Sas Crackman practice

Contents

[1 first.variable last.variable 4](#_Toc58153752)

[The following example is better: 5](#_Toc58153753)

[Equivalent coding: 6](#_Toc58153754)

[2 Read raw data @ missover dsm dlm 10](#_Toc58153755)

[Infile dsd 10](#_Toc58153756)

[Infile dlm 12](#_Toc58153757)

[Infile missover 12](#_Toc58153758)

[Input @ 13](#_Toc58153759)

[3 Libname dataset excel ‘’ 14](#_Toc58153760)

[4 Function MDY character versus numeric variable 16](#_Toc58153761)

[5 Infile obs 17](#_Toc58153762)

[6 Keep statement and keep option 18](#_Toc58153763)

[7 ODS excel file 19](#_Toc58153764)

[8 first.variable last variable 20](#_Toc58153765)

[9 The value is not in the range proc format value 21](#_Toc58153766)

[10 Character variable being automatically converted to the numeric variable 22](#_Toc58153767)

[11 The output of the proc means 24](#_Toc58153768)

[12 libname dataset ‘Dallas.xls’; 26](#_Toc58153769)

[13 Two ways to define arrays 27](#_Toc58153770)

[14 Data datasetname /debug; 28](#_Toc58153771)

[15. \_error\_ will be automatically dropped 30](#_Toc58153772)

[16 Day() function 30](#_Toc58153773)

[17 If then statement. Two statements after then 31](#_Toc58153774)

[19 WHERE= data set option do not perform any automatic conversion in comparisons 34](#_Toc58153775)

[21 subsetting if 36](#_Toc58153776)

[22 Proc contents work.all; 36](#_Toc58153777)

[23 libname prog2 'C:\Users\dude\Desktop\prog2'; 39](#_Toc58153778)

[24 infile DSD 40](#_Toc58153779)

[25 firstobs 41](#_Toc58153780)

[26 Proc sort 43](#_Toc58153781)

[27 Proc sort by descending 44](#_Toc58153782)

[28 Merge and rename 48](#_Toc58153783)

[29 Trailing @ 51](#_Toc58153784)

[30 Execution error 52](#_Toc58153785)

[31 Missing data at the end of the row 54](#_Toc58153786)

[32 Sum statement 56](#_Toc58153787)

[33 character format is case-sensitive 57](#_Toc58153788)

[34 proc format will generate a file stored in the catalog. 58](#_Toc58153789)

[35 (tough) Write to multiple data sets 58](#_Toc58153790)

[36 Informat mmddyy10. 62](#_Toc58153791)

[38 First.variable 65](#_Toc58153792)

[39 Array defines new character variables 67](#_Toc58153793)

[40 do while vs. do until 68](#_Toc58153794)

[41 CATX function 70](#_Toc58153795)

[42 Find function. ‘I’ and ‘T’ as modifiers and 5 as the starting position. 72](#_Toc58153796)

[43 Put function gives you character; input gives you numbers 78](#_Toc58153797)

[TRANWRD function 79](#_Toc58153798)

[input function 80](#_Toc58153799)

[Put function 82](#_Toc58153800)

[CHAR Function 83](#_Toc58153801)

[Syntax 84](#_Toc58153802)

[Arguments 84](#_Toc58153803)

[44. don’t forget the semicolons 85](#_Toc58153804)

[45. ods csvall file='c:\test.cvs'; 86](#_Toc58153805)

[46 mean(of Rev:) Use colons to represent a list of variables. 86](#_Toc58153806)

[47 Tables variable1\*variable2/nocol norow 88](#_Toc58153807)

[48 Column input 90](#_Toc58153808)

[49 An array cannot be referenced on a keep= data set option 94](#_Toc58153809)

[50 Interleaving data sets with by statement: which variable comes first. 96](#_Toc58153810)

[51 Proc contents data=\_all\_; 99](#_Toc58153811)

[52 Merge. In dataset option 102](#_Toc58153812)

[53 Retain statement initializes the value to missing by default. 103](#_Toc58153813)

[54 retain City 'Beverly Hills'; 107](#_Toc58153814)

[55. Sas date constant: no blank before d. 108](#_Toc58153815)

[56. Tables varible1\*variable2 111](#_Toc58153816)

[57 Sas date function 114](#_Toc58153817)

[58 Proc format; value name (no period) 117](#_Toc58153818)

[59 Delimited raw data file and truncover 118](#_Toc58153819)

[60 Drop option in the data step 121](#_Toc58153820)

[61 if Age not in(15,16) then 122](#_Toc58153821)

[62 ods html file='sales.html' 123](#_Toc58153822)

[63 Do until 124](#_Toc58153823)

[65 Permanently assign labels in the data step 125](#_Toc58153824)

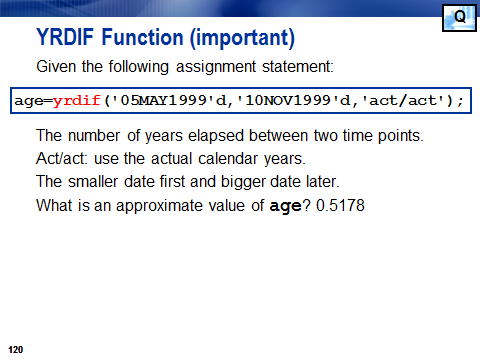
[66 do loop 127](#_Toc58153825)

[67 See a missing value as negative infinity 131](#_Toc58153826)

[68 Character variable vs. numeric variable. 133](#_Toc58153827)

[69 where Code like 'E\_U%'; 133](#_Toc58153828)

[70 Data step drop option 134](#_Toc58153829)



# 1 first.variable last.variable

On 2010/11/20, in 跟crackman做sas base认证试题, by crackman

Basically, this means

1.The following SAS program is submitted:

data WORK.TOTAL;

set WORK.SALARY;

by Department Gender;

if First.<\_insert\_code\_> then Payroll=0;

Payroll+Wagerate;

if Last.<\_insert\_code\_>;

run;

The SAS data set WORK.SALARY is currently ordered by Gender within Department.

Which inserted code will accumulate subtotals for each Gender within Department?

A. Gender

B. Department

C. Gender Department

D. Department Gender

Answer: A

Comment: We know that first.variable is like a dummy variable that identifies the first variable as 1 and others as 0.

## The following example is better:

**data** temp;

input group x;

cards;

1 23

1 34

1 .

1 45

2 78

2 92

2 45

2 89

2 34

2 76

3 31

4 23

4 12

;

**run**;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

The automatic variables first.group and last.group

are not saved with the data set. Here we write them

to data set variables to show their contents.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

**data** new;

set temp;

by group;

first=first.group;

last=last.group;

**run**;

**proc** **print**;

title 'Raw data along with first.group and last.group';

**run**;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

A common task in data cleaning is to identify

observations with a duplicate ID number. If we set

the data set by ID, then the observations which

are not duplicated will be both the first and the

last with that ID number. We can therefore write

any observations which are not both first.id and

last.id to a separate data set and examine them.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

**data** single dup;

set temp;

by group;

if first.group and last.group then output single;

else output dup;

**run**;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

We may also want to do data set processing within

each by group. In this example we construct the

cumulative sum of the variable X within each group.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

**data** cusum(keep=group sum);

set temp;

by group;

if first.group then sum=**0**;

sum+x;

if last.group then output;

**run**;

**proc** **print** data=cusum noobs;

title 'Sum of X within each group';

**run**;

Sum of X within each group 08:30 Wednesday, May 23, 2012 7

group sum

1 102

2 414

3 31

4 35

## Equivalent coding:

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

As an aside, if you simply want the sum of X within

each group, one of the many way of obtaining this

is with PROC PRINT.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

**proc** **print** data=temp;

title 'All data with X summed within each group';

by group;

sum x;

sumby group;

**run**;

**proc** **means** data=temp sum;

by group;

**run**;

All data with X summed within each group 8

08:30 Wednesday, May 23, 2012

-------------------------------------------- group=1 ---------------------------------------------

Obs x

1 23

2 34

3 .

4 45

----- ---

group 102

-------------------------------------------- group=2 ---------------------------------------------

Obs x

5 78

6 92

7 45

8 89

9 34

10 76

----- ---

group 414

-------------------------------------------- group=3 ---------------------------------------------

Obs x

11 31

-------------------------------------------- group=4 ---------------------------------------------

Obs x

12 23

13 12

----- ---

group 35

===

582

All data with X summed within each group 9

08:30 Wednesday, May 23, 2012

-------------------------------------------- group=1 ---------------------------------------------

The MEANS Procedure

Analysis Variable : x

Sum

ƒƒƒƒƒƒƒƒƒƒƒƒ

102.0000000

ƒƒƒƒƒƒƒƒƒƒƒƒ

-------------------------------------------- group=2 ---------------------------------------------

Analysis Variable : x

Sum

ƒƒƒƒƒƒƒƒƒƒƒƒ

414.0000000

ƒƒƒƒƒƒƒƒƒƒƒƒ

-------------------------------------------- group=3 ---------------------------------------------

Analysis Variable : x

Sum

ƒƒƒƒƒƒƒƒƒƒƒƒ

31.0000000

ƒƒƒƒƒƒƒƒƒƒƒƒ

-------------------------------------------- group=4 ---------------------------------------------

Analysis Variable : x

Sum

ƒƒƒƒƒƒƒƒƒƒƒƒ

35.0000000

ƒƒƒƒƒƒƒƒƒƒƒƒ

本题考察的是：first.var和last.var这两个知识点。 本题的意思，数据集SALAR已经按照department 和gender排序，

现在计算每一个department下每一个gender的某一个变量值的累积和。

根据本题的意思，写了一个模拟程序：

data crackman;

input department $ gender $ salary@;

datalines;

market f 6000

market m 5000

market f 5500

market m 8000

market f 6000

market m 7000

sales f 6000

seles m 4000

sales f 6000

seles m 4000

sales f 6000

seles m 4000

;

proc sort data=crackman;

by department gender;

run;

data result;

set crackman;/\*1\*/

by department gender;

if first.gender then subtotal=salary;/\*2\*/

else subtotal+salary;/\*3\*/

if last.gender;/\*4\*/

run;/\*5\*/

这个DATA RESULT 部分程序执行的过程是这样（不是编译过程）：/\*我把执行的语句分为标记1 2 3 4 5\*/

1.1–2–4–1（先读数据—判断first.gender是否为1，第一个肯定是1——所以直接跳过else到if last.gender，last.gender=0所以—set crackman，继续读数据)

2.1–2–3–4–1

3.1–2–3–4–5–1(因为是第三个了，所以last.gender=1 然后到run,输出到result.

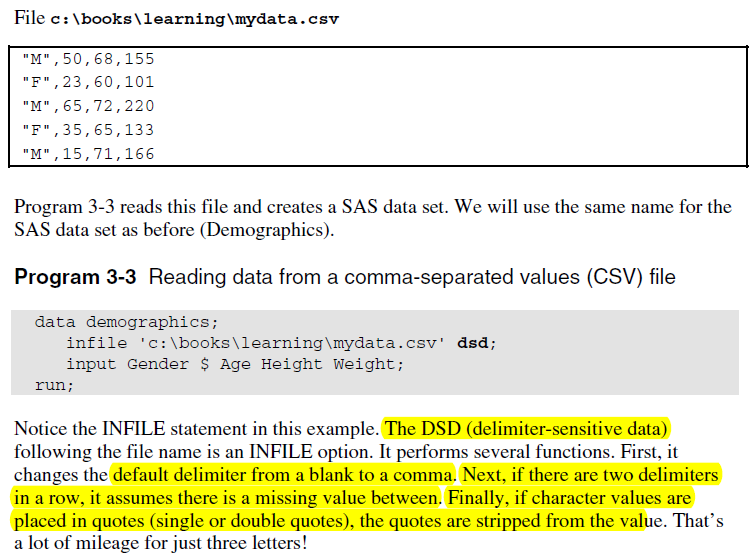
后面都是一样。

这里第三条语句其实===subtotal=subtotal+salary，只是这里简写了，体现程序的简洁性，但是对于入门同学来说增加了可读性的难度。

# 2 Read raw data @ missover dsm dlm

2.Given the following raw data records in TEXTFILE.TXT:  
   
  —-|—-10—|—-20—|—-30  
  John,FEB,13,25,14,27,Final  
  John,MAR,26,17,29,11,23,Current  
  Tina,FEB,15,18,12,13,Final  
  Tina,MAR,29,14,19,27,20,Current  
   
The following output is desired:  
   
  Obs  Name  Month  Status    Week1   Week2   Week3   Week4   Week5  
   
   1   John   FEB   Final       $13     $25     $14     $27       .  
   2   John   MAR   Current     $26     $17     $29     $11     $23  
   3   Tina   FEB   Final       $15     $18     $12     $13       .  
   4   Tina   MAR   Current     $29     $14     $19     $27     $20  
   
Which SAS program correctly produces the desired output?

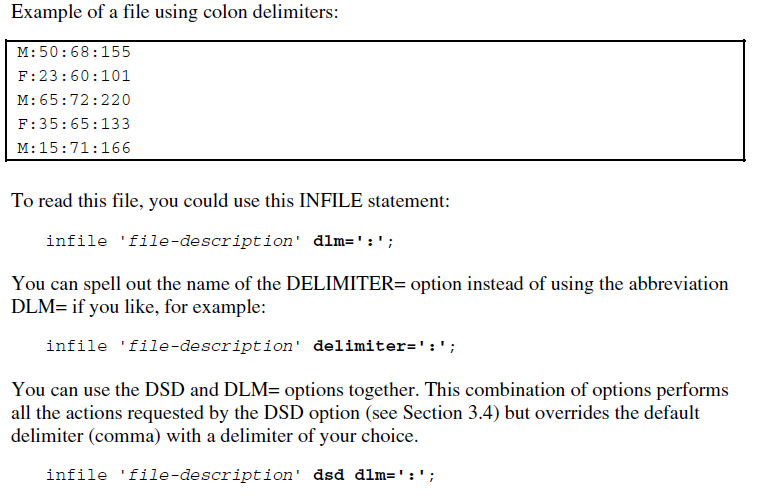
## Infile dsd

        A.  
 data WORK.NUMBERS;  
   length Name $ 4 Month $ 3 Status $ 7;  
   [infile](http://crackman.net/?tag=infile) ‘TEXTFILE.TXT’ dsd;  
   [input](http://crackman.net/?tag=input) Name $ Month $;  
   if Month=’FEB’ then [input](http://crackman.net/?tag=input) Week1 Week2 Week3 Week4 Status $;  
   else if Month=’MAR’ then [input](http://crackman.net/?tag=input) Week1 Week2 Week3 Week4 Week5 Status $;  
   format Week1-Week5 dollar6.;  
run;  
proc print data=WORK.NUMBERS;  
run;  
 A.DSD：规定若一个数据是由引号，那么SAS认为引号内的逗号也是属于字符数据；设定默认分隔符为逗号；连续两个分隔符之间的数据位缺失值；读入数据时去掉引号。  
input语句在读入观测时，先从第一行“John,FEB,13,25,14,27,Final”读取数据“John,FEB”就转入到了第二行“John,MAR,26,17,29,11,23,Current”，那么因为month=”FEB”是true  
所以接着在第二行“John,MAR,26,17,29,11,23,Current”，读入week1 week2，但是john mar都是字符型，而week1 week2默认为数值型，所以就是缺失值。继续读入week3 week4也就是26 17  
接着就是status 是29。读完之后回到infile，回到input,从第三行开始读”Tina,FEB,15,18,12,13,Final”,读入name month之后，转达到第四行“Tina,MAR,29,14,19,27,20,Current”。  
infile dlm

B.   
 data WORK.NUMBERS;  
   length Name $ 4 Month $ 3 Status $ 7;  
   infile ‘TEXTFILE.TXT’ dlm=’,’ missover;  
   [input](http://crackman.net/?tag=input) Name $ Month $;  
   if Month=’FEB’ then [input](http://crackman.net/?tag=input) Week1 Week2 Week3 Week4 Status $;  
   else if Month=’MAR’ then [input](http://crackman.net/?tag=input) Week1 Week2 Week3 Week4 Week5 Status $;  
   format Week1-Week5 dollar6.;  
run;  
proc print data=WORK.NUMBERS;  
run;

B.MISSOVER：阻止INPUT从下一个数据行中读入数据，未赋值的变量就是缺失值。  
其实从A看出，因为第一次INPUT只读入第一数据行两个值，指针转入到第二行了。那么MISSOVER是否可以阻止呢？肯定不能阻止，因为INPUT第一次就只读入两个数据，而且第一行的数据也是大于2个的，所以根本就无需阻止。自己读完两个数据之后自动转向第二行。不能让第一次和第二次的input指针保留在同一行中。

## Infile dlm



## Infile missover

Example 2: Handling Missing Values and Short Records with List Input

This example demonstrates how to prevent missing values from causing problems when you read the data with list input. Some data lines in this example contain fewer than five temperature values. Use the MISSOVER option so that these values are set to missing.

data weather;

infile datalines missover;

input temp1-temp5;

datalines;

97.9 98.1 98.3

98.6 99.2 99.1 98.5 97.5

96.2 97.3 98.3 97.6 96.5

;

SAS reads the three values on the first data line as the values of TEMP1, TEMP2, and TEMP3. The MISSOVER option causes SAS to set the values of TEMP4 and TEMP5 to missing for the first observation because no values for those variables are in the current input data record.

When you omit the MISSOVER option or use FLOWOVER, SAS moves the input pointer to line 2 and reads values for TEMP4 and TEMP5. The next time the DATA step executes, SAS reads a new line which, in this case, is line 3. This message appears in the SAS log:

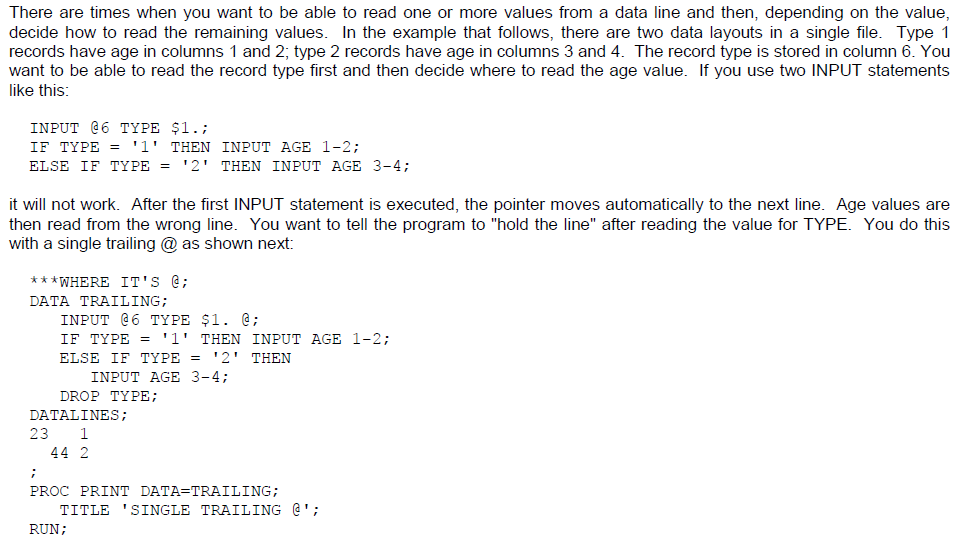
NOTE: SAS went to a new line when INPUT statement

reached past the end of a line.

   
     C.  
 data WORK.NUMBERS;  
   length Name $ 4 Month $ 3 Status $ 7;  
   infile ‘TEXTFILE.TXT’ dlm=’,';  
   [input](http://crackman.net/?tag=input) Name $ Month $ @;  
   if Month=’FEB’ then input Week1 Week2 Week3 Week4 Status $;  
   else if Month=’MAR’ then input Week1 Week2 Week3 Week4 Week5 Status $;  
   format Week1-Week5 dollar6.;  
run;  
proc print data=WORK.NUMBERS;  
run;

Input @

 So in the following example, @ will hold the line and ask SAS not to move to the next line.

