Paper S1-02-2013

SAS®: The Power of Macros

Audrey Yeo, Aviva USA, West Des Moines, Iowa

Abstract
The SAS® Macro facility is an extremely powerful tool that should be in the toolbox of every SAS programmer. However, without some proper training it is difficult to implement, or when it is implemented it often results in hard to understand code. On the other hand once you have mastered the macro facility, it opens up a whole new world. This paper will show some examples of creating using SAS macros to help simplify coding and reduce repetition or duplication of code.

Introduction

This paper will give some simple examples on what Macros can do for us.

Advantages of using Macros

Macros are extremely useful tools. It allows us to update codes easily. For example, if we have a code that creates multiple reports with dates as part of the title of the reports, it can be quite troublesome to go through the whole code to update the dates (assuming that we're creating the code on a daily, monthly, or yearly basis). We might even miss some of the dates that need to be updated. Macros will allow us to change the date in just one location and automatically updates the other date locations.

Also, Macro codes are transferable. Sometimes we find ourselves having to rewrite a code that was written before in another SAS code, and we know that this part of the code will be re-written again in SAS code C, SAS code D, and so on. We can create Macro codes so that we do not have to rewrite that part of the code, instead, we could just call the Macro functions in the other codes. This allows us to transfer part of a code from one code to another code easily.

Furthermore, if we have a long code that consists of repetitive codes, Macros can help us shorten the long codes by removing those repetitive codes. In addition, Macros allows us to apply logical statements **on** DATA steps (vs. within DATA steps) and finally, using Macros makes us sound smart.

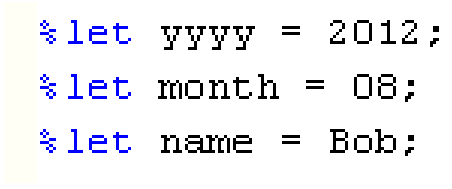
Macro Variables and Macro Statements

Macro variables are used to store and manipulate character strings. They follow the same naming rules as SAS. Macro variables are stored in memory.

Macro statements begin with a % sign, followed by a macro keyword and ends with a semicolon. It is used to assign values, substitute values and change macro variables.

%let *macro variable keyword* = assignment value;

Examples of macro variables and macro statements are as follows:



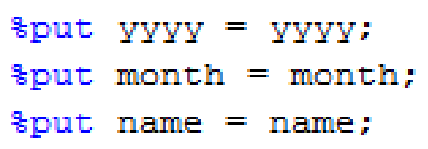
```
%let yyyy = 2012;
%let month = 08;
%let name = Bob;
```

In the examples above, we are setting macro variable keywords yyyy = 2012; month = 08; and name = Bob. Next step is to check and see whether the macro keywords were set to values that we've assigned to them.

Displaying Macro Variables

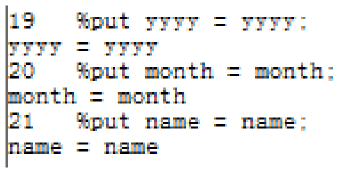
The %put statement will display the macro variable in the log window. %put is the easiest way to display and debug macros. In the first set of code and log output below, we used the %put statement to check the macro variables and we see in the corresponding log output, the output is not what we have assigned earlier.

Code1:



```
%put yyyy = yyyy;
%put month = month;
%put name = name;
```

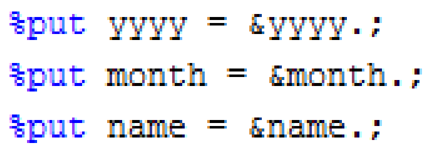
Log1:



```
%put yyyy = yyyy;
yyyy = yyyy
%put month = month;
month = month
%put name = name;
name = name
```

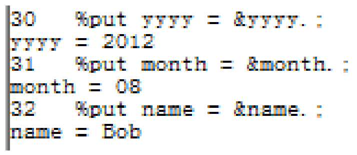
This is because, in order to activate the substitution, we have to put an ampersand (&) sign in front of the macro variable keyword. This can be seen in the second set of code and log output below. We see that by adding the ampersand sign before the macro variable keyword, the substitution was activated.

Code2:



```
%put yyyy = &yyyy.;
%put month = &month.;
%put name = &name.;
```

Log2:

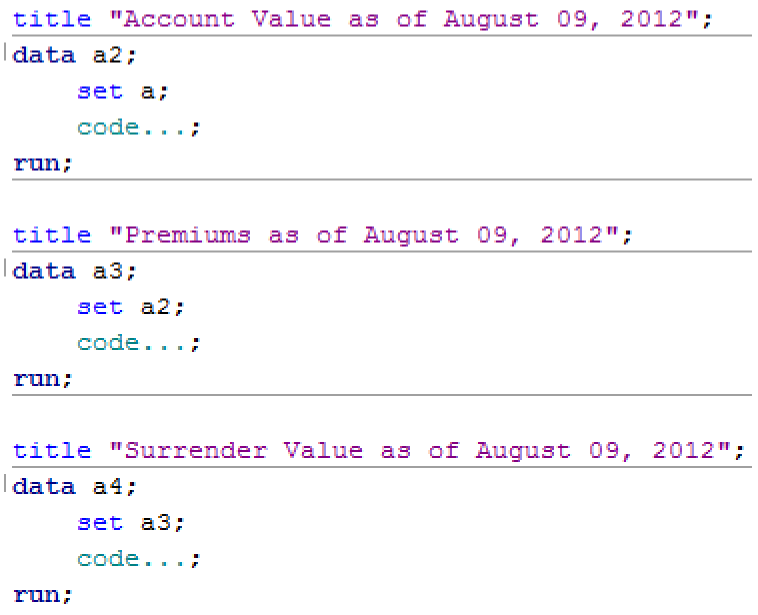


```
%put yyyy = &yyyy.;
yyyy = 2012
%put month = &month.;
month = 08
%put name = &name.;
name = Bob
```

Easy to Update Codes

This example shows how easy it is to update codes using Macros. Looking at code3 below, we see some repetitions in the code, specifically, the dates. This will work if we are producing the report only once.

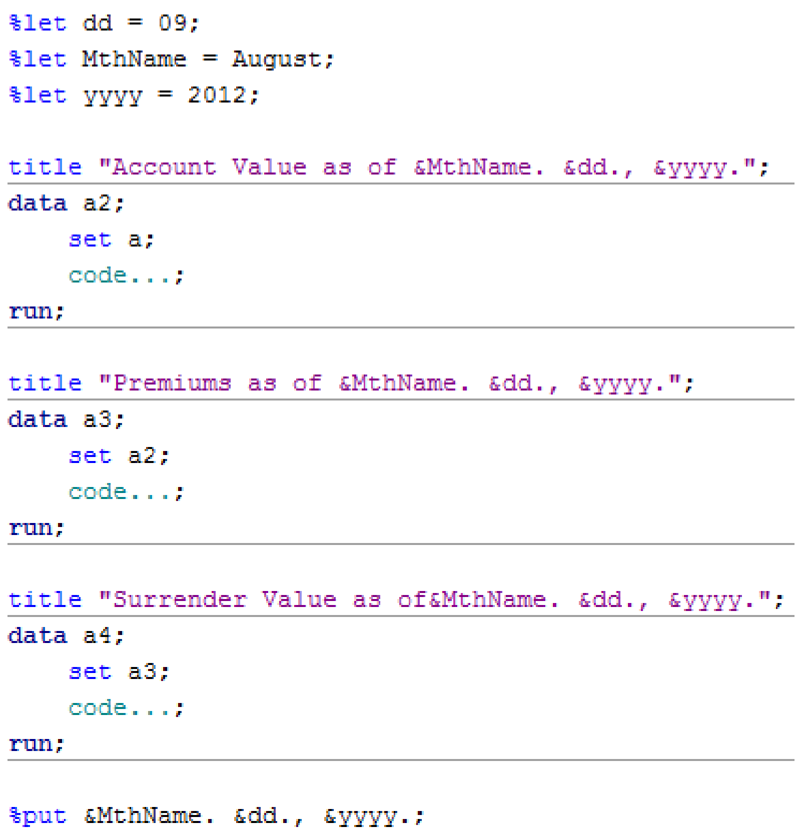
Code3:



```
title "Account Value as of August 09, 2012";
data a2;
     set a;
     code...;
run;

title "Premiums as of August 09, 2012";
data a3;
     set a2;
     code...;
run;

title "Surrender Value as of August 09, 2012";
data a4;
     set a3;
     code...;
run;
```

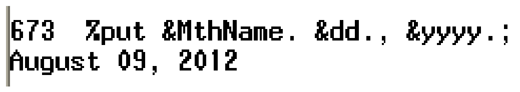
However, if this is not the case, and we need to create this report on a monthly basis, we have to go through the whole code to change the dates, and there may be times where we might miss some parts of the code. This is where Macros come in handy.

Code4:



```
%let dd = 09;
%let MthName = August;
%let yyyy = 2012;

title "Account Value as of &MthName. &dd., &yyyy.";
data a2;
    set a;
    code...;
run;

title "Premiums as of &MthName. &dd., &yyyy.";
data a3;
    set a2;
    code...;
run;

title "Surrender Value as of&MthName. &dd., &yyyy.";
data a4;
    set a3;
    code...;
run;

%put &MthName. &dd., &yyyy.;
```

Log4:

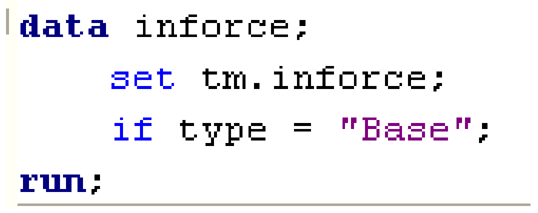


```
673  %put &MthName. &dd., &yyyy.;
August 09, 2012
```

Making Macro Codes Transferable

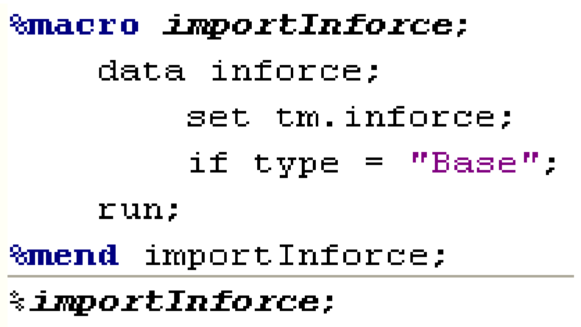
It can be intimidating to create a macro code if you have never tried that before. An easy way to create macro code is to first write the code as it is, as shown in code5. Once you have the code that you want to use repetitively, add a %macro *macro-code-name* at the beginning of the code and a %mend *macro-code-name* at the end of the code, as shown in the macro code below. That is how we create macro codes. To call the macro code, we just need to put a percent (%) sign in front of the *macro-code-name*.

Code5:



```
data inforce;
    set tm.inforce;
    if type = "Base";
run;
```

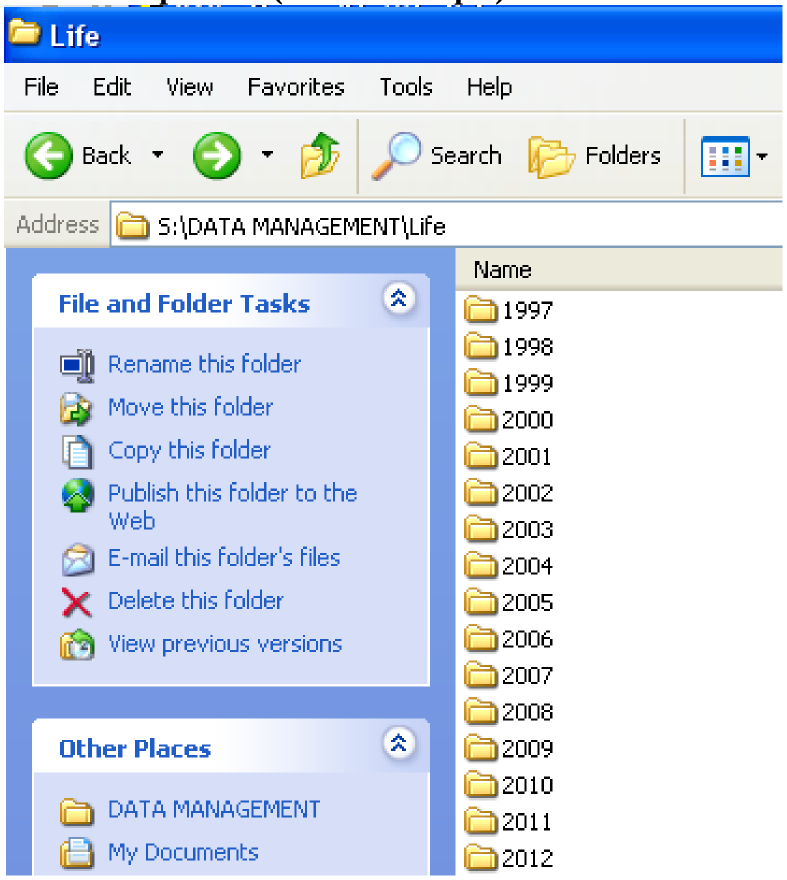
Macro Code1:



```
%macro importInforce;
    data inforce;
        set tm.inforce;
        if type = "Base";
    run;
%mend importInforce;
%importInforce;
```

Now we have a code that can be used repetitively without retyping the whole code. All we need to do is to call the macro code (%importInforce, in the example above) whenever you need it.

