PSP Familiarization Exercise 3: Program Size

Purpose

This exercise shows you how to count lines of code.

Contents

The exercise package contains a line-of-code counting standard and the source code for a small Pascal program.

Directions

First, read the line-of-code counting standard and mark every line that should be counted on the listing. Then count and number the lines you have marked. When you are finished, the instructor will hand out the exercise answer and check to see whether everyone got the correct answer. Ask about any differences and make sure you know how to count lines of code.

Instructor's Note:

If the students do not know the Pascal language, you will have to provide a LOC counting standard and sample program listing in a language they know.

Line-of-Code Counting Standard: Pascal

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Purpose</td>
<td>To guide engineers in determining the size of Pascal source programs.</td>
</tr>
</tbody>
</table>
| General       | ● Line-of-code (LOC) counting standards are arbitrary and language-dependent.  
                  ● For consistent size measurement, all software team members should use a single LOC counting standard.                          |
| Parentheses   | No statement or punctuation marks within '( š )' are counted.                                                                                  |
| Brackets      | No statement or punctuation marks within '{ š }' are counted.                                                                                   |
| Semicolon     | Every occurrence of a ';' is counted once.                                                                                                     |
| Period        | Every occurrence of a '.' that follows a terminating END statement is counted once.                                                            |
| Key Words     | Every occurrence of the following selected key words is counted once: BEGIN, CASE, DO, ELSE, END, IF, RECORD, REPEAT, THEN                        |
| Special       | Where there is no ';' at a line end, every statement preceding the following key words is counted once (if not already counted): ELSE, END, UNTIL |
| Comma         | Every occurrence of a ',' in the USES or VAR portions of the program is counted once.                                                          |
Program

1 {Spelling procedure; WSH; 06/29/91}
2 {Purpose: This procedure converts positive or zero integers to strings.}
3 {Usage instructions:
4 Call: SPELLING (YourNumber: integer; YourResult: string; YourError: 0)
5 Return: YourNumber: unchanged
6 If YourNumber negative, YourResult: blank character string
7 YourError = 1
8 If YourNumber >0 or 0, YourResult: string of characters for the number
9 YourError = 0}
10 {Implementation for the spelling procedure.}
11 PROCEDURE SPELLING(var InputInteger: integer; var SomeString: string;
12 var SomeError: integer);
13 var i, {index}
14 Length, {maximum number length}
15 Test, {a trial number}
16 SomeInteger, {a trial number}
17 N, {tens variable}
18 Offset: integer; {offset value to convert digits to ASCII characters}
19 Flag: boolean; {to show leading 0s}
20 SpelledNumber : string; {trial string, becomes output}
21 begin
22 YourError = 0;
23 SomeInteger := InputInteger;
24 SomeString := ";
25 Test := 0;
26 SpelledNumber := ";
27 N := 10000;
28 Length := 6;
29 Offset := 48;
30 i := 1;
31 Flag := false;
32 {Next, cycle through the integer from highest-order to lowest-order digits to
33 determine digit's value and add its character to SpelledNumber.}
34 if SomeInteger < 0 {Does not work for negative integers.}
35 then
36 SomeError := 1
37 else
38 repeat {until i := Length}
39 if SomeInteger > N {Is first digit >= 1?}
40 then
41 begin
42 Test := SomeInteger;
Test := Test div N; {Find the leading digit value.}
SomeInteger := SomeInteger mod(Test*N); {Eliminate the leading digit.}
Test := Test + Offset; {Set digit to its ASCII value.}
SpelledNumber := SpelledNumber + chr(Test); {Add the character to end of SpelledNumber.}
Flag := true; {Set flag to take care of non-leading zeros.}
end
else {Check for zero characters.}
  if Flag
    then {If an internal character was 0, insert '0'.}
      SpelledNumber := SpelledNumber + '0'
  else {If 0 or no character in leading positions, leave blank.}
    SpelledNumber := SpelledNumber + ' ';
  i := i + 1; {Step to the next digit.}
N := N div 10
until i = Length; {Continue until all digits tested}
SomeString := SpelledNumber; {Set output to resulting string}
end; {PROCEDURE SPELLING }

<- Exercise 3: Program Size

Exercise 3: Pascal Source Program

Exercise 4: Task and Schedule Planning ->

Contents  List of Forms  Tool Instructions