Assignment Kit for Size Counting Standard

Personal Software Process for Engineers: Part I

The Software Engineering Institute (SEI)
is a federally funded research and development center
sponsored by the U.S. Department of Defense and
operated by Carnegie Mellon University.

Personal Software Process for Engineers: Part I

Assignment Kit for the Size Counting Standard

Overview

Overview

This assignment kit covers the following topics.

Prerequisites

Prerequisite reading

- Chapter 3

Objectives

The objectives of Size Counting standard are to

- define the size counting standards that are appropriate for the programming language and environment that you will use
- provide a basis for developing a coding standard
- prepare for developing a program to count program size

Size counting standard requirements

Size counting standard requirements

Produce and document a standard for counting program size for the language and environment that you will use in this course.

Submit the completed standard using the format in the template on page 11 with your program 2 assignment package.

Example 1: Pascal size counting standard

Overview

The template is a simplified version of the SEI measurement framework.
- Use this template to describe important items.
- Tailor it to fit your needs or language.

We'll walk through two example size counting standards. The first example is for logical LOC for Pascal programs.

Completing the header

Complete the header as follows:
- the name you give this standard
- the language you are using
- your name
- the date you produced this standard (or revision)

Count type

Choices are logical and physical LOC.
- Logical LOC counts language elements.
- Physical LOC counts text lines.

For this counting standard, you are counting logical LOC.

Statement type

Use this section to define how you will count various types of statements. Consider the following:
- How are you going to count procedure declarations and function prototypes?
- How will you count compiler directives?
- Will you count blank lines or comments? Why or why not?

Clarifications

A fully operational standard generally requires many notes and comments. Use the clarification section for this purpose.

Continued on next page

Pascal LOC Counting Standard Template

Definition Name:	Example Pascal LOC Std.	Language:	Pascal
Author:	W. S. Humphrey	Date:	12/20/93

Count Type	**Type**	**Comments**
Physical/Logical	Logical	
Statement Type	**Included**	**Comments**
Executable	yes	
Nonexecutable:		
Declarations	yes	
Compiler Directives	yes	
Comments	no	
Blank lines	no	
Other elements		
Clarifications		**Examples/Cases**
Nulls	yes	continues, no-ops, ...
Empty statements	yes	";;", lone ;'s, etc.
Generic instantiators		
Begin...end	note 1	when executable
Begin...end	note 1	when not executable
Test conditions	yes	
Expression evaluation	yes	when used as sub program arguments
End symbols	notes 1,2	when terminating executable statements
End symbols	notes 1,2	when terminating declarations or bodies
Then, else, otherwise	note 1	
Elseif	note 1	
Keywords	notes 1,2	procedure division, interface, implementation
Labels	yes	branch destinations when on separate lines
Note 1		unless followed by ; or. or included in {}, count the following keywords once: BEGIN, CASE, DO, ELSE, END, IF, RECORD, REPEAT, THEN, UNTIL
Note 2		count every ; and . that is not within a {} or ()
Note 3		count each , between USES and the next ; or between VAR and the next ;

Example 2: C++ LOC counting standard

How many LOC?

Using the C++ LOC counting standard on page 7, how many LOC are in the following program fragment?

```
#include <stdio.h>

void main (void)
{
        int i,j;

        for (i = 0; i < 10; i++)
                for (j = 0; j < 10; j++)
                        cout << i << “, ” << j << endl;
}
```

Continued on next page

Example C++ Size Counting Standard

Definition Name:	Example C++ LOC std.	Language:	C++
Author:	W.S. Humphrey	Date:	12/20/93

Count Type	**Type**	**Comments**
Physical/Logical	Logical	
Statement Type	**Included**	**Comments**
Executable	Yes	
Nonexecutable:		
Declarations	Yes,Notes 3, 4	
Compiler Directives	Yes, Note 4	
Comments	No	
Blank lines	No	
Other elements		
Clarifications		**Examples/Cases**
Empty statements	yes	";;", lone ;'s, etc.
Begin...end	note 1	
Expression evaluation	yes	when used as sub program arguments
End symbols	notes 1,2	for terminating executable statements, declarations, bodies
Then, else, otherwise	note 1	
Elseif	yes	
Keywords	yes	procedure division, interface, implementation
Labels	yes	branch destinations when on separate lines
Note 1		Count once every occurrence of the following key words: CASE, DO, ELSE, ENUM, FOR, IF, PRIVATE, PUBLIC, STRUCT, SWITCH, UNION, WHILE
Note 2		count once every occurrence of the following: ; , {} or };
Note 3		count each variable or parameter declaration
Note 4		count once each #define, #ifdef, #include, etc. statement

Example 3: another size counting standard

Overview

For this example standard, the class will select from among the following product categories.

- documents
- interface screens and forms
- database program elements
- maintenance fixes
- any other requested category

We'll then walk through developing the selected size counting standard.

Completing the standard

We'll then walk through the completion of the standard as a class exercise.

Continued on next page

Size Counting Standard Template

Definition Name: _____ Language: _____
Author: _____ Date: _____

Count Type	**Type**	**Comments**
Physical/Logical		
Statement Type	**Included**	**Comments**
Executable		
Nonexecutable:		
Declarations		
Compiler Directives		
Comments		
Blank lines		
Other elements		
Clarifications		**Examples/Cases**
Note		
Note		
Note		
Note		

Evaluation criteria and suggestions

Evaluation criteria	Your standard must be • complete • legible
Suggestions	Keep your standards simple and short. Do not hesitate to copy or build on the PSP materials.

Size Counting Standard Template

Definition Name: _____ Language: _____
Author: _____ Date: _____

Count Type	**Type**	**Comments**
Physical/Logical		
Statement Type	**Included**	**Comments**
Executable		
Nonexecutable:		
Declarations		
Compiler Directives		
Comments		
Blank lines		
Other elements		
Clarifications		**Examples/Cases**
Note		
Note		
Note		
Note		