

*ARMY RESEARCH LABORATORY*



## **An Introduction to Python (A One-Hour Tour)**

**by Binh Q. Nguyen**

**ARL-TN-0328**

**July 2008**

**Approved for public release; distribution unlimited.**

## **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.

# **Army Research Laboratory**

Adelphi, MD 20783-1197

---

---

**ARL-TN-0328**

**July 2008**

## **An Introduction to Python (A One-Hour Tour)**

**Binh Q. Nguyen**

**Computational and Information Sciences Directorate, ARL**

**REPORT DOCUMENTATION PAGE**

*Form Approved  
OMB No. 0704-0188*

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing the burden, to Department of Defense, Washington Headquarters Services, Directorate for Information Operations and Reports (0704-0188), 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to any penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number.

**PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ADDRESS.**

<b>1. REPORT DATE (DD-MM-YYYY)</b> July 2008		<b>2. REPORT TYPE</b> Summary		<b>3. DATES COVERED (From - To)</b> FY08	
<b>4. TITLE AND SUBTITLE</b> An Introduction to Python (A One-Hour Tour)				<b>5a. CONTRACT NUMBER</b>	
				<b>5b. GRANT NUMBER</b>	
				<b>5c. PROGRAM ELEMENT NUMBER</b>	
<b>6. AUTHOR(S)</b> Binh Q. Nguyen				<b>5d. PROJECT NUMBER</b>	
				<b>5e. TASK NUMBER</b>	
				<b>5f. WORK UNIT NUMBER</b>	
<b>7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)</b> U.S. Army Research Laboratory ATTN: AMSRD-ARL-CI-NT 2800 Powder Mill Road Adelphi, MD 20783-1128				<b>8. PERFORMING ORGANIZATION REPORT NUMBER</b>  ARL-TN-0328	
<b>9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)</b>				<b>10. SPONSOR/MONITOR'S ACRONYM(S)</b>	
				<b>11. SPONSOR/MONITOR'S REPORT NUMBER(S)</b>	
<b>12. DISTRIBUTION/AVAILABILITY STATEMENT</b> Approved for public release; distribution unlimited.					
<b>13. SUPPLEMENTARY NOTES</b>					
<b>14. ABSTRACT</b> 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.					
<b>15. SUBJECT TERMS</b> The Python programming language, tutorial					
<b>16. Security Classification of:</b>			<b>17. LIMITATION OF ABSTRACT</b>	<b>18. NUMBER OF PAGES</b>	<b>19a. NAME OF RESPONSIBLE PERSON</b>
<b>a. REPORT</b>	<b>b. ABSTRACT</b>	<b>c. THIS PAGE</b>			<b>19b. TELEPHONE NUMBER (Include area code)</b>
U	U	U	U	20	Binh Q. Nguyen  (301) 394-1781

RDECOM 

# An Introduction to Python

(A One-Hour Tour)

Binh Q. Nguyen  
**U.S. Army Research Laboratory**  
 Adelphi, Maryland  
 June 2, 2008

TECHNOLOGY DRIVEN. WARFIGHTER FOCUSED.

---

---

---

---

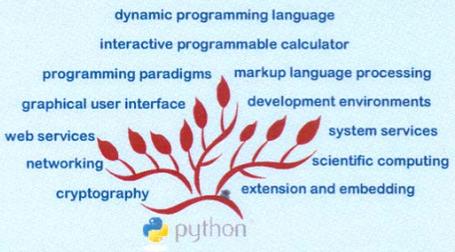
---

---

---

---

RDECOM **topics** 



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

python

June 2, 2008    An Introduction to Python    2  
 TECHNOLOGY DRIVEN. WARFIGHTER FOCUSED.

---

---

---

---

---

---

---

---

RDECOM **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

June 2, 2008    An Introduction to Python    3  
 TECHNOLOGY DRIVEN. WARFIGHTER FOCUSED.

---

---

---

---

---

---

---

---





**RDCOM conditional expression**

**x = A if <condition == True> else B**

equivalent form  
 if A <condition == True> :  
     x = A  
 else:  
     x = B

```
>>> MAXRANGE = 200
>>> for d in [ 166, 239, 192, 241, 207 ]:
    print d, ':', 'within range' if d < MAXRANGE else 'out of range'
166 : within range
239 : out of range
192 : within range
241 : out of range
207 : out of range
```

June 2, 2008      An Introduction to Python      10  
TECHNOLOGY DRIVEN. WARFIGHTER FOCUSED.

