

```

%_mprintto;
options notes nosource;
proc datasets lib=work nolist memtype=data kill; quit;
%put NOTE:
=====;
%put NOTE: Covance Study Number : 000000106326;
%put NOTE: Client Protocol ID   : ZRHR-PK-05-JP;
%put NOTE: Program Name        : t_device.sas;
%put NOTE: Purpose              : table of device events;
%put NOTE: ;
%put NOTE: Input Data           : ADAM.ADDE ADAM.ADSL;
%put NOTE: Output               : t_15_2_6_7(de);
%put NOTE: Macros Called        : _MPRINTTO;
%put NOTE: ;
%put NOTE: Programmed by        : cvn_jriley;
%put NOTE: Creation Date        : 2014-09-17;
%put NOTE: SAS Version          : 9.3;
%put NOTE: ;
%put NOTE: == Latest Run
=====;
%put NOTE: Run by                : &sysuserid;
%put NOTE: Date/Time             :
%sysfunc(putn(%sysfunc(date()),e8601da.))T%sysfunc(putn(%sysfunc(time()),
e86011z.));
%put NOTE: ;
%put NOTE: == Modification History
=====;
%put NOTE: Date      Initials   No. Reason;
%put NOTE: 24Sep2014   JR        1) Added menthol to title;
%put NOTE: 24Sep2014   JR        2) Removed extra line;
%put NOTE: ;
%put NOTE:
=====;
options notes source source2 nofullstimer validvarname=upcase missing='
';
ods _all_ close;
ods listing;

*=====;
* START OF PROGRAM CODE                                     ;
*=====;

/* Standard - just change the number to match the listing you're working
on. Also change the letters in the*/
/* bracket, eg ccb = current cigarette brands. Make sure to do this at
the top of the code too. */

      %let tflno=T_15_02_06_07(de);

/* Standard - leave this */
%let TFL_Part=%scan(&_SASPROGRAMFILE,-3,%str(/));

/* Standard - leave this */
data _null_;

```

```

    tmp("&TFL_Part";
    if tmp not in ("dev" "qc") then call symput("TFL_Part", "prod");
    call symput('TFLpath', compress("&_SASPROGRAMFILE", ""));
run;

*****;
* read in data ;
*****;

data dumtrts1; /*Use this to output any columns for which N=0*/
    attrib trtsega length =$40.
        trtsega length=8.;

    trtsega=5;
    trtsega='Enrolled not randomized';
    output;
run;

data adsl;
    set adam.adsl;
    where saffl = 'Y' and enrfl = 'Y';
    output;
    trtsega=99;
    trtsega='Overall Safety';
    output;
run;

proc freq data=adsl noprint;
    table trtsega*trtsega/ out =tota(drop=percent);
run;

data tot;
    merge tota(in=a) dumtrts1(in=b);
    by trtsega trtsega;
    if a or b;
    if b and not a then do;
        count = 0;
    end;

run;

data tot2;
    set tot;
    call symput('trt' || compress(put(trtsega,best.)),
compress(count));
run;

%macro test;
%if %sysfunc(exist(adam.adde))=0 %then %do;
data paging;
    page=1; flag=1; ln=1; roworder1=.; roworder2=.; rowtext1='';
    column=''; ord=.; sortord=.; n1=''; p1=''; e1=''; n2=''; p2=''; e2='';

```

```

n3=''; p3=''; e3=''; n4=''; p4=''; e4=''; n5=''; p5=''; e5=''; n99='';
p99=''; e99='';

```

```

output;

```

```

call symput("page", '1');

```

```

attrib n1 label = " n"

```

```

n2 label = " n"

```

```

n3 label = " n"

```

```

n4 label = " n"

```

```

n5 label = " n"

```

```

n99 label = " n"

```

```

p1 label = ' (%) '

```

```

p2 label = ' (%) '

```

```

p3 label = ' (%) '

```

```

p4 label = ' (%) '

```

```

p5 label = ' (%) '

```

```

p99 label = ' (%) '

```

```

e1 label = "Events"

```

```

e2 label = "Events"

```

```

e3 label = "Events"

```

```

e4 label = "Events"

```

```

e5 label = "Events"

```

```

e99 label = "Events";

```

```

run;

```

```

%end;

```

```

%else %if %sysfunc(exist(adam.adde)) %then %do;

```

```

%put "USER WARN" "ING: ADDE exists, update code.";

```

```

data de;

```

```

set adam.adde;

```

```

where anydefl='Y' and saffl = 'Y' and enrfl = 'Y';

```

```

run;

```

```

data device02;

```

```

set de;

```

```

attrib headtext1 length=$200.

