Stump the Programmer #62 – Understanding DATA Steps with Two SET Statements #1

By Art Carpenter

Most of us do not take advantage of the power of having two SET statements in a single DATA step. Perhaps for good reason – if we do not understand the processing flow of the DATA step. In the DATA step that creates the data set WORK.BOTH, answer the following:

1) How many observations will this data set contain and what will be the values of X and Y.
2) How many times will the word 'HERE' be written to the LOG?
3) How many times will the word 'THERE' be written to the LOG?
4) Bonus question: Why doesn't the use of the variable DONE cause an uninitialized variable warning?

The answer to an expanded version of this 'Stump the Programmer' question will be presented at the one day SANDS SAS conference on December 7, 2012.

```
data one;
   x=1; output;
   run;
data two;
   y=1; output;
   y=2; output;
   run;
data both;
   put _n_= 'HERE';
   set one;
   put 'THERE';
   do until(done);
      set two end=done;
      output;
   end;
   run;
```

Stump the Programmer #63 – Double SET Statement (Part 2)

By Art Carpenter

In the previous Stump the Programmer (December 2012), we examined the behavior of a DATA step with two SET statements, one of which was embedded within a DO loop. Building on that understanding, take a look at the third DATA step shown here, where we again find two SET statements.

The two incoming data sets do *not* have the same number of observations. How many observations will be in the new data set(WORK.NEW)? And perhaps more importantly, which PUT statement executes last and with what values of X and Y?

```
data a;
   do x =1to 3;
   output a;
   end;
   run;
data b;
   do y = 1 to 2;
   output b;
   end;
   run;

data new;
   set a;
   put 'after A ' x= y=;
   set b;
   put 'after B ' x= y=;
   run;
```

Stump the Programmer #64 – Double SET Statement (Part 3)

By Art Carpenter

In the previous two Stump the Programmers (December 2012 and March 2013), we examined the behavior of a DATA step with two SET statements. Use what you learned in those two problems, and the program below to answer the following questions:

1) Are the data sets DOUBLESET and ONESET the same?
2) Will they have the same variables and observations?
3) Will either be equivalent to the original data set (SASHELP.CLASS)?

```
data Males;
   set sashelp.class(where=(sex='M'));
   run;
data Females;
   set sashelp.class(where=(sex='F'));
   run;

data doubleset;
   set females;
   set males;
   run;

data oneset;
   set females males;
   run;
```

The next meeting's Stump the Programmer puzzle will be the final exam on double SET statements. Here it is:

Using the two data sets (MALES and FEMALES) created above, use a DATA step to perform a many to many merge matching each female with each male.

Stump the Programmer #65 – Double SET Statement (Part 4)

By Art Carpenter

In the previous three Stump the Programmers, we examined the behavior of a DATA step with two SET statements. Use what you learned in those problems to write a many to many merge using the DATA step and two SET statements.

```
* Write a DATA step that matches each female with each male (many to
many merge) – think dating service.
* This is trivial in a SQL step, but will test your understanding of a
DATA step with two SET statements.
*
* The final data set should have 90 (10 X 9) observations and four
columns (MNAME, MAGE, FNAME, and FAGE).
*   [name and age for each gender]
*************************************************************************
*********************************;
data Males;
   set sashelp.class(where=(sex='M'));
run;
data Females;
   set sashelp.class(where=(sex='F'));
run;

data ManyMany;
...... your code here .....
   set females;
...... your code here .....
   set males;
...... your code here .....
   run;
```