CHAPTER
20

The FORMS Procedure

Overview

The FORMS procedure produces labels for envelopes, mailing labels, external tape labels, file cards, and any other printer forms that have a regular pattern.

For each observation in the input SAS data set, PROC FORMS prints data in a rectangular block called a *form unit*. For example, a mailing label is a form unit.

Output 20.1 on page 481 illustrates a simple mailing list produced with PROC FORMS. The statements that produce the output follow. The OBS= data set option limits to six the number of observations that PROC FORMS processes.

```
options pagesize=60 linesize=64 nodate
        pageno=1;


filename labels 'external-file';


proc forms data=list(obs=6) file=labels
           align=0;
   line 1 name;
   line 2 street;
   line 3 city state zip;
run;
```

Output 20.1 Simple Mailing List Produced with PROC FORMS

```
Gabrielli, Theresa
24 Ridgetop Rd.
Westboro        MA 01581

Clayton, Aria
314 Bridge St.
Hanover         NH 03755

Dix, Martin L.
4 Shepherd St.
Norwich         VT 05055

Slater, Emily C.
2009 Cherry St.
York            PA 17407

Ericson, Jane
211 Clancey Court
Chapel Hill     NC 27514

An, Ing
95 Willow Dr.
Charlotte       NC 28211
```

Output 20.2 on page 482 is a customized version of the same mailing list. The statements that create this list

- □ invert the name so the first name appears first
- □ eliminate extra spaces between the city and state
- □ place three form units in each row
- □ make three copies of each form
- □ use only observations from the states in New England.

For an explanation of the program that produces these labels, see Example 3 on page 496.

Output 20.2 Customized Mailing List Produced with PROC FORMS

```


Theresa Gabrielli        Theresa Gabrielli        Theresa Gabrielli
24 Ridgetop Rd.          24 Ridgetop Rd.          24 Ridgetop Rd.
Westboro MA 01581        Westboro MA 01581        Westboro MA 01581


Aria Clayton             Aria Clayton             Aria Clayton
314 Bridge St.           314 Bridge St.           314 Bridge St.
Hanover NH 03755         Hanover NH 03755         Hanover NH 03755


Martin L. Dix            Martin L. Dix            Martin L. Dix
4 Shepherd St.           4 Shepherd St.           4 Shepherd St.
Norwich VT 05055         Norwich VT 05055         Norwich VT 05055
```

Procedure Syntax

Requirements: At least one LINE statement

Reminder: You can use the ATTRIB, FORMAT, and WHERE statements. See Chapter 3, "Statements with the Same Function in Multiple Procedures," for details. You can also use any global statements as well. See Chapter 2, "Fundamental Concepts for Using Base SAS Procedures," for a list.

PROC FORMS <*option(s)*>;
  BY <DESCENDING> *variable-1*
    <...<DESCENDING> *variable-n*>
    <NOTSORTED>;
  FREQ *variable*;
  LINE*line-number variable(s)* </ *option(s)*>;

To do this	Use this statement
Produce a separate set of forms for each BY group	BY
Treat observations as if they appear multiple times in the input data set	FREQ
Specify the information to print on a line of the form unit	LINE

PROC FORMS Statement

PROC FORMS <*option(s)*>;

To do this	Use this option
Specify the input data set	DATA=
Identify an external file for PROC FORMS to write to	FILE=
Control the dimensions of a form	
Specify the number of lines in a form unit	LINES=
Specify the number of columns across the form unit	WIDTH=
Control the placement of the forms	
Specify the number of form units to print across the page	ACROSS=
Specify the number of spaces to print between form units	BETWEEN=
Specify the number of lines to skip on a page before printing the first form unit	DOWN=
Specify the number of spaces to indent before printing the first form unit in each row	INDENT=

To do this	Use this option
Specify the number of form units to print down the page	NDOWN=
Specify the number of lines on a page of forms	PAGESIZE=
Specify the number of lines to skip between form units	SKIP=
Control the number of each form unit that PROC FORMS prints	
Specify the number of form units to produce for each observation in each set of form units	COPIES=
Specify the number of sets of form units to produce	SETS=
Control the placement of page eject characters	CC
Specify the number of lines of dummy form units to print	ALIGN=

Options

ACROSS=*form-units-per-line*
specifies the number of form units to print across the page. (See Figure 20.1 on page 490.)

