ABSTRACT

This paper illustrates the use of the PARMBUFF option in calculating the number of parameter values (N) supplied in a macro call. Once N is calculated, a predefined global macro variable is defined for each respective value of N. The primary advantage of using this macro with the PARMBUFF option is that a user need only supply the appropriate parameters. The macro Scanner calculates the value of N and subsequently invokes the user's macro via the %&VAR statement.

INTRODUCTION

When invoking a macro, parameter values are supplied to the macro by use of positional or keyword parameters. For some tasks, the number of parameter values may vary and cannot be predicted prior to invocation. Moreover, certain tasks may need to be performed an unknown number of times (N). These problems can be alleviated by using an iterative DO loop. However, when a large number of parameter values need to be supplied to a macro, iterative DO loops become tedious and inefficient. In these cases, keyword parameters become the parameters of choice for two reasons: one being for clarification, and the other to supply the parameter list with the maximum number (N) of parameter values. The Scanner macro presented in this paper shows how the use of the PARMBUFF option can be instrumental in calculating the value of N by counting the number of parameters that is passed to the macro.

MACRO SCANNER

A macro to generate title statements may be written as follows:

%MACRO TITLE(N,T1=T1,T2=T2,...,T10=T10);
  %DO i=1 %TO &N;
    TITLE&i="&T&i"
  %END;
%MEND TITLE;

Since the maximum number of TITLE statements that the SAS® System allows is ten, the parameter list uses keyword parameters to accommodate all ten TITLE statements even though only one may be used. This preset limit of ten allows this macro to be placed in the autocall facility for use with no revisions. This macro may be invoked in the following ways:

%TITLE(2,T1=ONE,T2=TWO)
%TITLE(4,T1=ONE,T2=TWO,T3=THREE,T4=FOUR)
ETC.

Note that the first parameter is a positional parameter which serves as the ending value in the DO loop. If the value of N is two, then two keyword parameters are defined in the call. Likewise, if the value of N is nine, then nine keyword parameters are defined in the call. However, if the value of N is eleven or greater, an error would result as the macro parameter list only accommodates ten titles. Moreover, the SAS System restricts the number of TITLE statements to ten.

In situations where the SAS System does not restrict the number of SAS statements, "padding" the parameter list with keyword parameters can be tedious. Suppose we use the INT function of base SAS software to truncate the decimal portion of the value of fifty variables. The macro to perform this task may look something like this:

%MACRO TRUNCATE(N,INT1=,INT2=, ...,INT50=);
  %DO i=1 %TO &N;
    INT&i=INT(&INT&i);
  %END;
%MEND TRUNCATE;

In contrast to the %TITLE macro, the value of N for macro %TRUNCATE is not restricted, that is, N is limitless. As can be seen, when the value of N is large, "padding" the parameter list is undesirable. The use of the PARMBUFF option eliminates not only the "padding" of the parameter list with keyword parameters, but also the passing of the value of N to end the DO loop.

859
Writing Macros with the PARMBUFF Option

Anthony B. Kreamer, Weston

As illustrated below, macro scanner uses the PARMBUFF option to assign the entire parameter list to the automatic local macro variable SYSPBUFF. Once the variable SYSPBUFF is defined, other macro functions are used to scan the value of SYSPBUFF and calculate the number of parameters in the macro call. The parameter number is then assigned to the value of N. Macro %SCANNER was written in such a manner that a user unfamiliar with the macro could invoke it as any other name-style macro call.

%MACRO SCANNER / PARMBUFF;
%GLOBAL N VAR;
%LOCAL COMMA LIST POS;
%LET LEN=LENGTH(&SYSPBUFF);
%LET POS=INDEX(&SYSPBUFF,=);
%LET LIST=SUBSTR(&SYSPBUFF,REVAL(&POS+1),
REVAL(LEN-(REVAL(&POS+1))));
%LET COMMA=
%LET INC=FIRST PASS;
%LET N=1;
%LET POS=INDEX(SQUOTE(&LIST),XQUOTE(&COMMA));
%DO WHILE (APPOS NE 0):
%LET "AIRLOOK"=ED FIRST PASS
THEN XDO;
%LET VAR=SUBSTR(XQUOTE(&LIST),1,
REVAL(REVAL(&POS-1)));
%LET INC=NOT FIRST PASS
THEN XDO;
%ELSE XDO;
%GLOBAL VARN;
%LET VARN=SUBSTR(XQUOTE(&LIST),1,
REVAL(REVAL(&POS-1)));
%LET N=REVAL(&N+1);
%END;
%LET POS=INDEX(SQUOTE(&LIST),XQUOTE(&COMMA));
%LET LIST=SUBSTR(XQUOTE(&LIST),REVAL(&POS+1));
%END;
%GLOBAL VARN;
%LET VARN=SUBSTR(XQUOTE(&LIST),1);
%INPUT VAR = VARN;
%INPUT N = &N;
%INPUT POSS = &VARN;
%END;
%GLOBAL VAR
%GLOBAL VAR
%GLOBAL VAR
%INPUT POSS = &VARN;
%END;
%VAR
%END SCANNER;

Macro Scanner is invoked by the following call:

%SCANNER(MACRO=TITLES,TITTLE ONE,TITTLE TWO)
%SCANNER(MACRO=TITLES,ONE,TWO,3,4,5,SIX)

Scanner subsequently invokes macro %TITLES (%&VAR). This is illustrated by the MACRO=TITLES in the call to scanner. At first glance, one may think that there are three parameters in the call, that is, a keyword parameter (MACRO=TITLES) followed by two positional parameters. Thus, an error would ensue because positional parameters must always come before keyword parameters. However, since the PARMBUFF option is in effect, the entire parameter list is treated as one parameter and assigned to the variables SYSPBUFF.

Now, the macro to generate title statements may read as follows:

%MACRO TITLES;
%DO I = 1 %TO &N;
TITLE&I "&VAR&I";
%END;
%END TITLES;

Note that there is no parameter list in the macro definition. Macro Scanner creates global variables N, VAR1, VAR2, ..., VAR&N internally and macro %TITLES uses these to generate SAS code.

SAS is a registered trademark of the SAS Institute., Cary, NC, USA.

AUTHOR'S ADDRESS:
Anthony B. Kreamer
ROY F. WESTON, INC.
Weston Way
West Chester, PA 19380