FLIPPING YOUR LIST:
Using The REVERSE Function To Break Up Text Strings

Lory Jones, Binghamton University

Sometimes the easiest way to do something is to turn things inside out. That is what I found out as part of my project to convert e-mail addresses from CMS® NAMES files to elm® aliases format for use on UNIX systems. The usual entry in a CMS NAMES file includes an e-mail nickname and the address to which it refers. A second type is a list of nicknames; sort of a named distribution list. An example might be:

nick.SWFolks lname.People in Snow White
list.Grumpy Sneezy Doc Bashful Sleepy
Dopey Happy Snowhite Charming

The equivalent in elm aliases format is:

SWFolks = People in Snow White = Grumpy, Sneezy, Doc, Bashful, Sleepy, Dopey, Happy, Snowhite, Charming

A CMS NAMES file can have records up to 255 characters. This means that the list field itself could be much longer than the 80 character limit I had decided to use for ease of network transfer and editing. Furthermore, it meant that once general formatting changes had been made, I would still need to break up each entry into meaningful parts. In many ways this is similar to wrapping text within margins.

In preparation for our task we will declare four string variables, each a bit less than our desired line length, and an index for an array of these variables, called L_Num, which we will initialize to 1.

*--------------------- Declare list vars;
Length
List list2 list3 list4 $ 76;
Retain
List list2 list3 list4 ' ', L_Num 1;
Array
ListPart(4) List list2 list3 list4;

The general approach taken was to check if the length of the list field in the CMS NAMES file is less than our target length. If so, one simply initializes the variable List in the main part of the data step. Otherwise we link to a routine, Makelist, which will put what fits into List, and the rest into the next piece of the ListPart array.

Makelist performs the following steps:

1. Check the index of ListPart (L_Num) and the length of the input field (HowLong) to see if we need to generate another output line.
2. If what is left fits, put it in the next line.
3. If not, find the position of the last comma prior to our desired line length (Spot), put that much text into the next output line, reset our list of names (Field), and its length, to what is left over, and return to the top of the loop.

The actual code from the program is:

*--------------------- MakeList: Break apart long lists;
MakeList: * Use up to four list parts;
Do While ((L_Num <= 4) & (HowLong > 1));
   If (HowLong <= 76) Then Do;
      ListPart(L_Num) = SubStr(Field, 1, HowLong);
      Field = ' '; HowLong = 0;
      End;
   Else Do; * Find last comma before col. 76, which is also the first comma in reversed string;
      Temp = REVERSE(SubStr(Field, 1, 76));
      Spot = index(Temp, ',');
      * Put the reverse of the rest of Temp into the next piece of ListPart;
      ListPart(L_Num) = REVERSE(SubStr(Temp, Spot));
      Len2 = Length(ListPart(L_Num));
      * Set Field to rest of line;
      Field = SubStr(Field, Len2 + 1);
      HowLong = Length(Field); L_Num + 1;
      End;
   End;
End;

*--------------------- If anything is left over, warn user;
If (HowLong > 1) Then Do;
   Put '!! List of names is too long.';
   Put 'Some names omitted.';
   End;
Return;

Step (3 of the Makelist routine is where we find our trick of the day. The approach used is to pull off the next 76 characters of the list, put it in a temporary vari-
able. Reverse it, find the first comma, and return the reverse of the rest of that section of the list! This seems bizarre at first, but actually is good example of how, sometimes, the best solution to a problem is to turn things inside-out.

How does this work? To illustrate with the present example lets assume we wanted to keep lines to 40 characters or less, and that the list of names was:

1 2 3 4
12345678901234567890123456789012345
Grumpy,Sneezy,Doc,Bashful,Sleepy,Dopey,Bappy,Snowhite,Charming

The reverse of the first 40 characters would be:
1 2 3 4
B,yepoD,ypeelS,lufhsaB,coD,yzeenS,ypmrG

The location of the first comma is column 2. Reversing the reversed string from column 2 to the end results in:
Grumpy,Sneezy,Doc,Bashful,Sleepy,Dopey,

which is exactly what we want. The length of this line is 39 characters. Thus, we take position 40 through the end of the original list as the next section to work on. The remainder is only 23 characters, so we simply write that much to the next line

Bappy,Snowhite,Charming

AUTHOR

Lary Jones
Computing Services
Binghamton University
Binghamton, NY 13902-6000
(607) 777-2879
LJones@BingSUNS.CC.Binghamton.EDU

REFERENCES

Jones, Lary 1993 Moving Your Names Without Losing an Alias, SUGI 18 Proceedings

CMS is a registered trademark of International Business Machines, Armonk, NY.
SAS is a register trademark of SAS Institute, Inc., Cary, NC
UNIX is a registered trademark of AT&T Bell Laboratories, Murray Hill, NJ