Alias: A=

Default: 1

Range: 1-200

Featured in: Example 1 on page 491

ALIGN=*number*
controls the number of alignment form units that print before the actual data values. Use the alignment form units, which consist solely of Xs, to check printer alignment.

Default: 8 with FILE=; 0 without FILE=

Interaction: If you use ACROSS=, the number of dummy form units that print is the product of the values of ACROSS= and ALIGN=.

Featured in: Example 1 on page 491

BETWEEN=*spaces-between-form-units*
specifies the number of spaces to print between form units. (See Figure 20.1 on page 490.)

Alias: B=

Default: 1

Range: 1-200

Featured in: Example 1 on page 491

CC
in continuous mode, writes a page-eject character at the top of the first page. In page mode, if you also specify FILE=, CC writes a page-eject character at the top of each page. (CC has no effect if you omit FILE=.) For a discussion of page mode and continuous mode, see "Modes of Operation" on page 490.

Tip: If you omit CC, PROC FORMS issues blank lines to go to the next page. We recommend that you always use CC with page-mode operation.

Featured in: Example 2 on page 493

COPIES=*number*
specifies the number of form units to produce for each observation in each set of form units. All copies of an observation appear together.

Alias: C=

Default: 1

Featured in: Example 3 on page 496

DATA=*SAS-data-set*
identifies the input SAS data set.

DOWN=*top-margin*
specifies the number of lines to skip on a page before printing the first form unit. (See Figure 20.1 on page 490.)

Alias: D=

Default: 1

Range: 1-200

Featured in: Example 1 on page 491

Note: When PROC FORMS writes to the procedure output file, it uses one line for each TITLE statement and leaves a blank line beneath the last title. Counting for the top margin begins at the next line. Thus, if you have two TITLE statements and specify DOWN=5, PROC FORMS begins printing the first form unit on each page on line 9. △

FILE=*fileref*
identifies an external file for PROC FORMS to write to. Use the FILENAME statement to associate an external file with a fileref (see *SAS Language Reference: Concepts*).

Alias: DDNAME=, D=

Default: If you omit FILE=, PROC FORMS writes to the procedure output file and selects page mode.

Interaction: If you use FILE= and do not specify the ALIGN= option, PROC FORMS uses ALIGN=8.

Interaction: When you use FILE=, PROC FORMS honors DOWN= only on the first page of form units.

Interaction: If you use FILE= with NDOWN= or PAGESIZE= or both, you select page mode. Otherwise, you select continuous mode.

Featured in: Example 1 on page 491

INDENT=*left-margin*
specifies the number of spaces to indent before printing the first form unit in each row. (See Figure 20.1 on page 490.)

Alias: I=

Default: 0

Range: 0-200

LINES=*form-unit-length*
specifies the number of lines in a form unit. (See Figure 20.1 on page 490.)

Alias: L=

Default: the largest number used with the LINE statement

Range: 1-200

NDOWN=*form-units-per-page*
specifies the number of form units to print down the page and selects page-mode operation. (See Figure 20.1 on page 490.)

Alias: ND=

Default: FLOOR((PAGESIZE–DOWN+SKIP)/(LINES+SKIP)) where FLOOR is a SAS function that returns the largest integer less than or equal to the value of the argument.

Interaction: If NDOWN= specifies a number of form units that is less than PAGESIZE= allows, PROC FORMS honors NDOWN=. If NDOWN= specifies a number of form units that is greater than PAGESIZE= allows, PROC FORMS adjusts the value of NDOWN= downwards to accommodate the page size.

