
- handling exceptions:
  try:  # do something that may raise an exception
  except <i>:  # encountering an exception, do this
  else:  # do this if an exception did not occur
  finally:  # do this whether an exception has occurred

recap on presented topics

- Python – a dynamic computer language
- built-in types: int, float, bool, complex, oct, hex, big numbers, string, list, tuple, set, dictionary
- operations: bitwise, binary, unary, boolean, conversion
- control flow statements:
  if ... else ... elif ... else, continue, break, pass, for, while
- frequently used functions: dir(), help(), range(), print()
- interactive programmable calculator
- programming paradigms:
  - procedural:
  - object-oriented:
  - functional:
  - file operations:
  - exceptions:

next topics

- modules – creating and importing
  - keywords
  - cryptography
  - Internet & web services
  - markup language processing
  - system services & networking
- scientific computing
- graphical user interface
- extension and embedding
- development environment
import random
n = random.randint(1, 10)  # [1, 10]
from random import *
n = randint(1,10)
from random import uniform
n = uniform(1,10)  # [1.0 .. 10.0]
import random as rand
n = rand.seed(10.0)

the “builtins” module

- automatically imported
- to see a list built-in objects:
  >>> dir(_builtins_)
  ['ArithmeticError', 'dir', 'enumerate', 'filter', 'getattr', 'globals', 'hasattr', 'id', 'input', 'locals', 'map', 'max', 'min', 'next', 'object', 'open', 'ord', 'property', 'range', 'repr', 'reversed', 'round', 'setattr', 'slice', 'sorted', 'str', 'sum', 'super', 'tuple', 'type', 'vars', 'zip']
- to see their descriptions:
  >>> help(_builtins_)
  ... lots of output...

modules & main applications

# importing module myutils
__name__ == "myutils"
import myutils as u
p = u.factorial(10)
q = u.search("xyz")
o = u.MyClass()
n = u.MAXNUM
...

# running myutils application
python myutils.py

# file: "myutils.py"
MAXNUM = 20
def fact(n):
    ... def search():
        ... class MyClass:
        ...
def test():
        ...

# main application
__name__ == __main__
if __name__ == "__main__":
    test()
from keyword import kwlist
for kw in kwlist:
    print kw,

def del def else except exec finally for from global if import in is lambda not or pass print raise return try while with yield

>>> import sha, md5, hmac
>>> sha.new(abc-xyz).hexdigest()
'55e8a20533a2b7c99761a0874b48a0413d164337'
>>> md5.new(abc-xyz).hexdigest()
'096252a7b87b04f2b792b04d5a40174'
>>> key = 'shared_secret_key'
>>> text = 'abc-xyz'
>>> hash_algorithm = 'hashlib.sha256'
>>> hmac.new(key, text, hash_algorithm).hexdigest()
'd23a59b00632576007c5e5327d966108c0b6e726780985f237e1512d39a43a53'

#(standard library)
>>> hashlib.sha1(abc-xyz).hexdigest()
'55e8a20533a2b7c99761a0874b48a0413d164337'
>>> hashlib.sha512(abc-xyz).hexdigest()
'd311bb080815ae77e0c0ae3706fb94c8059c690286cd66e0f2be98a21e7659d5f7aa37590daa9fd82e4f7b60c46e713eab187a92f0f0d3132fd3e493f258013'
>>> hashlib.md5(abc-xyz).hexdigest()
'096252a7b87b04f2b792b04d5a40174'
Internet & web services

```python
>>> import webbrowser as w
>>> w.open('http://www.yahoo.com')
>>> import urllib as u
>>> f = u.urlopen(url)
>>> x = f.read()
>>> f.close()
``` 

Other libraries: ftp, http, pop, smtp, ...

---

markup languages

- HTML parsers
  - HTML
  - XHTML
- XML parsers
  - DOM
  - SAX
  - Expat
- SGML parsers

---

system services

- os, sys, threading, socket,
- interprocess communications (IPC), ...
- os.getcwd(), os.getenv("HOME"), ...
- os.path.basename(p), os.path.isfile(f), ...
- sys.stderr, sys.stdout, sys.stdin, ...
- threading.Thread(): run(), isAlive(), ...
- socket(): ntohl(n), gethostname(), ...
- Popen (IPC): communicate(), poll(), wait(), ...