```

```

headorder1 length=8.;

```

```

headorder1=trtsega;

```

```

headtext1=trtsega;

```

```

output;

```

```

headorder1=99;

```

```

headtext1='Overall Safety';

```

```

output;

```

```

run;

```

```

proc sort data=device02;

```

```

by headorder1 headtext1;

```

```

run;

```

```

* Create an additional observation with missing VOL value for each table
section;

```

```

* This is used to ensure that all table rows are output, even for rows
with no device events;
data device03;
  set device02;
  by headorder1 headtext1;
  if not missing(aerel) and index(aerel, 'NOT RELATED') then aereln = 2;
  else aereln = 1;
  output;

  if first.headorder1 then do;
    subjid = .;
    dedecod='';
    output;
  end;
run;

/*This will give a list of all terms in ADDE which need to be coded
below*/
proc sort data=device03 out=allterms(keep=dedecod) nodupkey;
  by dedecod;
run;

* Create values for table rows;
data device04;
  set device03;
  length rowtext $200;
  * All device events;
  roworder1 = 1;
  roworder2 = 1;
  rowtext = 'Device events';
  output;
  *Adverse events relationship;
  roworder1 = 2;
  roworder2 = 1;
  rowtext = '$S={foreground=white} . $S={} AE relationship';
  output;
  * Related Adverse events;
  roworder1 = 2;
  roworder2 = 2;
  rowtext = '$S={foreground=white} . $S={} Related';
  if subjid = . or aereln = 1 then output;
  * Unrelated adverse events;
  roworder1 = 2;
  roworder2 = 3;
  rowtext = '$S={foreground=white} . $S={} Not related';
  if subjid = . or aereln = 2 then output;

  * Major device events;
  roworder1 = 3;
  roworder2 = 2;
  rowtext = '$S={foreground=white} . $S={} Major';
  if subjid = . or dese = 'MAJOR' then output;
  *Cigarette heater broken;
  roworder1 = 3;

```

```

roworder2 = 3;
rowtext = '$S={foreground=white} . $S={} CHARGING ISSUE';
if (desev='MAJOR' and dedecod = 'CHARGING ISSUE') then output;
    *Battery cigarette holder not charge;
roworder1 = 3;
roworder2 = 4;
rowtext = '$S={foreground=white} . $S={} DEVICE DIFFICULT TO SETUP OR
PREPARE';
if (desev='MAJOR' and dedecod = 'DEVICE DIFFICULT TO SETUP OR
PREPARE') then output;
    *Stops heating before end of smoking experience;
roworder1 = 3;
roworder2 = 5;
rowtext = '$S={foreground=white} . $S={} DEVICE INOPERABLE';
if (desev='MAJOR' and dedecod = 'DEVICE INOPERABLE') then output;
    *Battery malfunction;
roworder1 = 3;
roworder2 = 6;
ROWTEXT = '$S={foreground=white} . $S={} DEVICE OPERATES DIFFERENTLY
THAN EXPECTED';
if (desev='MAJOR' and dedecod = 'DEVICE OPERATES DIFFERENTLY THAN
EXPECTED') then output;
    *Cigarette can not be inserted into hole where should be heated.
cigarette can not be heated completly.;
roworder1 = 3;
roworder2 = 7;
rowtext = '$S={foreground=white} . $S={} DEVICE STOPS INTERMITTENTLY';
if (desev='MAJOR' and dedecod = 'DEVICE STOPS INTERMITTENTLY') then
output;
    *Does not charge when inserted into the mobil unit;
roworder1 = 3;
roworder2 = 8;
rowtext = '$S={foreground=white} . $S={} OUTPUT ISSUE';
if (desev='MAJOR' and dedecod = 'OUTPUT ISSUE') then output;
* Minor Device events;
roworder1 = 4;
roworder2 = 2;
rowtext = '$S={foreground=white} . $S={} Minor';
if subjid = . or dese = 'MINOR' then output;
    *Cigarette heater broken;
roworder1 = 4;
roworder2 = 3;
rowtext = '$S={foreground=white} . $S={} CHARGING ISSUE';
if (desev='MINOR' and dedecod = 'CHARGING ISSUE') then output;
    *Battery cigarette holder not charge;
roworder1 = 4;
roworder2 = 4;
rowtext = '$S={foreground=white} . $S={} DEVICE DIFFICULT TO SETUP OR
PREPARE';
if (desev='MINOR' and dedecod = 'DEVICE DIFFICULT TO SETUP OR
PREPARE') then output;
    *Stops heating before end of smoking experience;
roworder1 = 4;
roworder2 = 5;