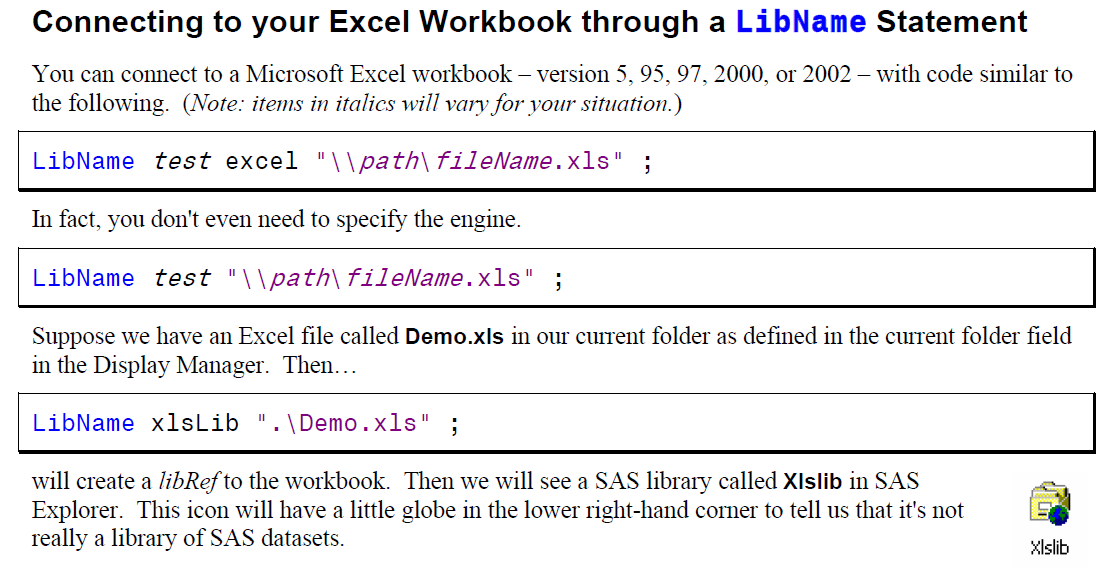
  
     D.  
 data WORK.NUMBERS;  
   length Name $ 4 Month $ 3 Status $ 7;  
   infile ‘TEXTFILE.TXT’ dsd @;  
   input Name $ Month $;  
   if Month=’FEB’ then input Week1 Week2 Week3 Week4 Status $;  
   else if Month=’MAR’ then input Week1 Week2 Week3 Week4 Week5 Status $;  
   format Week1-Week5 dollar6.;  
run;  
proc print data=WORK.NUMBERS;  
run;

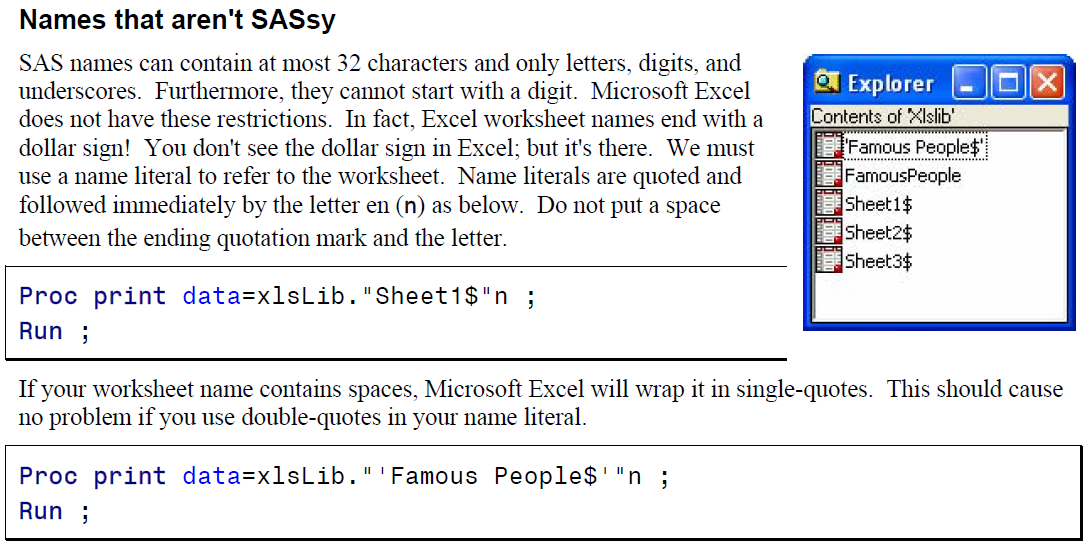
# 3 Libname dataset excel ‘’

3.The Excel workbook REGIONS.XLS contains the following four worksheets:  
     EAST  
     WEST  
     NORTH  
     SOUTH  
   
The following program is submitted:  
   
  libname MYXLS ‘regions.xls’;  
   
Which PROC PRINT step correctly displays the NORTH worksheet?  
     A. proc print data=MYXLS.NORTH;run;  
     B. proc print data=MYXLS.NORTH$;run;  
     C. proc print data=MYXLS.’NORTH’e;run;  
     D. proc print data=MYXLS.’NORTH$’n;run;

D is the correct one.

Basically, the above code establishes the excel libname and then print one of the worksheet. The key is to understand that north$ is the actual excel name, though we cannot see $. However, it is not a legitimate sas name. That’s why we put them in quotation marks, followed by a dollar sign. Dollar sign should also be the in the quotation marks. The quotation marks should be followed immediately by n, without blanks.





# 4 Function MDY character versus numeric variable

4.The following SAS program is submitted:

  data WORK.DATE\_INFO;  
     Day=”01″ ;  
     Yr=1960 ;  
     X=mdy(Day,01,Yr) ;  
  run;  
   
What is the value of the variable X?  
     A. the numeric value 0  
     B. the character value “01011960″  
     C. a missing value due to syntax errors  
     D. the step will not compile because of the character argument in the mdy function.

First of all, notice that the day here actually means the month.

More importantly, we should actually do the coding like this:

**data** WORK.DATE\_INFO;

Day=**01**;

Yr=**1960**;

X=mdy(Day,**01**,Yr) ;

**run**;

In the above code, day variable is properly treated as a numeric variable.

**data** WORK.DATE\_INFO;

Day='01';

Yr=**1960**;

X=mdy(Day,**01**,Yr) ;

**run**;

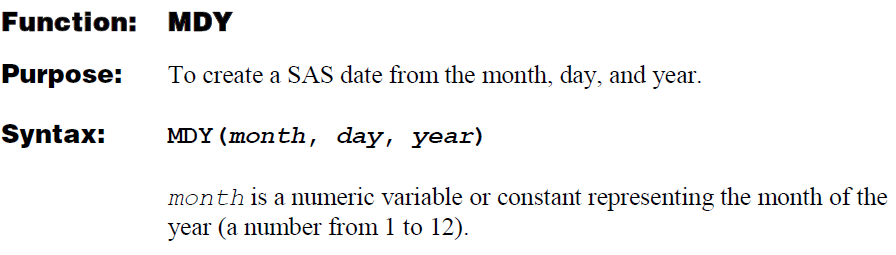
However, the above code still works, because SAS converted it. Check out the log:

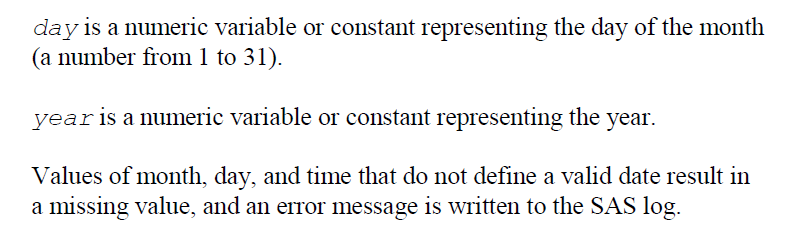
NOTE: Character values have been converted to numeric values at the places given by:

(Line):(Column).

178:12

Answer: A  
这里考察的是日期在SAS中如何储存的。  
SAS日期格式都是以数值型存在的。而且SAS根据1960年1月1号作为基础转化目标日期为数值型数据存储。  
X=MDY(DAY,01，YR)编译之后就是X=MDY(’01′,01,1960)，系统自动将字符型的数字‘01’转化为数字1，所以结果1960年1月1号，所以为0。





# 5 Infile obs

5.Which statement specifies that records 1 through 10 are to be read from the raw data file customer.txt?

     A. [infile](http://crackman.net/?tag=infile) ‘customer.txt’ 1-10;  
     B. [input](http://crackman.net/?tag=input) ‘customer.txt’ [stop@10](mailto:stop@10);  
     C. [infile](http://crackman.net/?tag=infile) ‘customer.txt’ [obs=](http://crackman.net/?tag=obs)10;  
     D. [input](http://crackman.net/?tag=input) ‘customer.txt’ stop=10;

Answer: C  
这道题考察的是INFILE语句后面的参数意义  
前几道题考过DSD MISSOVER DLM等  
此处只有C是对的。

**OBS=**   
Indicates which line in your raw data file should be treated as the last record to be read by SAS. This is a good option to use for testing your program. For example, you might use **obs=100** to just read in the first 100 lines of data while you are testing your program. When you want to read the entire file, you can remove the **obs=** option entirely.

A typical **infile** statement for reading a comma delimited file that contains the variable names in the first line of data would be:

Note that there is on stop option for the infile or input statement.

# 6 Keep statement and keep option

6.After a SAS program is submitted, the following is written to the SAS log:  
   
  101    data WORK.JANUARY;  
  102    set WORK.ALLYEAR([keep](http://crackman.net/?tag=keep)=product month num\_Sold Cost);  
  103    if Month=’Jan’ then output WORK.JANUARY;  
  104    Sales=Cost \* Num\_Sold;  
  105    [keep](http://crackman.net/?tag=keep)=Product Sales;  
                        —–  
                        22  
  ERROR 22-322: Syntax error, expecting one of the following: !,!!, &, \*, \*\*, +, -, , <=, <>, =, >, >=,  
                AND, EQ, GE, GT, IN, LE, LT, MAX, MIN, NE, NG, NL,NOTIN, OR, ^=, |, ||, ~=.  
  106  run;  
   
What changes should be made to the KEEP statement to correct the errors in the LOG?

     A. [keep](http://crackman.net/?tag=keep)=(Product Sales);  
     B. [keep](http://crackman.net/?tag=keep) Product, Sales;  
     C. [keep](http://crackman.net/?tag=keep)=Product, Sales;  
     D. [keep](http://crackman.net/?tag=keep) Product Sales;

Answer: D  
本道题考察的是KEEP语句的语法问题  
在括号内就是KEEP=  
如果不是就是KEEP

The following DATA step creates the same two data sets as the DATA step in the previous example, but it does not read the variable Total into the program data vector. Compare the SET statement here to the one in [Creating More Than One Data Set in a Single DATA Step](http://support.sas.com/documentation/cdl/en/basess/58133/HTML/default/a001291733.htm).

data services (keep=ServicesTotal ServicesPolice ServicesFire

ServicesWater\_Sewer)

admin (keep=AdminTotal AdminLabor AdminSupplies

AdminUtilities);

set city(drop=Total);

run;

proc print data=services;

title 'City Expenditures: Services';

run;

proc print data=admin;

title 'City Expenditures: Administration';

run;

In contrast with previous examples, the data set options in this example appear in both the DATA and SET statements. In the SET statement, the DROP= option determines which variables are omitted from the program data vector. In the DATA statement, the KEEP= option controls which variables are written from the program data vector to each data set being created.

Note:   Using a DROP or KEEP statement is comparable to using a DROP= or KEEP= option in the DATA statement. All variables are included in the program data vector; they are excluded when the observation is written from the program data vector to the new data set. When you create more than one data set in a single DATA step, using the data set options enables you to drop or keep different variables in each of the new data sets. A DROP or KEEP statement, on the other hand, affects all of the data sets that are created.  [cautionend]

# 7 ODS excel file

7.Which of the following choices is an unacceptable ODS destination for producing output that can be viewed in Microsoft Excel?

     A. MSOFFICE2K  
     B. [EXCEL](http://crackman.net/?tag=excel)XP  
     C. [CSV](http://crackman.net/?tag=csv)ALL  
     D. WINXP

Answer: D  
本题考察的是ODS destination。  
答案是D，前三个都可以再[EXCEL](http://crackman.net/?tag=excel)下打开。不过三者之间还是有区别。  
One of the benefits of using this tagset is that it parents from the MSOffice2K tagset, and therefore the output can contain both graphics and tables, unlike the ExcelXP destination, which cannot handle graphics currently. Tagsets.MSOffice2K\_x also adds options to perform many of the common tasks in Excel, much like ExcelXP. Some things that the ExcelXP tagset does very well, such as creating multiple worksheets per workbook, can be done here as well, but not as dynamically.

D is too ridiculous. A better false option would be D excel.

Here are the general ODS statements needed to generate XML output that is compatible with Excel 2002 and later:

ods \_all\_ close;

ods tagsets.ExcelXP file='*file-name.*xml' style=*style-name* ... ;

\* Your SAS procedure code here;

ods tagsets.ExcelXP close;

# 8 first.variable last variable

8.The SAS data set named WORK.SALARY contains 10 observations for each department,and is currently ordered by Department. The following SAS program is submitted:  
   
  data WORK.TOTAL;  
     set WORK.SALARY(keep=Department MonthlyWageRate);  
     by Department;  
     if First.Department=1 then Payroll=0;  
     Payroll+(MonthlyWageRate\*12); (This is the sum statement)  
     if Last.Department=1; (This is the subsetting if. SAS cannot go over this statement unless last.department=1)  
  run;

By sheer coincidence, we just learned this.

By department; will create two variables first.department and last.department.

payroll=0 is the initial value

Which statement is true?  
     A. The by statement in the DATA step causes a syntax error.  
     B. The statement Payroll+(MonthlyWageRate\*12); in the data step causes a syntax error.  
     C. The values of the variable Payroll represent the monthly total for each department in the WORK.SALARY data set.  
     D. The values of the variable Payroll represent a monthly total for all values of WAGERATE in the WORK.SALARY data set.

# 9 The value is not in the range proc format value

9. data course;  
[input](http://crackman.net/?tag=input) exam;  
datalines;  
50.1  
;  
run;

[proc format](http://crackman.net/?tag=proc-format);  
value score 1 – 50 = ‘Fail’  
            51 – 100 = ‘Pass’;  
run;

proc report data =course nowd;  
column exam;  
define exam / display format=score.;  
run;

What is the value for exam?  
A. Fail  
B. Pass  
C. 50.1  
D. No output

Answer: C  
其实这道题考察的是FORMAT的FUZZ  
结果是50.1，因为50.1 不在FORMAT VALUE的区间内。如果改成：  
[proc format](http://crackman.net/?tag=proc-format);  
value score（fuzz=.2） 1 – 50 = ‘Fail’  
            51 – 100 = ‘Pass’;  
run;  
大家看看！  
还有1-50是[1,50]  
51-100是[51,100]

USING PROC FORMAT TO FIND UNEXPECTED

VALUES

User defined formats can be used to list out unexpected

values. If a range of values are not mapped in

a PROC FORMAT, the labels will be the original values.

Here is an example:

**proc format;**

**value looky 370-870 = ‘370-870’**

**;**

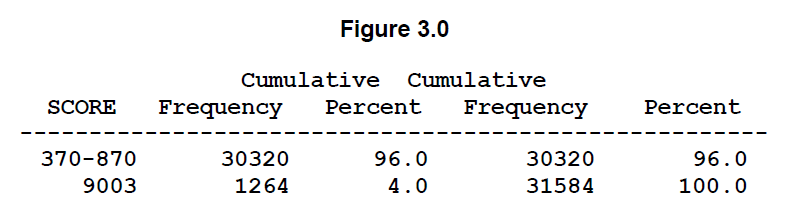
**proc freq data=cb;**

**tables score;**

**format score looky.;**

**run;**

The output of this code is shown in Figure 3.0 below:



Note that in the above output when the score is 9003 it is kept that way.

Ranges can include intervals such as:

<lower> **–** <higher>. means that the interval includes

both endpoints.

<lower> **<-** <higher>. means that the interval

includes higher endpoint, but not the lower

one.

<lower> **- <** <higher> means that the interval

includes lower endpoint, but not the higher

one.

<lower> **<- <** <higher> means that the interval

does not include either endpoint.

# 10 Character variable being automatically converted to the numeric variable

10.The following SAS program is submitted:  
   
 data WORK.RETAIL;  
   Cost=’$20.000′;  
   Discount=.10\*Cost;  
 run;  
   
What is the result?

     A. The value of the variable Discount in the output data set is 2000.No messages are written to the SAS log.  
     B. The value of the variable Discount in the output data set is 2000.A note that conversion has taken place is written to the SAS log.  
     C. The value of the variable Discount in the output data set is missing.A note in the SAS log refers to invalid numeric data.  
     D. The variable Discount in the output data set is set to zero.No messages are written to the SAS log.

**data** WORK.RETAIL;

Cost='20.000';

Discount=**.10**\*Cost;

**run**;

**data** WORK.RETAIL;

Cost='20.000';

Discount=**.10**\*Cost;

Put Discount;

**run**;

I guess crackman uses the put statement for some purpose, but we don’t really need here. As long as it is the standard numeric variable in the quotation marks, SAS will automatically convert it.

256 data WORK.RETAIL;

257 Cost='20.000';

258 Discount=.10\*Cost;

259 run;

NOTE: Character values have been converted to numeric values at the places given by:

(Line):(Column).

258:17

NOTE: The data set WORK.RETAIL has 1 observations and 2 variables.

NOTE: DATA statement used (Total process time):

real time 0.01 seconds

cpu time 0.01 seconds

260 data WORK.RETAIL;

261 Cost='20.000';

262 Discount=.10\*Cost;

263 Put Discount;

264 run;

NOTE: Character values have been converted to numeric values at the places given by:

(Line):(Column).

262:17

2

NOTE: The data set WORK.RETAIL has 1 observations and 2 variables.

NOTE: DATA statement used (Total process time):

real time 0.01 seconds

cpu time 0.00 seconds

# 11 The output of the proc means

11.Given the existing SAS program:  
   
     [proc format](http://crackman.net/?tag=proc-format);  
        value agegrp  
           low-12 =’Pre-Teen’  
           13-high = ‘Teen’;  
     run;  
   
     [proc means](http://crackman.net/?tag=proc-means) data=SASHELP.CLASS;  
        var Height;  
        class Sex Age;  
        format Age agegrp.;  
     run;  
   
Which statement in the [proc means](http://crackman.net/?tag=proc-means) step needs to be modified or added to generate the following results:  
   
                     Analysis Variable : Height

                    N  
   Sex  Age         Obs       Minimum         Maximum            Mean  
   ——————————————————————  
   F    Pre-Teen      3          51.3            59.8            55.8  
   
        Teen          6          56.5            66.5            63.0  
   
   M    Pre-Teen      4          57.3            64.8            59.7  
   
        Teen          6          62.5            72.0            66.8  
   ——————————————————————–  
     A. var Height / nobs min max mean maxdec=1;  
     B. [proc means](http://crackman.net/?tag=proc-means) data=SASHELP.CLASS maxdec=1 ;  
     C. [proc means](http://crackman.net/?tag=proc-means) data=SASHELP.CLASS min max mean maxdec=1;  
     D. output nobs min max mean maxdec=1;

Note that proc means by default does not report obs. So it is absolutely unnecessary to specify noobs.

The Default Descriptive Statistics. According to the following default output, we can observe that the default output includes the standard deviation

The SAS System 1

The MEANS Procedure

Analysis Variable : Integer

N Mean Std Dev Minimum Maximum

----------------------------------------------------------------------------------

10 5.5000000 3.0276504 1.0000000 10.0000000

Note that crackman’s output is a bit messy. N obs is not obs. We simply don’t have obs in the output of proc means.

The MEANS Procedure

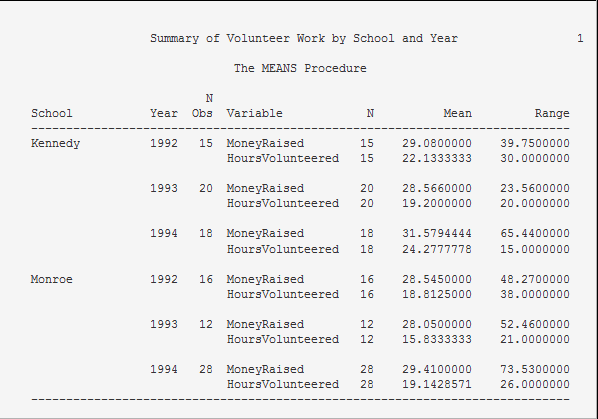
Variable N Mean Std Dev Minimum Maximum

ƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒ

Miles 8 909.5000000 458.2454115 200.0000000 1694.00

totmiles 8 3445.63 1916.92 976.0000000 7007.00

ƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒ



I tried the code like the following:

**proc** **means** data=freqmiles min max mean maxdec=**1**;

class code;

**run**;

The SAS System 08:53 Wednesday, May 30, 2012 45

The MEANS Procedure

N

Code Obs Variable Minimum Maximum Mean

ƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒ

MF 4 Miles 696.0 1694.0 1192.5

totmiles 976.0 7007.0 3179.0

SS 4 Miles 200.0 935.0 626.5

totmiles 2990.0 5478.0 3712.3

ƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒ

Note that if we don’t have the class statement, then it will not be Nobs. Once we have the class statement, it becomes nobs.