Featured in: Example 2 on page 493

PAGESIZE=*lines-per-page*
specifies the number of lines on a page of forms after allowing for TITLE statements and a blank line following the titles. (See Figure 20.1 on page 490.) It also selects page-mode operation.

Alias: PS=

Default: the system page size (with FILE=); inferred from the characteristics of the procedure output file and from title information (without FILE=)

Range: the value of DOWN= plus the value of LINES=, up to 10,000

Interaction: When you write to the procedure output, if the page size that you specify is greater than the page size specified by the SAS system option PAGESIZE=, PROC FORMS adjusts the PROC FORMS page size to accommodate the system page size.

Interaction: If the page size allows for more form units than NDOWN= specifies, PROC FORMS honors the NDOWN= option. If the page size does not allow for as many form units as NDOWN= specifies, PROC FORMS adjusts the value of NDOWN= to accommodate the page size.

SETS=*number*
specifies the number of sets of form units to produce. In page-mode operation, PROC FORMS starts each set on a new page.

Default: 1

Featured in: Example 2 on page 493

SKIP=*lines-between-form-units*
specifies the number of lines to skip between form units. (See Figure 20.1 on page 490.)

Alias: S=

Default: 1

Range: 1-200

Featured in: Example 1 on page 491

WIDTH=*form-unit-width*
specifies the number of columns across the form unit. PROC FORMS truncates values that do not fit in the specified width. (See Figure 20.1 on page 490.)

Alias: W=

Default: width of the widest line

Range: 1-255

Featured in: Example 1 on page 491

BY Statement

Produces a separate set of forms for each BY group.

Main discussion: “BY” on page 68

BY <DESCENDING> *variable-1*
<…<DESCENDING> *variable-n*>
<NOTSORTED>;

Required Arguments

variable
specifies the variable that the procedure uses to form BY groups. You can specify more than one variable. If you do not use the NOTSORTED option in the BY statement, the observations in the data set must either be sorted by all the variables that you specify, or they must be indexed appropriately. Variables in a BY statement are called *BY variables*.

Options

DESCENDING
specifies that the data set is sorted in descending order by the variable that immediately follows the word DESCENDING in the BY statement.

NOTSORTED
specifies that observations are not necessarily sorted in alphabetic or numeric order. The data are grouped in another way, for example, chronological order.

The requirement for ordering or indexing observations according to the values of BY variables is suspended for BY-group processing when you use the NOTSORTED option. In fact, the procedure does not use an index if you specify NOTSORTED. The procedure defines a BY group as a set of contiguous observations that have the same values for all BY variables. If observations with the same values for the BY variables are not contiguous, the procedure treats each contiguous set as a separate BY group.

How PROC FORMS Separates BY Groups

In page mode, the forms for each BY group begin on a new page. In continuous mode, BY groups are not separated.

FREQ Statement

Treats observations as if they appear multiple times in the input data set.

FREQ *variable*;

Required Arguments

variable

specifies a numeric variable whose value represents the frequency of each observation. If you use the FREQ statement, the procedure assumes that each observation in the input data set represents *n* observations, where *n* is the value of *variable*. If *n* is not an integer, the SAS System truncates it. If *n* is less than 1 (which includes missing), the procedure does not use that observation.

The sum of the frequency variable represents the total number of observations.

LINE Statement

Specifies the information to print on one line of the form unit. Use one LINE statement for each line of the form unit.

LINE *line-number variable(s) </ option(s)>*;

To do this	Use this option
Specify the number of spaces to indent the line within the form unit	INDENT=
Rotate the words in a character variable that contains a comma around the comma and remove the comma	LASTNAME
Remove extra blanks from the line so that one blank separates variables	PACK
Remove periods that represent missing values from a line that contains no other values.	REMOVE

Required Arguments

line-number

identifies the number of the line. You can specify lines in any order. You do not need a LINE statement for a blank line.