```

```

rowtext = '$S={foreground=white} . $S={} DEVICE INOPERABLE';
if (desev='MINOR' and dedecod = 'DEVICE INOPERABLE') then output;
    *Battery malfunction;
roworder1 = 4;
roworder2 = 6;
rowtext = '$S={foreground=white} . $S={} DEVICE OPERATES DIFFERENTLY
THAN EXPECTED';
if (desev='MINOR' and dedecod = 'DEVICE OPERATES DIFFERENTLY THAN
EXPECTED') then output;
    *Cigarette can not be inserted into hole where should be heated.
cigarette can not be heated completly.;
roworder1 = 4;
roworder2 = 7;
rowtext = '$S={foreground=white} . $S={} DEVICE STOPS INTERMITTENTLY';
if (desev='MINOR' and dedecod = 'DEVICE STOPS INTERMITTENTLY') then
output;
    *Does not charge when inserted into the mobil unit;
roworder1 = 4;
roworder2 = 8;
rowtext = '$S={foreground=white} . $S={} OUTPUT ISSUE';
if (desev='MINOR' and dedecod = 'OUTPUT ISSUE') then output;
run;

/*Check that all terms have been accounted for*/
proc sort data=device04 out=device04_check(keep=subjid dedecod);
    by dedecod;
run;

data termcheck;
    merge allterms(in=a) device04_check(in=b);
    by dedecod;
    if a and not b then put "WA" "RNING: Update code for DEDECODs: "
subjid= dedecod= ;
run;

data adsl1;
    set adsl;
    attrib headtext1 length=$200.
                headorder1 length=8.;

    headorder1=trtseql;
    headtext1=trtseql;
    drop trtseql trtseql;
run;

proc sql;
    create table results01 as
    select headorder1, headtext1, count(distinct usubjid) as treated
    from adsl1
    group by headorder1, headtext1;
quit;

```

```

proc sort data=device04 out=device04_a ;
    by headorder1 headtext1 roworder1 roworder2 rowtext usubjid
dedecod;
run;

proc sql;
    create table results02 as
    select headorder1, headtext1, roworder1, roworder2, rowtext, subjid,
count(dedecod) as events,
        count(distinct subjid) as subjects
    from device04_a
    group by headorder1, headtext1, roworder1, roworder2, rowtext;
quit;

data results03;
    merge results02(in=a) results01(keep=headorder1 headtext1 treated);
    by headorder1 headtext1;
    if a;
run;

data results04;
    set results03;
run;

proc sort data=results04;
    by headorder1 headtext1 roworder1 roworder2 rowtext;
run;

* Create data set with all combinations of row values and column values;
* This creates a data set with an observation for each table cell;
proc sql;
    create table results05 as
    select *
    from (select distinct headorder1, headtext1, roworder1, roworder2,
rowtext from results04);
quit;

* Sort the all combinations data set by section heading order, row order
and column order;
proc sort data=results05;
    by headorder1 headtext1 roworder1 roworder2 rowtext ;
run;

* Merge the results data set with the all combinations data set;
* This effectively adds observations with missing results for table cells
with no results;
* This allows text to be created for these table cells if necessary;
data results06;
    merge results04 results05;
    by headorder1 headtext1 roworder1 roworder2 rowtext ;
run;

* Convert results to text values for the summary table;
data results07;

```

```

set results06;
length text text2 text3 $200. ;
if (events = . and subjects = .) or missing(events) and
missing(subjects) then do;
    events = 0;
    subjects = 0;
end;
percent = 100 * subjects / treated;

/* if missing(roworder1) or roworder1 = 1 then delete;*/

    if roworder1 ne 1 and roworder2 = 1 then do;
        text='';
        text2='';
    end;
    else do;

        /*n value*/
        if missing(subjects) then text='0';
        else text=put(subjects,3.);

        /*% value*/
        if missing(percent) or percent=0 then text3='';
        else if percent=100 then text3='(100 %)';
        else if percent ge 10 then text3='(
'||compress(put(percent,8.1))||'%)';
        else if percent lt 10 then text3='(
'||compress(put(percent,8.1))||'%)';

        /*events value*/
        if missing(events) or events=0 then text2='';
        else text2=compress(put(events,3.));
    end;

    keep headorder1 headtext1 roworder1 roworder2 rowtext text text2
text3;
run;

proc sort data=results07 nodupkey;
    by headorder1 headtext1 roworder1 roworder2 rowtext text text2
text3;
run;

data dumtrts; /*Use this to output any columns for which N=0*/
    attrib headtext1 length =$200.
            rowtext length=$70.
            headorder1 length=8.;

    roworder1=1;
    roworder2=1;
    rowtext='Device events';

    headorder1=1;
    headtext1='THS 2.2 Menthol - mCC';