Remove Repetition (Folder example)



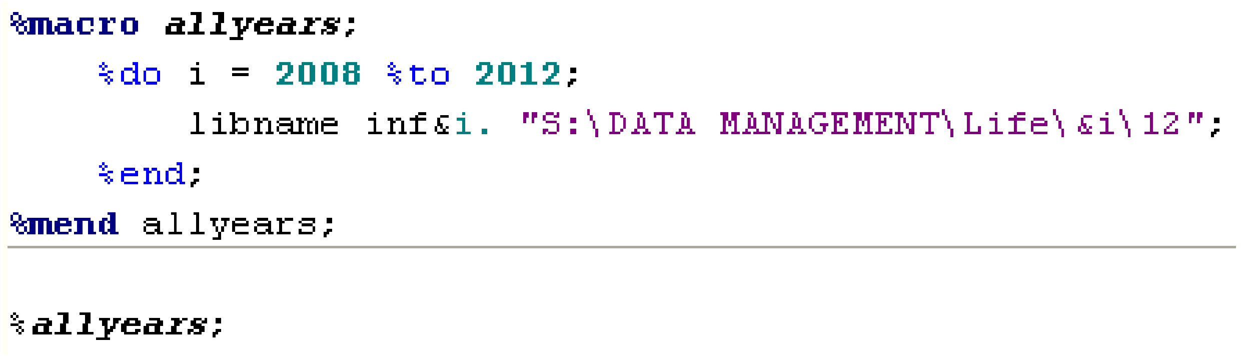
We have a folder that looks like the figure above, and in each folder, we have datasets that we want to access. In order to do this, you can type out the following code.

Code6:

```
libname inf2008 'S:\DATA MANAGEMENT\Life\2008\12';
libname inf2009 'S:\DATA MANAGEMENT\Life\2009\12';
libname inf2010 'S:\DATA MANAGEMENT\Life\2010\12';
libname inf2011 'S:\DATA MANAGEMENT\Life\2011\12';
libname inf2012 'S:\DATA MANAGEMENT\Life\2012\12';
```

Again, the code above works if we only need five years worth of information (2008 – 2012). What if we need ten, twenty, or even thirty years worth of information? Typing all of it will make it a long code. We could even mistype something. So what can we do then? Well, we can use macro and a do loop to remove the repetition. We see in code6 above that everything is the same except for the year, and the year is from 2008 to 2012.

Macro Code2:



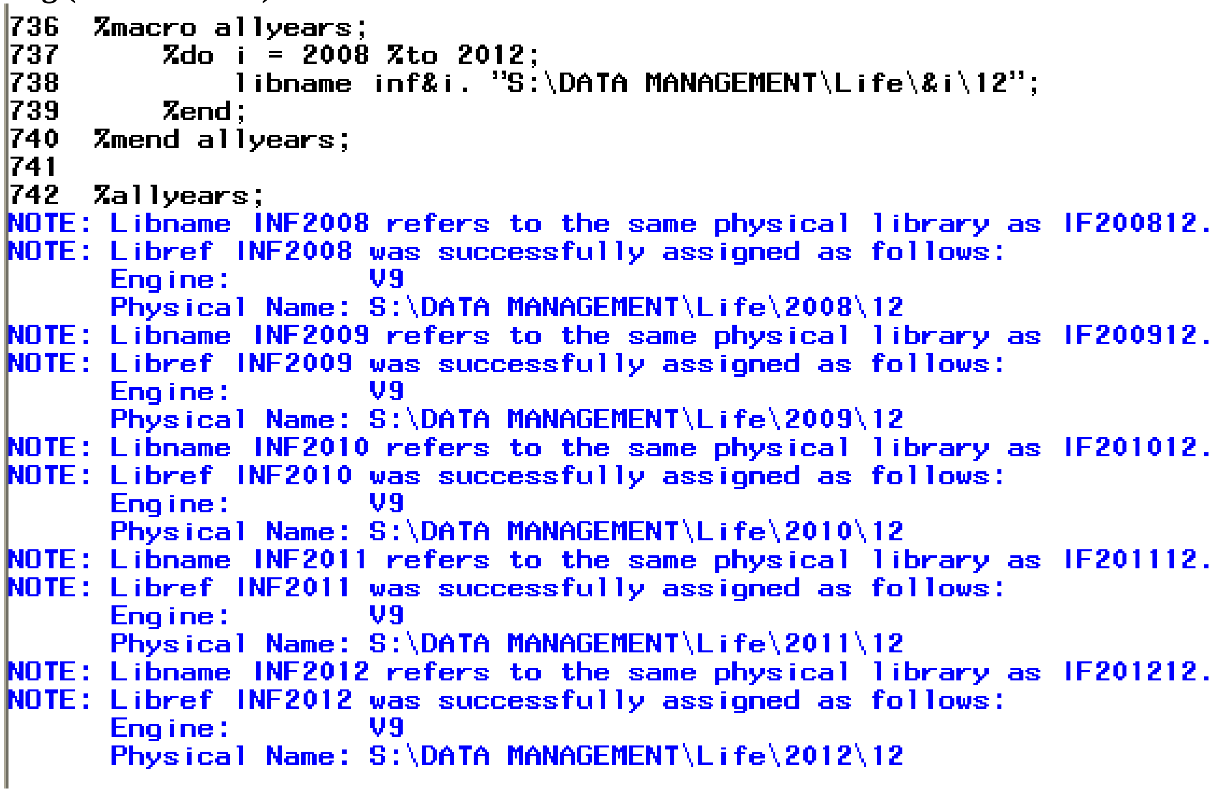
```
%macro allyears;
    %do i = 2008 %to 2012;
        libname inf&i. "S:\DATA MANAGEMENT\Life\&i\12";
    %end;
%mend allyears;

%allyears;
```

Looking at macro code2 above, we have the base code (as shown below).

```
libname inf____ "S:\Data Management\Life\____\12";
```

Using a do loop and setting *i* from 2008 to 2012, we’re able to remove the repetitive code. Please take note that we are able to do a do loop on the libname because macros allows us to do that outside of DATA steps. By invoking the macro code using *%allyears*, we see the log output below that both libname and libref are assigned successfully.

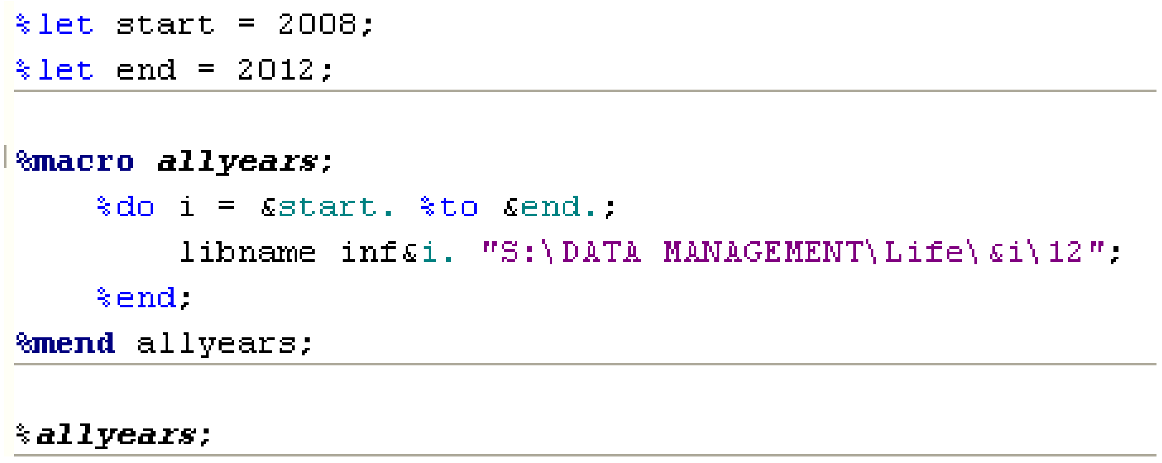
Log (Macro Code2):



```
%macro allyears;
    %do i = 2008 %to 2012;
        libname inf&i. "S:\DATA MANAGEMENT\Life\&i\12";
    %end;
%mend allyears;

%allyears;
NOTE: Libname INF2008 refers to the same physical library as IF200812.
NOTE: Libref INF2008 was successfully assigned as follows:
      Engine:        V9
      Physical Name: S:\DATA MANAGEMENT\Life\2008\12
NOTE: Libname INF2009 refers to the same physical library as IF200912.
NOTE: Libref INF2009 was successfully assigned as follows:
      Engine:        V9
      Physical Name: S:\DATA MANAGEMENT\Life\2009\12
NOTE: Libname INF2010 refers to the same physical library as IF201012.
NOTE: Libref INF2010 was successfully assigned as follows:
      Engine:        V9
      Physical Name: S:\DATA MANAGEMENT\Life\2010\12
NOTE: Libname INF2011 refers to the same physical library as IF201112.
NOTE: Libref INF2011 was successfully assigned as follows:
      Engine:        V9
      Physical Name: S:\DATA MANAGEMENT\Life\2011\12
NOTE: Libname INF2012 refers to the same physical library as IF201212.
NOTE: Libref INF2012 was successfully assigned as follows:
      Engine:        V9
      Physical Name: S:\DATA MANAGEMENT\Life\2012\12
```

To make the code more versatile, we can do the following.

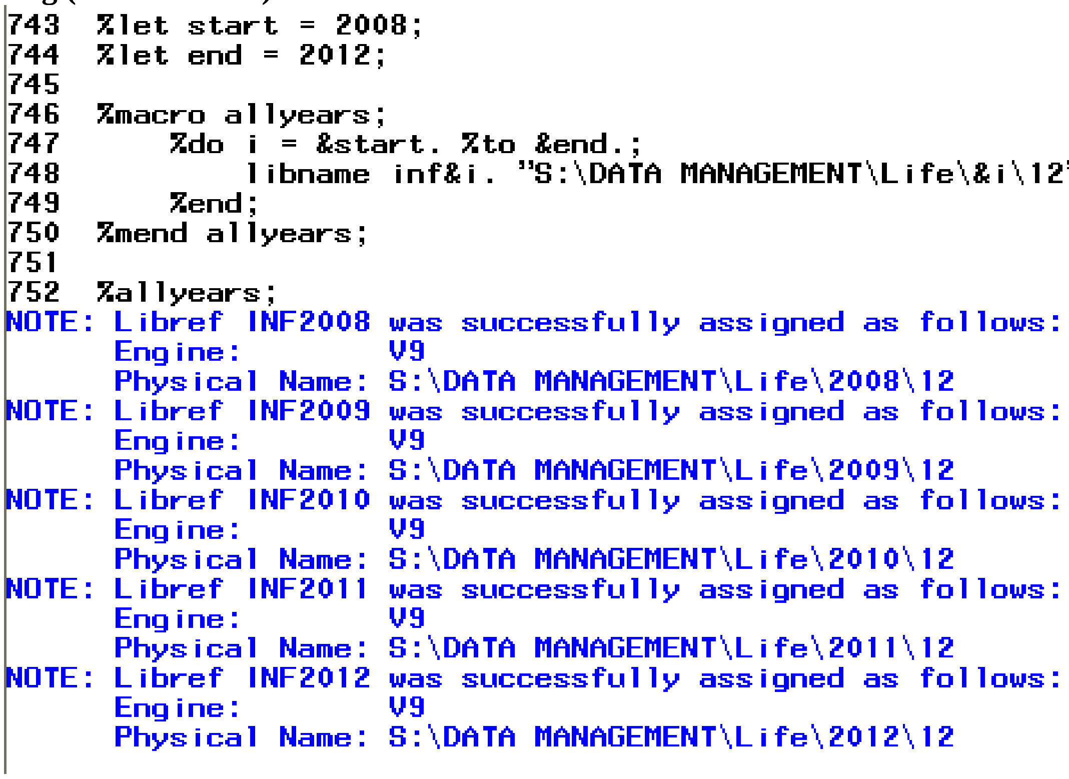
Macro Code3:



```
%let start = 2008;
%let end = 2012;

%macro allyears;
    %do i = &start. %to &end.;
        libname inf&i. "S:\DATA MANAGEMENT\Life\&i\12";
    %end;
%mend allyears;

%allyears;
```

Instead of hard-coding the start year and end year in the code, we can create a macro statement to assign the start year and end year. This can be put at the beginning of the code, and the macro code 3 can be placed anywhere in the code and we're still able to update the years without having to hunt down the do loop. The log output below shows that we get the same results as macro code2 above.

