Published: 10/21/68
(Supersedes: BY.2.04, 11/10/67;
BY.2.04, 04/03/67)

Identification

The equals convention
equalcomp, compcount, chop
R. J. Feiertag, C. Garman

Purpose

This routine provides a shorthand method of representing the latter of a pair of entry names, when many of their components are identical. A utility procedure (with the two names compcount and chop) is provided which can count the number of components in an entry name or place the components in elements of an array or do both.

Definition

The "=" is a character having a special meaning to some file system commands. In a pair of arguments the strings "=" or "==" may appear as components in the second argument with the following special meanings.

"=" - This component is replaced by the corresponding component in the previous argument. If there is no corresponding component (i.e., the second argument has more components than the first) then this component in the second argument is eliminated.

"==" - This string may only appear as the last component in an entry name. If it appears as other than the last component the remainder of the name is ignored. The "==" component is replaced by the corresponding component and all components to its right in the first argument. If the corresponding component does not exist then this component is eliminated.

Usage

```
call equalcomp (name, mask, newname);

dcl (name, mask, newname) char(33);   /*or char(33) var*/
```

name - entry name from which components are to be copied, described as first argument above.

mask - entry name containing "=" and "==" as components, described as second argument above.

newname - entry name created by replacing "=" and "==" by corresponding components.

Examples

name	mask	newname
fred.george	alfred.isaac	alfred.isaac
fred.george	alfred.=	alfred.george
fred	alfred.=	alfred
fred.george	=.=	fred.george
fred.george	=	fred
fred.george.abe	==	fred.george.abe
fred.george.abe	alfred.==	alfred.george.abe
fred.george.abe	alfred.==.isaac	alfred.george.abe

Note that in the last case "isaac" was ignored because it appeared after "==".

Implementation

A check is first made to see if there are any "=" or "==" components in mask. If there aren't any, newname is returned equal to mask. The components in name and mask are then put into arrays by calls to countcomp$both. A test is then made on each component. If a component of the array of mask is an "=" then it is replaced by the corresponding component in the array of name. If the component in the array of mask is a "==" then the remaining components in the array of mask are replaced by the remaining components in the array of name. The array of mask is then concatenated to form newname.

Usage

```
call compcount (name, count);

dcl  name char(*),                  /*entry name whose com-
                                      ponents are to be
                                      counted*/

     count fixed bin(17);           /*count returned by
                                      component*/
```

The number of components in name is returned as count. The number of components is the number of "."'s plus one.

```
call chop (name, array);

dcl  array (*) char (*) varying;  /*array into which
                                    components are to
                                    be placed*/
```

Each component in name is placed in an element of array in sequence, i.e., the ith component of name in the ith element of array. The length of array must be larger than the number of components in name. Any extra elements in array will remain unchanged.

```
call compcount$both (name, array, count);

     or

call chop$both (name, array, count);
```

Both count and array are filled as described above.

Implementation

name is converted to a fixed string by a call to cv_string$cs (BY.10.03). If appropriate, the string up to the first "." is placed in array by a call to substr and a counter, initially zero, is incremented by one. The operation is then performed on the remainder of the string until the string is exhausted.