CBCS
http://agrobiol.sggw.waw.pl/cbcs

Communications in Biometry and Crop Science
Vol. 4, No. 1, 2009, pp. 1–2

International Journal of the Faculty of Agriculture and Biology,
Warsaw University of Life Sciences, Poland

SOFTWARE TRICKS AND TIPS

SAS Macro Variables

Macro variables are part of the SAS/BASE installation. While they are used in the SAS Macro Language they are also very useful in everyday SAS programs. Unfortunately many web-based help sites start with rather silly examples only to move to complicated problems without giving the searching user a chance to get used to the idea. But even a novice can deploy macro variables successfully by obeying four simple rules.

(1) A macro variable is defined using the % **LET** statement. No quotation marks are needed for the string.

(2) A macro variable called in a program by preceding it with the ampersand **(&)** symbol. Upon encountering a macro variable, SAS will replace it with the string defined in step 1.

(3) Double quotes **(")** are used if the macro variable replaces a string enclosed in quotation marks as in the LIBNAME statement below.

(4) In compound strings (see DATAFILE statement below) macro variables are terminated with a **period**. SAS will then interpret the remainder as a regular string.

I have found macro variables very useful when clients bring their analysis projects on removable data storage devices (USB sticks). Because the Windows operating system uses relative addressing for such devices, the drive letter can and often will change from session to session. Defining the path at the beginning of a program through a macro variable will result in easier access, cleaner programming, and fewer headaches.

Here is a small program to demonstrate the utility of a macro variable. In the standard SAS color scheme for the programming window, terms colored in blue are SAS commands; black indicates the information provided by the user, and green program annotations.

```
OPTIONS symbolgen;
/*The symbolgen option forces SAS to write the resolution of the macro variable to the log window*/
 %LET path = G:\MYPROJECT\;
 LIBNAME RICE "&path";
 PROC IMPORT DATAFILE = "&path.SASdata.xls"
      OUT = RICE.SASdata DBMS = excel2000 REPLACE;
      RANGE = mydata;
RUN;
PROC PRINT; RUN;
```

If the drive letter changes, a simple adjustment in the definition for macro variable 'path' will take care of both the library assignment as well as the spreadsheet location. Macro variables have multiple uses, some of which I will describe in the next issue.

contributed by Edzard van Santen

Forage Breeding and Genetics, Dept. of Agronomy and Soils,
Auburn University, AL 36849-5412.
E-mail: vanedza@auburn.edu

Published online: 19 February 2009