---

---

---

---

---

---

---

---

---

---

**RDCOM bitwise ops & control flow**

```
>>> a = 0xff                    >>> n = 2**16
>>> b = 0x0f                    >>> while n:
>>> hex(a ^ b)                    print n,
'0xf0'                            n = n >> 1
>>> hex(a | b)                    65536 32768 16384 8192
'0xff'                            4096 2048 1024 512 256
>>> hex(a & b)                    128 64 32 16 8 4 2 1
'0xf'                             >>> n = 1
>>> hex(a >> 4)                   >>> while n <= 2**16:
'0xf'                            print n,
>>> hex(a << 8 | 0x0c)           n = n << 1
'0xff0c'                          1 2 4 8 16 32 64 128 256
>>> hex(a ^ a)                   512 1024 2048 4096 8192
'0x0'                            16384 32768 65536
```

June 2, 2008      An Introduction to Python      11  
TECHNOLOGY DRIVEN. WARFIGHTER FOCUSED.

---

---

---

---

---

---

---

---

---

---

**RDCOM built-in functions**

```
>>> x = [ 3, 1, 2, -1]
>>> print 'f=%5.1f i=%d s=%s' % (2.12, 4, 'abc'), x
f= 2.1 i=4 s=abc [3, 1, 2, -1]
>>> sorted(x); len(x)            # [-1, 1, 2, 3] 4
>>> max(x); min(x); sum(x)       # 3 -1 5
```

abs(), ord(), chr(), enumerate(), eval(), hex(),  
 id(), hash(), raw\_input(), open(), type(), zip(), ...

June 2, 2008      An Introduction to Python      12  
TECHNOLOGY DRIVEN. WARFIGHTER FOCUSED.

---

---

---

---

---

---

---

---

---

---

RDECOM **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)

```

procedural

June 2, 2008 An Introduction to Python 13  
TECHNOLOGY DRIVEN. WARFIGHTER FOCUSED.

---

---

---

---

---

---

---

---

RDECOM **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

```

object-oriented

June 2, 2008 An Introduction to Python 14  
TECHNOLOGY DRIVEN. WARFIGHTER FOCUSED.

---

---

---

---

---

---

---

---

RDECOM **inheritance** 

```

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

class PropModel: # signal propagation model
    def __init__(self, params):
        self.params = params
    def get_params(self): return self.params
    def get_LER(self): pass # no-op, do nothing

class FSL(PropModel): # Free-Space Loss model
    def __init__(self, params):
        PropModel.__init__(params)
    def get_LER(self): # overriding the base class method
        ...
    return LER # return Link-Error Rate

```

June 2, 2008 An Introduction to Python 15  
TECHNOLOGY DRIVEN. WARFIGHTER FOCUSED.

---

---

---

---

---

---

---

---

**more on str() ...**

```

>>> s='abc ... xyz'
>>> s.isupper()           #> False
>>> s.capitalize()       #> 'Abc ... xyz'
>>> s.upper()            #> 'ABC ... XYZ'
>>> s.endswith('z')      #> True
>>> s.startswith('a')    #> True
>>> s.endswith('xyz')    #> False
>>> s[2:5]               #> 'c .'
>>> s[5:]                 #> '...xyz'
>>> s.split()             #> ['abc', '...', 'xyz']
>>> s.find('c')           #> 2
>>> s.rfind('.')          #> 6
>>> s + '123'            #> 'abc ... xyz123'
>>> s = 'ab'*3           #> 'ababab'
>>> help(str)

```

June 2, 2008 16  
An Introduction to Python  
TECHNOLOGY DRIVEN. WARRIOR FIGHTER FOCUSED.

---

---

---

---

---

---

---

---

---

---

**more on list() ...**

```

>>> x = [1, 'a', 3, 'z'] # 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                   # x = [1, 3]
>>> x.append(4)         # add '4' to the end of the list x
>>> x.insert(1, 2)      # insert '2' to the 2nd position (i=1)
>>> x.append(5)         # add '5' to the end of the list x
>>> x                   # x = [1, 2, 3, 4, 5]
>>> x.extend(y)         # same as x = x + y
>>> x                   # x = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
>>> x[3:7]              # slice x[3:7]:= [4, 5, 6, 7]
>>> x=[1,2,3]*3        # x = [1, 2, 3, 1, 2, 3, 1, 2, 3]
>>> help(list)         # help(x), help(list)

```

June 2, 2008 17  
An Introduction to Python  
TECHNOLOGY DRIVEN. WARRIOR FIGHTER FOCUSED.