Range: An integer between 1 and the value of LINES= in the PROC FORMS statement

variable(s)

specifies one or more variables to print on this line of the form unit. The FORMS procedure inserts one space between each value. By default, the width of a variable's field in the form unit is the formatted length of that variable. Default formats are the length of the variable for character variables and BEST12. for numeric variables.

Interaction: If the length of all values in a line is longer than the value of WIDTH= specified in the PROC FORMS statement, PROC FORMS truncates the values (starting with the rightmost value in the line) to fit the WIDTH= value. For information on squeezing variables onto a line, see PACK on page 489.

Options

INDENT=*margin-within-form-unit*
specifies the number of spaces to indent the line within the form unit. Contrast this option to INDENT= in the PROC FORMS statement, which specifies the size of the left margin preceding the first form unit in each row.

Alias: I=

Featured in: Example 1 on page 491

LASTNAME
rotates the words in a character variable that contains a comma around the comma and removes the comma.

Alias: L

Featured in: Example 1 on page 491

PACK
removes extra blanks from the line so that one blank separates variables.

Alias: P

Tip: PACK can squeeze fields onto a form unit, but if the values for all the variables are long, you may lose an entire field. To avoid this problem, use a FORMAT statement to limit the space for each variable. For example, the following statement sets the field widths of the variables CITY and STATE to 20 and 2 columns, respectively:

```
format city $20. state $2.;
```

Featured in: Example 1 on page 491

REMOVE
removes periods that represent missing values from a line that contains no other values.

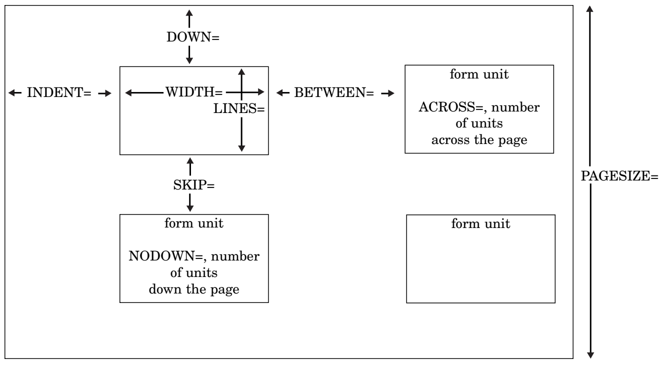
Alias: R

Concepts

Form Layout

The size and spacing of form units are controlled by options in the PROC FORMS statement, as illustrated in Figure 20.1 on page 490. (See also the discussion of these options on page 484.)

Figure 20.1 Sample Placement for Forms


DOWN=
INDENT=
WIDTH=
LINES=
BETWEEN=
form unit
ACROSS=, number of units across the page
SKIP=
form unit
NODOWN=, number of units down the page
form unit
PAGESIZE=

The values of the variables specified in LINE statements are formatted into a form unit that is WIDTH= columns wide and LINES= lines long. Values that do not fit into WIDTH= columns are truncated. ACROSS= form units are printed across the page, with BETWEEN= spaces between adjacent form units. The forms are indented INDENT= spaces from the left margin. SKIP= blank lines are printed between form units down the page.

Modes of Operation

PROC FORMS operates in two modes: continuous mode and page mode. Continuous mode is for forms that feed continuously through a printer, without the printer's needing to perform page ejects. Page mode is for forms that use separate pieces of paper for each form unit or for multiple form units (such as sheets of labels that come with 30 labels per sheet of paper).

By default, PROC FORMS uses page mode. To select continuous mode, you must specify FILE= and must not specify NDOWN= or PAGESIZE=.

In Continuous Mode, PROC FORMS Always Writes to an External File

When it writes in continuous mode, PROC FORMS

1 skips the number of lines specified by DOWN=
2 prints one form unit
3 skips the number of lines specified by SKIP=
4 repeats steps 2 and 3 until it uses all the data.