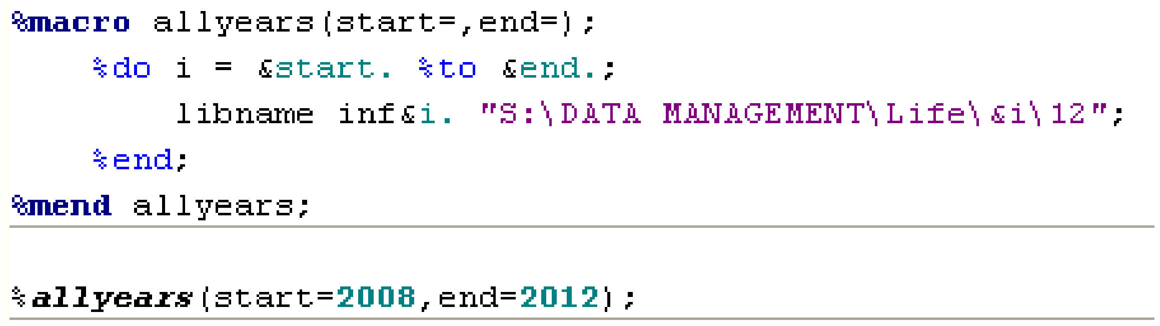
Log (Macro Code3):



```
%let start = 2008;
%let end = 2012;

%macro allyears;
    %do i = &start. %to &end.;
        libname inf&i. "S:\DATA MANAGEMENT\Life\&i\12";
    %end;
%mend allyears;

%allyears;
NOTE: Libref INF2008 was successfully assigned as follows:
      Engine:        V9
      Physical Name: S:\DATA MANAGEMENT\Life\2008\12
NOTE: Libref INF2009 was successfully assigned as follows:
      Engine:        V9
      Physical Name: S:\DATA MANAGEMENT\Life\2009\12
NOTE: Libref INF2010 was successfully assigned as follows:
      Engine:        V9
      Physical Name: S:\DATA MANAGEMENT\Life\2010\12
NOTE: Libref INF2011 was successfully assigned as follows:
      Engine:        V9
      Physical Name: S:\DATA MANAGEMENT\Life\2011\12
NOTE: Libref INF2012 was successfully assigned as follows:
      Engine:        V9
      Physical Name: S:\DATA MANAGEMENT\Life\2012\12
```

We could also set the parameters next to the macro-code call.

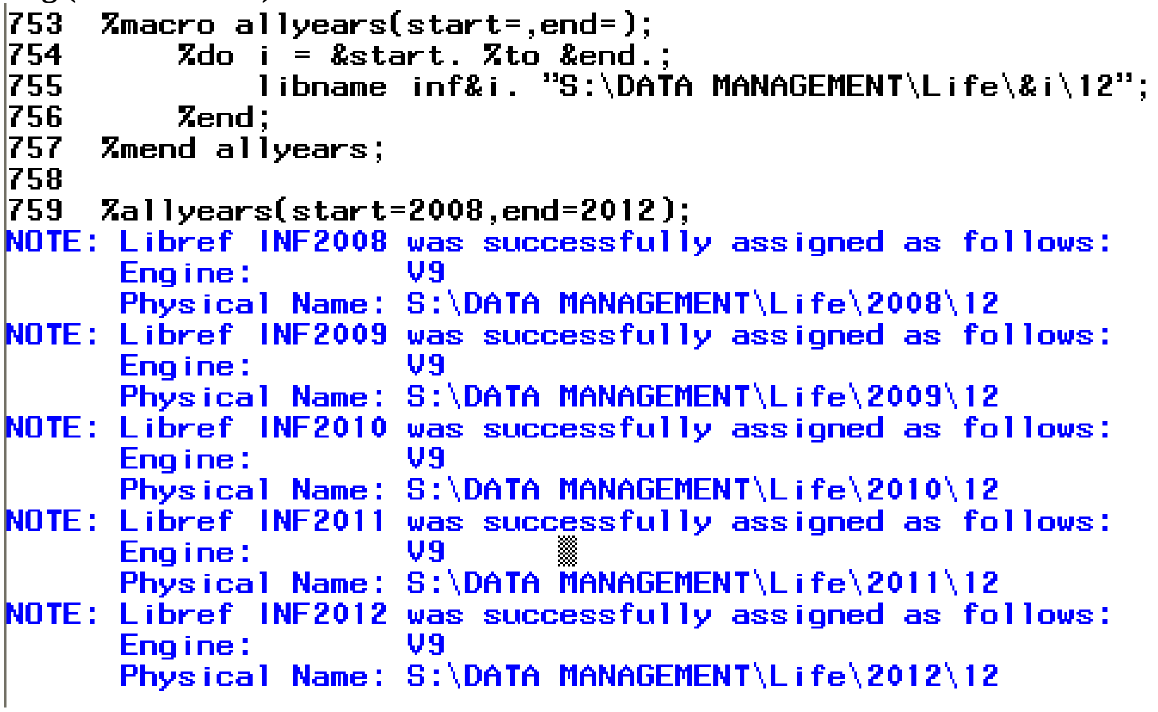
Macro Code4:



```
%macro allyears(start=,end=);
    %do i = &start. %to &end.;
        libname inf&i. "S:\DATA MANAGEMENT\Life\&i\12";
    %end;
%mend allyears;

%allyears(start=2008,end=2012);
```

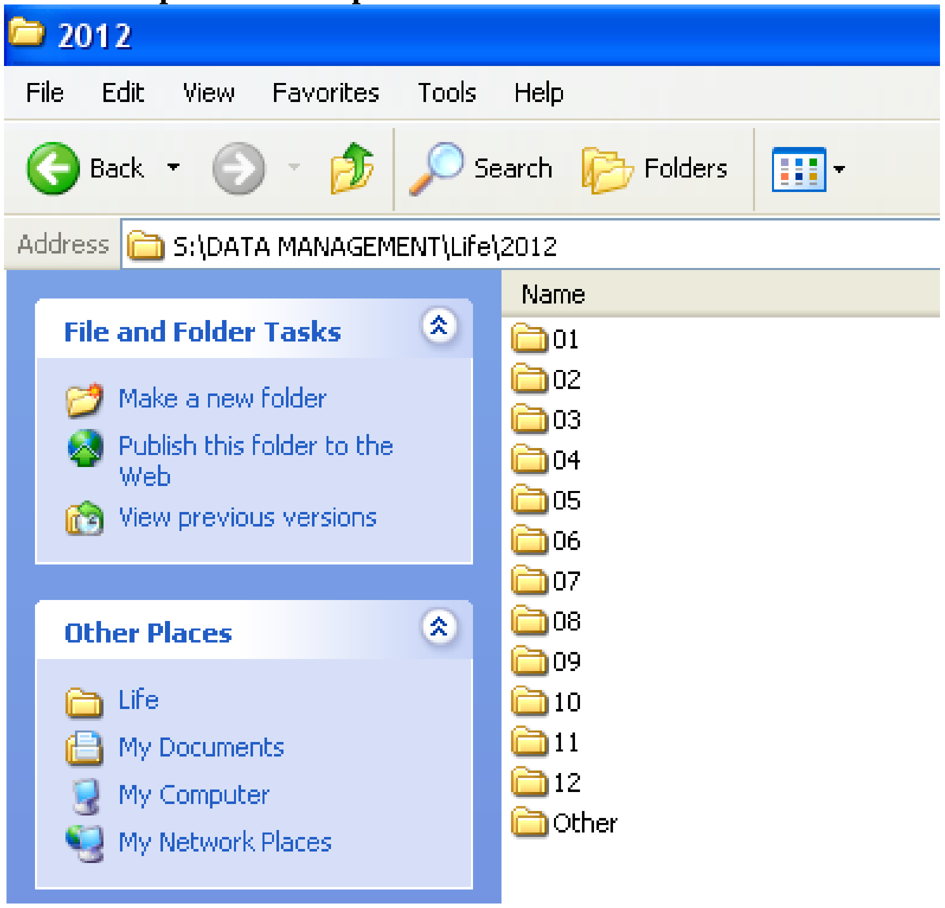
This way, if we are trying to set different sets of libnames throughout the code, we can do that easily. The log output shown below has the same output results as the macro codes before.

Log (Macro Code4):



```
%macro allyears(start=,end=);
    %do i = &start. %to &end.;
        libname inf&i. "S:\DATA MANAGEMENT\Life\&i\12";
    %end;
%mend allyears;

%allyears(start=2008,end=2012);
NOTE: Libref INF2008 was successfully assigned as follows:
      Engine:        V9
      Physical Name: S:\DATA MANAGEMENT\Life\2008\12
NOTE: Libref INF2009 was successfully assigned as follows:
      Engine:        V9
      Physical Name: S:\DATA MANAGEMENT\Life\2009\12
NOTE: Libref INF2010 was successfully assigned as follows:
      Engine:        V9
      Physical Name: S:\DATA MANAGEMENT\Life\2010\12
NOTE: Libref INF2011 was successfully assigned as follows:
      Engine:        V9
      Physical Name: S:\DATA MANAGEMENT\Life\2011\12
NOTE: Libref INF2012 was successfully assigned as follows:
      Engine:        V9
      Physical Name: S:\DATA MANAGEMENT\Life\2012\12
```

Remove Repetition Example 2:



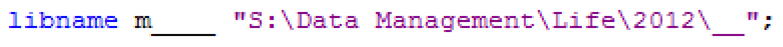
S:\DATA MANAGEMENT\Life\2012

01
02
03
04
05
06
07
08
09
10
11
12
Other

Now, let's say we want to access monthly data for the year 2012 (as seen in the figure above). Again, we could code it as shown below

```
libname m01 'S:\DATA MANAGEMENT\Life\2012\01';
libname m02 'S:\DATA MANAGEMENT\Life\2012\02';
libname m03 'S:\DATA MANAGEMENT\Life\2012\03';
libname m04 'S:\DATA MANAGEMENT\Life\2012\04';
libname m05 'S:\DATA MANAGEMENT\Life\2012\05';
libname m06 'S:\DATA MANAGEMENT\Life\2012\06';
libname m07 'S:\DATA MANAGEMENT\Life\2012\07';
libname m08 'S:\DATA MANAGEMENT\Life\2012\08';
libname m09 'S:\DATA MANAGEMENT\Life\2012\09';
libname m10 'S:\DATA MANAGEMENT\Life\2012\10';
libname m11 'S:\DATA MANAGEMENT\Life\2012\11';
libname m12 'S:\DATA MANAGEMENT\Life\2012\12';
```

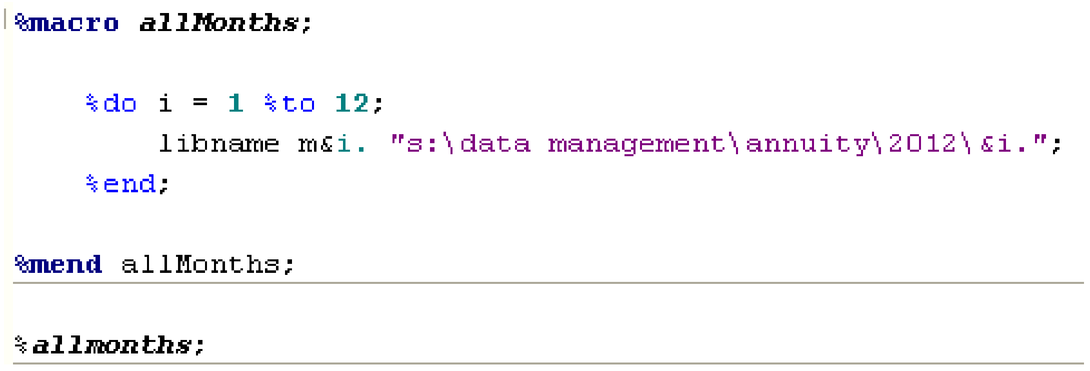
Or, we can use macros and do loop again to shorten and simplify the code. We'll start with the base code again, as shown below,



```
libname m____ "S:\Data Management\Life\2012\__";
```

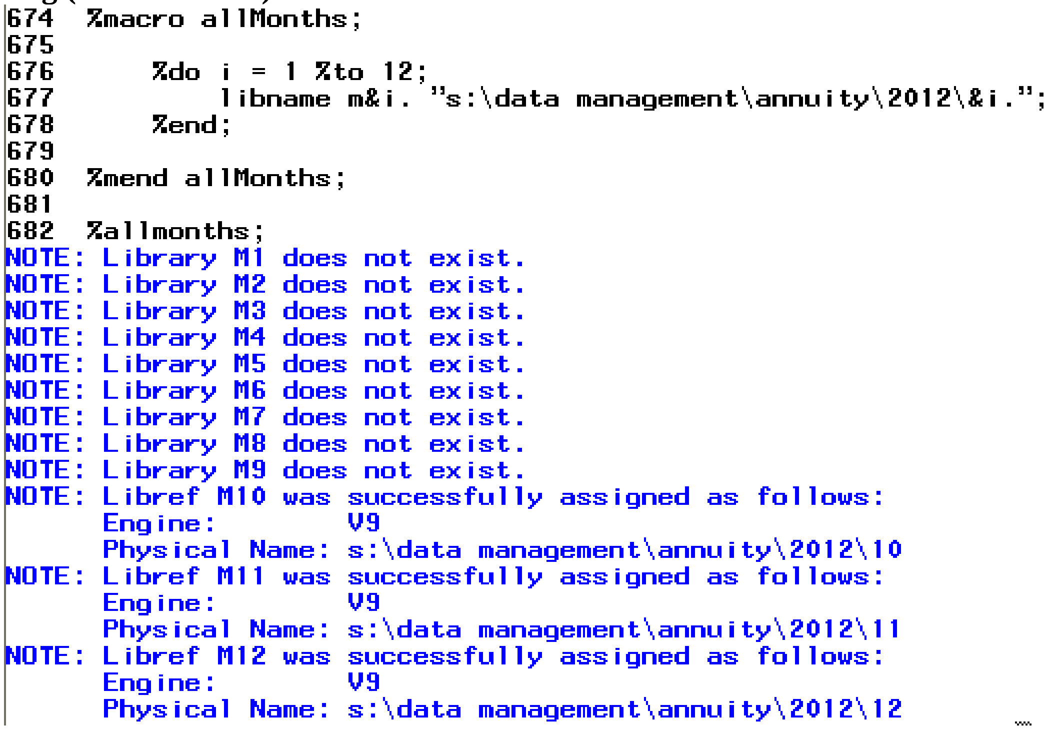
and adding a do loop into the mix, we have the macro code (seen in macro code5) below.