---

---

---

---

---

---

---

---

---

---

**stack & queue using list**

- `x = range(5)`      `# x:= [0,1, 2, 3, 4]`
  - `def size(x):`      `return len(x)`
  - `def is_empty():`   `return size(x) == 0`
- **stack – last in first out (LIFO)**
  - `push(5): x.append(5)` # `x:= [0,1, 2, 3, 4, 5]`
  - `pop(): x.pop()`    # `5, x:= [0,1, 2, 3, 4]`
- **queue – first in first out (FIFO)**
  - `enqueue(5): x.append(5)` # `x:= [0,1, 2, 3, 4, 5]`
  - `dequeue(): x.pop(0)`   # `0, x:= [1, 2, 3, 4, 5]`

June 2, 2008 18  
An Introduction to Python  
TECHNOLOGY DRIVEN. WARRIOR FIGHTER FOCUSED.

---

---

---

---

---

---

---

---

---

---

**tuple() and dict ()**

<b>Tuple</b>	<b>Dictionary</b>
>>> t = (1, 2, 3)	>>> d = {'A': 65, 'C': 0x43}
>>> 'a' in t	>>> d['B']=0x42
False	>>> d.keys()
>>> 2 in t	[ 'A', 'C', 'B' ]
True	>>> d.values() # [65, 67, 66]
>>> t[1]	>>> 'D' in d # False
2	>>> d.get('C') # 67 (0x43)

indexable  
immutable

unindexable  
mutable

June 2, 2008 An Introduction to Python TECHNOLOGY DRIVEN WARRIGHTER FOCUSED. 19

---

---

---

---

---

---

---

---

---

---

**set()**

```
>>> a = set(range(5))
>>> b = set(range(3,8))
>>> a.intersection(b)
set([3, 4])
>>> a.union(b)
set([0, 1, 2, 3, 4, 5, 6, 7])
>>> a-b
set([0, 1, 2])
>>> b-a
set([5, 6, 7])
```

Set a = { 0, 1, 2, 3, 4 }

Set b = { 3, 4, 5, 6, 7 }

June 2, 2008 An Introduction to Python TECHNOLOGY DRIVEN WARRIGHTER FOCUSED. 20

---

---

---

---

---

---

---

---

---

---

**handy built-in functions**

- range(5) => [0, 1, 2, 3, 4]
- range(2, 13, 3) => [2, 5, 8, 11]
- dir( [object] ) => list of names
- dir() => current names
- dir(str) => [ ..., 'isalnum', ..., 'replace', ..., 'strip', ... 'upper', 'zfill' ]
- help(str)
- help(str.strip)

June 2, 2008 An Introduction to Python TECHNOLOGY DRIVEN WARRIGHTER FOCUSED. 21

---

---

---

---

---

---

---

---

---

---

**RDECOM** creating instances 

**[a]<==>[b]<==>[c]**

```

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

    def get_links(self):
        return self.links

    def set_links(self, links):
        self.links = links

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

```

June 2, 2008 An Introduction to Python 22  
TECHNOLOGY DRIVEN. WARRIOR FIGHTER FOCUSED.

---

---

---

---

---

---

---

---

---

---

**RDECOM** functional programming 

**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, 9, 5, 7])
>>> map(float, set(filter(odd, r))) # [1.0, 3.0, 9.0, 5.0, 7.0]
# >>> [ float(x) for x in set ( filter(odd, r) ) ]
# >>> [ float(x) for x in set ( [ x for x in r if odd(x) ] ) ]

>>> def product(x, y): return x * y
>>> reduce( product, range(1, 6) ) # 1*2*3*4*5 = 120
>>> reduce(lambda x, y: x * y, range(1, 6) )

```

June 2, 2008 An Introduction to Python 23  
TECHNOLOGY DRIVEN. WARRIOR FIGHTER FOCUSED.

---

---

---

---

---

---

---

---

---

---

**RDECOM** file operation 

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

June 2, 2008 An Introduction to Python 24  
TECHNOLOGY DRIVEN. WARRIOR FIGHTER FOCUSED.

---

---

---

---

---

---

---

---

---

---

**RDECOM** 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 <>:
    ... # encountering an exception, do this
else:
    ... # do this if an exception did not occur
finally:
    ... # do this whether an exception has occurred

```

June 2, 2008 An Introduction to Python 25  
TECHNOLOGY DRIVEN. WARFIGHTER FOCUSED.

