Python CodeCount™ Counting Standard

University of Southern California

Center for Systems and Software Engineering

October , 2007

Revision Sheet

Date	Version	Revision Description	Author
10/28/2007	1.0	Consolidated Draft	CSSE
4/2/2008	1.1	Update section 3.0 (selection, iteration statements)	CSSE
4/14/2008	1.2	Update section 3.0 (jump, expression statements)	CSSE
1/2/2013	2.0	Updated document template	CSSE

Table of Contents

1. Definitions

1.1. **SLOC –** Source Lines of Code is a unit used to measure the size of software program. SLOC counts the program source code based on a certain set of rules. SLOC is a key input for estimating project effort and is also used to calculate productivity and other measurements.

1.2. **Physical SLOC –** One physical SLOC is corresponding to one line starting with the first character and ending by a carriage return or an end-of-file marker of the same line, and which excludes the blank and comment line.

1.3. **Logical SLOC –** Lines of code intended to measure "statements", which normally terminate by a semicolon (C/C++, Java, C#) or a carriage return (VB, Assembly), etc. Logical SLOC are not sensitive to format and style conventions, but they are language-dependent.

1.4. **Data declaration line or data line –** A line that contains declaration of data and used by an assembler or compiler to interpret other elements of the program.

The following table lists the Python keywords that denote data declaration lines:

class

Table 1 Data Declaration Types

1.5. **Compiler Directives –** A statement that tells the compiler how to compile a program, but not what to compile.

1.6. **Blank Line –** A physical line of code, which contains any number of white space characters (spaces, tabs, form feed, carriage return, line feed, or their derivatives).

1.7. **Comment Line –** A comment is defined as a string of zero or more characters that follow language-specific comment delimiter.

There are two styles of comments in Python

- # single line comment
- " " " this is a multiline comment which spawns many lines

A whole comment line may span one line and does not contain any compilable source code. An embedded comment can co-exist with compilable source code on the same physical line. Banners and empty comments are treated as types of comments.

1.8. **Executable Line of code –** A line that contains software instruction executed during runtime and on which a breakpoint can be set in a debugging tool. An instruction can be stated in a simple or compound form.

- An executable line of code may contain the following program control statements:
    - Selection statements (if, ? operator)
    - Iteration statements (for, while, do-until, foreach)
    - Empty statements (pass)
    - Jump statements (return, goto, last, next, exit function)
    - Expression statements (function calls, assignment statements, operations, etc.)
    - Block statements
- An executable line of code may not contain the following statements:
    - Whole line comments, including empty comments and banners
    - Blank lines

2. Checklist for source statement counts

PHYSICAL SLOC COUNTING RULES			
MEASUREMENT UNIT	ORDER OF PRECEDENCE	PHYSICAL SLOC	COMMENTS
Executable Lines	1	One per line	Defined in 1.8
Non-executable Lines			
Declaration (Data) Lines	2	One per line	Defined in 1.4
Compiler Directive	3	One per line	Defined in 1.5
Comments			Defined in 1.7
One their own lines	4	Not Included (NI)	
Embedded	5	NI	
Banners	6	NI	
Empty Comments	7	NI	
Blank Lines	8	NI	Defined in 1.6

LOGICAL SLOC COUNTING RULES				
NO.	STRUCTURE	ORDER OF PRECEDENCE	LOGICAL SLOC RULES	COMMENTS
R01	"for", "while" or "if" statement	1	Count Once	"while" is an independent statement.
R02	*do {…} while (…); statement*	2	Count Once	Braces {…} and semicolon ; used with this statement are not counted.
R03	Statements ending by a semicolon	3	Count once per statement, including empty statement	Semicolons within "for" statement are not counted.Semicolons used with R01 and R02 are not counted.
R04	Block delimiters, braces {…}	4	Count once per pair of braces {..}, except where a closing brace is followed by a semicolon, i.e. };or an opening brace comes after a keyword "else".	Braces used with R01 and R02 are not counted.Function definition is counted once since it is followed by {…}.
R05	Compiler Directive	5	Count once per directive	

3. Examples

EXECUTABLE LINES

SELECTION Statement

ESS1 – if, else if, else and nested if statements

GENERAL EXAMPLE	SPECIFIC EXAMPLE	SLOC COUNT

if <expression>:
        <statements>
	if password == "pass":
	    print "Access Granted"
		1
		1

if <expression>:
        <statement>
else:
        <statement>
	if password == "name":
	    print "Access Granted"
	else:
	    print "Access Denied"
		1
		1
		0
		1

if <expression>:
        <statements>
elif <expression>:
    <statements>
else:
    <statements>
	if num > 0:
	    print 'positive'
	elif num < 0:
	    print 'negative'
	else:
	    print 'zero'
		1
		1
		1
		1
		0
		1

if <expression>:
        <statements>
        <statements>
else:
        <statements>
 	