```



```

        output;
        headorder1=2;
        headtext1='mCC - THS 2.2 Menthol';
        output;
        headorder1=3;
        headtext1='THS 2.2 Menthol - NRT gum';
        output;
        headorder1=4;
        headtext1='NRT gum - THS 2.2 Menthol';
        output;
        headorder1=5;
        headtext1='Enrolled not randomized';
        output;

run;

data results07a;
    merge results07(in=a) dumtrts(in=b);
    by headorder1 headtext1 roworder1 roworder2 rowtext;
    if a or b;
    if b and not a then do;
        text='0';
        text2='';
        text3='';
    end;
run;

proc sort data=results07a;
    by roworder1 roworder2 rowtext;
run;

* Transpose the results;
proc transpose data=results07a out=results08_n prefix=n ;
    by roworder1 roworder2 rowtext ;
    id headorder1;
    idlabel headtext1;
    var text ;
run;

proc transpose data=results07a out=results08_p prefix=p ;
    by roworder1 roworder2 rowtext ;
    id headorder1;
    idlabel headtext1;
    var text3;
run;

proc transpose data=results07a out=results08_e prefix=e ;
    by roworder1 roworder2 rowtext ;
    id headorder1;
    idlabel headtext1;
    var text2 ;
run;

data results08;

```

```

merge results08_n results08_e results08_p;
by roworder1 roworder2;
n99n=input(n99,8.);
e99n=input(e99,8.);
run;

proc sort data=results08(where=(roworder1 in (3 4))) out=sorting;
by roworder1 descending n99n descending e99n;
run;

data sorting2;
set sorting;
by roworder1 descending n99n descending e99n;
sortord+1;

roworder2=99;
run;

data final;
set results08(where=(roworder1 not in (3 4))) sorting2;

array a [2] n2 n4;
do i=1 to 2;
if missing(a[i]) then a[i] ='0';
end;
run;

data labels;
set final;
attrib n1 label = "n"
n2 label = "n"
n3 label = "n"
n4 label = "n"
n5 label = "n"
n99 label = "n"
p1 label = '(%)'
p2 label = '(%)'
p3 label = '(%)'
p4 label = '(%)'
p5 label = '(%)'
p99 label = '(%)'
e1 label = "Events"
e2 label = "Events"
e3 label = "Events"
e4 label = "Events"
e5 label = "Events"
e99 label = "Events"
rowtext label = "Device text"
rowtext1 label = "Device text with format"

length=$400.;

rowtext=left(strip(tranwrd(rowtext,'$S={foreground=white} . $S={}
',''))));

```

```

        attrib wrap length = $400;
        if not index(rowtext, ' AE ') then
wrap=left(strip(uppercase(substr(rowtext,1,1))||lowercase(substr(rowtext,2))))
;
        else wrap = left(trim(rowtext));

if roworder2 ne 1 and rowtext ne 'Minor' and rowtext ne 'Major' and
index(rowtext, ' AE')=0 then do;
    i=32; *this is the max length allowed on a single line - change as
needed;
    if length(wrap)>i then do;
        nwraps = int(length(wrap)/i); *calculate how many lines the text
will wrap over;
        do while(nwraps > 0);
            fin=0;
            j = i*nwraps; *calculate starting point - loop will cycle
backwards from this point looking for a space;
            do while(fin=0 and j gt 1);
                if substr(wrap,j,1)=' ' then do;
                    wrap=substr(wrap,1,j-1) || "$n $S={foreground=white} .
$S={} " || substr(wrap,j+1);
                    fin=1;
                end;
                else j=j-1; *no space found - move back one character;
            end;
            nwraps=nwraps-1; *once this wrap is handled, move up a line
until all are handled (when nwraps = 0);
        end;
        rowtext1='$S={foreground=white} . $S={} ' || left(trim(wrap));
    end;
    else do;
        rowtext1='$S={foreground=white} . $S={} ' ||
left(trim(wrap));
    end;

end;
else do; rowtext1=rowtext; end;

if rowtext1 in('Major' 'Minor' 'AE relationship') then do;
    rowtext1='$S={foreground=white} . $S={} ' || left(trim(rowtext1));
end;
    flag = 1;

    if length(left(strip(e1)))=2 then e1= '$S={foreground=white}.$S={} '
|| left(strip(e1));
    if length(left(strip(e1)))=1 then e1= '$S={foreground=white}
.$S={} ' || left(strip(e1));
    if length(left(strip(e99)))=2 then e99=
'$S={foreground=white}.$S={} ' || left(strip(e99));
    if length(left(strip(e99)))=1 then e99= '$S={foreground=white}
.$S={} ' || left(strip(e99));

    if roworder1 = 1 then ord=1;