**proc** **means** data=freqmiles min max mean maxdec=**1**;

**run**;

The SAS System 08:53 Wednesday, May 30, 2012 46

The MEANS Procedure

Variable Minimum Maximum Mean

ƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒ

Miles 200.0 1694.0 909.5

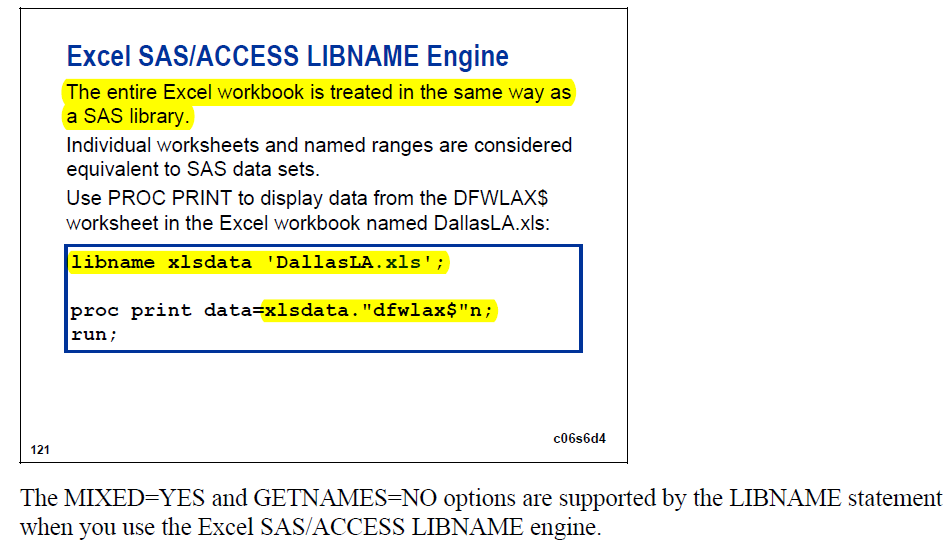
totmiles 976.0 7007.0 3445.6

ƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒƒ

# 12 libname dataset ‘Dallas.xls’;

12.The Excel workbook QTR1.XLS contains the following three worksheets:  
     JAN  
     FEB  
     MAR  
Which statement correctly assigns a library reference to the Excel workbook?  
     A. [libname](http://crackman.net/?tag=libname) qtrdata ‘qtr1.xls’;  
     B. [libname](http://crackman.net/?tag=libname) ‘qtr1.xls’ sheets=3;  
     C. [libname](http://crackman.net/?tag=libname) jan feb mar ‘qtr1.xls’;  
     D. [libname](http://crackman.net/?tag=libname) mydata ‘qtr1.xls’ WORK.heets=(jan,feb,mar);

The correct answer is ~~C~~ A? The reason is shown below.



Another noteworthy point is how to read the a certain worksheet from the excel library. It is

Proc print data=Xlsdata.’dfwlax$’n;

# 13 Two ways to define arrays

13.The following SAS program is submitted:  
   
 data WORK.TEST;  
    set WORK.MEASLES(keep=Janpt Febpt Marpt);  
    [array](http://crackman.net/?tag=array) Diff{3}  Difcount1-Difcount3;  
    [array](http://crackman.net/?tag=array) Patients{3} Janpt Febpt Marpt;  
 run;

   
What new variables are created?  
     A. Difcount1, Difcount2 and Difcount3  
     B. Diff1, Diff2 and Diff3  
     C. Janpt, Febpt, and Marpt  
     D. Patients1, Patients2 and Patients3

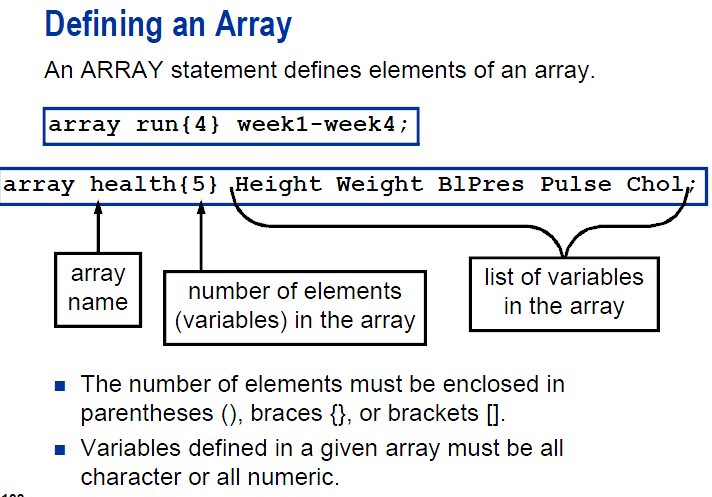
Crackman’s answer is right to the point. Two noteworthy points: there are two ways of defining the array names

**array run{4} week1-week4;**

**array health{5} Height Weight BlPres Pulse Chol;**

Answer: A

在编译过程中，SAS直接在PDV中，加入ARRAY里面定义的变量。  
Janpt Febpt Marpt这三个变量是来自WORK.MEASLES中的。而不是来自新定义的。



# 14 Data datasetname /debug;

14.Which of the following programs correctly invokes the DATA Step Debugger:  
        A.  
 data WORK.TEST [debug](http://crackman.net/?tag=debug);  
   set WORK.PILOTS;  
   State=scan(cityState,2,’ ‘);  
   if State=’NE’ then description=’Central’;  
run;  
   
     B.  
 data  WORK.TEST [debug](http://crackman.net/?tag=debug)ger;     
   set WORK.PILOTS;  
   State=scan(cityState,2,’ ‘);  
   if State=’NE’ then description=’Central’;  
run;  
   
     C.  
 data WORK.TEST / [debug](http://crackman.net/?tag=debug);     
   set WORK.PILOTS;  
   State=scan(cityState,2,’ ‘);  
   if State=’NE’ then description=’Central’;  
run;  
   
     D.  
 data  WORK.TEST / [debug](http://crackman.net/?tag=debug)ger;  
   set WORK.PILOTS;  
   State=scan(cityState,2,’ ‘);  
   if State=’NE’ then description=’Central’;  
run;

Answer: C

其实这里没有多说的，就是语句格式问题。/DEBUG

**INVOKING THE DEBUGGER**

Invoking the Data Step debugger is as simple as

adding a parameter to the DATA statement:

**DATA dsname / DEBUG ;**

When SAS encounters the DEBUG parameter on

the DATA statement, it changes the default

behavior for processing the Data step.

Normally, the user has no control over the execution

of the code that has been compiled. The default

behavior is to execute the entire Data step program

with no external access. Historically, the only

means available to view or modify the execution of

a Data step has been with hard coded PUT

statements or conditional logic. However, with the

DEBUG parameter, SAS first compiles the Data

step and then enters DEBUG mode so the user can

interact with the execution of the Data step. In

DEBUG mode, the user can view and control how

SAS executes the Data step program statements.

# 15. \_error\_ will be automatically dropped

15.Which statement is true concerning the SAS automatic variable \_ERROR\_?

     A. It cannot be used in an if/then condition.  
     B. It cannot be used in an assignment statement.  
     C. It can be put into a keep statement or keep= option.  
     D. It is automatically dropped.

If we want to put the error variable in the keep statement, you should use a regular assignment statement like error=\_error\_

It can be used in an if/then condition.

It will be automatically dropped, because it is a temporary variable.

看看如下程序：  
data crackman;  
[input](http://crackman.net/?tag=input) x y@@;  
if \_error\_=0 then k=1;  
e=\_error\_;  
keep x y k \_error\_ ;  
datalines;  
1 2 3 4 5 6 7 8 9 0  
;  
run;  
proc print;  
run;

这里有争议的是keep语句，keep语句操作应该是除了\_N\_和\_ERROR\_之外的变量。\_N\_和\_ERROR\_是PDV中默认为DROP。不会输出到CRACKMAN数据集中。  
只要是在PDV中的变量，都可以在DATA STEP  IF ELSE语句中进行比较做出判断。

# 16 Day() function

data WORK.DATE\_INFO;  
     X=’04jul2005′d;  
     DayOfMonth=day(x);  
     MonthOfYear=month(x);  
     Year=year(x);  
  run;  
   
What types of variables are DayOfMonth, MonthOfYear, and Year?

     A. DayOfMonth, Year, and MonthOfYear are character.  
     B. DayOfMonth, Year, and MonthOfYear are numeric.  
     C. DayOfMonth and Year are numeric. MonthOfYear is character.  
     D. DayOfMonth, Year, and MonthOfYear are date values.

本题考察的是对时间日期数据格式的把握以及函数DAY() MONTH() YEAR()的应用。  
在SAS系统里，日期型数据都是以数值型数据存在的，X=’04JUL2005′D，其实就是日期型数据，但是如果去掉D就是字符型数据了。  
DAY以及MONTH YEAR函数分别提取日期中天，月，年的具体数据，返回值都是数字。  
看看HELP:  
The DAY function produces an integer from 1 to 31 that represents the day of the month.

# 17 If then statement. Two statements after then

17.Given the following data step:  
   
  data WORK.GEO;  
     [infile](http://crackman.net/?tag=infile) datalines;  
     [input](http://crackman.net/?tag=input) City $20.;  
     if City=’[Tulsa](http://crackman.net/?tag=tulsa)’ then  
     State=’OK’;  
     Region=’Central’;  
     if City=’Los Angeles’ then  
     State=’CA’;  
     Region=’Western’;  
  datalines;  
  [Tulsa](http://crackman.net/?tag=tulsa)  
  Los Angeles  
  Bangor  
  ;  
  run;  
   
After data step execution, what will data set WORK.GEO contain?  
        A.  
 City           State   Region  
———–    —–   ——-  
[Tulsa](http://crackman.net/?tag=tulsa)          OK      Western  
Los Angeles    CA      Western  
Bangor                 Western  
   
     B.  
 City           State   Region  
———–    —–   ——-  
[Tulsa](http://crackman.net/?tag=tulsa)          OK      Western  
Los Angeles    CA      Western  
Bangor  
   
     C.  
 City           State   Region  
———–    —–   ——-  
[Tulsa](http://crackman.net/?tag=tulsa)          OK      Central  
Los Angeles    CA      Western  
Bangor                 Western  
   
     D.  
 City           State   Region  
———–    —–   ——-  
[Tulsa](http://crackman.net/?tag=tulsa)          OK      Central  
Los            CA      Western  
Bangor

Answer: A

本题考察的时候IF THEN语句的应用。  
原始题目可能存在一些错误。我做了一些更正。关键是大家要掌握这个知识点。下面来解释一下程序如何产生这个结果的过程。  
  data WORK.GEO;  
     infile datalines; /\*1\*/  
     [input](http://crackman.net/?tag=input) City $20.;/\*2\*/  
     if City=’[Tulsa](http://crackman.net/?tag=tulsa)’ then  
     State=’OK’;/\*3\*/  
     Region=’Central’;/\*4\*/  
     if City=’Los Angeles’ then  
     State=’CA’;/\*5\*/  
     Region=’Western’;/\*6\*/  
  datalines;  
  [Tulsa](http://crackman.net/?tag=tulsa)  
  Los Angeles  
  Bangor  
  ;  
  run;  
1.1（读入第一行数据）–2–3（IF判断为真，所以STATE=OK）–4（变量值REGION=CENTRAL）—5（判断为假，所以STATE的值维持OK不变）—6（替换原始的REGION变量值为WESTERN）  
2.1（读入第二行数据）–2–3（IF为假，直接执行第4语句，此时，STATE为缺失值）–4（变量值修改为CENTRAL）—5（判断为真，所以STATE的缺失值改为CA）—6（REGION变量值改为WESTERN）  
3.1（读入第三行数据）–2–3（IF为假，直接执行语句4，此时STATE为缺失值）–4（变量值修改为CENTRAL）—5（判断为真，所以STATE的缺失值改为CA）—6（REGION变量值改为WESTERN）。  
所以无论签名的region怎么变化，到后面的第六条语句总是在修改，随意region一直为一个值，就是western。  
对每一行数据读入并做完处理之后就输出到数据集GEO中。

**data** WORK.GEO;

infile datalines;

input City $20.;

if City='Tulsa' then

State='OK';

Region='Central';

if City='Los Angeles' then

State='CA';

Region='Western';

datalines;

Tulsa

Los Angeles

Bangor

;

**run**;

**proc** **print**;

**run**;

|  |
| --- |
| Current Output from Program |

| **Obs** | **City** | **State** | **Region** |
| --- | --- | --- | --- |
| **1** | Tulsa | OK | Western |
| **2** | Los Angeles | CA | Western |
| **3** | Bangor |  | Western |

The main trick is that the last statement Region='Western'; is in a sense unconditional. So it changes all regions to western.

The original code might be modified as follows:

**data** WORK.GEO;

infile datalines;

input City $20.;

if City='Tulsa' then do;

State='OK';

Region='Central';

end;

if City='Los Angeles' then do;

State='CA';

Region='Western';

end;

datalines;

Tulsa

Los Angeles

Bangor

;

**run**;

**proc** **print**;

**run**;

|  |
| --- |
| Current Output from Program |

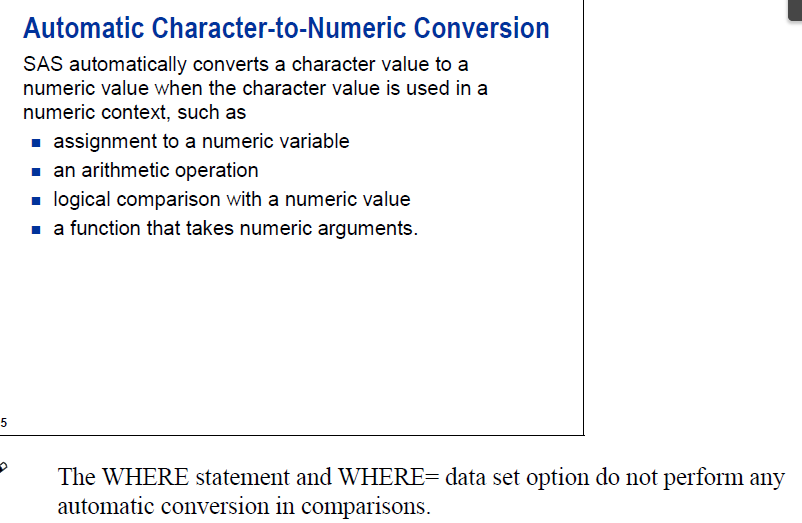
| **Obs** | **City** | **State** | **Region** |
| --- | --- | --- | --- |
| **1** | Tulsa | OK | Central |
| **2** | Los Angeles | CA | Western |
| **3** | Bangor |  |  |

# 19 WHERE= data set option do not perform any automatic conversion in comparisons

19.The SAS data set WORK.ONE contains a numeric variable named Num and a character variable named Char:  
   
  WORK.ONE  
   
  Num   Char  
  —   —-  
    1   23  
    3   23  
    1   77  
   
The following SAS program is submitted:  
   
  proc print data=WORK.ONE;  
     where Num=’1′;  
  run;  
   
What is output?  
        A.  
 Num   Char  
—   —-  
1     23  
   
     B.  
 Num   Char     
—   —-  
1     23  
1     77  
   
     C.  
 Num   Char      
—   —-  
1     23  
3     23  
1     77  
   
     D. No output is generated.

Answer: D  
本题考察的是数值型和字符型数据在表达式中的应用。  
答案是D  
因为WHERE语句无法找到满足条件的观测。  
NUM是数值型变量，表达式右边是字符型数据，肯定无法查询到满足条件的观测。

Crackman is right, but the real reason is as follows. The instructor claims that in the data step, the if statement could work.



# 21 subsetting if

21.Given the SAS data set WORK.PRODUCTS:  
   
  ProdId    Price    ProductType    Sales    Returns  
  ——    —–    ———–    —–    ——-  
  K12S      95.50    OUTDOOR           15          2  
  B132S      2.99    CLOTHING         300         10  
  R18KY2    51.99    EQUIPMENT         25          5  
  3KL8BY     6.39    **OUTDOOR**         125         15  
  DY65DW     5.60    **OUTDOOR**          45          5  
  DGTY23    34.55    EQUIPMENT         67          2  
   
The following SAS program is submitted:  
   
  data WORK.OUTDOOR WORK.CLOTH WORK.EQUIP;  
     set WORK.PRODUCTS;  
     [if](http://crackman.net/?tag=if) Sales GT 30;  
     [if](http://crackman.net/?tag=if) ProductType EQ ‘OUTDOOR’  then output WORK.OUTDOOR;  
     else [if](http://crackman.net/?tag=if) ProductType EQ ‘CLOTHING’ then output WORK.CLOTH;  
     else [if](http://crackman.net/?tag=if) ProductType EQ ‘EQUIPMENT’ then output WORK.EQUIP;  
  run;  
   
How many observations does the WORK.OUTDOOR data set contain?  
     A. 1  
     B. 2  
     C. 3  
     D. 6

B

That is what SAS proudly calls subsetting if. First, choose all the sales that are strictly greater than 30: all the highlighted records. Then among those chosen ones, choose outdoor: the bold ones.

其实这里考察的是IF ELSE语句以及OUTPUT语句的应用.  
在这里一定要记住一个执行过程，就是通过SET每读入一条语句之后，例如第一行：“ K12S      95.50    OUTDOOR           15          2 ”  
IF THEN 语句开始作出判断，IF 为真才执行THEN后面的语句，否则就是ELSE的语句操作。  
具体的流程可以参考签名几个题目的解析。

# 22 Proc contents work.all;

22.Which step displays a listing of all the data sets in the [WORK library](http://crackman.net/?tag=work-library)?  
     A. [proc contents](http://crackman.net/?tag=proc-contents) lib=WORK run;  
     B. [proc contents](http://crackman.net/?tag=proc-contents) lib=WORK.all;run;  
     C. [proc contents](http://crackman.net/?tag=proc-contents) data=WORK.\_all\_; run;  
     D. [proc contents](http://crackman.net/?tag=proc-contents) data=WORK \_ALL\_; run;

The following codes are equivalent. Both will show us the contents of the work library.

**proc** **contents** data=WORK.\_all\_ nods; **run**;

**proc** **contents** data=\_all\_ nods; **run**;

This turns off the descriptive portion.

<libref.>\_ALL\_

gives you information about all SAS data sets that have the type or types specified by the MEMTYPE= option. libref refers to the SAS library. The default for libref is the libref of the procedure input library.

* If you are using the \_ALL\_ keyword, you need read access to all read-protected SAS data sets in the SAS library.
* DATA=\_ALL\_ automatically prints a listing of the SAS files that are contained in the SAS library. Note that for SAS views, all librefs that are associated with the views must be assigned in the current session in order for them to be processed for the listing.

|  |  |  |  |
| --- | --- | --- | --- |
| Default: | | most recently created data set in your job or session, from any SAS library. | |
| Tip: | | If you specify a read-protected data set in the DATA= option but do not give the read password, by default the procedure looks in the PROC DATASETS statement for the read password. However, if you do not specify the DATA= option and the default data set (last one created in the session) is read protected, the procedure does not look in the PROC DATASETS statement for the read password. | |
| Featured in: | | [Describing a SAS Data Set](http://support.sas.com/documentation/cdl/en/proc/61895/HTML/default/a002473456.htm)  Note that the following codes pro | |
| Current Output from Program | |

The CONTENTS Procedure

| **Directory** | |
| --- | --- |
| **Libref** | WORK |
| **Engine** | V9 |
| **Physical Name** |  |
| **Filename** |  |

| **#** | **Name** | **Member Type** | **File Size** | **Last Modified** |
| --- | --- | --- | --- | --- |
| **1** | AGENTS2 | DATA | 13312 | 01Jun12:13:02:15 |
| **2** | FUTURE | DATA | 5120 | 01Jun12:13:25:29 |
| **3** | GEO | DATA | 5120 | 01Jun12:13:33:51 |
| **4** | STUDENTS | DATA | 17408 | 01Jun12:10:16:07 |

|  |
| --- |
| Current Output from Program |

The CONTENTS Procedure

| **Directory** | |
| --- | --- |
| **Libref** | WORK |
| **Engine** | V9 |
| **Physical Name** |  |
| **Filename** |  |

| **#** | **Name** | **Member Type** | **File Size** | **Last Modified** |
| --- | --- | --- | --- | --- |
| **1** | AGENTS2 | DATA | 13312 | 01Jun12:13:02:15 |
| **2** | FUTURE | DATA | 5120 | 01Jun12:13:25:29 |
| **3** | GEO | DATA | 5120 | 01Jun12:13:33:51 |
| **4** | STUDENTS | DATA | 17408 | 01Jun12:10:16:07 |

考察的是[proc contents](http://crackman.net/?tag=proc-contents)过程语句格式。  
Proc contents lists the structure of the specified SAS data set. The information includes the names and types (numeric or character) of the variables in the data set. The most common form of usage is

[proc contents](http://crackman.net/?tag=proc-contents) data=second;  
run;

This lists the information for the data set second. The invocation

[proc contents](http://crackman.net/?tag=proc-contents) data=\_all\_;  
run;

lists the contents of all data sets in the current catalog.

For further information see the [online help](http://crackman.net/?tag=online-help) under [SAS SYSTEM](http://crackman.net/?tag=sas-system) HELP–DATA MANAGEMENT–CONTENTS or [the SAS Procedures Guide](http://crackman.net/?tag=the-sas-procedures-guide).

