Demystifying Inherited Programs

Have you ever inherited a program that leaves you scratching your head trying to figure it out? If you have not, chances are that some time in the future you will. While initially the program may work and not require any changes, this may not always be the case. At some point the program may require maintenance due to new requirements, system upgrades, data issues, etc. At that time it will be necessary to figure out what is really happening so you can modify the program.

You have inherited the program provided in Figure 1. After an initial review of the program you have determined there is no documentation and while the code executes, it is not very readable.

Figure 1:

```
options nosource ls=80;
%macro macro1;
data z (keep=deptnum full);
length full $16;
set y;
full=fname !! lname;
format deptnum $deptfmt.;
label full='Employee name';
label deptnum='Department name';
label empid='employee id';
%Mend;
filename deptnam 'c:\temp\deptname.txt' ;
filename fmtpgm 'c:\temp\deptfmt.sas';
Data x;
infile deptnam  end=eof pad;
input @1 deptno $3. @5 deptname $15.;
file   fmtpgm;
if _n_ = 1 then
put "proc format; value $deptfmt";
put "'" deptno   "' = '"  deptname "'";
if eof then put "; run;";
run;
%include fmtpgm;
data y;
input deptnum $ empid $ fname $ lname $;
datalines;
101 1001 steve last
102 1002 fred  derf
103 1003 mike ekim
104 1004 mary gold
;
```

```
run;
%macro1;
proc print label;
title 'original report';
run;
```

You decide to execute the program to see if the resulting log (Figure 2) and output (Figure 3) will help in deciphering the process.

Figure 2:

```
1    options nosource ls=80;
NOTE: The infile DEPTNAM is:
      FILENAME=c:\temp\deptname.txt,
      RECFM=V,LRECL=256
NOTE: The file FMTPGM is:
      FILENAME=c:\temp\deptfmt.sas,
      RECFM=V,LRECL=256
NOTE: 4 records were read from the infile DEPTNAM.
      The minimum record length was 11.
      The maximum record length was 14.
NOTE: 6 records were written to the file FMTPGM.
      The minimum record length was 6.
      The maximum record length was 27.
NOTE: The data set WORK.X has 4 observations and 2 variables.
NOTE: The DATA statement used 0.28 seconds.
NOTE: %INCLUDE (level 1) file FMTPGM is file c:\temp\deptfmt.sas.
NOTE: Format $DEPTFMT has been output.
NOTE: The PROCEDURE FORMAT used 0.33 seconds.
NOTE: %INCLUDE (level 1) ending.
NOTE: The data set WORK.Y has 4 observations and 4 variables.
NOTE: The DATA statement used 0.17 seconds.
NOTE: The data set WORK.Z has 4 observations and 2 variables.
NOTE: The DATA statement used 0.17 seconds.
NOTE: The PROCEDURE PRINT used 0.16 seconds.
```

Figure 3:

```
         original report
         Employee      Department
OBS        name           name
 1     steve   last    Purchasing
 2     fred    derf    Payroll
 3     mike    ekim    Personnel
 4     mary    gold    Shipping
```

The original SAS log is very hard to decipher. We are left with the following questions:

1. How are the data sets X, Y, and Z created?
2. What data is within each of these data sets?
3. What code is being executed from the included file FMTPGM?

At this point there are several steps we can take to make the code more readable and to begin to demystify the program:

1. Add RUN statements to explicitly see the end of SAS steps.
2. Indent statements within each of the SAS steps.
3. Turn on all source OPTIONS such as SOURCE, SOURCE2, MPRINT, and SYMBOLGEN to show more information in the SAS log which will be useful for tracing logic.
4. Draw a rough data flow, or program flowchart, to help track where data comes from, and where it goes.
5. Examine each step from the top down, noting data sources and transformations.
6. Ask questions. Even if the original programmer is no longer around someone may know about the code.

We need to see what is really happening during the program execution. To accomplish this we turn on the options SOURCE, SOURCE2, MPRINT, and SYMBOLGEN and run the program again. The resulting log is provided in Figure 4.