Macro Code5:



```
%macro allMonths;

    %do i = 1 %to 12;
        libname m&i. "s:\data management\annuity\2012\&i.";
    %end;

%mend allMonths;

%allmonths;
```

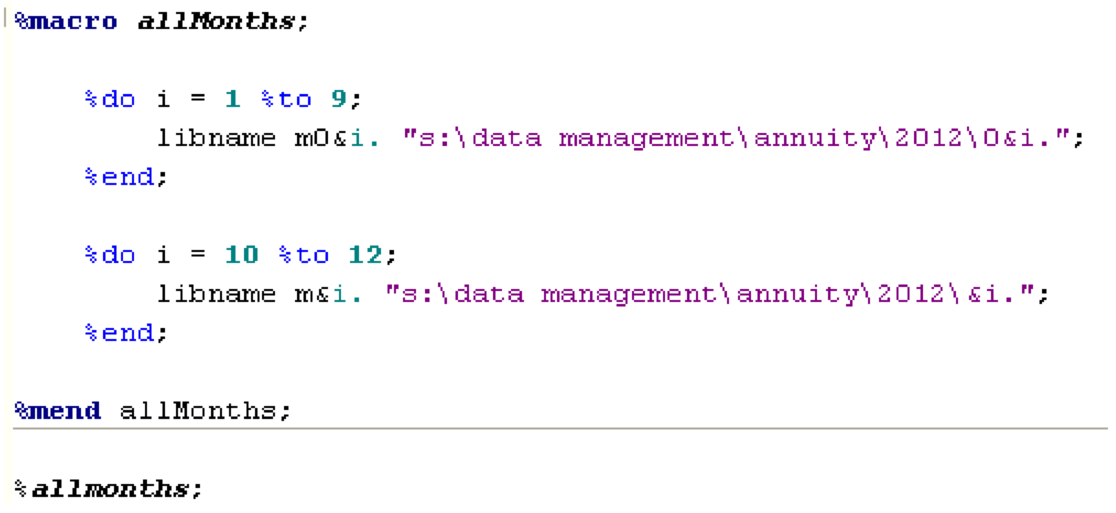
Log (Macro Code5):



```
%macro allMonths;

    %do i = 1 %to 12;
        libname m&i. "s:\data management\annuity\2012\&i.";
    %end;

%mend allMonths;

%allmonths;
NOTE: Library M1 does not exist.
NOTE: Library M2 does not exist.
NOTE: Library M3 does not exist.
NOTE: Library M4 does not exist.
NOTE: Library M5 does not exist.
NOTE: Library M6 does not exist.
NOTE: Library M7 does not exist.
NOTE: Library M8 does not exist.
NOTE: Library M9 does not exist.
NOTE: Libref M10 was successfully assigned as follows:
      Engine:        V9
      Physical Name: s:\data management\annuity\2012\10
NOTE: Libref M11 was successfully assigned as follows:
      Engine:        V9
      Physical Name: s:\data management\annuity\2012\11
NOTE: Libref M12 was successfully assigned as follows:
      Engine:        V9
      Physical Name: s:\data management\annuity\2012\12
```

Looking at the log output for macro code5, we see that only libref M10, M11, and M12 was assigned successfully, whereas libref M1 to M9 is no successful. Looking at the figure above, we see that this is because the folders are named 01 to 09, as opposed to, 1 to 9.

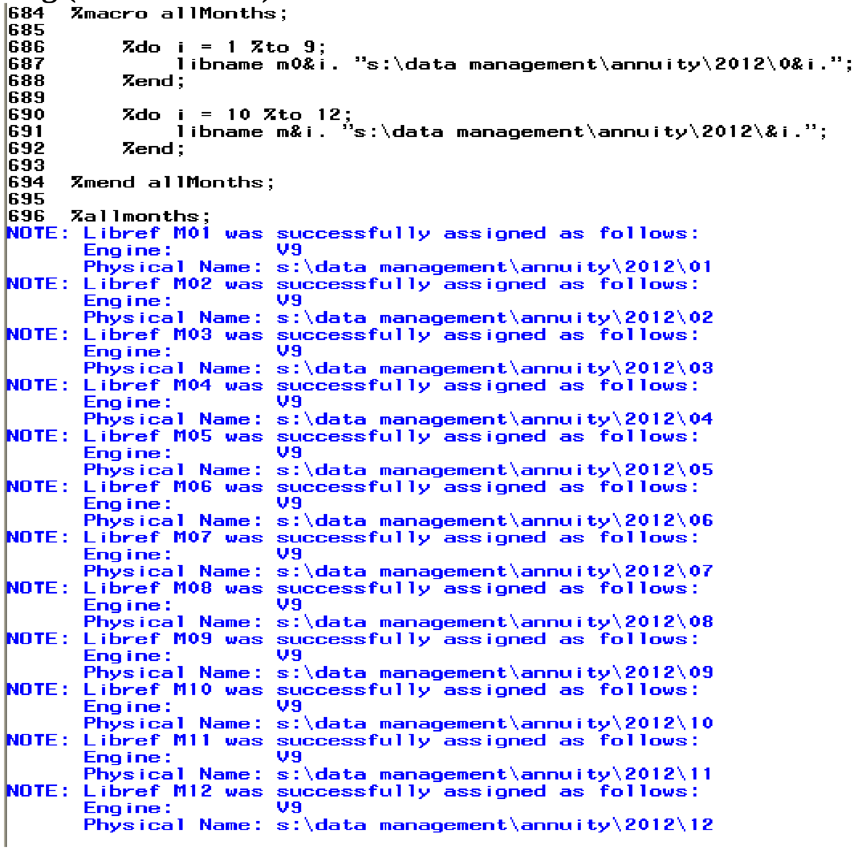
Macro Code6:



```
%macro allMonths;

    %do i = 1 %to 9;
        libname m0&i. "s:\data management\annuity\2012\0&i.";
    %end;

    %do i = 10 %to 12;
        libname m&i. "s:\data management\annuity\2012\&i.";
    %end;

%mend allMonths;

%allmonths;
```

Making a slight change to the code by adding a second do loop; one for *i* from 1 to 9, and the second from 10 to 12, like above, we're able to correct the problem. The libref for 1 to 9 has a 0 at the front, while libref 11 to 12 stays the same. Running the code produced the log output below, and here, we see that all the librefs are assigned successfully.

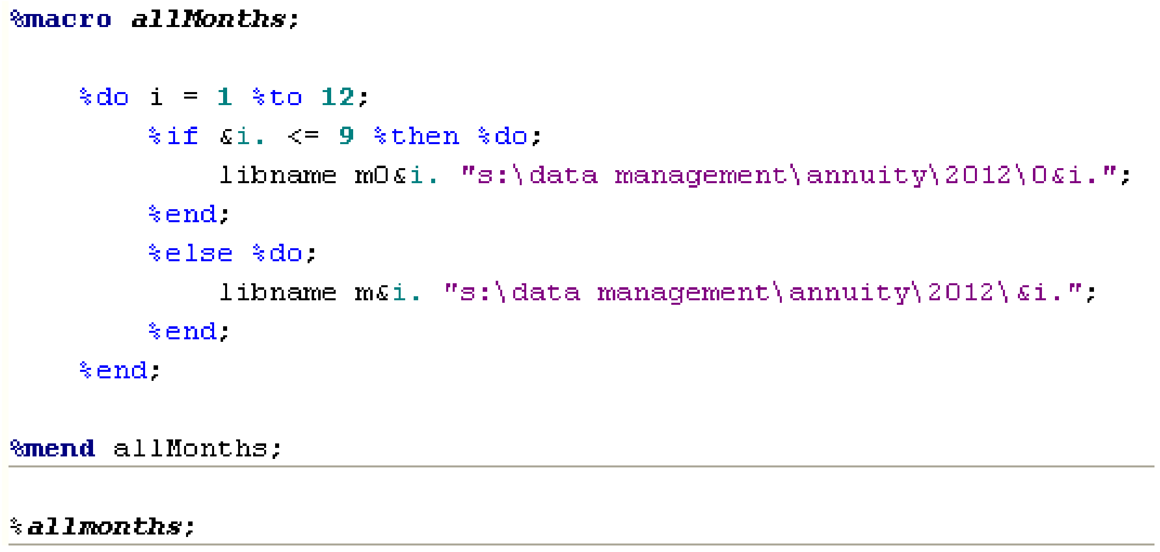
Log (Macro Code6):



```
%macro allMonths;

    %do i = 1 %to 9;
        libname m0&i. "s:\data management\annuity\2012\0&i.";
    %end;

    %do i = 10 %to 12;
        libname m&i. "s:\data management\annuity\2012\&i.";
    %end;

%mend allMonths;

%allmonths;
NOTE: Libref M01 was successfully assigned as follows:
      Engine:        V9
      Physical Name: s:\data management\annuity\2012\01
NOTE: Libref M02 was successfully assigned as follows:
      Engine:        V9
      Physical Name: s:\data management\annuity\2012\02
NOTE: Libref M03 was successfully assigned as follows:
      Engine:        V9
      Physical Name: s:\data management\annuity\2012\03
NOTE: Libref M04 was successfully assigned as follows:
      Engine:        V9
      Physical Name: s:\data management\annuity\2012\04
NOTE: Libref M05 was successfully assigned as follows:
      Engine:        V9
      Physical Name: s:\data management\annuity\2012\05
NOTE: Libref M06 was successfully assigned as follows:
      Engine:        V9
      Physical Name: s:\data management\annuity\2012\06
NOTE: Libref M07 was successfully assigned as follows:
      Engine:        V9
      Physical Name: s:\data management\annuity\2012\07
NOTE: Libref M08 was successfully assigned as follows:
      Engine:        V9
      Physical Name: s:\data management\annuity\2012\08
NOTE: Libref M09 was successfully assigned as follows:
      Engine:        V9
      Physical Name: s:\data management\annuity\2012\09
NOTE: Libref M10 was successfully assigned as follows:
      Engine:        V9
      Physical Name: s:\data management\annuity\2012\10
NOTE: Libref M11 was successfully assigned as follows:
      Engine:        V9
      Physical Name: s:\data management\annuity\2012\11
NOTE: Libref M12 was successfully assigned as follows:
      Engine:        V9
      Physical Name: s:\data management\annuity\2012\12
```

Instead of using two do loops, we can still get by with using one do loop. This can be done by adding if-else statements, as shown in macro code7 below,

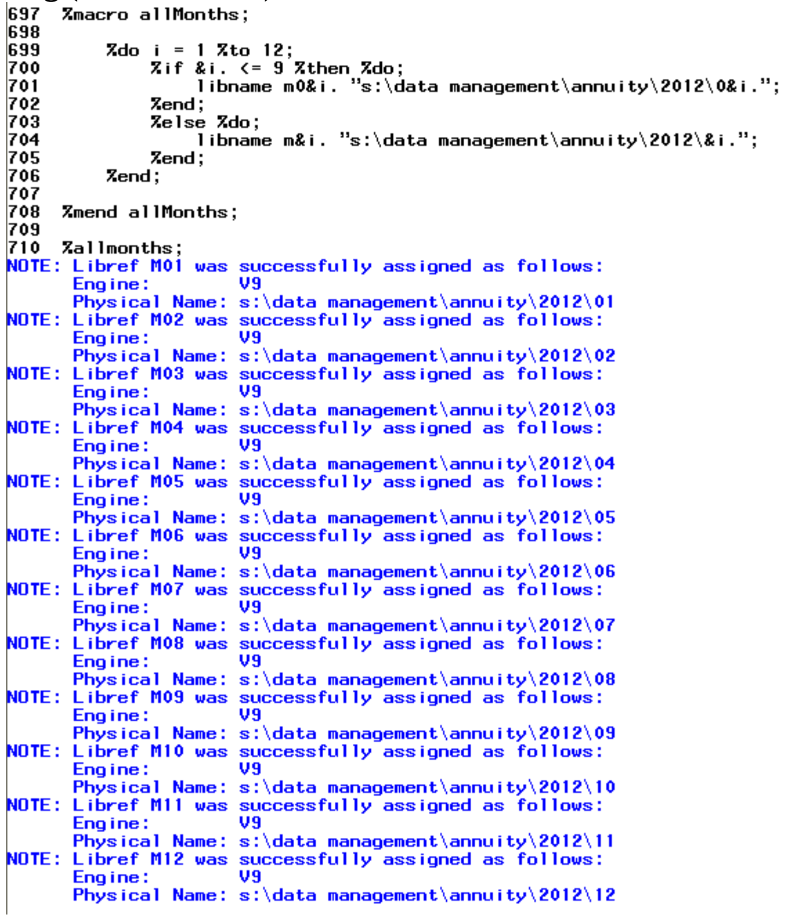
Macro Code7:



```
%macro allMonths;

    %do i = 1 %to 12;
        %if &i. <= 9 %then %do;
            libname m0&i. "s:\data management\annuity\2012\0&i.";
        %end;
        %else %do;
            libname m&i. "s:\data management\annuity\2012\&i.";
        %end;
    %end;

%mend allMonths;

%allmonths;
```

Using a do loop for *i* from 1 to 12, we can set that if *i* is from 1 to 9, then add a zero at the front, else, do not add a zero in the front. The log output for macro code7 is shown below and the result is the same as the result from macro code7.