# 23 libname prog2 'C:\Users\dude\Desktop\prog2';

23.Which is a valid LIBNAME statement?  
     A. [libname](http://crackman.net/?tag=libname) “\_SAS\_data\_library\_location\_”;  
     B. sasdata [libname](http://crackman.net/?tag=libname) “\_SAS\_data\_library\_location\_”;  
     C. libname sasdata “\_SAS\_data\_library\_location\_”;  
     D. libname sasdata sas “\_SAS\_data\_library\_location\_”;

C

For example, I used this one:

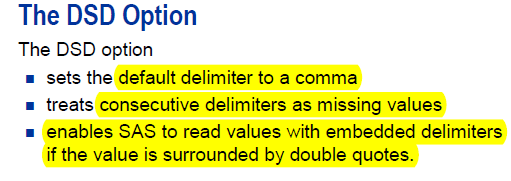
libname prog2 'C:\Users\dude\Desktop\prog2';

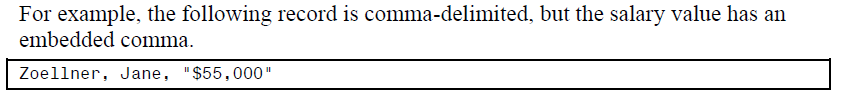
本题考察的是libname语句建立逻辑库。

LIBNAME libref <engine> ‘SAS-data-library’   
LIBNAME libref <engine> (‘SAS-data-library-1′ <,…’SAS-data-library-n’)>;   
语句格式如上，显示Libname，然后是名字，可有可无的<engine>.  
D选项中sas本身就是错误，即使engine里面也没有sas这个参数

# 24 infile DSD

24.Given the following raw data records:  
   
  —-|—-10—|—-20—|—-30  
  Susan\*12/29/1970\*10  
  Michael\*\*6  
   
The following output is desired:  
   
  Obs  employee   bdate  years  
   1   Susan       4015    10  
   2   Michael        .     6  
   
Which SAS program correctly reads in the raw data?  
        A.  
 data employees;  
   [infile](http://crackman.net/?tag=infile) ‘file specification’ dlm=’\*';  
   [input](http://crackman.net/?tag=input) employee $ bdate : mmddyy10. years;  
run;  
   
     B.  
 data employees;  
   [infile](http://crackman.net/?tag=infile) ‘file specification’ [dsd](http://crackman.net/?tag=dsd)=’\*';  
   [input](http://crackman.net/?tag=input) employee $ bdate mmddyy10. years;  
run;  
   
     C.  
 data employees;  
   [infile](http://crackman.net/?tag=infile) ‘file specification’ dlm [dsd](http://crackman.net/?tag=dsd);  
   [input](http://crackman.net/?tag=input) employee $ bdate mmddyy10. years;  
run;  
   
     D.  
 data employees;  
   [infile](http://crackman.net/?tag=infile) ‘file specification’ dlm=’\*’ dsd;  
   input employee $ bdate : mmddyy10. years;  
run;



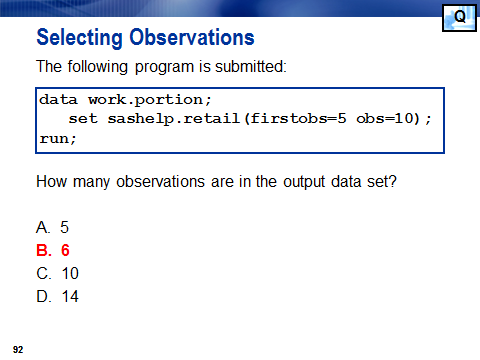


# 25 firstobs

25.Given the following code:  
  
proc print data=SASHELP.CLASS(firstobs=5 obs=15);  
    where Sex='M';  
run;   
  
How many observations will be displayed?

     A. 11  
     B. 15  
     C. 10 or fewer   
     D. 11 or fewer

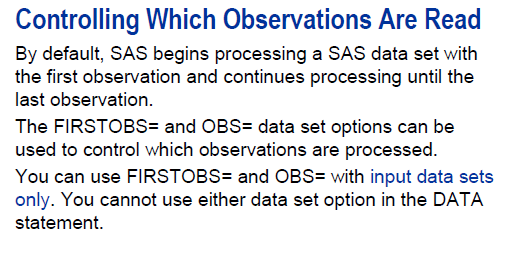
I chose A, because of the following slide.

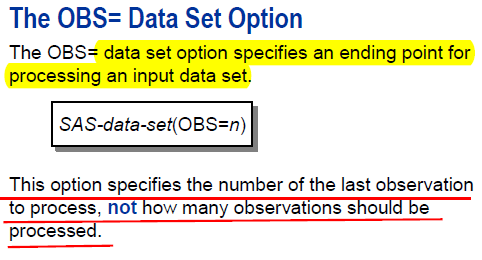


Unfortunately, I neglected the where statement:

where Sex='M';

Due to the above where statement, we should choose D. Among these 11 observations, there might be sex=’F’. Then those observations will not be displayed.





# 26 Proc sort

26.Which step sorts the observations of a permanent SAS data set by two variables and stores the sorted observations in a temporary SAS data set?

A.

proc sort out=EMPLOYEES data=EMPSORT;

by Lname and Fname;

run;

B.

proc sort data=SASUSER.EMPLOYEES out=EMPSORT;

by Lname Fname;

run;

C.

proc sort out=SASUSER.EMPLOYEES data=WORK.EMPSORT;

by Lname Fname;

run;

D.

proc sort data=SASUSER.EMPLOYEES out=SASUSER.EMPSORT;

by Lname and Fname;

run;

本题考察几个：1.临时数据集和永久数据集的差别 2.OUT输出语句和BY语句的语法。  
每一个SAS数据集都有二级目录，第一级是逻辑库，第二级是数据集名称；  
一般来说WORK逻辑库里面的数据集，可以默认不用谢WORK，A就是属于WORK数据集里面的数据集，是临时数据集。  
题目要求的是permanent SAS data set永久数据集。  
C项其实就是语法错误，显示DATA=然后再是OUT= 。  
D是BY语句错误，不需要加and.

# 27 Proc sort by descending

27.Given the SAS data set WORK.TEMPS:   
  
Day Month Temp   
--- ----- ----   
    1 May     75  
   15 May     70  
   15 June    80  
    3 June    76  
    2 July    85  
   14 July    89  
  
The following program is submitted:  
  
proc sort data=WORK.TEMPS;  
     by descending Month Day;  
run;  
  
proc print data=WORK.TEMPS;  
run;  
  
Which output is correct?

        A.  
Obs    Day    Month    Temp  
---    ---    -----    ----  
1       2    July      85   
2      14    July      89   
3       3    June      76   
4      15    June      80   
5       1    May       75   
6      15    May       7  
  
     B.  
Obs    Day    Month    Temp  
---    ---    -----    ----  
1       1    May       75   
2       2    July      85   
3       3    June      76   
4      14    July      89   
5      15    May       70   
6      15    June      80   
  
     C.  
Obs    Day    Month    Temp  
---    ---    -----    ----  
1       1    May       75   
2      15    May       70   
3       3    June      76   
4      15    June      80   
5       2    July      85   
6      14    July      89   
  
     D.  
Obs    Day    Month    Temp  
---    ---    -----    ----  
1      15    May       70   
2       1    May       75   
3      15    June      80   
4       3    June      76   
5      14    July      89   
6       2    July      85   
  
Answer: C  
这里考察的是by语句descending命令的作用范围.  
程序如下：  
data temps;  
input day month $ temp@;  
datalines;  
1 may 75  
15 may 70  
15 june 80  
3 june 76  
2 july 85  
14 july 89  
;  
run;  
proc sort data=WORK.TEMPS;  
     by descending month day;  
run;  
proc print data=WORK.TEMPS;  
run;

descending只是作用与在他之前的变量，也就是只是对于month是这样，而DAY则是默认为ASCENDING升序。

I chose A, because I thought July=7 and may=5. So July should go before may in the descending order. However, here july and may are seen as character variables. So M should be in front of J, in the descending order.

**data** temps;

input day month $ temp@;

datalines;

1 may 75

15 may 70

15 june 80

3 june 76

2 july 85

14 july 89

;

**run**;

**proc** **sort** data=WORK.TEMPS;

by descending month day;

**run**;

**proc** **print** data=WORK.TEMPS;

**run**;

**data** temps;

input day month $ temp@;

datalines;

1 may 75

15 may 70

15 june 80

3 june 76

2 july 85

14 july 89

;

**run**;

**proc** **sort** data=WORK.TEMPS;

by month day;

**run**;

**proc** **print** data=WORK.TEMPS;

**run**;

|  |
| --- |
| The SAS System |

| **Obs** | **day** | **month** | **temp** |
| --- | --- | --- | --- |
| **1** | 1 | may | 75 |
| **2** | 15 | may | 70 |
| **3** | 3 | june | 76 |
| **4** | 15 | june | 80 |
| **5** | 2 | july | 85 |
| **6** | 14 | july | 89 |

|  |
| --- |
| The SAS System |

| **Obs** | **day** | **month** | **temp** |
| --- | --- | --- | --- |
| **1** | 2 | july | 85 |
| **2** | 14 | july | 89 |
| **3** | 3 | june | 76 |
| **4** | 15 | june | 80 |
| **5** | 1 | may | 75 |
| **6** | 15 | may | 70 |

# 28 Merge and rename

28.Given the SAS data set WORK.P2000:

Location    Pop2000  
——–    ——-  
Alaska      626931  
Delaware    783595  
Vermont     608826  
Wyoming     493782

and the SAS data set WORK.P2008:

State       Pop2008  
——–    ——-  
Alaska      686293  
Delaware    873092  
Wyoming     532668

The following output is desired:

Obs     State      Pop2000    Pop2008    Difference  
1     Alaska       626931     686293       59362  
2     Delaware     783595     873092       89497  
3     Wyoming      493782     532668       38886

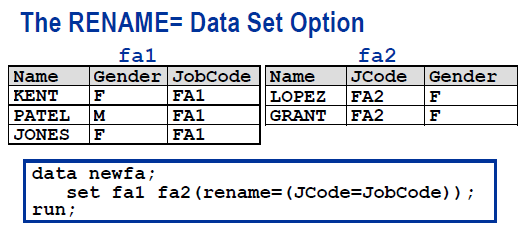
Which SAS program correctly combines the data?  
A.  
data compare;  
merge WORK.P2000(in=\_a Location=State)  
WORK.P2008(in=\_b);  
by State;  
if \_a and \_b;  
Difference=Pop2008-Pop2000;  
run;

B.  
data compare;  
merge WORK.P2000(rename=(Location=State))  
WORK.P2008;  
by State;  
if \_a and \_b;  
Difference=Pop2008-Pop2000;  
run;

C.  
data compare;  
merge WORK.P2000(in=\_a rename=(Location=State))  
WORK.P2008(in=\_b);  
by State;  
if \_a and \_b;  
Difference=Pop2008-Pop2000;  
run;

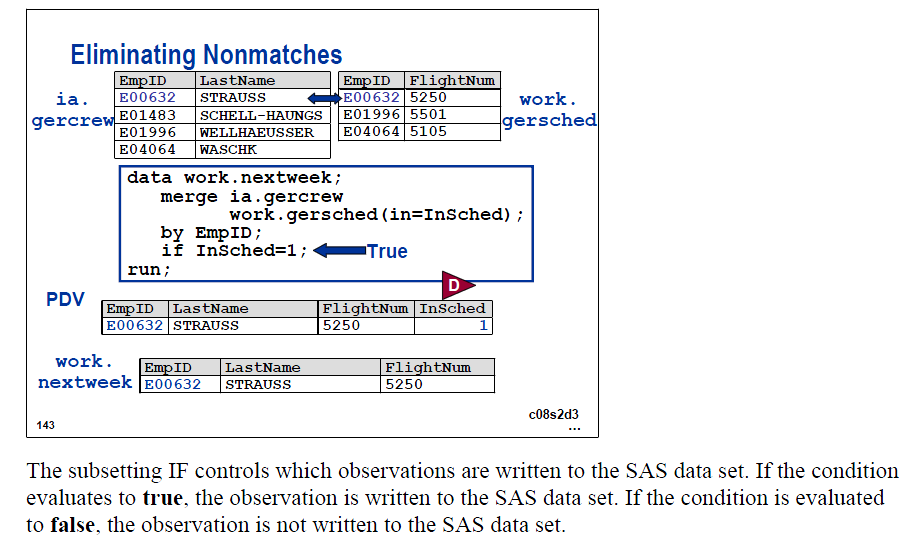
D.  
data compare;  
merge WORK.P2000(in=\_a) (rename=(Location=State))  
WORK.P2008(in=\_b);  
by State;  
if \_a and \_b;  
Difference=Pop2008-Pop2000;  
run;  
Answer: C

Crackman’s remark is perfect.



The old name is on the left.

Also, we must have an equality sign after rename.



这道题考察的是MERGE语句。  
首先是语法格式，MERGE date-set-name<(options)>  
D项就是OPTIONS选项分开放在两个括号内，错误  
对于RENAME命令应该是，rename=（oldname=newname）A出现rename错误  
B未知变量\_a和\_b；  
IN=\_A和IN=\_B，分别表示的是最终数据集中其中观测是否来自数据集WORK.P2000，标记变量为\_a,是否来自数据集WORK.P2008  
，标记变量为\_b。如果\_A 和\_B都为1 表示来自数据集WORK.2000和WORK.P2008，为0表示不是来自该数据集。

下面来解释一下merge过程对数据集的操作并产生新数据集的过程。  
1.在开始执行数据步之前，SAS识别两个数据集的名称创建PDV，PDV包括了这两个数据集中所有的变量。  
WORK.P2000中的state(location)    Pop2000  
work.p2008中的POP2008。  
记住两个数据集中重复的变量PDV中只会出现一次。  
2.接着SAS寻找BY变量，STATE，在两个数据集中的都出现的第一个值，这里是alaska。  
3.接下来SAS从数据集P2000中读入ALASKA同一数据行中其他数据，然后继续读入ALASKA在数据集P2008同一数据行的其他数据。  
读完数据之后输出到新数据集中，在DATA 步中建立的变量，\_A=1 \_B=1，因为BY语句中的STATE都存在两个数据集中。  
这个时候PDV中的变量依然保留中的。  
4.继续读入BY语句中STATE第二个值，Delaware重复3的过程。  
5.继续读入BY语句中STATE第三个值，Vermont。此时在P2008中没有这个值，所以POP2008为缺失值。  
6.IF \_A AND \_B 等价于IF \_A=1 AND \_B=1 ;在SAS里面，1=true,0=FALSE,所以，\_A和\_B就是1 AND 1，也就是TREU AND TRUE 结果就是TRUE。IF这里祈起到删选数据观测的作用。

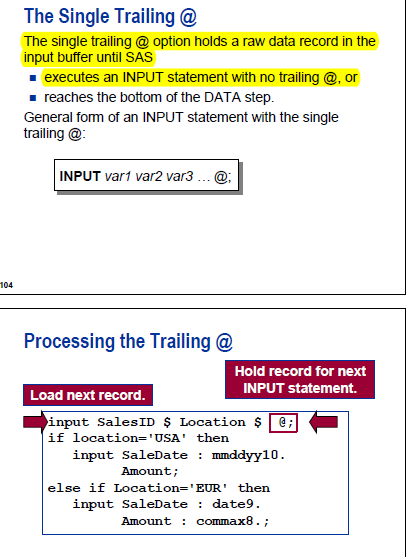
# 29 Trailing @

29.The following SAS program is sumbitted:  
   
  data WORK.INFO;  
     [infile](http://crackman.net/?tag=infile) ‘DATAFILE.TXT’;  
     [input](http://crackman.net/?tag=input) @1 Company $20. @25 State $2. @;  
     if State=’ ‘ then [input](http://crackman.net/?tag=input) @30 Year;  
     else [input](http://crackman.net/?tag=input) @30 City Year;  
     [input](http://crackman.net/?tag=input) NumEmployees;  
  run;  
   
How many raw data records are read during each iteration of the DATA step?

     A. 1  
     B. 2  
     C. 3  
     D. 4

Answer: B

这里考察的是@的使用，具体可以参考文章：  
跟crackman做[sas base认证试题](http://crackman.net/?tag=sas-base%e8%ae%a4%e8%af%81%e8%af%95%e9%a2%98" \t "_blank" \o "查看 sas base认证试题 中的全部文章)（2）  
<http://crackman.net/?p=240>  
INPUT语句中 @1 @25表示是列项，指针移到第1列读取数据，然后移到第25列，继续读数据。

I chose A. Stupid! I did understand what Crackman said, but neglected      [input](http://crackman.net/?tag=input) NumEmployees;  


# 30 Execution error

30.You're attempting to read a raw data file and you see the following messages displayed in the SAS Log:

NOTE: Invalid data for Salary in line 4 15-23.

RULE: ----+----1----+----2----+----3----+----4----+----5--

4 120104 F 46#30 11MAY1954 33

Employee\_Id=120104 employee\_gender=F Salary=. birth\_date=-2061 \_ERROR\_=1 \_N\_=4

NOTE: 20 records were read from the infile 'c:\employees.dat'.

The minimum record length was 33.

The maximum record length was 33.

NOTE: The data set WORK.EMPLOYEES has 20 observations and 4 variables.

What does it mean?

A. A compiler error, triggered by an invalid character for the variable Salary.

B. An execution error, triggered by an invalid character for the variable Salary.

C. The 1st of potentially many errors, this one occurring on the 4th observation.

D. An error on the INPUT statement specification for reading the variable Salary.

I knew the highlighted part is wrong. I did not know what’s execution error, but I should know that it was not caused by input statement.

Answer: B  
这里先给大家解释哈两个名词：  
编译阶段（Compilation Phase）:在这个阶段，系统扫描每个语句检查它是否有语法错误。大部分语法错误导致系统无法对数据步作进一步的处理。在编译阶段将建立要创建的数据集的描述部分。  
语法检查的主要内容：  
        漏掉或错拼的关键词  
        无效的变量名  
        遗漏或错误的符号  
        无效的选择项      
在内存中建立程序数据列PDV  
         用于建立SAS系统的数据集，一次只处理一个观测  
         两个自动变量  
          \_N\_ 记录DATA步执行的次数  
          \_ERROR\_指示出错信息.  0表示无错误,1表示有错误  
建立数据集的描述部分  
       数据集名  
       观测数和变量个数  
       变量名及其属性  
执行阶段（Execution Phase）:若数据步编译成功，就开始执行阶段。在这个阶段对源数据文件的每一条记录斗执行一次数据步，除非在程序中指明其它处理方式。在这个阶段建立数据集的数据部分。

执行顺序  
 PDV中外部为题初始化为缺省值  
 输入每条记录至输入缓冲器,按INPUT语句读至PDV  
 按数据步的其它语句处理后存入PDV  
 在数据步结束时缺省地将PDV的内容作为一条观测  
    写入新的数据集  
 回到数据步的开始.使PDV中外部变量初始化为缺省值  
 对源文件中每条记录都按上述步骤执行一次  
 当对源文件最后一条记录执行结束后,数据步执行完成  
对上述两个名字了解之后，很容理解答案为B。是在读取数据的时候出现的错误。

# 31 Missing data at the end of the row

31. Given the following raw data records in DATAFILE.TXT:

----|----10---|----20---|----30

Kim,Basketball,Golf,Tennis

Bill,Football

Tracy,Soccer,Track

The following program is submitted:

data WORK.SPORTS\_INFO;

length Fname Sport1-Sport3 $ 10;

infile 'DATAFILE.TXT' dlm=',';

input Fname $ Sport1 $ Sport2 $ Sport3 $;

run;

proc print data=WORK.SPORTS\_INFO;

run;

Which output is correct based on the submitted program?

A.

Obs Fname Sport1 Sport2 Sport3

1 Kim Basketball Golf Tennis

2 Bill Football

3 Tracy Soccer Track

B.

Obs Fname Sport1 Sport2 Sport3

1 Kim Basketball Golf Tennis

2 Bill Football Football Football

3 Tracy Soccer Track Track

C.

Obs Fname Sport1 Sport2 Sport3

1 Kim Basketball Golf Tennis

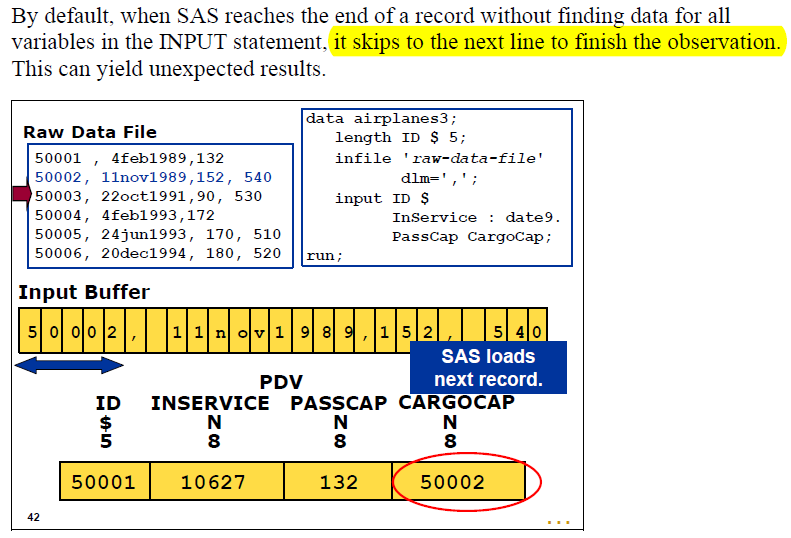
2 Bill Football Tracy Soccer

D.

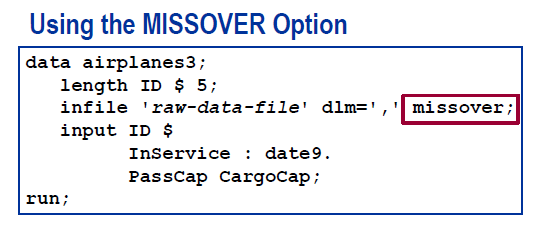
Obs Fname Sport1 Sport2 Sport3

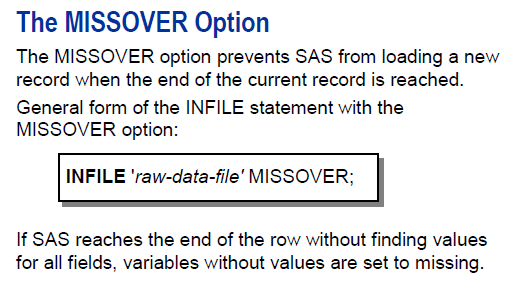
1 Kim Basketball Golf Tennis

2 Bill Football



In order to solve the problem, we should specify the missover option in the infile statement.