---
extension and embedding

- extension – calling C/C++ functions from Python programs
- embedding – calling Python functions from C/C++ programs
- open-source tools:
  - SWIG Simplified Wrapper & Interface Generator
  - Boost.Python C++ and Python interoperability
  - Pyrex Mixing Python & C to generate extensions
  - T2py Fortran to Python interface generator
  - ...
  - http://wiki.python.org/moin/IntegratingPythonWithOtherLanguages

scientific computing & visualization

- math – standard library
- add-on:
  - SAGE – math
  - SymPy – symbolic math
  - PyChem – multivariate analysis
  - NumPy – numerical Python
  - SciPy – scientific Python
  - PyOpenGL – binding to OpenGL
  - VTK – visual tool kit
  - ...

graphical user interface

- Tkinter – the standard GUI of Python
- other tool kits:
  - wxPython
  - pyKDE
  - pyGtk
  - PyQt
  - ...

June 2, 2006
An Introduction to Python
A Tkinter application

```python
from tkinter import *
from random import randint, choice
SIZE = 200
canvas = Canvas(Tk(), w=SIZE, height=SIZE, bg="lightblue")
```
```
```
```html
for i in range(300):
    x, y = randint(0, SIZE), randint(0, SIZE)
```
```python
d = randint(5, 10)
canvas.create_oval(x, y, x+d, y+d, fill=choice(colors))
canvas.pack()
canvas.mainloop()
```

Turtle graphics

```python
import turtle
```
```
```
```html
import turtle
```
```
```python
colors=['green', 'yellow', 'orange', 'red', 'magenta', 'blue', 'lightblue']
```
```
```
```
```
```python
ncolors = len(colors)
```
```
```
```python
w, h = 220
```
```
```
```python
PEN_WIDTH = 20
```
```
```
```python
turtle.setup(width=x, height=y)
turtle.title("Rainbow Colors by BON")
```
```
```
```python
pen = turtle.Pen()
```
```
```
```python
pen.up()
```
```
```
```python
x, y = pen.position()
```
```
```
```python
pen.goto(x, (PEN_WIDTH - h)/2)
```
```
```
```python
pen.down()
```
```
```
```python
pen.width(PEN_WIDTH)
```
```
```
```python
for i in range(16, 0, -1):
```
```
```
```python
pen.color(colors[ncolors % ncolors])
```
```
```
```python
ten.circle((PEN_WIDTH*1/2)
```
```
```python
turtle.done()
```
```
```

development tools

- Freely available
  - IDLE — comes with Python
  - IPython — comes with SciPy
  - DrPython — requires wxPython
  - Eclipse+PyDEV — requires Java
  - ...
- Commercial IDE
  - Wing IDE
  - Komodo IDE
  - ...

June 2, 2008
An Introduction to Python
topics presented

dynamic programming language
interactive programmable calculator
programming paradigms
graphical user interface
web services
networking
encryption and embedding

http://www.python.org

- tutorial for the current version (2.5.2)
- library reference
- language reference
- news & announcements
- links to other relevant sites
- binary and source files for downloading
  - Windows™
  - Linux™
  - ...

The End

Thanks

lmingyey@arl.army.mil
<table>
<thead>
<tr>
<th>No. of Copies</th>
<th>Organization</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 (Elect Copy)</td>
<td>ADMNSTR DEFNS TECHL INFO CTR ATTN DTIC OCP 8725 JOHN J KINGMAN RD STE 0944 FT BELVOIR VA 22060-6218</td>
</tr>
<tr>
<td>1</td>
<td>US ARMY RSRCH LAB ATTN AMSRD ARL CI OK TP TECHL LIB T LANDFRIED BLDG 4600 ABERDEEN PROVING GROUND MD 21005-5066</td>
</tr>
<tr>
<td>22</td>
<td>US ARMY RSRCH LAB ATTN AMSRD ARL CI NT D GWYN ATTN AMSRD ARL CI A TOTH ATTN AMSRD ARL CI J GOWENS ATTN AMSRD ARL CI N G RACINE ATTN AMSRD ARL CI NT B NGUYEN (5 HC, 1 ELECT) ATTN AMSRD ARL CI NT B RIVERA ATTN AMSRD ARL CI NT G CIRINCIONE ATTN AMSRD ARL CI NT K MARCUS ATTN AMSRD ARL CI NT L M SCOTT ATTN AMSRD ARL CI NT N IVANIC ATTN AMSRD ARL CI NT R HARDY ATTN AMSRD ARL CI NT R PRESSLEY ATTN AMSRD ARL CI OK T TECHL PUB ATTN AMSRD ARL CI OK TL TECHL LIB ATTN AMSRD ARL CI R NAMBURU ATTN AMSRD ARL SE R P AMIRTHARAJ ATTN IMNE ALC IMS MAIL &amp; RECORDS MGMT ADELPHI MD 20783-1197</td>
</tr>
<tr>
<td>TOTAL</td>
<td>24 (22 HC and 2 Elect)</td>
</tr>
</tbody>
</table>