Log (Macro Code7):



```
%macro allMonths;

    %do i = 1 %to 12;
        %if &i. <= 9 %then %do;
            libname m0&i. "s:\data management\annuity\2012\0&i.";
        %end;
        %else %do;
            libname m&i. "s:\data management\annuity\2012\&i.";
        %end;
    %end;

%mend allMonths;

%allmonths;
NOTE: Libref M01 was successfully assigned as follows:
      Engine:        V9
      Physical Name: s:\data management\annuity\2012\01
NOTE: Libref M02 was successfully assigned as follows:
      Engine:        V9
      Physical Name: s:\data management\annuity\2012\02
NOTE: Libref M03 was successfully assigned as follows:
      Engine:        V9
      Physical Name: s:\data management\annuity\2012\03
NOTE: Libref M04 was successfully assigned as follows:
      Engine:        V9
      Physical Name: s:\data management\annuity\2012\04
NOTE: Libref M05 was successfully assigned as follows:
      Engine:        V9
      Physical Name: s:\data management\annuity\2012\05
NOTE: Libref M06 was successfully assigned as follows:
      Engine:        V9
      Physical Name: s:\data management\annuity\2012\06
NOTE: Libref M07 was successfully assigned as follows:
      Engine:        V9
      Physical Name: s:\data management\annuity\2012\07
NOTE: Libref M08 was successfully assigned as follows:
      Engine:        V9
      Physical Name: s:\data management\annuity\2012\08
NOTE: Libref M09 was successfully assigned as follows:
      Engine:        V9
      Physical Name: s:\data management\annuity\2012\09
NOTE: Libref M10 was successfully assigned as follows:
      Engine:        V9
      Physical Name: s:\data management\annuity\2012\10
NOTE: Libref M11 was successfully assigned as follows:
      Engine:        V9
      Physical Name: s:\data management\annuity\2012\11
NOTE: Libref M12 was successfully assigned as follows:
      Engine:        V9
      Physical Name: s:\data management\annuity\2012\12
```

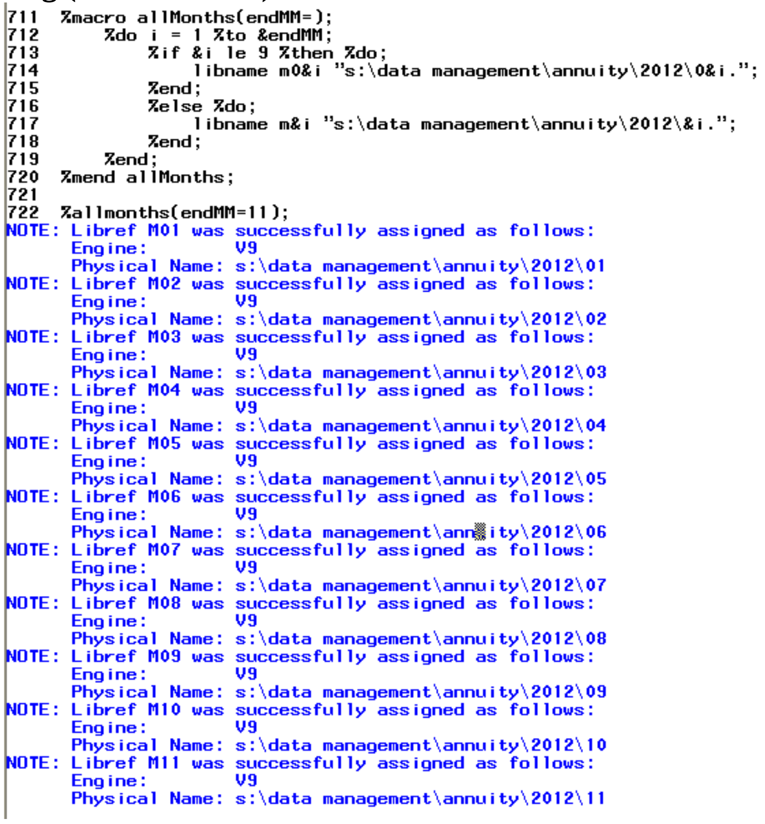
Again, we can use parameter calls to set the months instead of hard coding it.

Marco Code8:

```
%macro allMonths(endMM=);
    %do i = 1 %to &endMM;
        %if &i le 9 %then %do;
            libname m0&i "s:\data management\annuity\2012\0&i.";
        %end;
        %else %do;
            libname m&i "s:\data management\annuity\2012\&i.";
        %end;
    %end;
%mend allMonths;

%allmonths(endMM=11);
```

If we want the last month to be set as November instead of December, using the parameter call, we can set the last month to be anything we want easily, rather than searching through the whole code to change it. The output of the code shows that we have successfully assigned eleven librefs, from January to November.

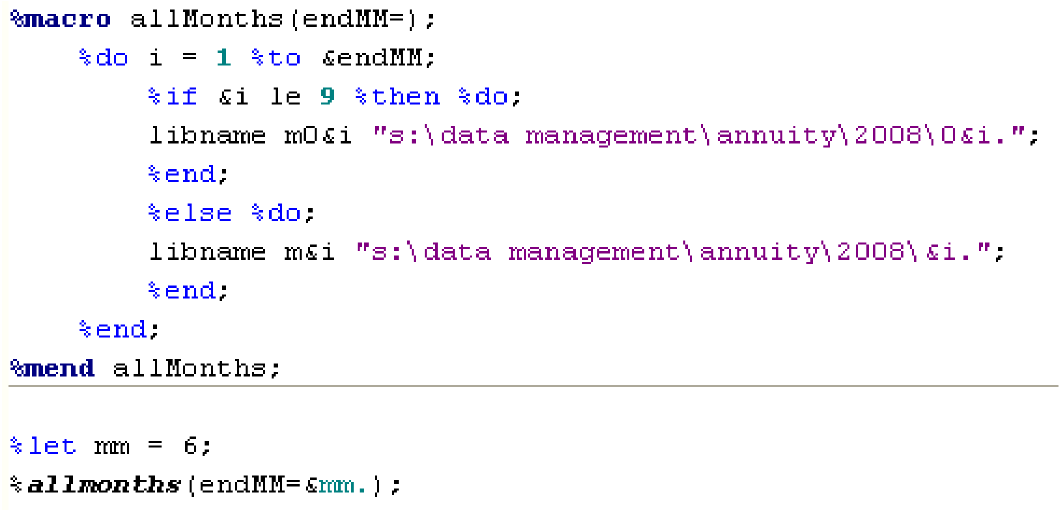
Log (Macro Code8):



```
%macro allMonths(endMM=);
    %do i = 1 %to &endMM;
        %if &i le 9 %then %do;
            libname m0&i "s:\data management\annuity\2012\0&i.";
        %end;
        %else %do;
            libname m&i "s:\data management\annuity\2012\&i.";
        %end;
    %end;
%mend allMonths;

%allmonths(endMM=11);
NOTE: Libref M01 was successfully assigned as follows:
      Engine:        V9
      Physical Name: s:\data management\annuity\2012\01
NOTE: Libref M02 was successfully assigned as follows:
      Engine:        V9
      Physical Name: s:\data management\annuity\2012\02
NOTE: Libref M03 was successfully assigned as follows:
      Engine:        V9
      Physical Name: s:\data management\annuity\2012\03
NOTE: Libref M04 was successfully assigned as follows:
      Engine:        V9
      Physical Name: s:\data management\annuity\2012\04
NOTE: Libref M05 was successfully assigned as follows:
      Engine:        V9
      Physical Name: s:\data management\annuity\2012\05
NOTE: Libref M06 was successfully assigned as follows:
      Engine:        V9
      Physical Name: s:\data management\annuity\2012\06
NOTE: Libref M07 was successfully assigned as follows:
      Engine:        V9
      Physical Name: s:\data management\annuity\2012\07
NOTE: Libref M08 was successfully assigned as follows:
      Engine:        V9
      Physical Name: s:\data management\annuity\2012\08
NOTE: Libref M09 was successfully assigned as follows:
      Engine:        V9
      Physical Name: s:\data management\annuity\2012\09
NOTE: Libref M10 was successfully assigned as follows:
      Engine:        V9
      Physical Name: s:\data management\annuity\2012\10
NOTE: Libref M11 was successfully assigned as follows:
      Engine:        V9
      Physical Name: s:\data management\annuity\2012\11
```

Another way of assigning librefs using macros is as follows.

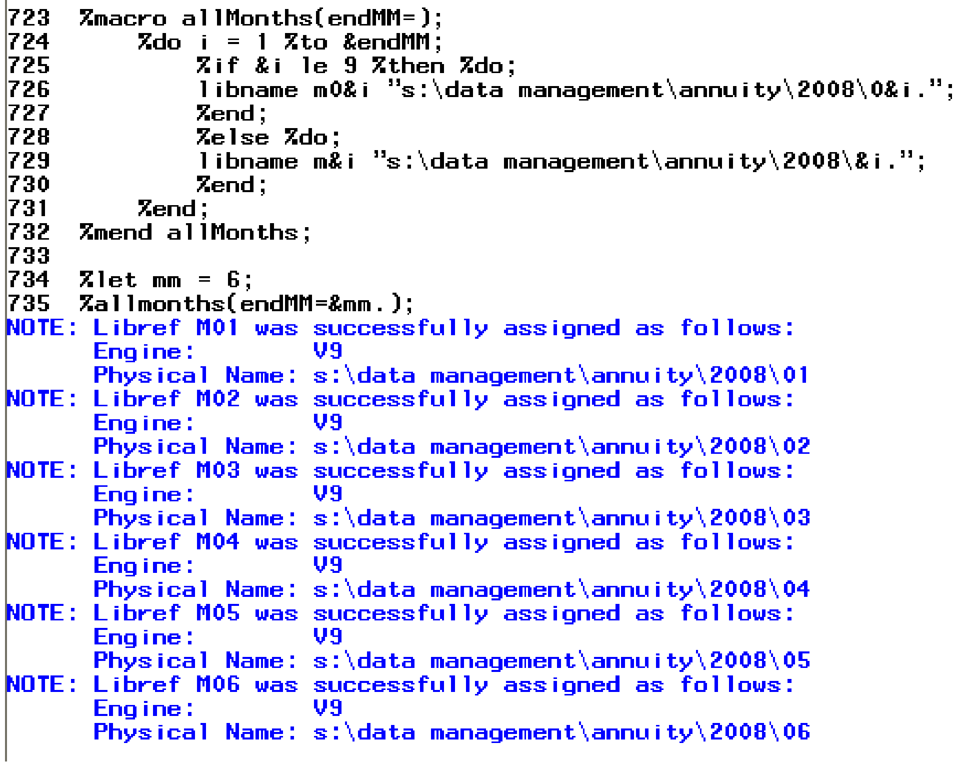
Macro Code9:



```
%macro allMonths(endMM=);
    %do i = 1 %to &endMM;
        %if &i le 9 %then %do;
        libname m0&i "s:\data management\annuity\2008\0&i.";
        %end;
        %else %do;
        libname m&i "s:\data management\annuity\2008\&i.";
        %end;
    %end;
%mend allMonths;

%let mm = 6;
%allmonths(endMM=&mm.);
```

What we did in macro code9 is to assign a macro variable to the parameter call. Here, we only want to assign librefs from January to June. The log output below shows that all six librefs are assigned successfully.