Answer: C

**data** WORK.SPORTS\_INFO;

length Fname Sport1-Sport3 $ **10**;

infile 'E:\software textbook\SAS\sas exam\DATAFILE.TXT' dlm=',';

input Fname $ Sport1 $ Sport2 $ Sport3 $;

**run**;

**proc** **print** data=WORK.SPORTS\_INFO;

**run**;

NOTE: 3 records were read from the infile 'E:\software textbook\SAS\sas exam\DATAFILE.TXT'.

The minimum record length was 13.

The maximum record length was 26.

NOTE: SAS went to a new line when INPUT statement reached past the end of a line.

NOTE: The data set WORK.SPORTS\_INFO has 2 observations and 4 variables.

NOTE: DATA statement used (Total process time):

real time 0.10 seconds

cpu time 0.03 seconds

| **Obs** | **Fname** | **Sport1** | **Sport2** | **Sport3** |
| --- | --- | --- | --- | --- |
| **1** | Kim | Basketball | Golf | Tennis |
| **2** | Bill | Football | Tracy | Soccer |

这里考察的还是INPUT语句读入数据时的换行问题。  
INPUT语句在读入数据行时，根据PDV中的变量，依次读入数据，如果该数据行数据小于变量数，换行继续读入数据到PDV中。如果数据行太长，超过了变量数，那么也会自动换到下一个数据行。所以，读取的数据应该是如下过程：  
第一次：  
 Fname    Sport1        Sport2    Sport3  
 Kim      Basketball    Golf      Tennis  
此时换行到第二行了，也就是“Bill,Football”  
第二次：  
  Fname    Sport1        Sport2    Sport3  
  Bill     Football  
此时SPORT2和SPORT3是没有了，就换行到第三行 “Tracy,Soccer,Track ”  
将[Tracy Soccer](http://crackman.net/?tag=tracy-soccer" \t "_blank" \o "查看 Tracy Soccer 中的全部文章) 放在sport2和sport3中。  
等于第二次的时候：PDV中的第二个观测是：  
Fname    Sport1        Sport2    Sport3  
Bill     Football      Tracy     Soccer  
继续换行，但是已经到最后了，所以就只有输出两行观测了。

# 32 Sum statement

32.Consider the following data step:

data WORK.NEW;

set WORK.OLD;

Count+1;

run;

The variable Count is created using a sum statement. Which statement regarding this variable is true?

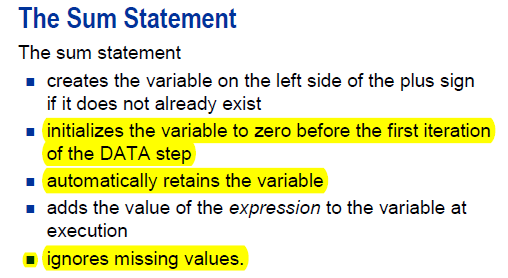
A. It is assigned a value 0 when the data step begins execution.

B. It is assigned a value of missing when the data step begins execution.

C. It is assigned a value 0 at compile time.

D. It is assigned a value of missing at compile time.

I chose the correct answer, without knowing what’s really wrong with A.



这道题考察的是SAS表达式。  
COUNT+1等价于  
RETAIN COUNT 0;  
COUNT=COUNT+1;

I am not sure whether crackman is totally right—what if there are missing data?

# 33 character format is case-sensitive

33.The following SAS program is submitted:

data WORK.TEST;

set WORK.PILOTS;

if Jobcode='Pilot2' then Description='Senior Pilot';

else Description='Unknown';

run;

The value for the variable Jobcode is: PILOT2.What is the value of the variable Description?

A. PILOT2

B. Unknown

C. Senior Pilot

D. ' ' (missing character value)

Answer: B

这道题考察的是IF语句判断字符时注意大小写的问题。

在字符变量中，大小写的Pilot2 和PILOT2是不一样的，所以答案就是B。

# 34 proc format will generate a file stored in the catalog.

34.A user-defined format has been created using the FORMAT procedure.How is it stored?

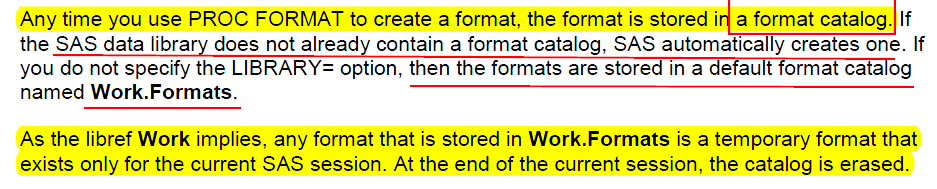
A. in a SAS catalog

B. in a memory resident lookup table

C. in a SAS dataset in the WORK library

D. in a SAS dataset in a permanent SAS data library

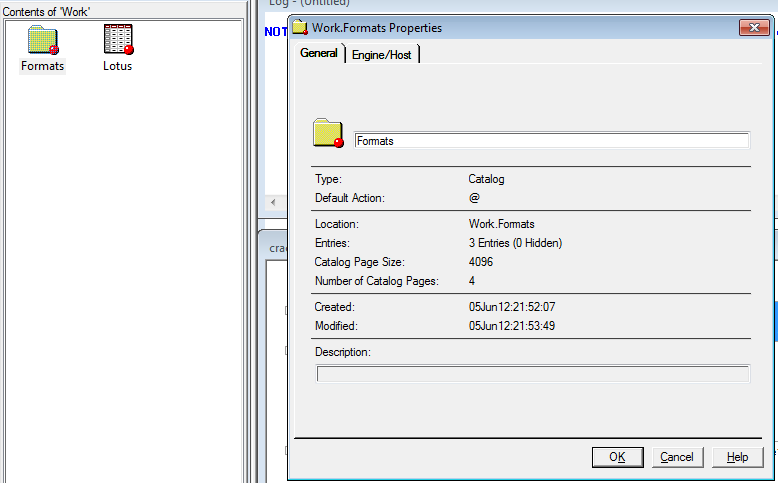
这里考察的是format [macro](http://crackman.net/?tag=macro)这类编译之后在SAS系统存储的形式。  
在SAS系统里面，文件有主要有两种，一个TABLE 一个CATALOG.  
例如：format编译就以[catalog](http://crackman.net/?tag=catalog)形式存在，macro也是。



Note that by default it is in the work library, but not as a dataset. Instead it is a catalog.

I don’t really know the difference between a catalog and a dataset, but I ran the code and got this:

**PROC** **FORMAT**; VALUE wgt **1**='Obese' **2**='Overweight' **3**='Normal';

VALUE dir **1**='Ascending' **2**='Descending'; VALUE dev **1**='stairs' **2**='escalate'; **run**;

------------------------------------------------------------------

# 35 (tough) Write to multiple data sets

35.given the SAS data set SASDATA.TWO:

X Y

-- --

5 2

3 1

5 6

The following SAS program is submitted:

data SASUSER.ONE SASUSER.TWO OTHER;

set SASDATA.TWO;

if X eq 5 then output SASUSER.ONE;

if Y lt 5 then output SASUSER.TWO;

output;

run;

What is the result?

A.

data set SASUSER.ONE has 5 observations

data set SASUSER.TWO has 5 observations

data set WORK.OTHER has 3 observations

B.

data set SASUSER.ONE has 2 observations

data set SASUSER.TWO has 2 observations

data set WORK.OTHER has 1 observations

C.

data set SASUSER.ONE has 2 observations

data set SASUSER.TWO has 2 observations

data set WORK.OTHER has 5 observations

D. No data sets are output. The DATA step fails execution due to syntax errors.

This is probably the most difficult one that I have ever run into so far. I don't even blame myself for getting this one wrong.

Crackman’s answer is certainly correct. His explanation should be also right.

**DATA** two;

INPUT x y;

DATALINES;

5 2

3 1

5 6

;**run**;

**PROC** **PRINT**; **RUN**;

**data** ONE TWO OTHER;

set TWO;

if X eq **5** then output ONE;

if Y lt **5** then output TWO;

output;

**run**;

title'one';

**proc** **print** data=one;**run**;

title'two';

**proc** **print** data=two;**run**;

title 'other';

**proc** **print** data=other;**run**;

|  |
| --- |
| one |

| **Obs** | **x** | **y** |
| --- | --- | --- |
| **1** | 5 | 2 |
| **2** | 5 | 2 |
| **3** | 3 | 1 |
| **4** | 5 | 6 |
| **5** | 5 | 6 |

|  |
| --- |
| two |

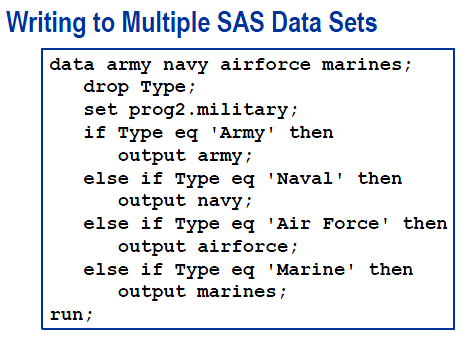
| **Obs** | **x** | **y** |
| --- | --- | --- |
| **1** | 5 | 2 |
| **2** | 5 | 2 |
| **3** | 3 | 1 |
| **4** | 3 | 1 |
| **5** | 5 | 6 |

|  |
| --- |
| other |

| **Obs** | **x** | **y** |
| --- | --- | --- |
| **1** | 5 | 2 |
| **2** | 3 | 1 |
| **3** | 5 | 6 |

这道题是一个非常有意思的题。  
其实考察的就是两个output语句对数据输出的影响。  
下面对数据的执行过程进行解析：  
程序编译之后，建立了PDV。  
当从数据集SASDATA.TWO中读入第一条数据，  
开始IF判断，判断为真就是OUTPUT到SASUSER.ONE数据集中。但是这里没有交代如果为假将会怎么样。我个人认为即使是假，也不会做任何处理。因为没有在假这个条件下的执行语句。  
继续判断，如果Y lt 5 then output SASUSER.TWO;  
记住一个点，也是PDV中，一条观测的在PDV中存在周期问题，如果OUTPUT出去了，是不是PDV中这一条数据就立即消失，全部为默认值了？这里就是 关键所在，我个人觉得这个时候PDV没有中新的SET SASDATA.TWO或者INPUT时候，PDV应该是保留上次的值。那么到了  
OUTPUT语句时，其实默认输出到三个数据集中。  
所以导致5 2 这条观测以及5 6 这条被输出两次到SASUSER.ONE，加上3 1这一条是通过最后一条OUTPUT语句输出的，所以就是5条。  
同样SASUSER.TWO也是5条 OTHER是3条

Note that the above question is much more difficult than the prep textbook:



# 36 Informat mmddyy10.

36.Given the contents of the raw data file 'EMPLOYEE.TXT':

----+----10---+----20---+----30--

Xing 2 19 2004 ACCT

Bob 5 22 2004 MKTG

Jorge 3 14 2004 EDUC

The following SAS program is submitted:

data WORK.EMPLOYEE;

infile 'EMPLOYEE.TXT';

input

@1 FirstName $

@15 StartDate

@25 Department $;

run;

Which SAS informat correctly completes the program?

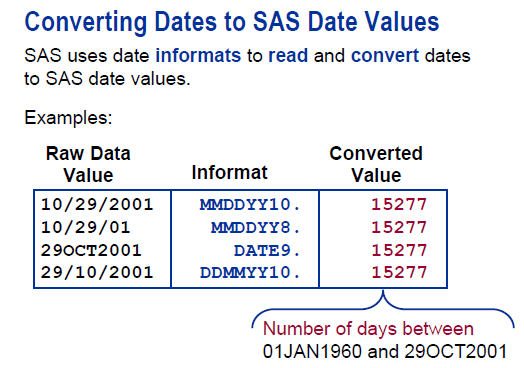
A. date9.

B. mmddyy10.

C. ddmmyy10.

D. mondayyr10.

I got this one wrong because I forgot that the forward slash should be counted. By the way, it is pretty clear that it is mmdd not ddmm.



本题考察的是时间日期数据的格式问题。

完成的程序是：  
data WORK.EMPLOYEE;  
infile ‘EMPLOYEE.TXT’ ;  
format startdate mmddyy10.;/\*谢谢FISH的提示和纠正\*/  
[input](http://crackman.net/?tag=input)  
@1  FirstName $  
@15 StartDate  
@25 Department $;  
run;

MMDDYY10.数据格式：  
08 11 2009  
8  11 2009  
11 12 2009  
08-11-2009  
08/11/2009  
等等这些格式都可以读取。  
@1其实列指针指向第一列，@15就是指向第十五列读取数据。

37.The SAS data set Fed.Banks contains a variable Open\_Date which has

been assigned a permanent label of "Open Date". Which SAS program temporarily

replaces the label "Open Date" with the label "Starting Date" in the output?

A.

proc print data=SASUSER.HOUSES label;

label Open\_Date "Starting Date";

run;

B.

proc print data=SASUSER.HOUSES label;

label Open\_Date="Starting Date";

run;

C.

proc print data=SASUSER.HOUSES;

label Open\_Date="Starting Date";

run;

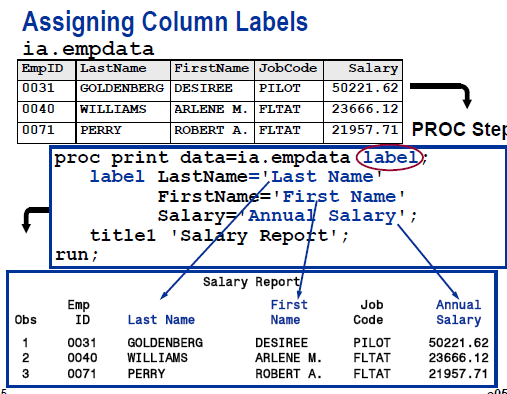
D.

proc print data=SASUSER.HOUSES;

Open\_Date="Starting Date";

run;

这道题考察LABEL语句  
本题的意思数据集SASUSER.HOUSES中变量OPEC\_DATE本身已经有了一个LABEL，现在想在输出时更改此变量标签为“Starting Date”  
这里考察就是LEBEL语句格式，很简单，答案就是B。  
LABEL varname=”label name”;



If you don’t specify the label option, then the output will still use the variable names, instead of the labels.

# 38 First.variable

38.Given the SAS data set WORK.ONE:

X Y Z

- - --

1 A 27

1 A 33

1 B 45

2 A 52

2 B 69

3 B 70

4 A 82

4 C 91

The following SAS program is submitted:

data WORK.TWO;

set WORK.ONE;

by X Y;

if First.Y;

run;

proc print data=WORK.TWO noobs;

run;

Which report is produced?

A.

X Y Z

-- -- --

1 B 45

2 A 52

2 B 69

3 B 70

4 A 82

4 C 91

B.

X Y Z

-- -- --

1 A 27

1 B 45

2 A 52

2 B 69

3 B 70

4 A 82

4 C 91

C.

X Y Z

-- -- --

1 A 33

1 B 45

2 A 52

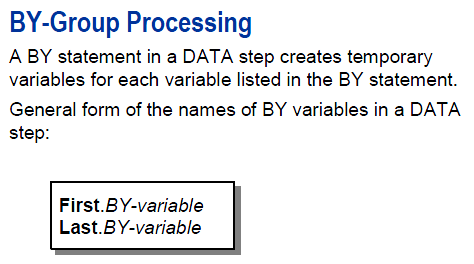
2 B 69

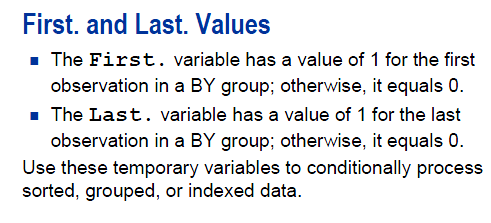
3 B 70

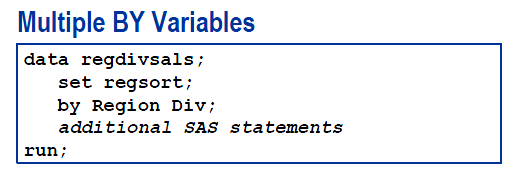
4 A 82

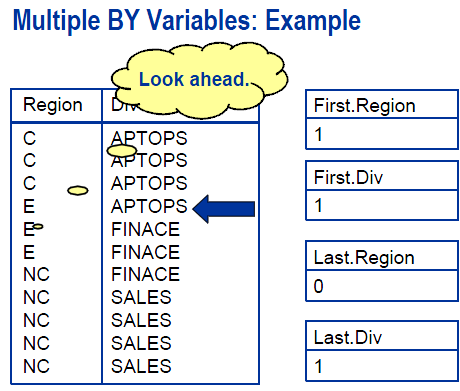
4 C 91

D. The PRINT procedure fails because the data set WORK.TWO is not created in the DATA step.









！未经本文作者同意不得用于商业应用。  
这道题考察了BY语句在SET过程中的作用。  
BY X Y；语句表达的是对X，Y分组  
if first.y 意思是同样在X=1的条件，Y第一次等于A，那么FIRST.Y=1，第二次等于A就是0，第一次等于B就是1，第二次等于B就是0。  
那么接着X=2的条件，第一次出现某个Y值，那么FIRST.Y=1，第二次就是0。一次类推。  
这里的IF FIRST.Y语句筛选出了X Y组合不重复的观测对象。

这里有几个问题，如果我在数据集中添加一条：2 B 79，那么LOG提示会是什么？如何改变这个问

题？如果我只是单纯的分组不排序，BY语句中作如何改变？

# 39 Array defines new character variables

39.The following SAS program is submitted:

data WORK.AUTHORS;

array Favorites{3} $ 8 ('Shakespeare','Hemingway','McCaffrey');

run;

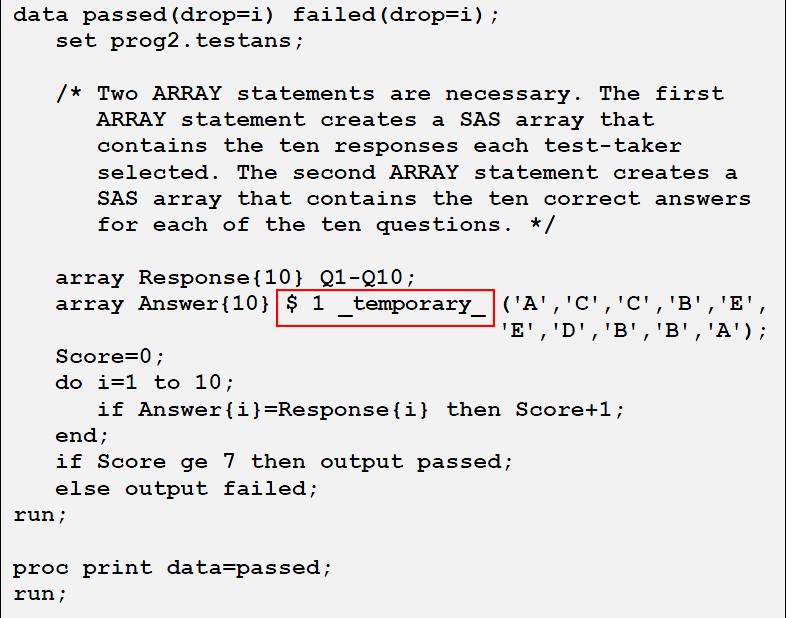
What is the value of the second variable in the dataset WORK.AUTHORS?

A. Hemingway

B. Hemingwa

C. ' ' (a missing value)

D. The program contains errors. No variables are created.

本道题考察的是ARRAY与变量之间的关系。  
在DATA 步中，ARRYA可以创建新变量。  
这里创建了三个变量，变量名分别是  
Shakespeare,Hemingway,McCaffrey  
每一个变量都是都是字符型数据，长度为8。  


Note that in the above code, we uses $1 to specify the length of the character variable. Also, we used the \_temporary\_ option.

# 40 do while vs. do until

40.The following SAS program is submitted:

data WORK.PRODUCTS;

Prod=1;

do while(Prod LE 6);

Prod + 1;

end;

run;

What is the value of the variable Prod in the output data set?