NOTE: complexity is not considered
	if x < 0:
	      x = 0
	      print 'Negative'
	else:
	      print 'Positive'
		1
		1
		1
		0
		1

ESS2 – try-except-finally

GENERAL EXAMPLE	SPECIFIC EXAMPLE	SLOC COUNT

try:
        <do something>
except Exception:
    <handle the error>
finally:
    <cleanup>
	try:
	    try: 1/0
	    except:
	       print "exception"
	except ZeroError:
	    print "divide-by-0"
		1
		1
		1
		1
		1
		1

ITERATION Statement

EIS1 – for

GENERAL EXAMPLE	SPECIFIC EXAMPLE	SLOC COUNT

for <expression>:
    <statement>
 	
NOTE: “for” statement counts as one, no matter how many optional expressions it contains
	for x in a:
	  print x,
	
	for x in a:
	{
	  print ‘x’
	}
		1
		1
		
		1
		0
		1
		0

EIS2 – while

GENERAL EXAMPLE	SPECIFIC EXAMPLE	SLOC COUNT

while <boolean expression>:
  <statement>
	while x <= 100:
	   print x
	   x += 1
		1
		1
		1

JUMP Statement

EJS1 – return

GENERAL EXAMPLE	SPECIFIC EXAMPLE	SLOC COUNT

return *expression*	def knights():
	    title = 'Sir'
	    action = (lambda x: title + ' ' + x)
	    return action
	act = knights()
	print act('robin')
		1
		1
		1
		1
		1
		1

EJS2 – break

GENERAL EXAMPLE	SPECIFIC EXAMPLE	SLOC COUNT

break	for x in range(1, 11):
	  if x == 5:
	    break
	  print x,
	print "\nBroke out of loop at x =", x
		1
		1
		1
		1
		1

EJS3 – exit function

GENERAL EXAMPLE	SPECIFIC EXAMPLE	SLOC COUNT
os.exit(int return_code)	def outahere():	1
	import os	1
	print 'Bye os world'	1
	os._exit(99)	1
	print 'Never reached'	1
	if __name__ == '__main__': outahere()	2

EJS4 – continue

GENERAL EXAMPLE	SPECIFIC EXAMPLE	SLOC COUNT
	for x in range(1, 11):	1
	if x == 5:	1
continue	continue	1
	print x,	1
	print "\nUsed continue to skip printing the value 5"	1

EXPRESSION Statement

EES1 – function call

GENERAL EXAMPLE	SPECIFIC EXAMPLE	SLOC COUNT
<function_name> (<parameters>);	read_file (name);	1

EES2 – assignment statement

GENERAL EXAMPLE	SPECIFIC EXAMPLE	SLOC COUNT
assignment_stmt = (target_list "=")+ expression_list	x = y	1
	name = “file1”	1
target_list = target ("," target)* [","]	a, b, c = 1,2,3	1
target = identifier | "(" target_list ")" | "[" target_list "]" | attributeref | subscription | slicing		

EES3 – empty statement (is counted as it is considered to be a placeholder for something to call attention)

GENERAL EXAMPLE	SPECIFIC EXAMPLE	SLOC COUNT
Pass	if month == 1:	1
	pass	1
	else:	1
	print "late"	1

EES4 – Explicit line joining

GENERAL EXAMPLE	SPECIFIC EXAMPLE	SLOC COUNT
<expression> \	bar = 'this is ' \	1
<expression> \	'one long string ' \	
<expression>	'that is split ' \	
	'across multiple lines'	
	print bar	1

EES5 – Implicit line joining (Expressions in parentheses, square brackets, or curly braces can be split over more than one physical line without using backslashes)

GENERAL EXAMPLE	SPECIFIC EXAMPLE	SLOC COUNT
(<expression>,	day = ['mon', 'tue',	1
…	'wed', 'thur', 'fri',	
<expression>)	'sat', 'sun']	
[<expression>,	def node(name):	1
…	return {	1
<expression>]	'Parent' : None,	
	'LeftChild' : None,	
{<expression>,	'RightChild' : None,	
…	'LeftRoutingTable' : list(),	
<expression>}	'Name' : name,	
	'Level' : 0	
	}	0

DECLARATION OR DATA LINES

DDL1 – **class**

GENERAL EXAMPLE	SPECIFIC EXAMPLE	SLOC COUNT
class ClassName:	class MyClass:	0
<statement-1>	i = 12345	1
.	def f(self):	1
.	return 'hello'	1
.		
<statement-N>		