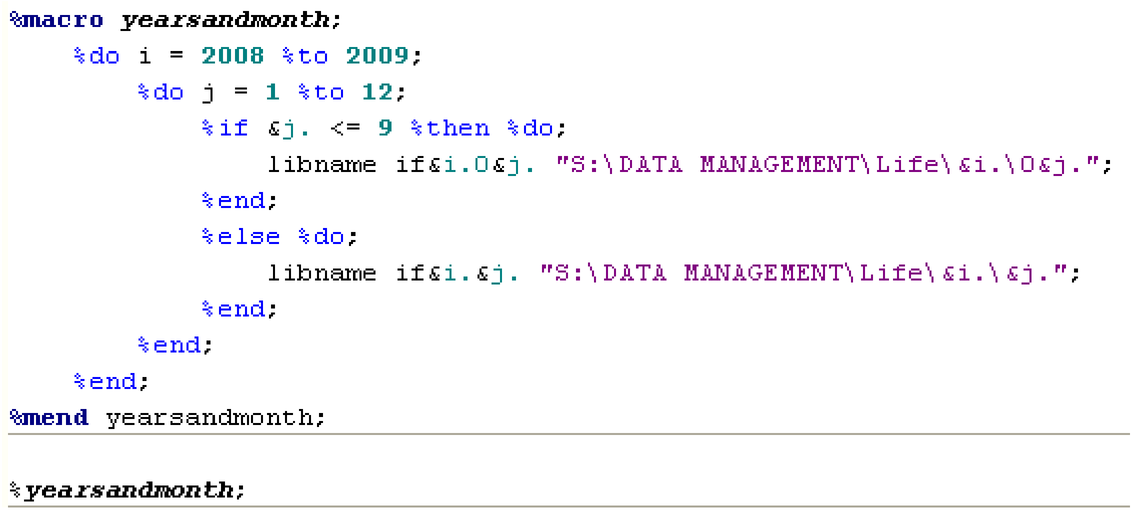
Log (Macro Code9):



```
%macro allMonths(endMM=);
    %do i = 1 %to &endMM;
        %if &i le 9 %then %do;
        libname m0&i "s:\data management\annuity\2008\0&i.";
        %end;
        %else %do;
        libname m&i "s:\data management\annuity\2008\&i.";
        %end;
    %end;
%mend allMonths;

%let mm = 6;
%allmonths(endMM=&mm.);
NOTE: Libref M01 was successfully assigned as follows:
      Engine:        V9
      Physical Name: s:\data management\annuity\2008\01
NOTE: Libref M02 was successfully assigned as follows:
      Engine:        V9
      Physical Name: s:\data management\annuity\2008\02
NOTE: Libref M03 was successfully assigned as follows:
      Engine:        V9
      Physical Name: s:\data management\annuity\2008\03
NOTE: Libref M04 was successfully assigned as follows:
      Engine:        V9
      Physical Name: s:\data management\annuity\2008\04
NOTE: Libref M05 was successfully assigned as follows:
      Engine:        V9
      Physical Name: s:\data management\annuity\2008\05
NOTE: Libref M06 was successfully assigned as follows:
      Engine:        V9
      Physical Name: s:\data management\annuity\2008\06
```

What if we want to set librefs from January to December for both year 2008 and year 2009. We can type out all 24 lines of code, or we can use macros and do loops to create the librefs for us.

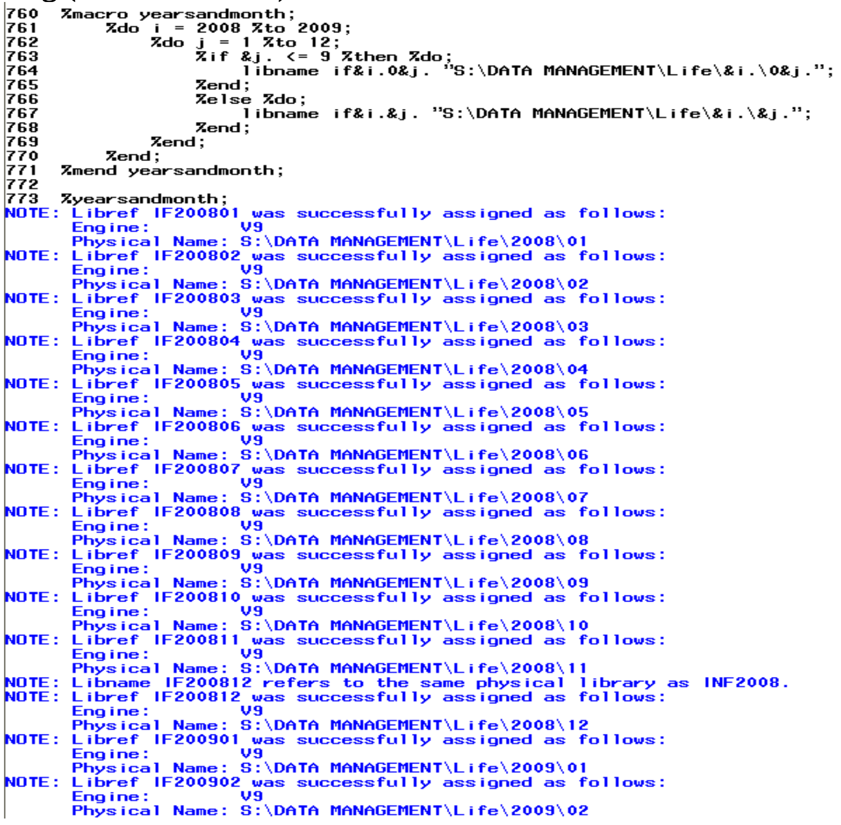
Macro Code10:



```
%macro yearsandmonth;
    %do i = 2008 %to 2009;
        %do j = 1 %to 12;
            %if &j. <= 9 %then %do;
                libname if&i.0&j. "S:\DATA MANAGEMENT\Life\&i.\0&j.";
            %end;
            %else %do;
                libname if&i.&j. "S:\DATA MANAGEMENT\Life\&i.\&j.";
            %end;
        %end;
    %end;
%mend yearsandmonth;

%yearsandmonth;
```

To do that, we need to use two do loops, one to iterate for year, and a second do loop, nested within the first, for month iterations, like macro code10 above. The log output below shows that all librefs are assigned successfully.

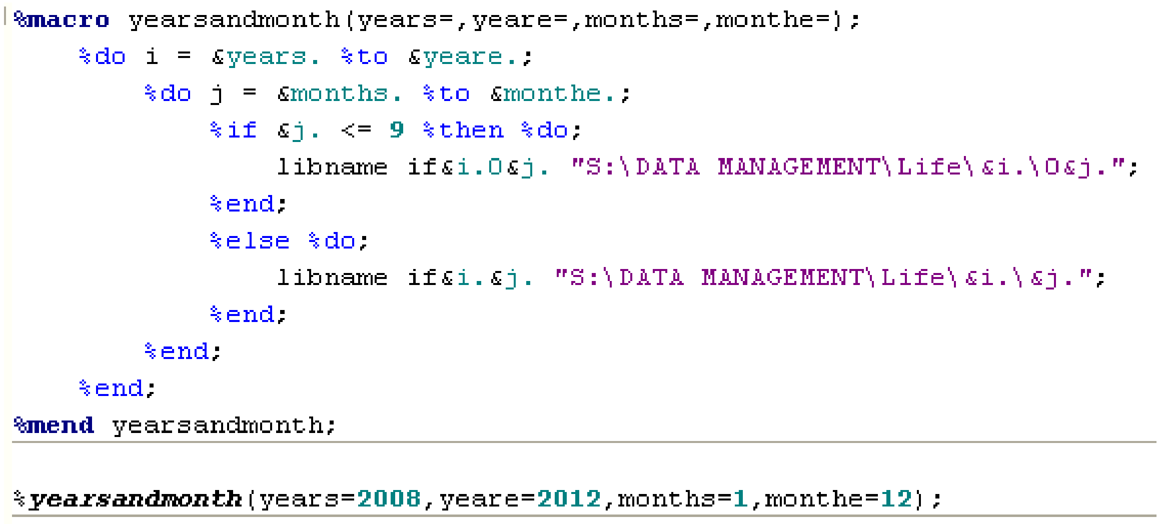
Log (Macro Code10):



```
%macro yearsandmonth;
    %do i = 2008 %to 2009;
        %do j = 1 %to 12;
            %if &j. <= 9 %then %do;
                libname if&i.0&j. "S:\DATA MANAGEMENT\Life\&i.\0&j.";
            %end;
            %else %do;
                libname if&i.&j. "S:\DATA MANAGEMENT\Life\&i.\&j.";
            %end;
        %end;
    %end;
%mend yearsandmonth;

%yearsandmonth;
NOTE: Libref IF200801 was successfully assigned as follows:
      Engine:        V9
      Physical Name: S:\DATA MANAGEMENT\Life\2008\01
NOTE: Libref IF200802 was successfully assigned as follows:
      Engine:        V9
      Physical Name: S:\DATA MANAGEMENT\Life\2008\02
NOTE: Libref IF200803 was successfully assigned as follows:
      Engine:        V9
      Physical Name: S:\DATA MANAGEMENT\Life\2008\03
NOTE: Libref IF200804 was successfully assigned as follows:
      Engine:        V9
      Physical Name: S:\DATA MANAGEMENT\Life\2008\04
NOTE: Libref IF200805 was successfully assigned as follows:
      Engine:        V9
      Physical Name: S:\DATA MANAGEMENT\Life\2008\05
NOTE: Libref IF200806 was successfully assigned as follows:
      Engine:        V9
      Physical Name: S:\DATA MANAGEMENT\Life\2008\06
NOTE: Libref IF200807 was successfully assigned as follows:
      Engine:        V9
      Physical Name: S:\DATA MANAGEMENT\Life\2008\07
NOTE: Libref IF200808 was successfully assigned as follows:
      Engine:        V9
      Physical Name: S:\DATA MANAGEMENT\Life\2008\08
NOTE: Libref IF200809 was successfully assigned as follows:
      Engine:        V9
      Physical Name: S:\DATA MANAGEMENT\Life\2008\09
NOTE: Libref IF200810 was successfully assigned as follows:
      Engine:        V9
      Physical Name: S:\DATA MANAGEMENT\Life\2008\10
NOTE: Libref IF200811 was successfully assigned as follows:
      Engine:        V9
      Physical Name: S:\DATA MANAGEMENT\Life\2008\11
NOTE: Libname IF200812 refers to the same physical library as INF2008.
NOTE: Libref IF200812 was successfully assigned as follows:
      Engine:        V9
      Physical Name: S:\DATA MANAGEMENT\Life\2008\12
NOTE: Libref IF200901 was successfully assigned as follows:
      Engine:        V9
      Physical Name: S:\DATA MANAGEMENT\Life\2009\01
NOTE: Libref IF200902 was successfully assigned as follows:
      Engine:        V9
      Physical Name: S:\DATA MANAGEMENT\Life\2009\02
```

To make the code more versatile, we can use parameter calls to do that.

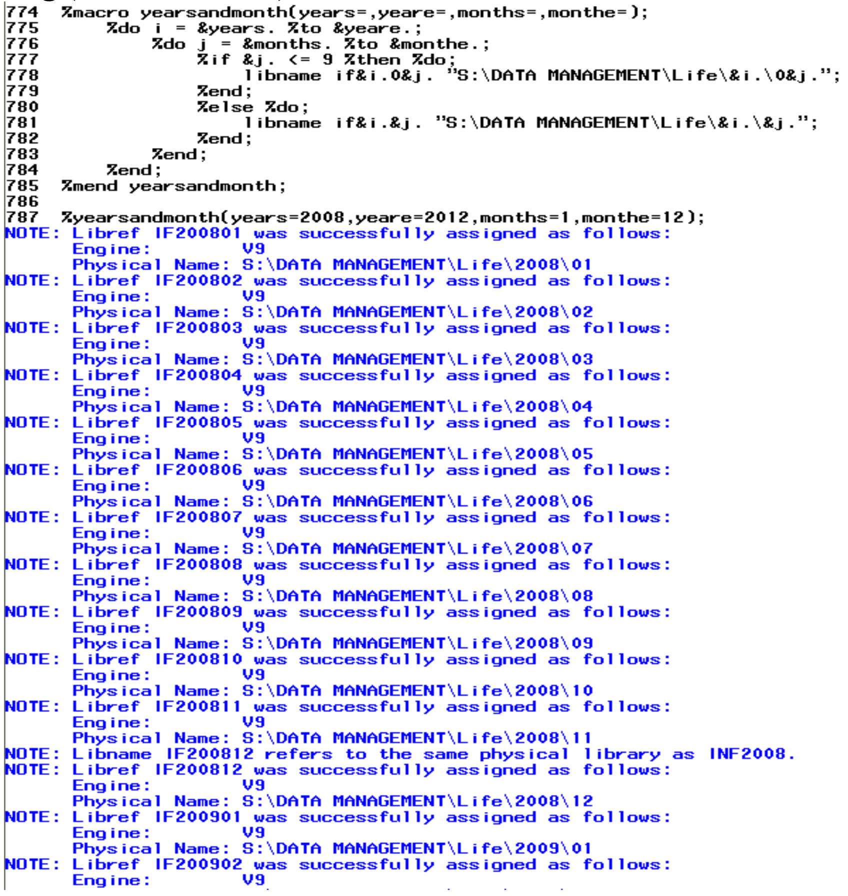
Macro Code11:



```
%macro yearsandmonth(years=,yeare=,months=,monthe=);
    %do i = &years. %to &yeare.;
        %do j = &months. %to &monthe.;
            %if &j. <= 9 %then %do;
                libname if&i.0&j. "S:\DATA MANAGEMENT\Life\&i.\0&j.";
            %end;
            %else %do;
                libname if&i.&j. "S:\DATA MANAGEMENT\Life\&i.\&j.";
            %end;
        %end;
    %end;
%mend yearsandmonth;

%yearsandmonth(years=2008,yeare=2012,months=1,monthe=12);
```