A. 6

B. 7

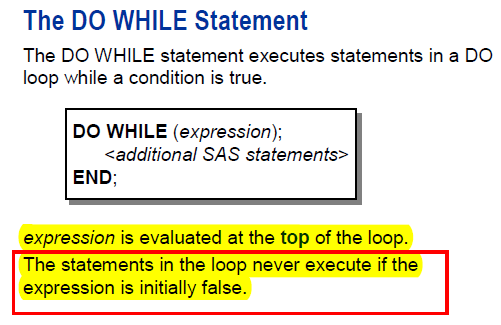
C. 8

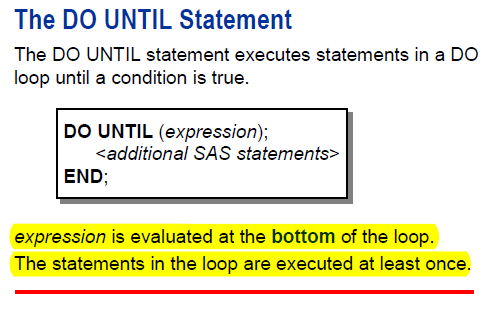
D. . (missing numeric)

这道题考察的是DO WHILE语句中循环执行条件的判断时点。  
到底是先判断再执行还是先执行在判断？  
DO WHILE是先判断然后执行，也就是说PROD=1,先判断 PROD LE 6是否为TRUE，然后决定是否执行循环语句。  
循环每次执行完，PROD自动加1，也就是当PROD LE 6的为TRUE时，PROD最大值就是6，超过6就不执行循环内的语句，就是PROD加1了，但是PROD等于6的时候，依然要执行循环，LE就是小与等于的意思。所以最终PROD=7。

如果改成：  
data WORK.PRODUCTS;  
   Prod=1;  
   [do until](http://crackman.net/?tag=do-until)(Prod LE 6);  
     Prod + 1;  
   end;  
 run;  
那么PROD=2，为什么等于2？  
因为DO UNTIL是执行后再判断，PROD执行一次之后变成2,2 LE 6是true，所以终止执行！。UNTIL里面的表达式是循环终止的判断条件，如果UNTIL里面为真，那么就终止执行DO LOOP；而WHILE的表达式是循环继续的判断条件，为真继续执行DO LOOP。

英文解释：  
The DO UNTIL statement executes statements in a DO loop repetitively until a condition is true, checking the condition after each iteration of the DO loop. The DO WHILE statement evaluates the condition at the top of the loop; the DO UNTIL statement evaluates the condition at the bottom of the loop.





If the initial statement is true, then SAS goes through the data step once.

# 41 CATX function

41.Given the raw data record in the file phone.txt:

----|----10---|----20---|----30---|

Stevens James SALES 304-923-3721 14

The following SAS program is submitted:

data WORK.PHONES;

infile 'phone.txt';

input EmpLName $ EmpFName $ Dept $ Phone $ Extension;

<\_insert\_code\_>

run;

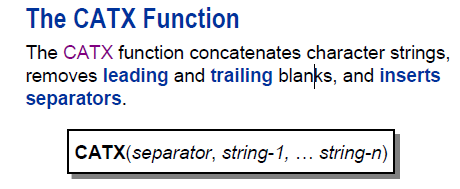
Which SAS statement completes the program and results in a value of "James Stevens" for the variable FullName?

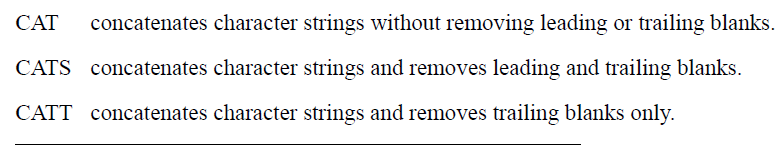
A. FullName=CATX(' ',EmpFName,EmpLName);

B. FullName=CAT(' ',EmpFName,EmpLName);

C. FullName=EmpFName!!EmpLName;

D. FullName=EmpFName + EmpLName;





**The instructor jokes that cat is an old lazy cat who does not do anything, but just put two strings together.**

**CATS is a stripper, taking off all pieces.**

**CATX is our favorite: taking away the leading and trailing blanks, but also adding the separator in between.**

**Note that we seem to have to specify the separator for catx: we cannot assume it is by default a blank.**

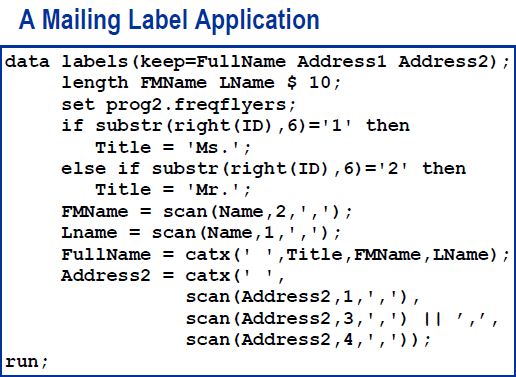
本题考察的是函数[CAT](http://crackman.net/?tag=cat" \t "_blank" \o "查看 CAT 中的全部文章)X的用法。  
先看看SAS 对CATX的解释：  
去除变量前后的空格符，插入分隔符，返回连接后的字符串。  
请思考：是先处理要准备连接的每一个变量的前后空格符，还是连接号之后再除去连接后的变量的前后空格符？连接后的变量默认长度是好多呢？这个函数与CAT函数的差别是什么呢？  
Removes leading and trailing blanks, inserts delimiters, and returns a concatenated character string.

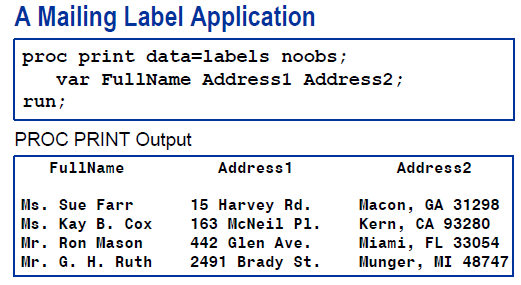
——————————————————————————–  
Syntax  
CATX(delimiter, item-1 <, …item-n>)

Arguments

delimiter  
specifies a character string that is used as a delimiter between concatenated items.

item  
specifies a constant, variable, or expression, either character or numeric. If item is numeric, then its value is converted to a character string by using the BESTw. format. In this case, SAS does not write a note to the log. For more information, see The Basics.





# 42 Find function. ‘I’ and ‘T’ as modifiers and 5 as the starting position.

42.The following SAS program is submitted:

data WORK.ONE;

Text='Australia, US, Denmark';

Pos=find(Text,'US','i',5);

run;

What value will SAS assign to Pos?

A. 0

B. 1

C. 2

**D. 12**

**I don’t blame myself for missing this one. In our textbook, the modifier is ‘I’ or ‘T’. So I did not know that we could use the lower case. Of course, I tried the sas. It turns out that we could use both.**

**data** WORK.ONE;

Text='Australia, US, Denmark';

Pos=find(Text,'US','i',**5**);

**run**;

**proc** **print**;**run**;

**data** WORK.ONE;

Text='Australia, US, Denmark';

Pos=find(Text,'US','t',**5**);

**run**;

**proc** **print**;**run**;

**data** WORK.ONE;

Text='Australia, US, Denmark';

Pos=find(Text,'US','I',**5**);

**run**;

**proc** **print**;**run**;

|  |
| --- |
| The SAS System |

| **Obs** | **Text** | **Pos** |
| --- | --- | --- |
| **1** | Australia, US, Denmark | 12 |

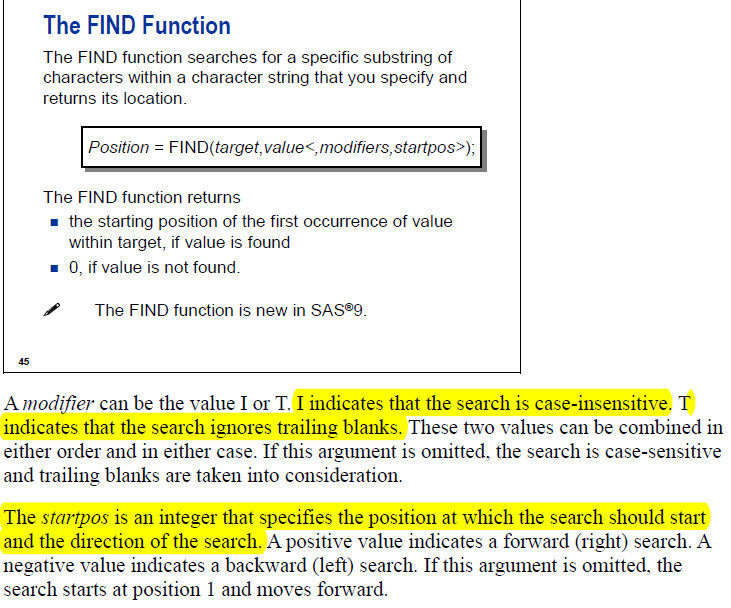
|  |
| --- |
| The SAS System |

| **Obs** | **Text** | **Pos** |
| --- | --- | --- |
| **1** | Australia, US, Denmark | 12 |

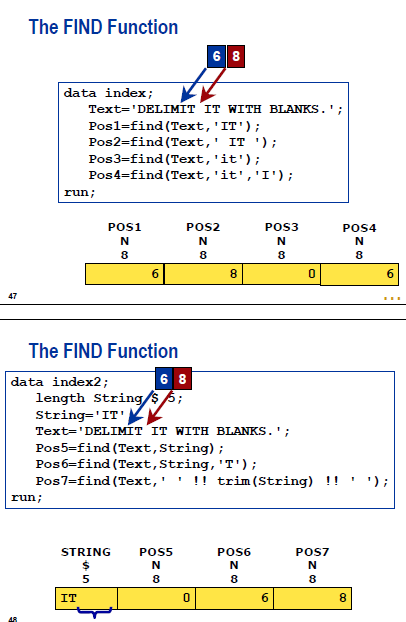
|  |
| --- |
| The SAS System |

| **Obs** | **Text** | **Pos** |
| --- | --- | --- |
| **1** | Australia, US, Denmark | 12 |

Note that ‘T’ has no effect here, because there is no trailing blanks in the string: ‘US’.







本题考察的是SAS中FIND函数的应用。  
POS=FIND(Text,’US’,'i’,5)解释为：  
从[Australia](http://crackman.net/?tag=australia), US, [Denmark](http://crackman.net/?tag=denmark) 字符串的第五个字符开始，忽略字符串中大小写，寻找字符串US在源字符串中的起始位置，并返回起始位置的数字给POS。  
几个问题值得思考，第一个是如果没有’i'这个参数，结果会怎么样？如果把5改成1，那么结果又如何？  
看看SAS官网网站的解释：  
Syntax  
FIND(string,substring<,modifiers><,startpos>)   
FIND(string,substring<,startpos><,modifiers>)

Searches for a specific substring of characters within a character string.

Arguments

string  
specifies a character constant, variable, or expression that will be searched for substrings.

Tip: Enclose a literal string of characters in quotation marks.

substring  
is a character constant, variable, or expression that specifies the substring of characters to search for in string.

Tip: Enclose a literal string of characters in quotation marks.

modifiers  
is a character constant, variable, or expression that specifies one or more modifiers. The following modifiers can be in uppercase or lowercase:

i ignores character case during the search. If this modifier is not specified, FIND only searches for character substrings with the same case as the characters in substring.  
   
t trims trailing blanks from string and substring.

Note:   If you want to remove trailing blanks from only one character argument instead of both (or all) character arguments, use the TRIM function instead of the FIND function with the T modifier.   
   
Tip: If modifier is a constant, enclose it in quotation marks. Specify multiple constants in a single set of quotation marks. Modifier can also be expressed as a variable or an expression.

startpos  
is a numeric constant, variable, or expression with an integer value that specifies the position at which the search should start and the direction of the search.

By the way, crackman asks us to think about what would happen if we drop ‘i’ and use 1.

As long as we have the starting position as 5, dropping ‘i’ has no effect.

If we drop 5, then the value will change:

**data** WORK.ONE;

Text='Australia, US, Denmark';

Pos=find(Text,'US','I');

**run**;

**proc** **print**;**run**;

|  |
| --- |
| The SAS System |

| **Obs** | **Text** | **Pos** |
| --- | --- | --- |
| **1** | Australia, US, Denmark | 2 |

**Obviously, they gave us the position of the highlighted u.**

# 43 Put function gives you character; input gives you numbers

43.Given the SAS data set WORK.ORDERS:

WORK.ORDERS

order\_id customer shipped

-------- ------------ ---------

9341 Josh Martin 02FEB2009

9874 Rachel Lords 14MAR2009

10233 Takashi Sato 07JUL2009

The variable order\_id is numeric; customer is character; and shipped is numeric, contains a SAS date value,and is shown with the DATE9. format.

A programmer would like to create a new variable, ship\_note,that shows a character value with the order\_id,shipped date, and customer name.

For example, given the first observation ship\_note would have the value "Order 9341 shipped on 02FEB2009 to Josh Martin".

Which of the following statement will correctly create the value and assign it to ship\_note?

A. ship\_note=catx(' ','Order',order\_id,'shipped on',input(shipped,date9.),'to',customer);

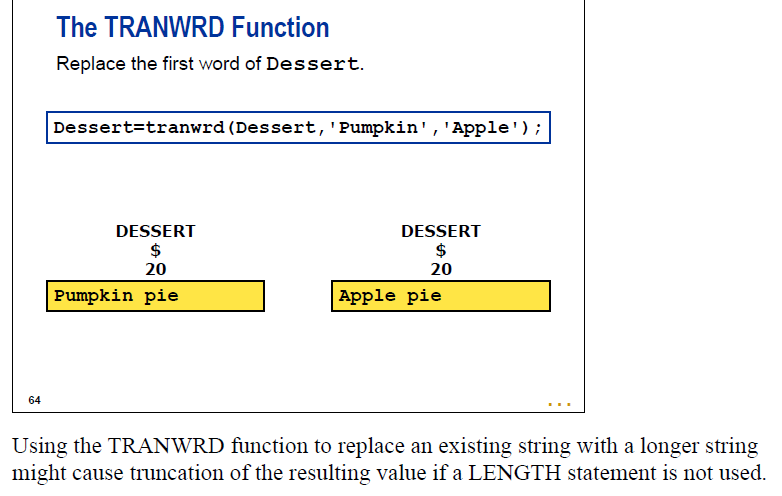
B. ship\_note=catx(' ','Order',order\_id,'shipped on',char(shipped,date9.),'to',customer);

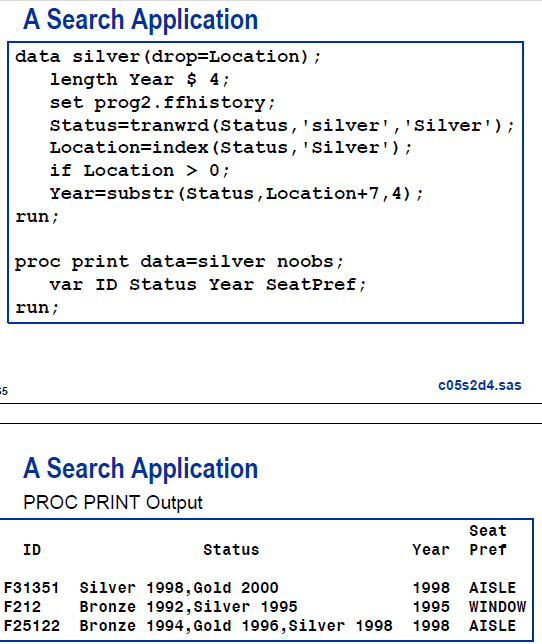
C. ship\_note=catx(' ','Order',order\_id,'shipped on',**transwrd**(shipped,date9.),'to',customer);

D. ship\_note=catx(' ','Order',order\_id,'shipped on',put(shipped,date9.),'to',customer);

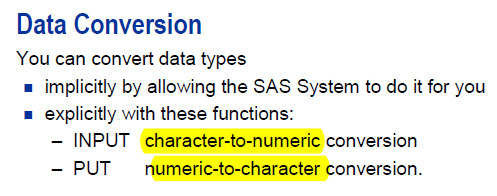
Note that there is no such function as the one in red. It might be a typo.

## TRANWRD function

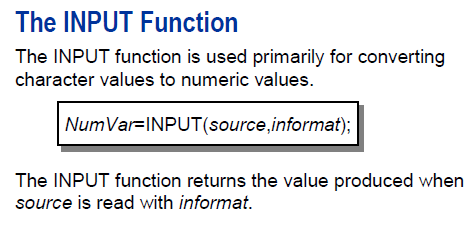


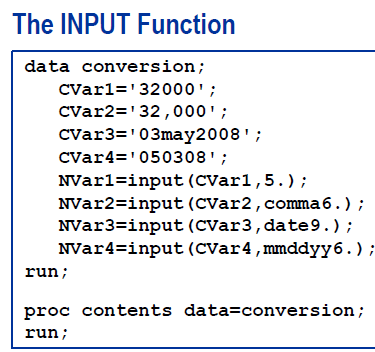


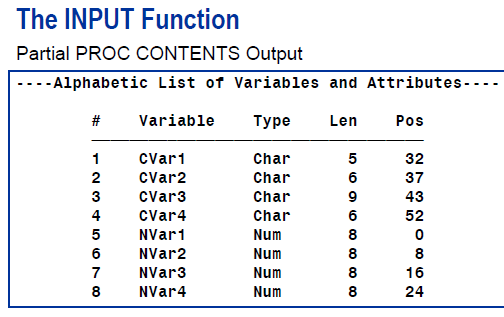
## input function



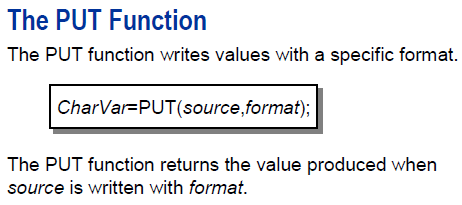
Input will give you numeric, while put will give you character.

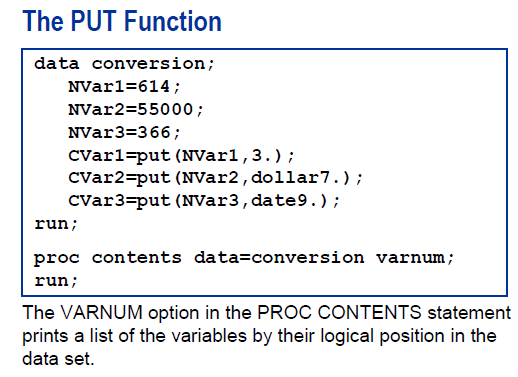


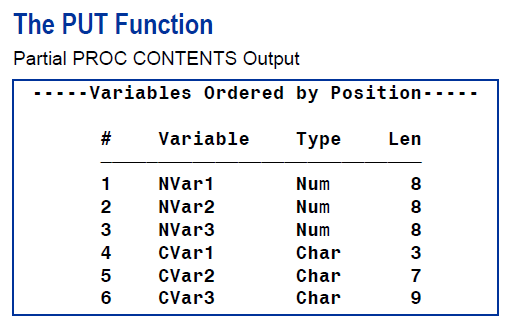




## Put function







## CHAR Function

Returns a single character from a specified position in a character string.

|  |  |
| --- | --- |
| Category: | Character |

|  |
| --- |
| [Syntax](http://support.sas.com/documentation/cdl/en/lrdict/64316/HTML/default/a003082990.htm#a003105743) |
| [Arguments](http://support.sas.com/documentation/cdl/en/lrdict/64316/HTML/default/a003082990.htm#a003105744) |
| [Details](http://support.sas.com/documentation/cdl/en/lrdict/64316/HTML/default/a003082990.htm#a003105745) |
| [Comparisons](http://support.sas.com/documentation/cdl/en/lrdict/64316/HTML/default/a003082990.htm#a003105746) |
| [Examples](http://support.sas.com/documentation/cdl/en/lrdict/64316/HTML/default/a003082990.htm#a003105747) |
| [See Also](http://support.sas.com/documentation/cdl/en/lrdict/64316/HTML/default/a003082990.htm#a003105748) |

|  |
| --- |
|  |
| Syntax |

|  |
| --- |
| CHAR(string, position) |

### Arguments

string

specifies a character constant, variable, or expression.

position

is an integer that specifies the position of the character to be returned.

|  |
| --- |
|  |
| Details |

In a DATA step, the default length of the target variable for the CHAR function is 1.

If position has a missing value, then CHAR returns a string with a length of 0. Otherwise, CHAR returns a string with a length of 1.

If position is less than or equal to 0, or greater than the length of the string, then CHAR returns a blank. Otherwise, CHAR returns the character at the specified position in the string.

|  |
| --- |
|  |
| Comparisons |

The CHAR function returns the same result as SUBPAD(string, position, 1). While the results are the same, the default length of the target variable is different.

|  |
| --- |
|  |
| Examples |

The following example shows the results of using the CHAR function.

options pageno=1 ps=64 ls=80 nodate;

data test;

retain string "abc";

do position = -1 to 4;

result=char(string, position);

output;

end;

run;

proc print noobs data=test;

run;

Output from the CHAR Function

The SAS System 1

string position result

abc -1

abc 0

abc 1 a

abc 2 b

abc 3 c

abc 4

# 44. don’t forget the semicolons

44.The following SAS program is submitted:

data ONE TWO SASUSER.TWO

set SASUSER.ONE;

run;

Assuming that SASUSER.ONE exists, how many temporary and permanent SAS data sets are created?

A. 2 temporary and 1 permanent SAS data sets are created

B. 3 temporary and 2 permanent SAS data sets are created

C. 2 temporary and 2 permanent SAS data sets are created

D. there is an error and no new data sets are created

Answer: D

本题考察的是语句结尾注意分号的问题。  
在SAS里面一般一条语句都是以分号作为结束标志。  
本程序中第一条DATA步 后没分号。所以答案为D。

Karma! When coding, I forget semicolons at times. No wonder, I did not notice that error.

# 45. ods csvall file='c:\test.cvs';

45.The following SAS program is submitted:

ods csvall file='c:\test.cvs';

proc print data=WORK.ONE;

var Name Score Grade;

by IdNumber;

run;

ods csvall close;

What is produced as output?