```

```

        else if roworder1 = 2 then ord=2;
        else if roworder1 = 3 then ord=3;
        else if roworder1 = 4 then ord=5;
        else put "WA" "RNING: Unexpected roworder: " roworder1= ;

run;

proc sql noprint;

create table table.t_15_02_06_07 as
select rowtext, rowtext1, n1, n2, n3, n4, n5, n99, e1, e2, e3, e4, e5,
e99, p1, p2, p3, p4, p5, p99
from labels
order by ord, roworder1, roworder2, sortord;

quit;

data paging;
    set labels;
        by ord roworder1 roworder2 sortord;
            if (first.roworder1 and ln gt 10) or ln gt 12 then ln=1;
/*Amend to look presentable, and avoid page overflows*/
    else ln+1;
    if ln=1 then page+1;
    call symput("page",compress(put(page,best.)));

run;
%end;
%mend;
%test;

/* Standard - leave this */
options number nodate orientation=landscape papersize=&p_pgsz missing='
';
ods escapechar='$';
%let linetop = \brdrt\brdrs\brdrw30; * needs to be 1.5pt so calculated
in twips (1/20 pt) ;
%let linebot = \brdrb\brdrs\brdrw30;
/* Standard - macro for paging */
%macro outrtf(blankn=, halfblnk=);

%if &halfblnk=N %then %let halfblnk=;
%else %if &halfblnk=Y %then %let halfblnk=\~;

ods path stdlib.tl06326 (read) ;
ods results off;
ods rtf toc_data
file="/cvn/projects/prj/data/000000106326/TFL/&TFL_Part./&tflno..rtf"
style=tl06326 startpage=yes headery=1440 footery=1440 ;
ods noproctitle;
%do i=1 %to &page;

```

```

title ;
footnote;
%let wd=0;
%let noobs=0;

data comp;
    set paging end=eof;
    where page=&i;
    if missing(column) then call symput('noobs', 1);
    /* Amend title as needed */
    _firtitl="Table 15.2.6.7 Summary of THS 2.2 Menthol Device
Events and Malfunctions - Safety Population"; /* 1) JR 24Sep2014 */
    _upcas=(length(_firtitl)-
length(compress(_firtitl,'ABCDEFGHIJKLMNOPQRSTUVWXYZ')))/2;
    len=&blankn.-length("(Page &i of &page)");
    if eof then do;
        call symput('_FSRTITL', trim(left(_firtitl)));
        call symput('_blankn', compress(put(len,best.)));
    end;
    drop _firtitl _upcas len;
run;

```

```

ods listing close;
ods proclabel = ' ';
* most set up in template others below;
* title arial 12pt bold with 12pt paragraph space below;
* all headers to be arial 11pt bold;
* data arial 10pt;
* headers to be central, text values left aligned and numeric centered
around decimal point;
/* Update with your variables as needed */
proc report data = comp headline headskip missing nowd split = '$' %if
&i=1 %then %do; contents=' ' %end; %else %do; contents='' %end;;
    column flag page ORD roworder1 roworder2 sortord rowtext1
("Sequence &linebot" ("THS 2.2 Menthol$- mCC $(N=&trt1) &linebot" n1 p1
e1)
("mCC -$THS 2.2 Menthol$(N=&trt2) &linebot" n2 p2 e2)
("THS 2.2 Menthol$- NRT gum $(N=&trt3) &linebot" n3 p3 e3)
("NRT gum -$THS 2.2 Menthol$(N=&trt4) &linebot" n4 p4 e4)
("Enrolled Not$Randomized$(N=&trt5) &linebot" n5 p5 e5))
("Overall$Safety$(N=&trt99) &linebot" n99 p99 e99); ;
    define flag          / order order = internal noprint;
    define page          / order order = internal noprint;
    define ord           / order order = internal noprint;
    define roworder1     / order order = internal noprint;
    define roworder2     / order order = internal noprint;
    define sortord       / order order = internal noprint;
    define rowtext1      / display style={just=left
cellwidth=3.5cm}' ';
    define n1            / display style={just=d cellwidth=0.3cm}
style(header)={just=right} ;

