ABSTRACT
This is a discussion on what items may be left behind when a user is running multiple SAS® programs during a single SAS session and how, using a macro that I have developed (and keep revising), the user can clean up the “garbage” so that some possible errors can be avoided. These errors could be as simple as incorrect titles and footnotes up to the wrong data being accessed.

INTRODUCTION
Multitasking—many of us do this in our day-to-day jobs. When programming using SAS® there are times when we work on many programs and/or projects at once. We may be in the midst of programming a table, when someone else will ask us to rerun a previous program with some minor modifications. Someone else wants an ad hoc request to make a graph out of some data. However, our LIBNAMES references and WORK directories are already set up with the data we originally intended to work on for the day. How can we get around this problem, making sure we are referencing the correct data, reading the correct logs, and reviewing the right output?

TAKE OUT THE GARBAGE
SAS will let us open multiple PC SAS sessions at once. Doing this, a user could then load up one program, run it, review the logs and output to make sure everything looks right, and send off the results to those that need it. This may suffice for some, but it could lead to many sessions being open at once, and could be a drain on resources depending on how much power your PC has. Multiple sessions could get confusing and make it difficult to find the session you need to run. Multiple sessions also leads to a user completely losing his/her profile on the second (and subsequent) open session.

An easier way may be to have one session open and load up the individual programs at once. Run the one you need, check outputs, and then close that particular program when you are done. This will leave much “garbage” behind as you try and run the next program. To make sure that you are using the correct data, and looking at the right log and output, you first want to make sure of a few things:

1. Your LIBNAMES references are correct.
2. You have cleared your log and output screens so that the results you get are for just the program you have run.
3. You have correctly reinitialized other items to certain values (e.g. titles are cleared out).

We need a way to get rid of all this previous “litter” so that we are starting with a clean slate.

SOLUTION
To take care of the various areas of SAS where “junk” can collect, I developed the %RESTART macro. I use it to clean up these potential problem areas.

When a user has a few programs open at once, various issues can come up that jeopardize getting the correct outcomes that are expected.

1. Are the LIBNAMES assigned to the correct paths that contain the data the user wants for this particular program, or are they pointing to data from another program? This especially becomes a problem if the user reuses names for program uniformity.
2. Is the log cleared so that when the program is run, all Log output will refer to only the program currently running?
3. Is the output screen cleared so that all output to be reviewed is relevant to only the program currently submitted?
4. Is the WORK library empty so that all intermediate data sets that are produced come from the current program?
5. Is the format catalog correctly formatting data for this program?
6. Are there any macros in the SASMACR library that need to be reassigned/recompiled?

%RESTART takes care of all these items so that when a user submits a program after running this macro, the user can be assured that all datasets, formats, logs, output, etc. that are produced are relevant to just the current program, and not a mish-mash of data from past programs and the current one.
DETAILS OF HOW THE MACRO WORKS
The macro defaults to clear/remove everything—logs, output screens, macros, etc. All of the parameters can be overwritten, and this is explained later.

CLEARING THE OUTPUT SCREEN AND THE LOG
The first, and easiest, “trash” to take out involves the Log and Output screens. We use the Display Manager statement to accomplish this. The DM statement submits SAS Program Editor, Log, Procedure Output or text editor commands as SAS statements. Two simple statements let us easily clear out the log and output screen.

```sas
dm log "clear";
dm output "clear";
```
Alternatively, you can use a different DM statement that will not only clear the output screen but will also remove any leftover “residue” in the results window.
```
Dm ‘odsresults’ clear;
```

CLEARING THE LIBNAME REFERENCES
Removing the LIBNAME references involved a DATA step that obtained all the LIBNAMES from the SASHELP.VSLIB view. Using a PROC SQL statement on the SASHELP.VSLIB view, the names were put into macro variables that were later called on to be removed by using the code: `libname &new&i clear; ‘&new&i’` was the macro variable reference containing the LIBNAME reference, and a loop was used to loop through all the references and clear out the “waste”.