Figure 4:

```
3    options source source2 mprint symbolgen ls=80;
4    %macro macro1;
5    data z (keep=deptnum full);
6    length full $16;
7    set y;
8    full=fname !! lname;
9    format deptnum $deptfmt.;
10   label full='Employee name';
11   label deptnum='Department name';
12   label empid='employee id';
13   %Mend;
14   filename deptnam 'c:\temp\deptname.txt' ;
15   filename fmtpgm 'c:\temp\deptfmt.sas';
16   Data x;
17   infile deptnam  end=eof pad;
18   input @1 deptno $3. @5 deptname $15.;
19   file   fmtpgm;
20   if _n_ = 1 then
21   put "proc format; value $deptfmt";
22   put "'" deptno   "' = '"  deptname  "'";
23     if eof then put "; run;"; run;
```

```
NOTE: The infile DEPTNAM is:
      FILENAME=c:\temp\deptname.txt,
      RECFM=V,LRECL=256
NOTE: The file FMTPGM is:
      FILENAME=c:\temp\deptfmt.sas,
      RECFM=V,LRECL=256
NOTE: 4 records were read from the infile DEPTNAM.
      The minimum record length was 11.
      The maximum record length was 14.
NOTE: 6 records were written to the file FMTPGM.
      The minimum record length was 6.
      The maximum record length was 27.
NOTE: The data set WORK.X has 4 observations and 2 variables.
NOTE: The DATA statement used 0.27 seconds.
24   %include fmtpgm;
NOTE: %INCLUDE (level 1) file FMTPGM is file c:\temp\deptfmt.sas.
25  +proc format;
26  +value $deptfmt
27  +'101 ' = 'Purchasing '
28  +'102 ' = 'Payroll '
29  +'103 ' = 'Personnel '
30  +'104 ' = 'Shipping '
31  +;
NOTE: Format $DEPTFMT has been output.
32  +run;
NOTE: The PROCEDURE FORMAT used 0.27 seconds.
NOTE: %INCLUDE (level 1) ending.
33   data y;
34   input deptnum $ empid $ fname $ lname $;
35   datalines;
NOTE: The data set WORK.Y has 4 observations and 4 variables.
NOTE: The DATA statement used 0.17 seconds.
36   ;
37   run;
38   %macro1;
MPRINT(MACRO1):   DATA Z (KEEP=DEPTNUM FULL);
MPRINT(MACRO1):   LENGTH FULL $16;
MPRINT(MACRO1):   SET Y;
MPRINT(MACRO1):   FULL=FNAME !! LNAME;
MPRINT(MACRO1):   FORMAT DEPTNUM $DEPTFMT.;
MPRINT(MACRO1):   LABEL FULL= 'Employee name';
MPRINT(MACRO1):   LABEL DEPTNUM= 'Department name';
MPRINT(MACRO1):   LABEL EMPID= 'employee id';
NOTE: The data set WORK.Z has 4 observations and 2 variables.
NOTE: The DATA statement used 0.17 seconds.
```

```
47   proc print label;
48   title 'original report';
49   run;
NOTE: The PROCEDURE PRINT used 0.11 seconds.
```

Now comes our challenge. We have been asked to make the following changes:

- Add a new department – 105 (Receiving)
- Add a new employee – Penn Teller
- Show the employee id on the report

We determine that to add a department we must add a record to the file deptname.txt. We also determine that to add an employee we must add a record to the instream data used to build the data set Y. But, how do we add employee id to the report?

By looking through the more complete log created with the options, several things become apparent. We see the data is manipulated in a macro and three data sets, X, Y, and Z, are created. We determine that the data from data set Z is output in the report. We also see the data set only keeps the department and name variables. To include the employee id on the report we must keep it on the data set.

As we noticed, the data sets do not have meaningful names. This makes it more difficult to trace the data through the process. The program also does not contain any comments or documentation explaining the program. We can take the following steps to further clean up the program:

1. Add comments to provide an overview of the program and identify data inputs and outputs.
2. Add comments describing each step.
3. Keep a log of modifications.
4. Use meaningful data set and variable names.
5. Explicitly name all data sets when used.
6. Use clean coding styles, such as indentation, for readability.

We have made the necessary changes, given the data sets meaningful names, cleaned up the code, and included comments. The revised program is provided in Figure 5.