A. A file named test.cvs that can only be opened in Excel.

B. A text file named test.cvs that can be opened in Excel or in any text editor.

C. A text file named test.cvs that can only be opened in a text editor.

D. A file named test.cvs that can only be opened by SAS.

Answer: B

本题考察的是CSV文件格式打开问题，可以用EXCEL也可以[用TEXT EDITOR](http://crackman.net/?tag=%e7%94%a8text-editor)打开。

# 46 mean(of Rev:) Use colons to represent a list of variables.

46.Given the SAS data set WORK.ONE:

Obs Revenue2008 Revenue2009 Revenue2010

--- ----------- ----------- -----------

1 1.2 1.6 2.0

The following SAS program is submitted:

data WORK.TWO;

set WORK.ONE;

Total=mean(of Rev:);

run;

What value will SAS assign to Total?

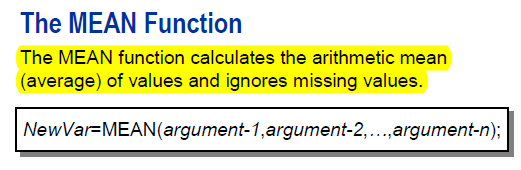
A. 3

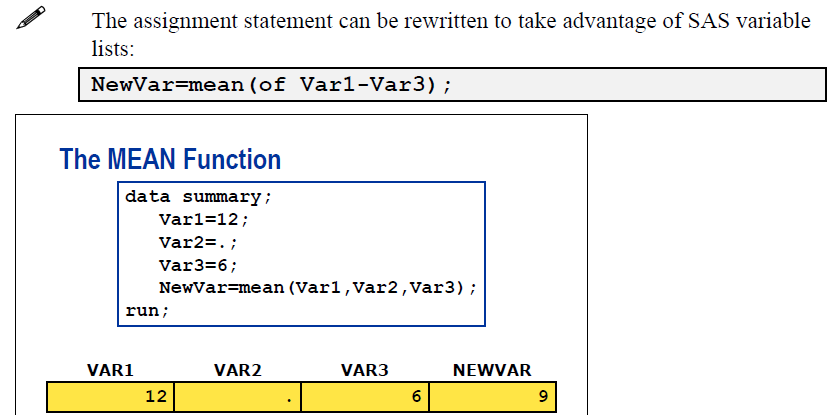
B. 1.6

C. 4.8

D. The program fails to execute due to errors.

I got this one right by guessing. SAS textbook and online help session don’t talk about this.





Anyway, I ran the program and it was perfect.

**DATA** one;

INPUT Revenue2008 Revenue2009 Revenue2010;

DATALINES;

1.2 1.6 2.0

;

;**run**;

**proc** **print**;**run**;

**data** WORK.TWO;

set WORK.ONE;

Total=mean(of Rev:);

**run**;

**proc** **print**;**run**;

|  |
| --- |
| The SAS System |

| **Obs** | **Revenue2008** | **Revenue2009** | **Revenue2010** |
| --- | --- | --- | --- |
| **1** | 1.2 | 1.6 | 2 |

|  |
| --- |
| The SAS System |

| **Obs** | **Revenue2008** | **Revenue2009** | **Revenue2010** | **Total** |
| --- | --- | --- | --- | --- |
| **1** | 1.2 | 1.6 | 2 | 1.6 |

本题考察的是变量简写的问题。  
TOTAL=MEAN(OF REV:) 等价于 TOTAL=MEAN(OF REVENUE2009-REVENUE2010) 或者TOTAL=MEAN(REVENUE2008，REVENUE2009，REVENUE2010).  
REV:表示是以REV开头的所有变量。  
可以考虑一下，如果是REV结尾的所有变量，该如何操作呢？

# 47 Tables variable1\*variable2/nocol norow

47.The following output is created by the FREQUENCY procedure:

The FREQ Procedure

Table of region by product

region product

Frequency|

Percent |

Row Pct |

Col Pct |corn |cotton |oranges | Total

---------+--------+--------+--------+

EAST | 2 | 1 | 1 | 4

| 22.22 | 11.11 | 11.11 | 44.44

| 50.00 | 25.00 | 25.00 |

| 50.00 | 33.33 | 50.00 |

---------+--------+--------+--------+

SOUTH | 2 | 2 | 1 | 5

| 22.22 | 22.22 | 11.11 | 55.56

| 40.00 | 40.00 | 20.00 |

| 50.00 | 66.67 | 50.00 |

---------+--------+--------+--------+

Total 4 3 2 9

44.44 33.33 22.22 100.00

Which TABLES option(s) would be used to eliminate the

row and column counts and just see the frequencies and percents?

A. norowcount nocolcount

B. freq percent

**C. norow nocol**

D. nocounts

**data** CellCounts;

input R C Count @@;

datalines;

1 1 5 1 2 3

2 1 4 2 2 3

;**run**;

**proc** **print**;**run**;

**proc** **freq** data=CellCounts;

tables R\*C;

**run**;

**proc** **freq** data=CellCounts;

tables R\*C /nocol norow;

**run**;

|  |
| --- |
| The SAS System |

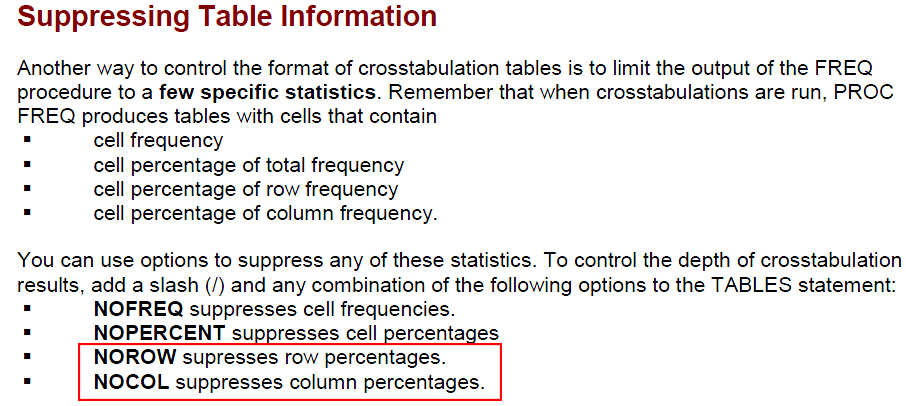
The FREQ Procedure

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  | | --- | --- | --- | --- | --- | | |  | | --- | | **Frequency** | | **Percent** | | **Row Pct** | | **Col Pct** | | | | **Table of R by C** | | | | | --- | --- | --- | --- | | **R** | **C** | | | | **1** | **2** | **Total** | | **1** | |  | | --- | | 1 | | 25.00 | | 50.00 | | 50.00 | | |  | | --- | | 1 | | 25.00 | | 50.00 | | 50.00 | | |  | | --- | | 2 | | 50.00 | |  | |  | | | **2** | |  | | --- | | 1 | | 25.00 | | 50.00 | | 50.00 | | |  | | --- | | 1 | | 25.00 | | 50.00 | | 50.00 | | |  | | --- | | 2 | | 50.00 | |  | |  | | | **Total** | |  | | --- | | 2 | | 50.00 | | |  | | --- | | 2 | | 50.00 | | |  | | --- | | 4 | | 100.00 | | |

|  |
| --- |
| The SAS System |

The FREQ Procedure

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  | | --- | --- | --- | | |  | | --- | | **Frequency** | | **Percent** | | | | **Table of R by C** | | | | | --- | --- | --- | --- | | **R** | **C** | | | | **1** | **2** | **Total** | | **1** | |  | | --- | | 1 | | 25.00 | | |  | | --- | | 1 | | 25.00 | | |  | | --- | | 2 | | 50.00 | | | **2** | |  | | --- | | 1 | | 25.00 | | |  | | --- | | 1 | | 25.00 | | |  | | --- | | 2 | | 50.00 | | | **Total** | |  | | --- | | 2 | | 50.00 | | |  | | --- | | 2 | | 50.00 | | |  | | --- | | 4 | | 100.00 | | |



Answer: C  
本题考察的是PROC FREQ过程步中TABLE语句的参数。  
答案是NOROW NOCOL。  
具体看官方解释：  
<http://support.sas.com/documentation/cdl/en/procstat/63104/HTML/default/viewer.htm#procstat_freq_sect010.htm>

# 48 Column input

48.The following SAS program is submitted:

data WORK.TEST;

**drop City;**

infile datalines;

input

Name $ 1-14 /

Address $ 1-14 /

City $ 1-12 ;

if City='New York ' then .;

else input;

datalines;

Joe Conley

123 Main St.

Janesville

WI

Jane Ngyuen

555 Alpha Ave.

New York

NY

Jennifer Jason

666 Mt. Diablo

Eureka

CA

;

What will the data set WORK.TEST contain?

**A.**

Name Address State

-------------- ---------------- ------

Joe Conley 123 Main St.

Jane Ngyuen 555 Alpha Ave. NY

Jennifer Jason 666 Mt. Diablo

B.

Name Address City State

-------------- ---------------- ----------- ------

Joe Conley 123 Main St. Janesville

Jane Ngyuen 555 Alpha Ave. New York NY

Jennifer Jason 666 Mt. Diablo Eureka

C.

Name Address State

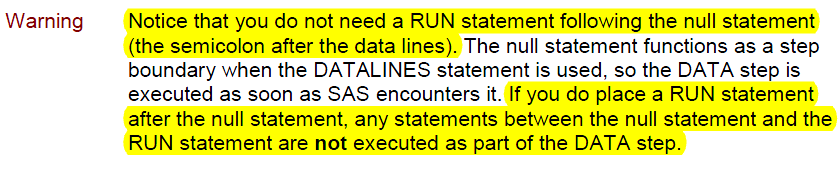
-------------- ---------------- ------

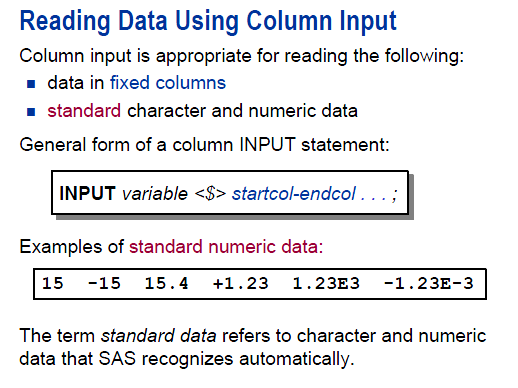
Jane Ngyuen 555 Alpha Ave. NY

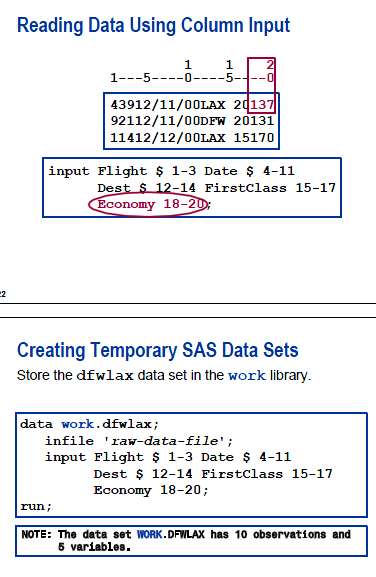
D. O observations,there is a syntax error in the data step.

Answer:A

What a pity! I neglected the line **drop City; Also note that after datalines….; we don't need run;**







这道题考察的时候INPUT语句中读入数据的指针走向，现在解析如下：  
程序编译后，建立PDV，此时只有三个Name Address City ，在DATA步执行语句中加入另外一个变量STATE到PDV中。  
 Name $ 1-14 /  
 Address $ 1-14 /  
 City $ 1-12 ;  
按照指定的列项读取每一行数据，/表示换行读取。  
当CITY数据读入之后，指针进入下一行，就是WI，此时，根据判断，进入ELSE中的执行语句INPUT，此时的INPUT没有指定此行数据读入PDV中那个变量，等于是让指针释放此行数据，进入下一个数据行。那么STATE为默认缺失值。重复前面的过程。

* Unique Post
* **data** WORK.TEST;
* drop City;
* infile datalines;
* input
* Name $ **1**-**14** /
* Address $ **1**-**14** /
* City $ **1**-**12** ;
* if City='New York' then input @**1** State $2.;
* else input;
* datalines;
* Joe Conley
* 123 Main St.
* Janesville
* WI
* Jane Ngyuen
* 555 Alpha Ave.
* New York
* NY
* Jennifer Jason
* 666 Mt. Diablo
* Eureka
* CA
* ;
* **proc** **print**;**run**;

|  |
| --- |
| * The SAS System |

| **Obs** | **Name** | **Address** | **State** |
| --- | --- | --- | --- |
| **1** | Joe Conley | 123 Main St. |  |
| **2** | Jane Ngyuen | 555 Alpha Ave. | NY |
| **3** | Jennifer Jason | 666 Mt. Diablo |  |

# 49 An array cannot be referenced on a keep= data set option

49.The following SAS program is submitted:

data WORK.TOTALSALES(keep=MonthSales{12});

set WORK.MONTHLYSALES(keep=Year Product Sales);

array MonthSales{12};

do i=1 to 12;

MonthSales{i}=Sales;

end;

drop i;

run;

The program fails execution due to syntax errors.

What is the cause of the syntax error?

A. An array cannot be referenced on a keep= data set option.

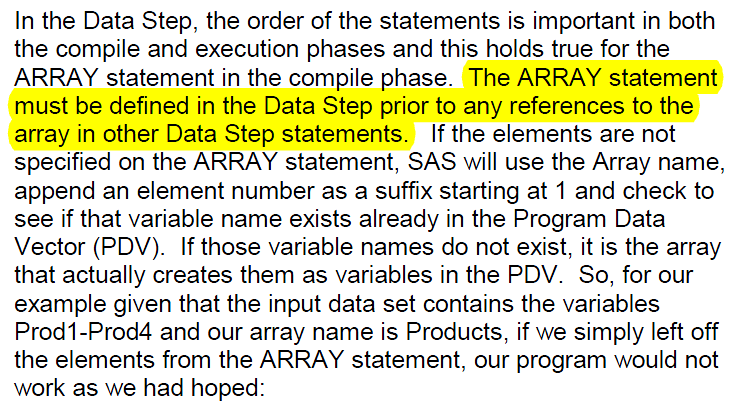
B. The keep= data set option should be (keep=MonthSales\*).

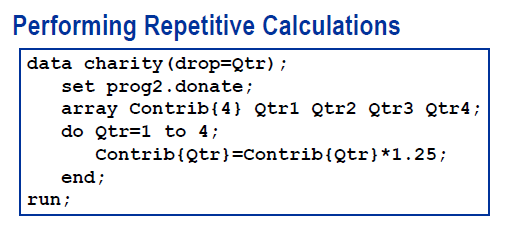
C. The keep= data set option should be the statement KEEP MonthSales{12}.

D. The variable MonthSales does not exist.

Answer:A

本题考察的是数组以及KEEP语句。  
KEEP语句中必须是具体变量，不能是数组。  
另外数组MonthSales{12}，并没有说明这个数组内部的成员名称。通过下标来引用数组成员的。





# 50 Interleaving data sets with by statement: which variable comes first.

50.Given the SAS data set WORK.ONE:

Id Char1

--- -----

111 A

158 B

329 C

644 D

and the SAS data set WORK.TWO:

Id Char2

--- -----

111 E

538 F

644 G

The following program is submitted:

data WORK.BOTH;

set WORK.ONE WORK.TWO;

by Id;

run;

What is the first observation in SAS data set WORK.BOTH?

A. Id Char1 Char2

--- ----- -----

111 A

B.

Id Char1 Char2

--- ----- -----

111 E

C.

Id Char1 Char2

--- ----- -----

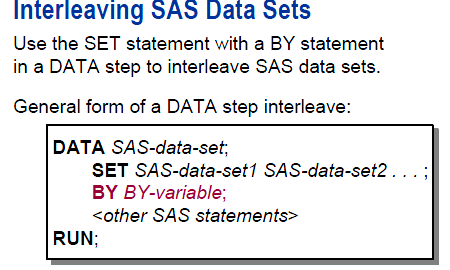
111 A E

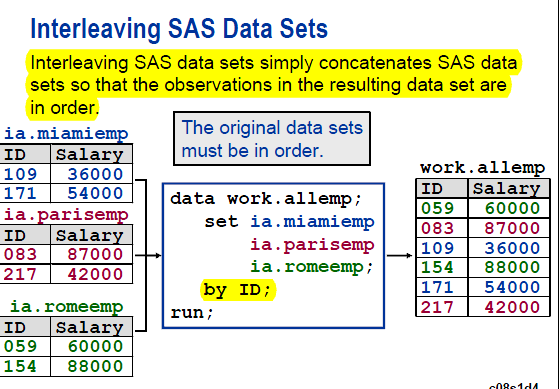
D.

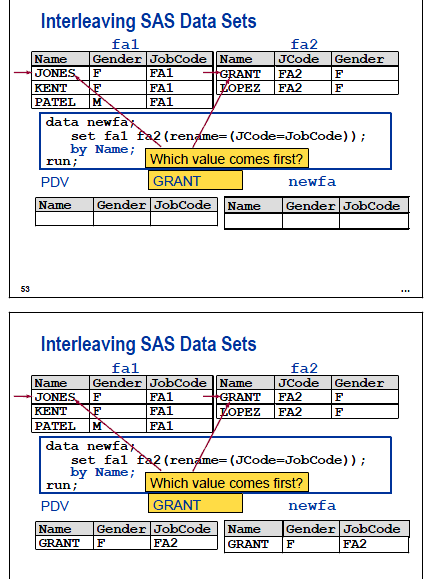
Id Char1 Char2

--- ----- -----

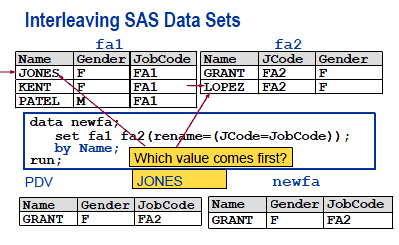
644 D G



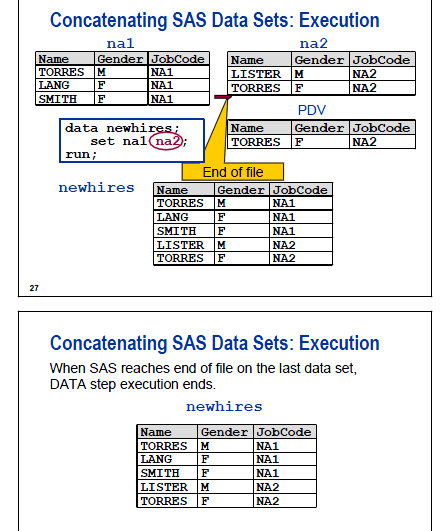




This above slide is noteworthy. Although Fa1 is before fa2 in the set statement, grant comes first because of the by name; statement.



Compare the above situation to the one without the by statement:



Answer: A  
  
本题主要是考察的是DATA步，SET语句 建立PDV时的问题。  
程序执行过程中，根据SET语句建立PDV，因为有两个数据集，ONE和TWO，ONE中两个变量，TWO中也是两个变量，其中ID相同，所以PDV中包括这三个变量，注意一点，我前面的文章都是说PDV里面的包括的变量，是除了\_N\_,\_ERROR\_等自动变量的。  
在读入数据时，首先是从ONE读取到PDV中。读完之后输出到数据集BOTH中，构成第一条观测。因为ONE数据集没有变量CHAR2，所以CHAR2默认为缺失值。  
继续从TWO数据集中读入数据到PDV中，读完之后输出到数据集BOTH中，构成第二条观测，因为TWO没有CHAR1变量，所以就为缺失值。  
最后由BY语句进行了分组。  
因为ONE TWO中ID的顺序都是降序的，所以BY语句在这里主要起到的是分组功能。  
大家可以思考一下 把SET改成MERGE是什么结果？  
那么MERGE SQL合并数据集上的差别是什么呢？

# 51 Proc contents data=\_all\_;

51.The following program is submitted:

proc contents data=\_all\_;

run;

Which statement best describes the output from the submitted program?

A. The output contains only a list of the SAS data sets that are contained in the WORK library.

B. The output displays only the contents of the SAS data sets that are contained in the WORK library.

C. The output displays only the variables in the SAS data sets that are contained in the WORK library.

D. The output contains a list of the SAS data sets that are contained in the WORK library and displays the contents of those data sets.

Answer:  D

I forgot that it includes the list of sas data sets.