---

---

---

---

---

---

---

---

---

---

**RDECOM** 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 ... elif ... else, continue, break, pass, for, while
- frequently used functions: dir(), help(), range(), print()
- interactive programmable calculator
- programming paradigms
  - procedural: def func(): return
  - object-oriented: class, inheritance, self, '.' notation
  - functional: filter(), map(), reduce(), lambda
- file operations: open(), read(), write(), close(), ...
- exceptions: try ... except ... else ...finally

June 2, 2008 An Introduction to Python 26  
TECHNOLOGY DRIVEN. WARFIGHTER FOCUSED.

---

---

---

---

---

---

---

---

---

---

**RDECOM** 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

June 2, 2008 An Introduction to Python 27  
TECHNOLOGY DRIVEN. WARFIGHTER FOCUSED.

---

---

---

---

---

---

---

---

---

---

**RDECOM** modules ... 

```

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)

```

June 2, 2008 An Introduction to Python TECHNOLOGY DRIVEN WARFIGHTER FOCUSED. 28

---

---

---

---

---

---

---

---

**RDECOM** the "builtins" module 

- automatically imported
- to see a list builtin objects:  

```
>>> dir(__builtins__)
['ArithmeticError', ..., 'zip']
```
- to see their descriptions:  

```
>>> help(__builtins__)
...lots of output...
```

June 2, 2008 An Introduction to Python TECHNOLOGY DRIVEN WARFIGHTER FOCUSED. 29

---

---

---

---

---

---

---

---

**RDECOM** modules & main applications 

```

# importing module myutils #file: "myutils.py"
# __name__ == myutils      MAXNUM = 20
import myutils as u        def fact(n): ...
p = u.fact(10)             def search(x): ...
q = u.search('xyz')       class MyClass: ...
o = u.MyClass()           def test(): ...
n = u.MAXNUM
...
# running myutils application #main application
python myutils.py          if __name__ ==
# __name__ == __main__     '.__main__':
                           test()

```

June 2, 2008 An Introduction to Python TECHNOLOGY DRIVEN WARFIGHTER FOCUSED. 30

---

---

---

---

---

---

---

---

RDECOM **key words** 

```
>>> from keyword import kwlist
>>> for kw in kwlist:
    print kw,
and as assert break class continue
def del elif else except exec finally for
from global if import in is lambda not
or pass print raise return try while
with yield
```

June 2, 2008 An Introduction to Python 31  
TECHNOLOGY DRIVEN. WARFIGHTER FOCUSED.

---

---

---

---

---

---

---

---

RDECOM **cryptographic modules ...** 

**sha, md5, hashlib, hmac**

```
>>> import sha, md5, hmac
>>> sha.new("abc-xyz").hexdigest()
'55e6a20533a2b7c99761d0874b48a0413d164337'
>>> md5.new("abc-xyz").hexdigest()
'096252a7b87b04f2b7928c04d5a40174'
>>> key = 'shared_secret_key'
>>> text = 'abc-xyz'
>>> hash_algorithm = hashlib.sha256
>>> hmac.new(key, text, hash_algorithm).hexdigest()
'4b23a5d8f806332180acffc57d986108cb0e7d1678098
5f237e1512d39a43a63'
```

June 2, 2008 An Introduction to Python 32  
TECHNOLOGY DRIVEN. WARFIGHTER FOCUSED.

---

---

---

---

---

---

---

---

RDECOM **... cryptographic modules** 

```
>>> import hashlib  #(standard library)
>>> hashlib.sha1("abc-xyz").hexdigest()
'55e6a20533a2b7c99761d0874b48a0413d164
337'
>>> hashlib.sha512("abc-xyz").hexdigest()
'd311bb080815ae77e0cae3706fb94c8059c69
0286cd566cf2be98a21e7659d5f7aa37590d
a94f082e4f37b60c46e713eab187a93f59632
3132fd3e493f258013'
>>> hashlib.md5("abc-xyz").hexdigest()
'096252a7b87b04f2b7928c04d5a40174'
```

June 2, 2008 An Introduction to Python 33  
TECHNOLOGY DRIVEN. WARFIGHTER FOCUSED.

---

---

---

---

---

---

---

---

**RDECOM** Internet & web services 

```

>>> import webbrowser as w
>>> w.open('http://www.yahoo.com')
>>> import urllib as u
>>> f = u.urlopen(url)
>>> x = f.read()
>>> f.close()
>>> u.urlretrieve(url, filename='xyz.htm')

```

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

June 2, 2008 An Introduction to Python 34  
TECHNOLOGY DRIVEN. WARFIGHTER FOCUSED.