Figure 5:

```
/*****************************************************************/
/* PROJECT:   Report Department Employees                       */
/* PROGRAM:   c:\programs\dept_emp_list.sas                     */
/*                                                               */
/* DESCRIPTION:                                                  */
/*   Create list of all employees within each department.       */
/*                                                              */
/* INPUT:                                                        */
/*   FILEREF   DSN                                    TYPE       */
```

```
 /*   --------  --------------------------------------  -------   */
 /*   deptnam   c:\temp\deptname.txt                    TEXT      */
 /*                                                               */
 /* WRITTEN BY: SYSTEMS SEMINAR CONSULTANTS                       */
 /*             MADISON,  WI                                      */
 /*             (608) 278-9964                                    */
 /*                                                               */
 /*****************************************************************/
 /* CHANGES:                                                      */
 /*   DATE       BY     CHANGE                                    */
 /*   06/15/2012 Teresa Added receiving department and employee    */
 /*                     Penn Teller, and included employee id on   */
 /*                     the final report.                         */
 /*                                                               */
 /*****************************************************************/
options source source2
        mprint symbolgen linesize=80;
  /*****************************************************/
  /* start macro1, this reads dataset empnam and creates */
  /* the fullnam dataset for the report.               */
  /*****************************************************/
%macro macro1;
  data fullnam (keep=deptnum empid full);
    length full $16;
    set empnam;
    /* create full name from first and last   */
    /* trimming extra blanks after first name */
    full = trim(fname) !! ' ' !! lname;

    /* assign the format we created */
    format deptnum $deptfmt.;

    /* assign labels to name, deptnumber, and empid */
    label full='Employee name';
    label deptnum='Department name';
    label empid='employee id';
  run;
%Mend macro1;

  /* assign file with dept numbers/names for format */
filename deptnam 'c:\temp\deptname2.txt' ;

  /* assign file to write out proc format code */
filename fmtpgm 'c:\temp\deptfmt.sas';

  /*****************************************************/
  /* create the code for the department name format       */
  /* this is the file with the department number and name */
  /* we use the end of file option (end=) to allow        */
  /* writing last line of code in proc format, and the    */
  /* pad to prevent problems reading input lines that are */
  /* too short.                                           */
  /******************************************************/
Data _null_;
  infile deptnam  end=eof pad lrecl=50;
  /* read in a record */
  input @1  deptno   $3.
        @5  deptname $15.
        ;
  /* send output from put statements to file */
  file fmtpgm;

  /* if we are processing the first record, write */
  /* out the SAS code for the proc format first   */
  if _n_ = 1 then
     put "proc format; value $deptfmt";

  /* for each record on the department name file, */
  /* write out a record for the values in the     */
  /* format. This builds a line looking like:     */
  /*   '1001' = 'Purchasing'                      */
  xline = "'" !! deptno !! "' = '" !! deptname !! "'";
  put xline;

  /* if we have processed the last record then write */
  /* out the ending statements for the proc format   */
  if eof then
    put "; run;";
run;

  /* include and run the proc format step in file fmtpgm */
%include fmtpgm;

  /* step to read employee data and create dataset empnam */
data empnam;
  input deptnum $ empid $ fname $ lname $;
datalines;
101 1001 steve last
102 1002 fred  derf
103 1003 mike ekim
104 1004 mary gold
```

```
105 1005 penn teller
;
run;

  /* macro reads empnam and creates fullnam */
  /* and assigns the format we created      */
%macro1;

  /* this step prints our report */
proc print data=fullnam label;
 title 'revised report';
run;
```

The new results are provided in Figure 6.

Figure 6:

```
             revised report
       Employee      Department    employee
OBS      name           name          id
1     steve last     Purchasing      1001
2     fred derf      Payroll         1002
3     mike ekim      Personnel       1003
4     mary gold      Shipping        1004
5     penn teller    Receiving       1005
```

Besides allowing us to complete the requested changes we have hopefully helped anyone else who will read this program in the future. An important thing to remember is to assume every program you write or change will be used in the future. Whether it is you or someone else, clean code and documentation are invaluable to the next person who needs to work with the program.