By default, in continuous mode the first eight form units are dummy form units that consist solely of Xs. These forms give the printer operator a chance to align the printer before real form units begin to print. Use ALIGN= to alter the number of dummy form units. Once the dummy form units are aligned to the physical forms, the file prints correctly. Carriage control characters are unnecessary.

In Page Mode, PROC FORMS Can Write Either to an External File or to the Procedure Output File

In page mode, PROC FORMS

1 goes to the top of a new page
2 skips the number of lines specified by DOWN=

3 prints the number of form units specified by NDOWN= down the page, or if you omit NDOWN=, prints the maximum number of form units allowed by the page size

4 repeats steps 1 to 3 until it uses all the data.

When PROC FORMS has written as many form units as you specified, either it writes a blank line for each line remaining on the page (as determined by the PAGESIZE= option) or it writes a page-eject character. If you are writing to the procedure output file, PROC FORMS always writes the page-eject characters. If you have specified FILE=, PROC FORMS by default writes blank lines, but if you specify the CC option, it writes page eject characters instead.

In page mode, the easiest way to ensure proper alignment is to specify the number of form units to print down the page with the NDOWN= option and to use CC to write a page-eject character at the beginning of each page. If you omit CC, be sure that the page size is set correctly. If it isn't, the number of blank lines that PROC FORMS writes will not take you to the top of the next page.

Note: We recommend that you always use CC when you use page mode with the FILE= option. △

CAUTION:

The procedure output file contains some things that you may not want on your forms. If you omit the FILE= option, the output from PROC FORMS goes to the procedure output file. If the DATE and NUMBER options are in effect, the output will contain dates and page numbers. If any titles or footnotes are defined, they will appear in the output as well. △

Examples

The examples in this chapter assume alignment for the forms that they use. You must experiment to determine how to align your form units with your forms.

Example 1: Printing a Single Form Unit for Each Observation

Procedure features:

PROC FORMS statement options:

ACROSS=
ALIGN=
BETWEEN=
DOWN=
FILE=
SKIP=
WIDTH=

LINE statement options:

INDENT=
LASTNAME
PACK

Other features:

SORT procedure

This example uses PROC FORMS to print one set of mailing labels consisting of one copy of the form unit for each observation.

Program

The data set LIST contains names and mailing addresses. PROC SORT sorts the data by ZIP code.

```
options nodate pageno=1 linesize=80 pagesize=60;
data list;
   input Name $ 1-19 Street $ 20-39 City $ 40-54
         State $ 55-56 Zip $ 59-63;
   datalines;
Ericson, Jane      211 Clancey Court   Chapel Hill    NC  27514
Dix, Martin L.     4 Shepherd St.      Norwich        VT  05055
Gabrielli, Theresa 24 Ridgetop Rd.     Westboro       MA  01581
Clayton, Aria      314 Bridge St.      Hanover        NH  03755
Archuleta, Ruby    Box 108             Milagro        NM  87429
Misiewicz, Jeremy  43-C Lakeview Apts. Madison        WI  53704
Ahmadi, Hafez      5203 Marston Way    Boulder        CO  80302
Jacobson, Becky    7 Lincoln St.       Tallahassee    FL  32312
An, Ing            95 Willow Dr.       Charlotte      NC  28211
Slater, Emily C.   2009 Cherry St.     York           PA  17407
;

proc sort data=list;
   by zip;
run;
```

The FILENAME statement associates the name LABELS with the external file that will receive the output from PROC FORMS.

```
filename labels 'external-file';
```

FILE= sends the output to the file associated with the fileref LABELS. Because neither NDOWN= nor PAGESIZE= is specified, PROC FORMS uses continuous mode. WIDTH= sets the width of the form units to 24 to provide enough room for all the variables on each line. ACROSS= writes three form units across each page. BETWEEN= puts four blank characters between adjacent form units. DOWN= skips two lines at the top of the file so that the form units and the forms align correctly. SKIP= skips two lines between form units to maintain the proper alignment. ALIGN= prints two lines of dummy form units.

```
proc forms data=list file=labels
     width=24
     across=3
     between=4
     down=2
```

```
      skip=2
      align=2;
```