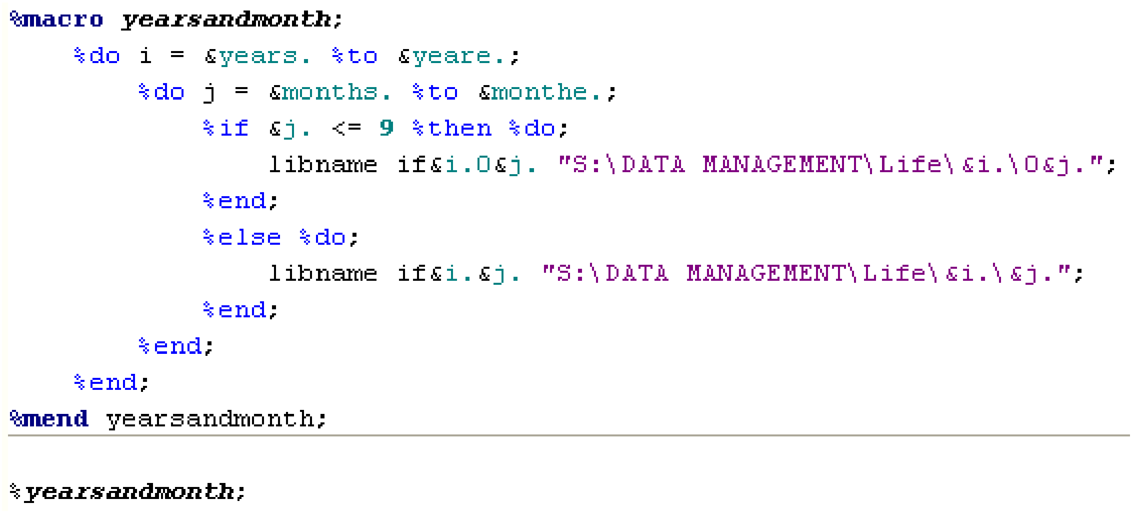
By using the parameter call, we can easily update the code or even used the macro code multiple times in a code.

Log (Macro Code11):



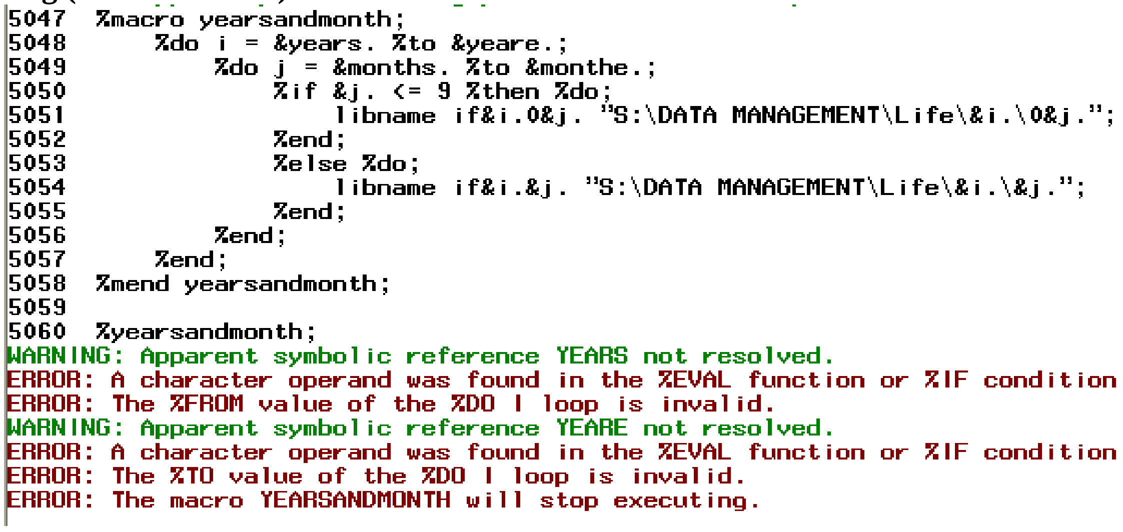
```
%macro yearsandmonth(years=,yeare=,months=,monthe=);
    %do i = &years. %to &yeare.;
        %do j = &months. %to &monthe.;
            %if &j. <= 9 %then %do;
                libname if&i.0&j. "S:\DATA MANAGEMENT\Life\&i.\0&j.";
            %end;
            %else %do;
                libname if&i.&j. "S:\DATA MANAGEMENT\Life\&i.\&j.";
            %end;
        %end;
    %end;
%mend yearsandmonth;

%yearsandmonth(years=2008,yeare=2012,months=1,monthe=12);
NOTE: Libref IF200801 was successfully assigned as follows:
      Engine:        V9
      Physical Name: S:\DATA MANAGEMENT\Life\2008\01
NOTE: Libref IF200802 was successfully assigned as follows:
      Engine:        V9
      Physical Name: S:\DATA MANAGEMENT\Life\2008\02
NOTE: Libref IF200803 was successfully assigned as follows:
      Engine:        V9
      Physical Name: S:\DATA MANAGEMENT\Life\2008\03
NOTE: Libref IF200804 was successfully assigned as follows:
      Engine:        V9
      Physical Name: S:\DATA MANAGEMENT\Life\2008\04
NOTE: Libref IF200805 was successfully assigned as follows:
      Engine:        V9
      Physical Name: S:\DATA MANAGEMENT\Life\2008\05
NOTE: Libref IF200806 was successfully assigned as follows:
      Engine:        V9
      Physical Name: S:\DATA MANAGEMENT\Life\2008\06
NOTE: Libref IF200807 was successfully assigned as follows:
      Engine:        V9
      Physical Name: S:\DATA MANAGEMENT\Life\2008\07
NOTE: Libref IF200808 was successfully assigned as follows:
      Engine:        V9
      Physical Name: S:\DATA MANAGEMENT\Life\2008\08
NOTE: Libref IF200809 was successfully assigned as follows:
      Engine:        V9
      Physical Name: S:\DATA MANAGEMENT\Life\2008\09
NOTE: Libref IF200810 was successfully assigned as follows:
      Engine:        V9
      Physical Name: S:\DATA MANAGEMENT\Life\2008\10
NOTE: Libref IF200811 was successfully assigned as follows:
      Engine:        V9
      Physical Name: S:\DATA MANAGEMENT\Life\2008\11
NOTE: Libname IF200812 refers to the same physical library as INF2008.
NOTE: Libref IF200812 was successfully assigned as follows:
      Engine:        V9
      Physical Name: S:\DATA MANAGEMENT\Life\2008\12
NOTE: Libref IF200901 was successfully assigned as follows:
      Engine:        V9
      Physical Name: S:\DATA MANAGEMENT\Life\2009\01
NOTE: Libref IF200902 was successfully assigned as follows:
      Engine:        V9
```

Macro Code12:



```
%macro yearsandmonth;
    %do i = &years. %to &yeare.;
        %do j = &months. %to &monthe.;
            %if &j. <= 9 %then %do;
                libname if&i.0&j. "S:\DATA MANAGEMENT\Life\&i.\0&j.";
            %end;
            %else %do;
                libname if&i.&j. "S:\DATA MANAGEMENT\Life\&i.\&j.";
            %end;
        %end;
    %end;
%mend yearsandmonth;

%yearsandmonth;
```

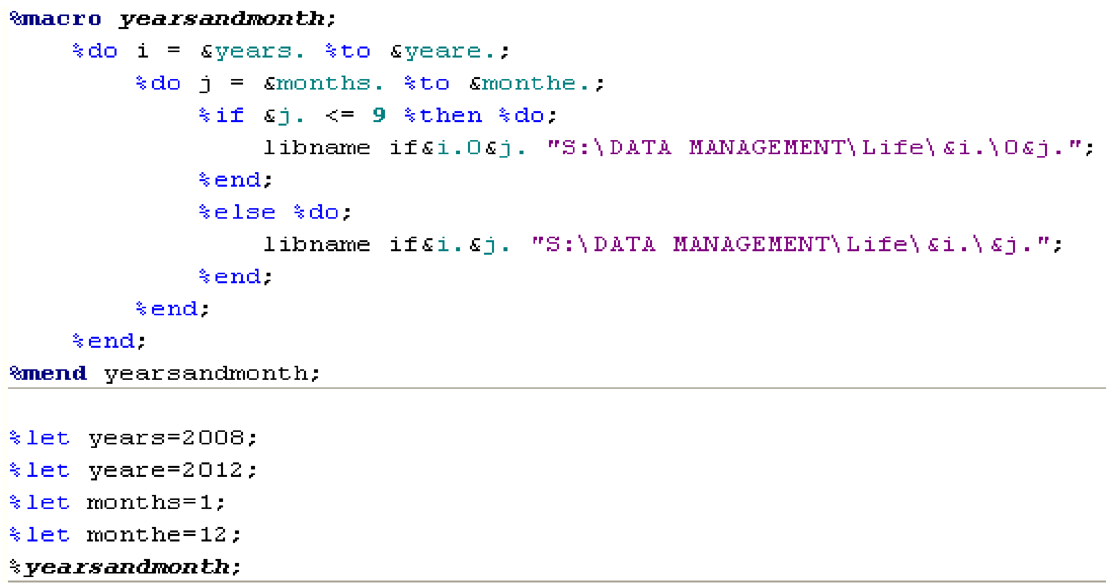
Log (Macro Code12):



```
%macro yearsandmonth;
    %do i = &years. %to &yeare.;
        %do j = &months. %to &monthe.;
            %if &j. <= 9 %then %do;
                libname if&i.0&j. "S:\DATA MANAGEMENT\Life\&i.\0&j.";
            %end;
            %else %do;
                libname if&i.&j. "S:\DATA MANAGEMENT\Life\&i.\&j.";
            %end;
        %end;
    %end;
%mend yearsandmonth;

%yearsandmonth;
WARNING: Apparent symbolic reference YEARS not resolved.
ERROR: A character operand was found in the %EVAL function or %IF condition
ERROR: The %FROM value of the %DO I loop is invalid.
WARNING: Apparent symbolic reference YEARE not resolved.
ERROR: A character operand was found in the %EVAL function or %IF condition
ERROR: The %TO value of the %DO I loop is invalid.
ERROR: The macro YEARSANDMONTH will stop executing.
```

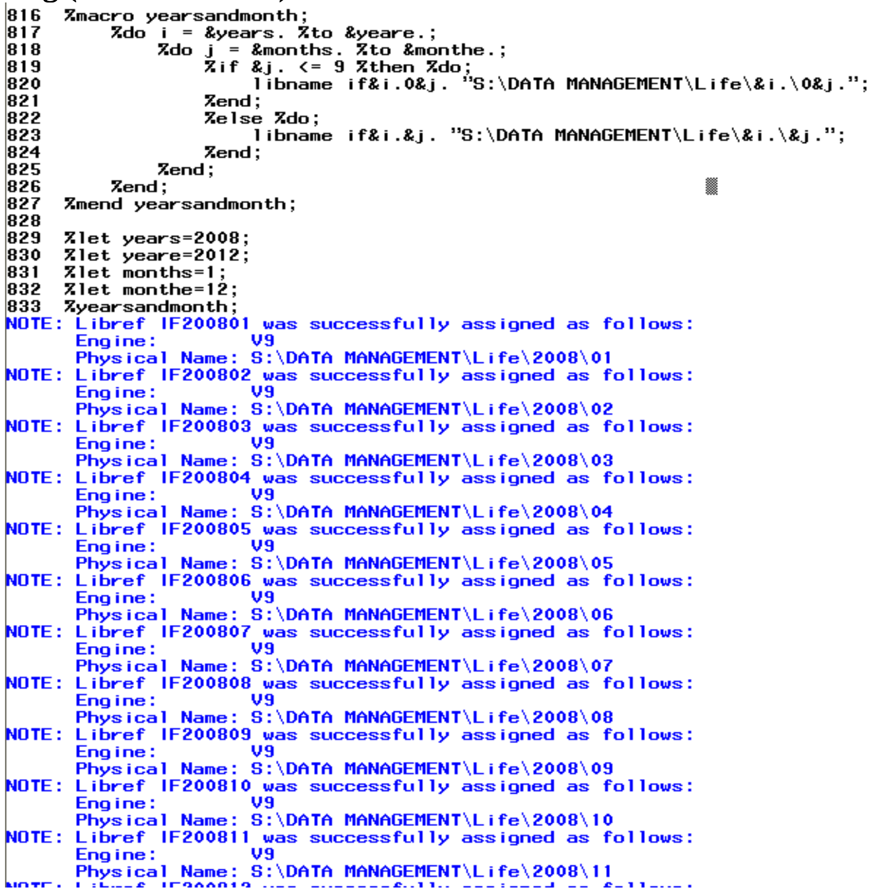
The log output for macro code12 shows some error messages. This is because we did not set the beginning and end year; and beginning and end month.