---

---

---

---

---

---

---

---

**RDECOM** markup languages 

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

June 2, 2008 An Introduction to Python 35  
TECHNOLOGY DRIVEN. WARFIGHTER FOCUSED.

---

---

---

---

---

---

---

---

**RDECOM** 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(), ...

June 2, 2008 An Introduction to Python 36  
TECHNOLOGY DRIVEN. WARFIGHTER FOCUSED.

---

---

---

---

---

---

---

---

**RDECOM** 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*
  - f2py *Fortran to Python interface generator*
  - ...
  - <http://wiki.python.org/moin/IntegratingPythonWithOtherLanguages>

June 2, 2008 An Introduction to Python 37  
TECHNOLOGY DRIVEN. WARFIGHTER FOCUSED.

---

---

---

---

---

---

---

---

**RDECOM** 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
  - ...

June 2, 2008 An Introduction to Python 38  
TECHNOLOGY DRIVEN. WARFIGHTER FOCUSED.

---

---

---

---

---

---

---

---

**RDECOM** graphical user interface 

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

June 2, 2008 An Introduction to Python 39  
TECHNOLOGY DRIVEN. WARFIGHTER FOCUSED.

---

---

---

---

---

---

---

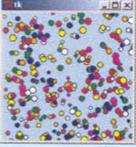
---

**RDCOM** A Tkinter application 

```

from Tkinter import *
from random import randint, choice
SIZE = 200
canvas = Canvas(Tk(), w=SIZE,
                height=SIZE, bg='lightblue')
colors=['green','yellow','orange',
        'red','magenta','blue','white']
for i in range(300):
    x, y = randint(0, SIZE), randint(0, SIZE)
    d = randint(5, 10)
    canvas.create_oval(x, y, x+d,y+d, fill=choice(colors))
canvas.pack()
canvas.mainloop()

```



June 2, 2008 An Introduction to Python 40  
TECHNOLOGY DRIVEN. WARFIGHTER FOCUSED.

---

---

---

---

---

---

---

---

---

---

**RDCOM** Turtle graphics 

```

import turtle
colors=['green','yellow','orange','red','magenta','blue','lightblue']
ncolors = len(colors)
w = h = 220
PEN_WIDTH = 20
turtle.setup( width=w, height=h )
turtle.title("Rainbow Colors by BQN")
pen = turtle.Pen()
pen.up()
x, y = pen.position()
pen.goto(x, (PEN_WIDTH - h)/2)
pen.down()
pen.width(PEN_WIDTH)
for i in range(10, 0, -1):
    pen.color(colors[i % ncolors])
    pen.circle( PEN_WIDTH * i / 2 )
turtle.done()

```



June 2, 2008 An Introduction to Python 41  
TECHNOLOGY DRIVEN. WARFIGHTER FOCUSED.

---

---

---

---

---

---

---

---

---

---

**RDCOM** 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 42  
TECHNOLOGY DRIVEN. WARFIGHTER FOCUSED.

---

---

---

---

---

---

---

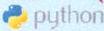
---

---

---

**RDECOM** **topics presented** 

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



June 2, 2008    An Introduction to Python    43  
TECHNOLOGY DRIVEN. WARFIGHTER FOCUSED.

---

---

---

---

---

---

---

---

**RDECOM** <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™
  - ...

June 2, 2008    An Introduction to Python    44  
TECHNOLOGY DRIVEN. WARFIGHTER FOCUSED.

---

---

---

---

---

---

---

---

**RDECOM** 



**Thanks**

[bnguyen@arl.army.mil](mailto:bnguyen@arl.army.mil)

June 2, 2008    An Introduction to Python    45  
TECHNOLOGY DRIVEN. WARFIGHTER FOCUSED.

---

---

---

---

---

---

---

---

No. of Copies	Organization
1 (Elect Copy)	ADMNSTR DEFNS TECHL INFO CTR ATTN DTIC OCP 8725 JOHN J KINGMAN RD STE 0944 FT BELVOIR VA 22060-6218
1	US ARMY RSRCH LAB ATTN AMSRD ARL CI OK TP TECHL LIB T LANDFRIED BLDG 4600 ABERDEEN PROVING GROUND MD 21005-5066
22	US ARMY RSRCH LAB ATTN AMSRD AR 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 & RECORDS MGMT ADELPHI MD 20783-1197
TOTAL	24 (22 HC and 2 Elect)