The LINE statements specify the variables to place on each line. LASTNAME removes the comma from Name and writes the first name before the last name. PACK removes extra blank characters between City and State. INDENT= indents Zip by 15 spaces.

```
    line 1 name / lastname;
    line 2 street;
    line 3 city state / pack;
    line 4 zip / indent=15;
 run;
```

Output

```


XXXXXXXXXXXXXXXXXXXXXXXX    XXXXXXXXXXXXXXXXXXXXXXXX    XXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXX    XXXXXXXXXXXXXXXXXXXXXXXX    XXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXX    XXXXXXXXXXXXXXXXXXXXXXXX    XXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXX    XXXXXXXXXXXXXXXXXXXXXXXX    XXXXXXXXXXXXXXXXXXXXXXXX


XXXXXXXXXXXXXXXXXXXXXXXX    XXXXXXXXXXXXXXXXXXXXXXXX    XXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXX    XXXXXXXXXXXXXXXXXXXXXXXX    XXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXX    XXXXXXXXXXXXXXXXXXXXXXXX    XXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXX    XXXXXXXXXXXXXXXXXXXXXXXX    XXXXXXXXXXXXXXXXXXXXXXXX


Theresa Gabrielli           Aria Clayton                Martin L. Dix
24 Ridgetop Rd.             314 Bridge St.              4 Shepherd St.
Westboro MA                 Hanover NH                  Norwich VT
               01581                       03755                       05055


Emily C. Slater             Jane Ericson                Ing An
2009 Cherry St.             211 Clancey Court           95 Willow Dr.
York PA                     Chapel Hill NC              Charlotte NC
               17407                       27514                       28211


Becky Jacobson              Jeremy Misiewicz            Hafez Ahmadi
7 Lincoln St.               43-C Lakeview Apts.         5203 Marston Way
Tallahassee FL              Madison WI                  Boulder CO
               32312                       53704                       80302


Ruby Archuleta
Box 108
Milagro NM
               87429
```

Example 2: Printing Two Sets of Mailing Labels

Procedure features:

PROC FORMS statement options:
- ALIGN=
- CC
- FILE=
- NDOWN=
- SETS=

Data set:
LIST on page 492

This example uses page mode and SETS= to produce two sets of mailing labels. Each sheet of labels holds four rows of two labels.

Program

The FILENAME statement associates the name LABELS with the external file that will receive the output from PROC FORMS.

```
filename labels 'external-file';
```

```
options nodate pageno=1 linesize=80 pagesize=60 ;
```

FILE= sends the output to the file associated with the fileref LABELS. NDOWN= prints four rows of form units on each page. CC writes carriage control characters to the file specified by FILE=. WIDTH= sets the width of the form units to 24 to provide enough room for the variables on each line. ACROSS= writes two form units across each page. BETWEEN= puts 20 blank characters between adjacent form units. DOWN= skips two lines at the top of each page so that the form units and the forms align correctly. SKIP= skips three lines between form units to maintain the proper alignment. ALIGN= suppresses the printing of dummy form units. SETS= writes two sets of form units. Each set begins on a new page.

```
proc forms data=list file=labels
           ndown=4
           cc
           width=24
           across=2
           between=20
           down=2
           skip=3
           align=0
           sets=2;
```

The LINE statements specify the variables to place on each line. PACK removes extra blank characters between City and State.

```
   line 1 name;
   line 2 street;
   line 3 city state zip / pack;
run;
```