CLEARING THE FORMATS
PROC DATASETS helps us to manage our files in a number of ways. We can copy files, rename files, delete files, etc. I used the procedure to clear out the formats that I may still have around from other programs. A simple PROC DATASETS step was used to access the format catalog and delete all the formats
```
proc datasets memtype=catalog;
   delete formats;
run;
quit;
```

DELETING TEMPORARY DATASETS
One area where a lot of “rubbish” is left is the WORK library. Every time a program is run and datasets are created, this library grows and grows. Using another simple PROC DATASETS statement, I accessed the WORK library and removed all the temporary datasets.
```
proc datasets lib=work memtype=data kill;
run;
quit;
```

REMOVING MACROS FROM THE SASMACR CATALOG
The CATALOG procedure is used to manage entries in your SAS catalogs. You can create listings, copy a catalog, etc. Using PROC CATALOG, I removed all the macros in the WORK.SASMACR folder with the exception of the %RESTART macro that we are currently using. One thing to keep in mind is that this step may not be needed if this macro (and others) are compiled and loaded in your autoexec file. If that is the case, then you may not need to do this step, or you may not have to include the save statement since the macro is not being used from the SASMACR catalog.
```
proc catalog catalog=work.sasmacr force;
   save restart/et=MACRO;
run;
quit;
```

REMOVING GLOBAL MACRO VARIABLES
Sometimes global macro variables are initialized, and they usually stay around until the end of the session. If there are global macro variables in the SASHELP.VMACRO view that we wish to clear, we use a PROC SQL procedure, along with %symexist and %symdel, to keep a list of the global macros we wish to remove and delete that list.
```
proc sql noprint;
   select name into :KeepList separated by ','
   from sashelp.vmacro
   where upcase(scope)='GLOBAL';
```
run;
quit;
%if %symexist(Keeplist)=1 %then %symdel &Keeplist KeepList;

REMOVE THE TITLES AND FOOTNOTES
There are many times when I forget to remove titles and footnotes from previous programs. I would run a program and my output would contain a title and/or a footnote that had no bearing on what I was presently doing. Reassigning the first title and footnote statements to null values takes care of removing ALL titles and footnotes potentially left from a previous program.

    title1;
    footnote1;

KEEPING CERTAIN PARAMETERS
The macro has a bunch of default parameters which are all set to ‘Y’, which means everything will be removed; however, the macro is flexible enough so that if a user would like to keep certain items, they can change a certain parameter(s) and then the particular data or screen will not be affected. You should make sure that you know what items you want to remove and which items you may want to save ahead of time.

For example, when you submit a program you get some output. You may then realize you want to resubmit the program, but wish to keep the previous output along with the output from the second run. Changing one parameter that is passed in the macro will allow you to do this (in this case changing the outputclear parameter to an N will not clear the output produced from the first run).

EXPANSION
The %RESTART macro has the added benefit of being able to be expanded on. When first developed, I was only clearing out titles, footnotes, logs, and the output screen. But as I came to think of other areas that I would want to clean up, it became apparent that this macro could be added to and made more powerful. I have since added a lot of functionality to it, yet I do not think that it is now “finished”. As SAS comes out with new functionality, it is possible that I may, or other users may, want to add pieces to the macro. This is highly encouraged.

CONCLUSION
Working on multiple projects seems to be the norm of today—running programs, revising parameters, analyzing output, ad hoc requests, etc. Minimizing the possible ways that the right program(s) is working with the wrong data is paramount to getting final, correct results to clients, whether internally or externally. The %RESTART macro is a tool to help with this potential issue. With it, we can help remove all types of “debris” that may be left over from previous programs.

ACKNOWLEDGEMENTS
I would like to thank Howard Proskin, who encourages all of his employees to grow through a professional development initiative he implemented; Mike Molter, my former manager, Ken Borowiak, a former colleague, and both William Murphy and William Csont, present colleagues, all of whom helped critique this paper.

Contact information
Adam LaManna
Howard M. Proskin and Associates, Inc.
300 Red Creek Drive
Suite 220
Rochester, NY 14623-4263
585-359-2420
alamanna@hmproskin.com

SAS and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.