```

```

        define p1          / display style={just=d cellwidth=1.2cm}
style(header)={just=center} ;
        define e1          / display style={just=left
cellwidth=1.2cm} style(header)={just=l} ;
        define n2          / display style={just=d cellwidth=0.3cm}
style(header)={just=right} ;
        define p2          / display style={just=d cellwidth=1.2cm}
style(header)={just=center} ;
        define e2          / display style={just=left
cellwidth=1.2cm } style(header)={just=l} ;
        define n3          / display style={just=d cellwidth=0.3cm}
style(header)={just=right} ;
        define p3          / display style={just=d cellwidth=1.2cm}
style(header)={just=center} ;
        define e3          / display style={just=left
cellwidth=1.2cm} style(header)={just=l} ;
        define n4          / display style={just=d cellwidth=0.3cm}
style(header)={just=right} ;
        define p4          / display style={just=d cellwidth=1.2cm}
style(header)={just=center} ;
        define e4          / display style={just=left
cellwidth=1.2cm} style(header)={just=l} ;
        define n5          / display style={just=d cellwidth=0.3cm}
style(header)={just=right} ;
        define p5          / display style={just=c cellwidth=1.2cm}
style(header)={just=center} ;
        define e5          / display style={just=left
cellwidth=1.2cm} style(header)={just=l} ;
        define n99         / display style={just=d cellwidth=0.5cm}
style(header)={just=right} ;
        define p99         / display style={just=d cellwidth=1.2cm}
style(header)={just=center} ;
        define e99         / display style={JUST=left
cellwidth=1.2cm} style(header)={just=l} ;

```

```

break before flag / page %if &i=1 %then %do;
contents("&_fsrtitl" %end; %else %do; contents='' %end;;

```

```

break after page / page;
%if &NOOBS./==*/ NE 1 %then %do; /* 2) JR 24Sep2014 */
compute after ord;
    line "";
endcomp;
%end;

```

```

compute before page / style={protectspecialchars=off};
    line "&linetop";
endcomp;

```

```

compute after page/style={just=center cellwidth=5cm
protectspecialchars=off};
    %if &NOOBS.=1 %then %do;
        line "No device events were recorded";
    %end;

```

```

        line " ";
    %end;
endcomp;

compute before _page_ / style={just=left protectspecialchars=off};
    line "\b\fs24\sa24&_FSRTITL." ; * \b = bold, \fs24 is font
size 12pt, \sa24 is space after 12pt;

    line "&linebot";
endcomp;

compute after _page_ / style={just=left protectspecialchars=off
pretext="&linetop."};
    line 'Note: mCC = menthol conventional cigarettes; NRT gum =
Nicotine Replacement Therapy gum; THS = Tobacco Heating System';
    line 'Note: Enrolled Not Randomized refers to all subjects
who were enrolled but not randomized. Overall Safety refers to all
subjects exposed to THS 2.2 Menthol or NRT gum.';
    line 'Note: Percentages are based on the number of subjects
indicated in the column header (N).';
    line ' ';
    line 'Appendix 15.3.6.2';
    line "Path: &TFLpath." &_blankn.*"\~\~" "(Page &i of &page)";
;
    line "Program Run: &sysdate   &sysuserid   Program Status:
&status";
endcomp;
run;
%end;
ods rtf close;
ods results on;
ods path sashelp.tmplmst (read);

%mend ;

%outrtf(blankn=70, halfblnk=N);
%macro test2;
%if %sysfunc(exist(adam.adde)) %then %do;
ods listing;
proc printto print = "&table./t_15_02_06_07.lst" new;
run;

proc contents data = table.t_15_02_06_07 varnum;
run;
ods listing close;
%end;
%mend;
%test2;

proc printto ; run;
*=====;
* END OF PROGRAM CODE ;
*=====;

```