Output

```


Gabrielli, Theresa                          Clayton, Aria
24 Ridgetop Rd.                             314 Bridge St.
Westboro MA 01581                           Hanover NH 03755



Dix, Martin L.                              Slater, Emily C.
4 Shepherd St.                              2009 Cherry St.
Norwich VT 05055                            York PA 17407



Ericson, Jane                               An, Ing
211 Clancey Court                           95 Willow Dr.
Chapel Hill NC 27514                        Charlotte NC 28211



Jacobson, Becky                             Misiewicz, Jeremy
7 Lincoln St.                               43-C Lakeview Apts.
Tallahassee FL 32312                        Madison WI 53704
```

```
Ahmadi, Hafez                               Archuleta, Ruby
5203 Marston Way                            Box 108
Boulder CO 80302                            Milagro NM 87429
```

```

Gabrielli, Theresa                          Clayton, Aria
24 Ridgetop Rd.                             314 Bridge St.
Westboro MA 01581                           Hanover NH 03755



Dix, Martin L.                              Slater, Emily C.
4 Shepherd St.                              2009 Cherry St.
Norwich VT 05055                            York PA 17407



Ericson, Jane                               An, Ing
211 Clancey Court                           95 Willow Dr.
Chapel Hill NC 27514                        Charlotte NC 28211



Jacobson, Becky                             Misiewicz, Jeremy
7 Lincoln St.                               43-C Lakeview Apts.
Tallahassee FL 32312                        Madison WI 53704
```

```
Ahmadi, Hafez                              Archuleta, Ruby
5203 Marston Way                           Box 108
Boulder CO 80302                           Milagro NM 87429
```

Example 3: Writing Multiple Copies of a Label within a Single Set of Labels

Procedure features:

PROC FORMS statement options:

COPIES=

LINE statement options:

LASTNAME
PACK

Data set: LIST on page 492

This example writes one set of mailing labels that consists of three copies of each form unit. It selects only those observations with addresses in one of the New England states.

Program

The FILENAME statement associates the name LABELS with the external file that will receive the output from PROC FORMS.

```
filename labels 'external-file';
```

```
options pagesize=60 pageno=1 nodate linesize=80;
```

FILE= sends the output to the file associated with the fileref LABELS. NDOWN= prints five rows of form units on each page. CC writes carriage control characters to the file specified by FILE=. ALIGN= suppresses the printing of dummy form units. WIDTH= sets the width of the form units to 24 to provide enough room for the variables on each line. ACROSS= writes three form units across each page. DOWN= skips two lines at the top of each page so that the form units and the forms align correctly. SKIP= skips two lines between form units to maintain the proper alignment. COPIES= writes three copies of each form unit.

```
proc forms data=list file=labels
           ndown=5
           cc
           align=0
           width=24
           across=3
           down=2
```

```
               skip=2
               copies=3;
```

The LINE statements specify the variables to place on each line. LASTNAME removes the comma from Name and writes the first name before the last name. PACK removes extra blank characters between City and State.

```
   line 1 name / lastname;
   line 2 street;
   line 3 city state zip / pack;
```

The WHERE statement selects observations where State is one of the New England states.

```
   where state in('ME', 'NH', 'VT', 'MA', 'CT', 'RI');
run;
```

Output

```


Theresa Gabrielli        Theresa Gabrielli        Theresa Gabrielli
24 Ridgetop Rd.          24 Ridgetop Rd.          24 Ridgetop Rd.
Westboro MA 01581        Westboro MA 01581        Westboro MA 01581


Aria Clayton             Aria Clayton             Aria Clayton
314 Bridge St.           314 Bridge St.           314 Bridge St.
Hanover NH 03755         Hanover NH 03755         Hanover NH 03755


Martin L. Dix            Martin L. Dix            Martin L. Dix
4 Shepherd St.           4 Shepherd St.           4 Shepherd St.
Norwich VT 05055         Norwich VT 05055         Norwich VT 05055
```

The correct bibliographic citation for this manual is as follows: SAS Institute Inc., *SAS® Procedures Guide, Version 8,* Cary, NC: SAS Institute Inc., 1999. 1729 pp.

SAS® Procedures Guide, Version 8

ISBN 1–58025–482–9

SAS Institute Inc., SAS Campus Drive, Cary, North Carolina 27513.

1st printing, October 1999