Macro Code13:



```
%macro yearsandmonth;
    %do i = &years. %to &yeare.;
        %do j = &months. %to &monthe.;
            %if &j. <= 9 %then %do;
                libname if&i.0&j. "S:\DATA MANAGEMENT\Life\&i.\0&j.";
            %end;
            %else %do;
                libname if&i.&j. "S:\DATA MANAGEMENT\Life\&i.\&j.";
            %end;
        %end;
    %end;
%mend yearsandmonth;

%let years=2008;
%let yeare=2012;
%let months=1;
%let monthe=12;
%yearsandmonth;
```

Log (Macro Code13):

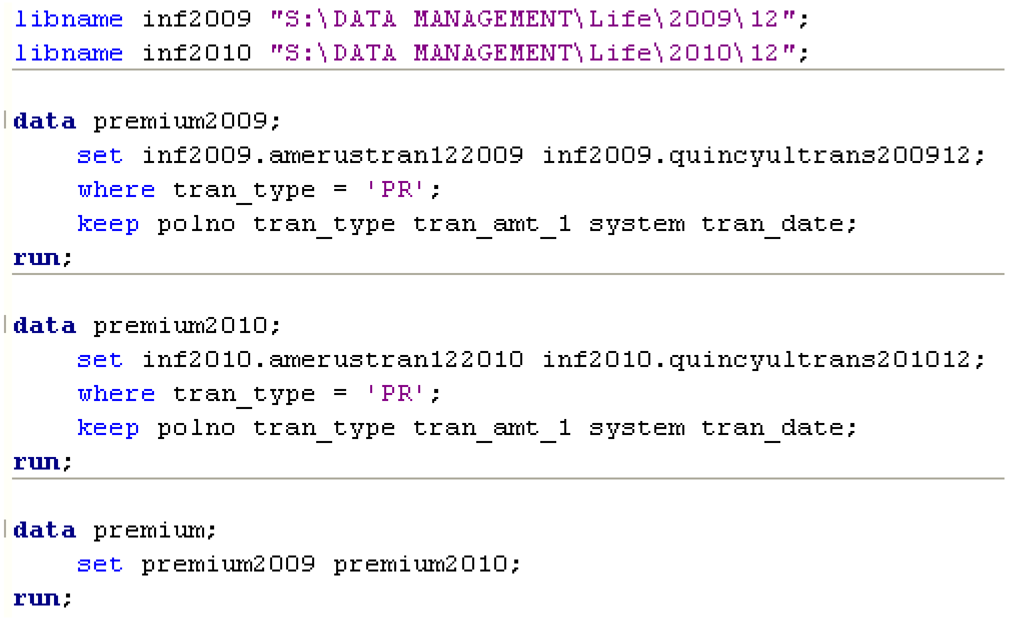


```
%macro yearsandmonth;
    %do i = &years. %to &yeare.;
        %do j = &months. %to &monthe.;
            %if &j. <= 9 %then %do;
                libname if&i.0&j. "S:\DATA MANAGEMENT\Life\&i.\0&j.";
            %end;
            %else %do;
                libname if&i.&j. "S:\DATA MANAGEMENT\Life\&i.\&j.";
            %end;
        %end;
    %end;
%mend yearsandmonth;

%let years=2008;
%let yeare=2012;
%let months=1;
%let monthe=12;
%yearsandmonth;
NOTE: Libref IF200801 was successfully assigned as follows:
      Engine:        V9
      Physical Name: S:\DATA MANAGEMENT\Life\2008\01
NOTE: Libref IF200802 was successfully assigned as follows:
      Engine:        V9
      Physical Name: S:\DATA MANAGEMENT\Life\2008\02
NOTE: Libref IF200803 was successfully assigned as follows:
      Engine:        V9
      Physical Name: S:\DATA MANAGEMENT\Life\2008\03
NOTE: Libref IF200804 was successfully assigned as follows:
      Engine:        V9
      Physical Name: S:\DATA MANAGEMENT\Life\2008\04
NOTE: Libref IF200805 was successfully assigned as follows:
      Engine:        V9
      Physical Name: S:\DATA MANAGEMENT\Life\2008\05
NOTE: Libref IF200806 was successfully assigned as follows:
      Engine:        V9
      Physical Name: S:\DATA MANAGEMENT\Life\2008\06
NOTE: Libref IF200807 was successfully assigned as follows:
      Engine:        V9
      Physical Name: S:\DATA MANAGEMENT\Life\2008\07
NOTE: Libref IF200808 was successfully assigned as follows:
      Engine:        V9
      Physical Name: S:\DATA MANAGEMENT\Life\2008\08
NOTE: Libref IF200809 was successfully assigned as follows:
      Engine:        V9
      Physical Name: S:\DATA MANAGEMENT\Life\2008\09
NOTE: Libref IF200810 was successfully assigned as follows:
      Engine:        V9
      Physical Name: S:\DATA MANAGEMENT\Life\2008\10
NOTE: Libref IF200811 was successfully assigned as follows:
      Engine:        V9
      Physical Name: S:\DATA MANAGEMENT\Life\2008\11
NOTE: Libref IF200812 was successfully assigned as follows:
```

By assigning macro statements with staring year as 2008, ending year as 2012, beginning month as January and ending month as December, we are able to successfully create the librefs that we need, as shown in the log output for macro code13.

Now that we've seen some short examples such as macro assignment statements, how to build macro codes that are transferable, how to remove repetitive codes, using do loops and if-else statements, let's put everything together.

Code7:



```
libname inf2009 "S:\DATA MANAGEMENT\Life\2009\12";
libname inf2010 "S:\DATA MANAGEMENT\Life\2010\12";

data premium2009;
    set inf2009.amerustran122009 inf2009.quincyultrans200912;
    where tran_type = 'PR';
    keep polno tran_type tran_amt_1 system tran_date;
run;

data premium2010;
    set inf2010.amerustran122010 inf2010.quincyultrans201012;
    where tran_type = 'PR';
    keep polno tran_type tran_amt_1 system tran_date;
run;

data premium;
    set premium2009 premium2010;
run;
```

Code7 creates a grand premium dataset, called premium, which contains premium information from December 2009 and December 2010, pulled from amerustran122009, quincyultrans200912, amerustran122010 and quincyultrans201012 respectively.

This is a simple and easy code to write since it only needs two years worth of information. What if we need five, ten, even fifteen years of information? That is going to be a long and repetitive code. Macro code 14 below shows an equivalent code for code7 but minus the repetitiveness by using macros, do loops, and if-else statements.

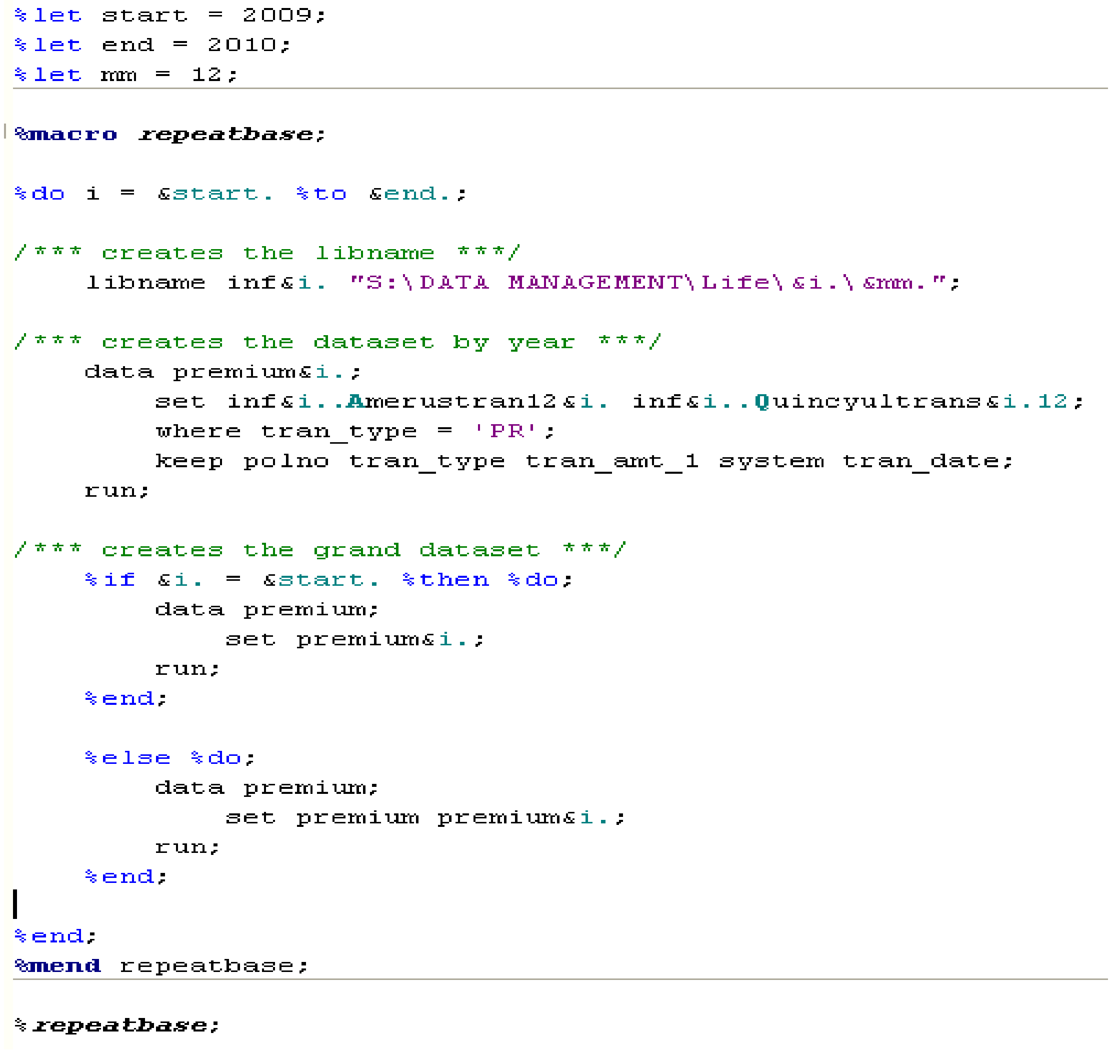
We see in code7 that everything is the same except for the years (2009 and 2010). So we'll start by creating macro assignments for 2009 and 2010 as the start and end years.

Using a do loop with start macro keyword assigned to 2009, we create a libname called inf2009. We then create a dataset called premium2009 by pulling premium transactions from inf2009.amerustran122009 and inf2009.quincyultrans200912. Next, since *i*, which is 2009 is equals to &starts. (2009), we will set the premium datasets as premium2009. This will end if-else statement and end the do loop.

Next, the end macro keyword is assigned as 2010. The inf2010 libname is then created. We then pull the premium transaction information from inf2010.amerustran122010 and inf2010.quincyultrans201012. Since *i*, which is assigned as 2010 now, does not equal to &start. (2009), the if-else statement will skip the if section of the code and go to the else section of the code and set the premium dataset as the previous premium dataset (premium2009) and premium2010.

The if-else statement is needed in the code or else the premium dataset will always be overwritten with the new premium dataset.

Macro Code14:



```
%let start = 2009;
%let end = 2010;
%let mm = 12;

%macro repeatbase;

%do i = &start. %to &end.;

/*** creates the libname ***/
    libname inf&i. "S:\DATA MANAGEMENT\Life\&i.\&mm.";

/*** creates the dataset by year ***/
    data premium&i.;
        set inf&i..Amerustran12&i. inf&i..Quincyultrans&i.12;
        where tran_type = 'PR';
        keep polno tran_type tran_amt_1 system tran_date;
    run;

/*** creates the grand dataset ***/
    %if &i. = &start. %then %do;
        data premium;
            set premium&i.;
        run;
    %end;

    %else %do;
        data premium;
            set premium premium&i.;
        run;
    %end;

%end;
%mend repeatbase;

%repeatbase;
```

Conclusion

To summarize, Macros is a very powerful and useful tool. It not only allows us to update our codes easily, but also makes our codes easily transferred between different codes. In addition, Macros also helps us shorten our codes by allowing us to remove repetitive parts. Finally, it also enables us to apply logical statements on DATA steps as oppose to just using logical statements with DATA steps.

Contact Information

Name: Audrey Yeo
Enterprise: Aviva USA
Address: 7700 Mills Civic Parkway
City, State, ZIP: West Des Moines, IA 50316
Work Phone: 515-342-3759
E-mail: audrey.yeo@avivausa.com