```
CH.PRP    EPL     05/27/    2214.7

char_to_print_pos: proc(string, n, cvec, ivec, n_pos);

/* This procedure maps the canonical-form character-string in STRING
   into its representation in a pair of concurrent vectors,
   CVEC and IVEC, one element per "print position",
   and returns the count of the number of print-positions of STRING in
   N_POS.

   If an element of CVEC is non-0, its value represents
   the number of characters in the print-position; if an element of CVEC
   is 0 the print position is occupied by a blank or part of an RHT.

   For non-zero elements of CVEC, the corresponding element of IVEC
   indicates the linear subscript within STRING of the first character
   of the print-position. */

dcl process(6) label,
          (n_pos, ci, si, n, type, blanks, bi, cvec(*), ivec(*)) fixed,
          this_chr char (1),
          (get_rel_count, type_func) external entry(char(1)) fixed,
          string char (*) var;

/* This will hold until SUBSTR becomes a built-in function */

dcl substr external entry(char(*), fixed, fixed) char(131071) var;

/* This will have to do until INITIAL comes along */

          process(1) = p1;
          process(2) = p2;
          process(3) = p3;
          process(4) = p4;
          process(5) = p5;
          process(6) = p6;

          ci = 1;
          cvec(1) = 0;

first:    do si = 1 to n;

/* NOTE: the value of SI is changed in sections P2 and P3 below,
   to obtain/skip-over the count-characters of RHT and RVT. */

            this_chr = substr(string, si, 1);
            type = type_func(this_chr);

            go to process(type);

/* control continues below, after internal function which initializes
   successive elements of CVEC and IVEC */

check_ivec: proc;

          if cvec(ci) = 0 then
chk:      do;
            ivec(ci) = si;
            cvec(ci+1) = 0;
          end chk;
end check_ivec;
```

```
/* code to process various types of characters, reached from transfer list
   PROCESS */

p1:         /* character is backspace */

            ci = ci - 1;
            cvec(ci) = cvec(ci) + 1;

            go to end_first;

p2:         /* character is RHT */

            si = si + 1;
            blanks = get_rel_count(substr(string, si, 1));

 proc2a:    do bi = 1 to blanks;
              ci = ci + 1;
              cvec(ci) = ƀ;
            end proc2a;

            go to end_first;

p3:         /* character is RVT, skip next character for count */

            call check_ivec;
            si = si + 1;
            cvec(ci) = cvec(ci) + 2;

            go to end_first;

p4:         /* character is blank */

            cvec(ci+1) = ƀ;

            go to inc_ci;

p5:         /* character is non-spacing control character */

            call check_ivec;
            cvec(ci) = cvec(ci) + 1;

            go to end_first;

p6:         /* character is graphic */

            call check_ivec;
            cvec(ci) = cvec(ci) + 1;

 inc_ci:    ci = ci + 1;
```

```
/* control goes eventually to end_first while in loop,
   either from inc_ci or by direct transfer */

end_first:

        end first;

/* Return number of print-positions to calling procedure */

        n_pos = ci - 1;

end char_to_print_pos;
```

```
PRP.CH    EPL    05/27/    2226.2

print_pos_to_char: proc(string, cvec, ivec, n_pos, out_string, n_out);

/* This procedure maps the print-position representation generated by the
   procedure CHAR_TO_PRINT_POS
   back to canonical form, using the contents of CVEC, IVEC,
   and STRING to form a new canonical-form character-string, OUT_STRING,
   of length N_OUT. */

dcl (i, n_pos, n_out, j,cvec(*), ivec(*)) fixed,
        (string, out_string) char (*) varying,
        one_char char (1),
        spec_char$rht_char char (1) external,
        put_rel_count external entry(fixed) char(1);

/* This will hold until SUBSTR becomes a built-in function */

dcl substr external entry(char(*), fixed, fixed) char(131071) var;

/* initialize output variables */

        n_out = 0;
        out_string = "" /* null string */ ;

/* main loop to re-pack character string from print-position notation */

outer:    do i = 1 to n_pos;

/* NOTE:  I is incremented within this loop, at the DO-statement
   labeled COUNT */

        if cvec(i) = 0 then

/* one or more blanks to be inserted */

 blanks:    do;
            if cvec(i+1) ¬= 0 | i >= n_pos then

/* 0 followed by ¬0 is simple blank */

 one_bl:     do;
               out_string = out_string || " " ;
               n_out = n_out + 1;
             end one_bl;

            else
```

```
/* here to generate RHT */

  many:       do;

   count:        do j = 2 by 1 while (cvec(i+1) = 0 & i < n_pos);
                   i = i + 1;
                 end count;

                 one_char = put_rel_count(j);
                 out_string = out_string || spec_char¢rht_char || one_char;
                 n_out = n_out + 2;

              end many;

            end blanks;

          else

/* here to concatenate output string with indexed sub-string */

 simple:    do;
              out_string = out_string || substr(string, ivec(i), cvec(i));
              n_out = n_out + cvec(i);
            end simple;

          end outer;

end print_pos_to_char;
```

```
GETREL     EPL     05/24/     135 .2

get_rel_count: proc(in_char) fixed;

/* This procedure is used as a function to obtain the count from
   the character following the relative horizontal- and vertical-tabs
   (RHT and RVT), returning a fixed-point value.

   NOTE:  This procedure uses UNSPEC to do its [dirty] work... */

dcl in_char char(1), i fixed;

          unspec(i) = in_char;

          return (i);

end get_rel_count;
```

PUTREL   EPL   05/24/   1438.1

```
put_rel_count: proc(fixed_in) char(1);

/* This procedure is used as a function to create the
   relative-count character for RHT and RVT,
   given a fixed-point argument.

   NOTE:  This procedure uses UNSPEC to do its [unclean] work... */

dcl fixed_in fixed, temp char(1);

          unspec(temp) = fixed_in;

          return (temp);

end put_rel_count;
```

Printed Repre$^{se}_{\wedge}$ntation

$$A\,B\,\underline{C}\qquad D^{E}_{F}\quad G_{\lrcorner}$$

Character String

?, commercial?

* $$\begin{array}{cccccccccccccccccc} A & B & C & {}^{b}{}_{s} & \_ & {}^{r}{}^{h}{}_{t} & \textcircled{2} & D & {}^{h}{}_{L_r} & E & {}^{b}{}_{s} & {}^{h}{}_{L_f} & {}^{h}{}_{L_f} & F & {}^{s}{}_{p} & {}^{h}{}_{L_r} & G & {}^{n}{}_{L} \\ 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 & 14 & 15 & 16 & 17 & 18 \end{array}$$

Vector Representation:

Count (CVEC): 1 1 3 0 0 1 6 0 2 1

Index (IVEC): 1 2 3 — — 8 9 — 16 18

* Key to Graphic Symbols:

Upper-Case Characters	A	Arbitrary Graphics
Lower-Case Symbols	${}^{r}{}^{h}{}_{t}\ {}^{r}{}^{v}{}_{t}$	Relative Horizontal, Vertical Tab
	${}^{b}{}_{s}$	Backspace
	${}^{h}{}_{L_f}\ {}^{h}{}_{L_r}$	Half-Line-Feed Forward, Reverse
	${}^{n}{}_{L}$	New-Line
Circled Numbers	$\textcircled{2}$	Count Character of ${}^{r}{}^{h}{}_{t}\ {}^{r}{}^{n}{}_{t}$