本题考察的是PROC CONTENTS。  
答案为D，具体可以在运行该语句后得到结果。  
包括了：list也就是WORK下的数据集的目录；然后每一个数据集的contents，里面包括variables等信息。

|  |
| --- |
| The SAS System |

The CONTENTS Procedure

| **Directory** | |
| --- | --- |
| **Libref** | WORK |
| **Engine** | V9 |
| **Physical Name** | C:\Users |
| **Filename** | C:\Users\ |

| **#** | **Name** | **Member Type** | **File Size** | **Last Modified** |
| --- | --- | --- | --- | --- |
| **1** | CELLCOUNTS | DATA | 5120 | 06Jun12:21:30:05 |
| **2** | ONE | DATA | 5120 | 06Jun12:20:57:42 |
| **3** | RAW | DATA | 5120 | 06Jun12:21:25:12 |
| **4** | TEST | DATA | 5120 | 06Jun12:21:48:29 |
| **5** | TWO | DATA | 5120 | 06Jun12:20:58:16 |

|  |
| --- |
| The SAS System |

The CONTENTS Procedure

|  |  |  |  |
| --- | --- | --- | --- |
| **Data Set Name** | WORK.CELLCOUNTS | **Observations** | 4 |
| **Member Type** | DATA | **Variables** | 3 |
| **Engine** | V9 | **Indexes** | 0 |
| **Created** | Wednesday, June 06, 2012 09:30:05 PM | **Observation Length** | 24 |
| **Last Modified** | Wednesday, June 06, 2012 09:30:05 PM | **Deleted Observations** | 0 |
| **Protection** |  | **Compressed** | NO |
| **Data Set Type** |  | **Sorted** | NO |
| **Label** |  |  |  |
| **Data Representation** | WINDOWS\_64 |  |  |
| **Encoding** | wlatin1 Western (Windows) |  |  |

| **Engine/Host Dependent Information** | |
| --- | --- |
| **Data Set Page Size** | 4096 |
| **Number of Data Set Pages** | 1 |
| **First Data Page** | 1 |
| **Max Obs per Page** | 168 |
| **Obs in First Data Page** | 4 |
| **Number of Data Set Repairs** | 0 |
| **Filename** | C:\Users\ |
| **Release Created** | 9.0301M0 |
| **Host Created** | X64\_7PRO |

| **Alphabetic List of Variables and Attributes** | | | |
| --- | --- | --- | --- |
| **#** | **Variable** | **Type** | **Len** |
| **2** | C | Num | 8 |
| **3** | Count | Num | 8 |
| **1** | R | Num | 8 |

# 52 Merge. In dataset option

52.Given the SAS data set WORK.EMP\_NAME:

Name EmpID

---- -----

Jill 1864

Jack 2121

Joan 4698

John 5463

Given the SAS data set WORK.EMP\_DEPT:

EmpID Department

----- ----------

2121 Accounting

3567 Finance

4698 Marketing

5463 Accounting

The following program is submitted:

data WORK.ALL;

merge WORK.EMP\_NAME(in=Emp\_N)

WORK.EMP\_DEPT(in=Emp\_D);

by Empid;

if (Emp\_N and not Emp\_D) or (Emp\_D and not Emp\_N);

run;

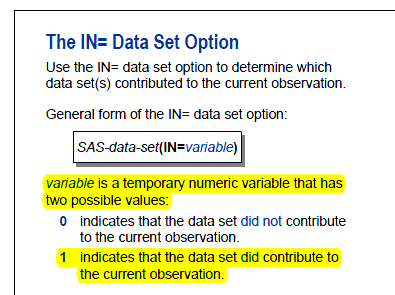
How many observations are in data set WORK.ALL after submitting the program?

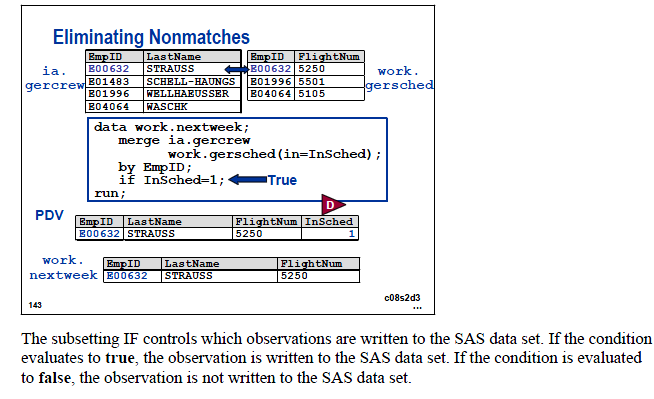
A. 1

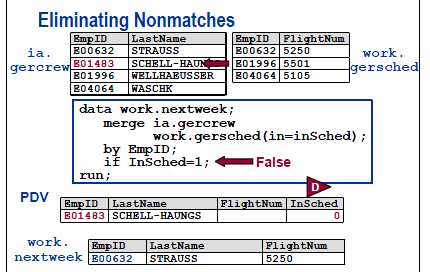
B. 2

C. 3

D. 5







# 53 Retain statement initializes the value to missing by default.

53.The following SAS program is submitted:

data WORK.TOTAL\_SALARY;

retain Total;

set WORK.SALARY;

by Department;

if First.Department

then Total=0;

Total=sum(Total, Wagerate);

if Last.Total;

run;

What is the initial value of the variable Total?

A. 0

B. Missing

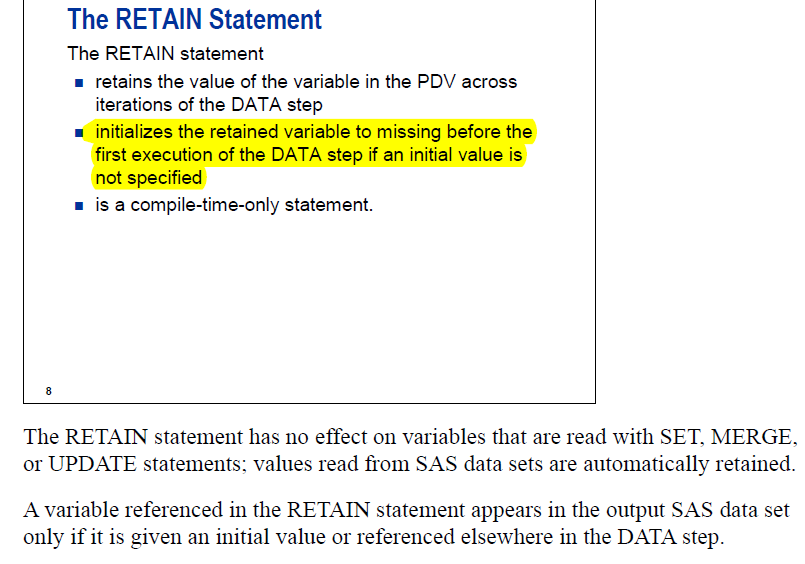
C. The value of the first observations Wagerate

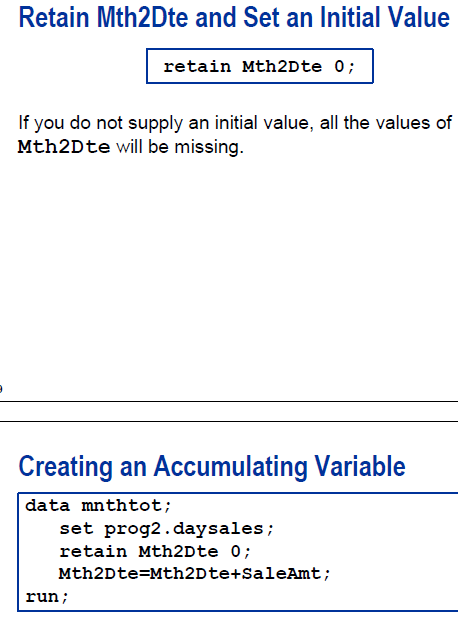
D. Cannot be determined from the information given

Answer: B

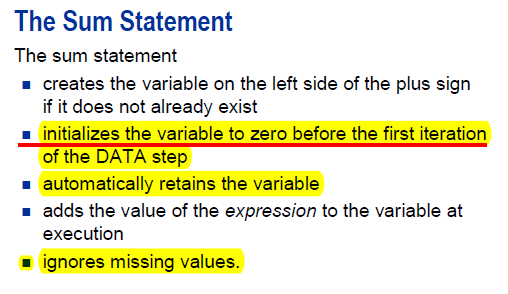
本文属于博客：[http://crackman.net](http://crackman.net/) 版权归作者所有，欢迎转载！如有转载，请务必注明出处！未经本文作者同意不得用于商业应用。

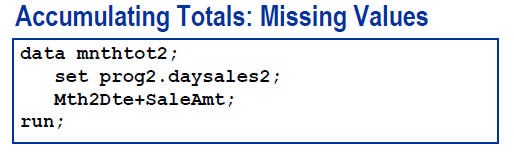
这里考察的是[retain](http://crackman.net/?tag=retain) 语句。  
看看SAS HELP文档中对retain语句的解释：  
RETAIN <element-list(s) <initial-value(s) |  
(initial-value-1) | (initial-value-list-1) >  
< . . . element-list-n <initial-value-n |  
(initial-value-n ) | (initial-value-list-n)>>>;  
这里的retain total;  
只有total(element-list) 但是没有initial-value,如果忽略initial-value，那么就是默认为缺失值。  
If you omit initial-value, the initial value is missing





**Compare the retain statement with the sum statement. Notice that the retain statement initializes the variable to missing before the execution of the data step, if no initial value is specified. By contrast, sum statement initializes the value to zero before the first iteration of the data step.**





# 54 retain City 'Beverly Hills';

54.Consider the following data step:

data WORK.TEST;

set SASHELP.CLASS(obs=5);

retain City 'Beverly Hills';

State='California';

run;

The computed variables City and State have their values assigned using two different methods, a RETAIN statement and an Assignment statement. Which statement regarding this program is true?

A. The RETAIN statement is fine, but the value of City will be truncated to 8

bytes as the LENGTH statement has been omitted.

B. Both the RETAIN and assignment statement are being used to initialize new

variables and are equally efficient. Method used is a matter of programmer preference.

C. The assignment statement is fine, but the value of City will be truncated

to 8 bytes as the LENGTH statement has been omitted.

D. City's value will be assigned one time, State's value 5 times.

**I got this one right, but was very surprised to find that the initial value of the retain statement could be a character**.

Answer: D

其实这里考察的RETAIN到底是在非执行语句还是执行语句。  
RETAIN的作用是初始化变量为“特定的默认值”，是非执行语句，在编译过程中，PDV中用“特定的默认值”替代SAS系统默认值。在DATA步中，加 入STATE进入PDV中，STATE=’California’; 是赋值语句，也就是执行语句。每一次从SASHELP.CLASS读取数据之后，都要执行一次赋值语句。

|  |
| --- |
| Syntax |

|  |
| --- |
| RETAIN <element-list(s) <initial-value(s) |  (initial-value-1) | (initial-value-list-1) >  < . . . element-list-n <initial-value-n |  (initial-value-n ) | (initial-value-list-n)>>>; |

initial-value

specifies an initial value, numeric or **character**, for one or more of the preceding elements.

|  |  |
| --- | --- |
| Tip: | If you omit initial-value, the initial value is missing. Initial-value is assigned to all the elements that precede it in the list. All members of a variable list, therefore, are given the same initial value. |

This RETAIN statement retains the values of nine variables and sets their initial values:

retain month1-month5 1 year 0 a b c 'XYZ';

The values of MONTH1 through MONTH5 are set initially to 1; YEAR is set to 0; variables A, B, and C are each set to the character value XYZ .

# 55. Sas date constant: no blank before d.

55.The following SAS program is submitted:

data WORK.DATE\_INFO;

X="01Jan1960" D ;

run;

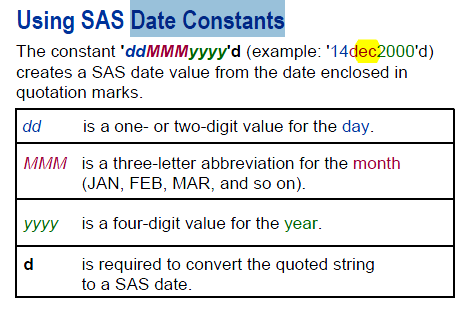
Variable X contains what value?

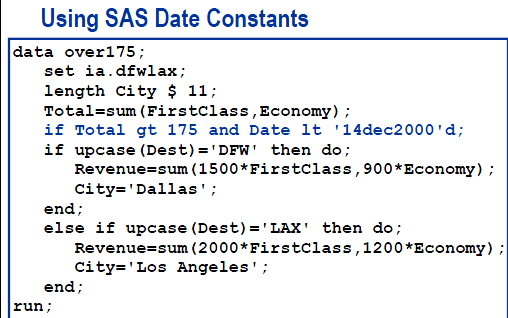
A. the numeric value 0

B. the character value "01Jan1960"

C. the date value 01011960

**D. the code contains a syntax error and does not execute.**





**data** WORK.DATe;

X='01Jan1960' D;

**run**;

**proc** **print**;**run**;

245 data WORK.DATe;

246 X='01Jan1960' D;

-

22

ERROR 22-322: Syntax error, expecting one of the following: !, !!, &, \*, \*\*, +, -, /, <, <=, <>, =,

>, ><, >=, AND, EQ, GE, GT, IN, LE, LT, MAX, MIN, NE, NG, NL, NOTIN, OR, ^=, |, ||,

~=.

247 run;

NOTE: Character values have been converted to numeric values at the places given by:

(Line):(Column).

246:8

NOTE: The SAS System stopped processing this step because of errors.

WARNING: The data set WORK.DATE may be incomplete. When this step was stopped there were 0

observations and 2 variables.

WARNING: Data set WORK.DATE was not replaced because this step was stopped.

**data** WORK.DATE\_INFO;

X=**'01Jan1960'D**;

**run**;

**proc** **print**;**run**;

253 data WORK.DATE\_INFO;

254 X='01Jan1960'D;

255 run;

NOTE: The data set WORK.DATE\_INFO has 1 observations and 1 variables.

NOTE: DATA statement used (Total process time):

real time 0.00 seconds

cpu time 0.01 seconds

256 proc print;run;

NOTE: There were 1 observations read from the data set WORK.DATE\_INFO.

NOTE: PROCEDURE PRINT used (Total process time):

real time 0.07 seconds

cpu time 0.03 seconds

|  |
| --- |
| The SAS System |

| **Obs** | **X** |
| --- | --- |
| **1** | 0 |

Crackman’s comment is perfect. It is so difficult to notice that there is a blank. However, if you look closely the color is different.

1. tsxn 说道：

[2011/10/09 10:52](http://crackman.net/?p=372#comment-452)

hi , Crackman, thanks for your detailed explanation of each question . I learned lots here.  
For this question, I tried to use Capital D to do the same thing, it works in SAS9. I don’t know why.

thanks

[回复](http://crackman.net/?p=372&replytocom=452#respond)

1. [*crackman*](http://crackman.net) 说道：

[2011/10/09 15:49](http://crackman.net/?p=372#comment-453)

错误的原因是因为  
data WORK.DATE\_INFO;  
X=”01Jan1960″D;  
run;  
X表达式中的D与引号”的距离

[回复](http://crackman.net/?p=372&replytocom=453#respond)

# 56. Tables varible1\*variable2

56.The following output is created by the FREQUENCY procedure:

The FREQ Procedure

Table of region by product

region product

Frequency|

Percent |

Row Pct |

Col Pct |corn |cotton |oranges | Total

---------+--------+--------+--------+

EAST | 2 | 1 | 1 | 4

| 22.22 | 11.11 | 11.11 | 44.44

| 50.00 | 25.00 | 25.00 |

| 50.00 | 33.33 | 50.00 |

---------+--------+--------+--------+

SOUTH | 2 | 2 | 1 | 5

| 22.22 | 22.22 | 11.11 | 55.56

| 40.00 | 40.00 | 20.00 |

| 50.00 | 66.67 | 50.00 |

---------+--------+--------+--------+

Total 4 3 2 9

44.44 33.33 22.22 100.00

Which TABLES statement was used to completed the following program

that produced the output?

proc freq data=sales;

<\_insert\_code\_>

run;

A. tables region product;

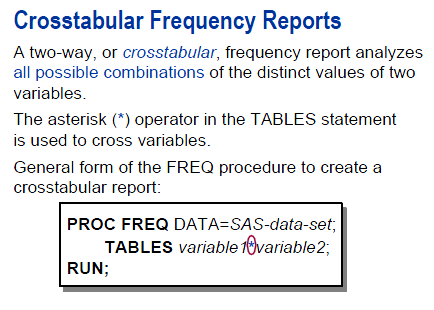
B. tables region,product

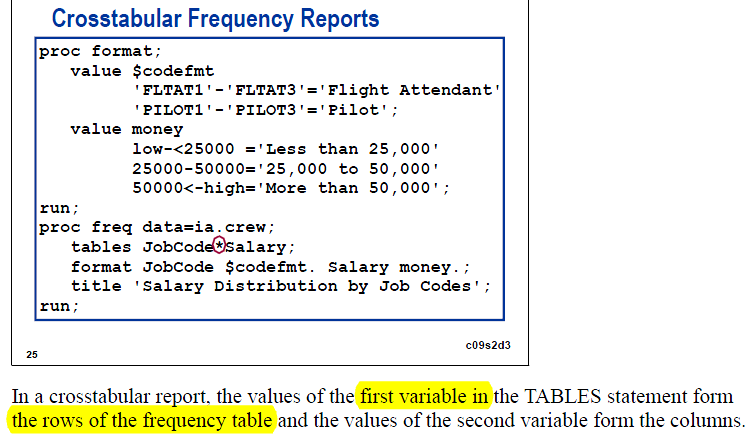
C. tables region/product;

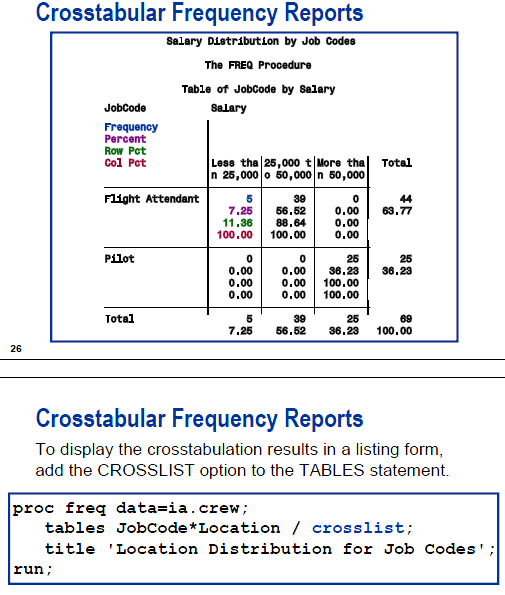
D. tables region\*product;

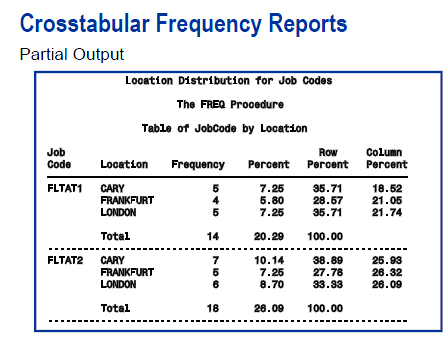
Answer: D

这里考察的是FREQ过程中TABLE语句。  
看看官方的解释：  
<http://support.sas.com/documentation/cdl/en/procstat/63104/HTML/default/viewer.htm#procstat_freq_sect010.htm>  
   
不过可以思考一下如果是第一个语句，结果会是如何？region和product各一个table









# 57 Sas date function

57.Given the SAS data set WORK.ONE:

N BeginDate

- ---------

1 09JAN2010

2 12JAN2010

The following SAS program is submitted:

data WORK.TWO;

set WORK.ONE;

Day=<\_insert\_code\_>;

format BeginDate date9.;

run;

The data set WORK.TWO is created, where Day would be 1 for Sunday, 2 for Monday, 3 for Tuesday, ... :

WORK.TWO

N BeginDate Day

- --------- ---

1 09JAN2010 1

2 12JAN2010 4

Which expression successfully completed the program and creates the variable Day?

A. day(BeginDate)

B. weekday(BeginDate)

C. dayofweek(BeginDate)

D. getday(BeginDate,today())

Answer: B

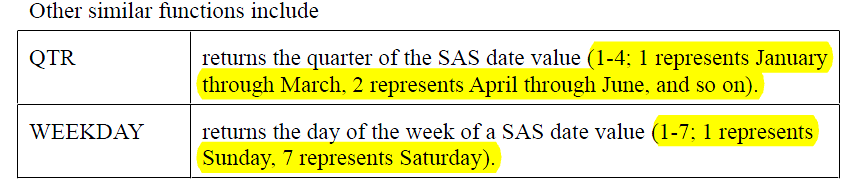
这里考察的是函数weekday的用法。  
此函数是活的某日期中的某天在一周星期中的数值。星期天就是1，星期六就是7。

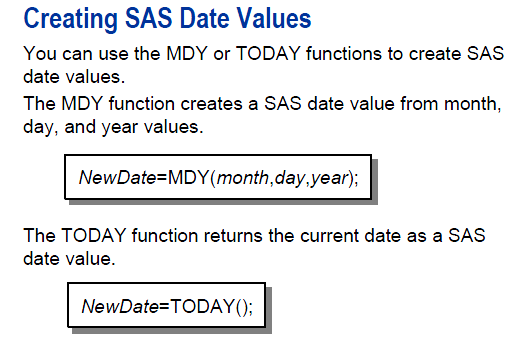
**DAY( *date* )**

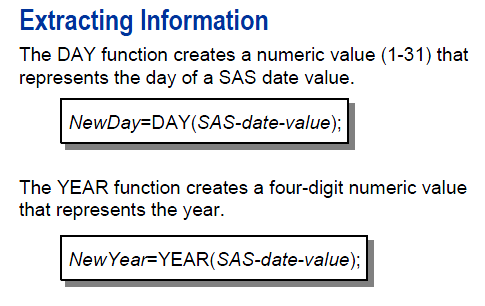
returns the day of the month from a SAS date value.

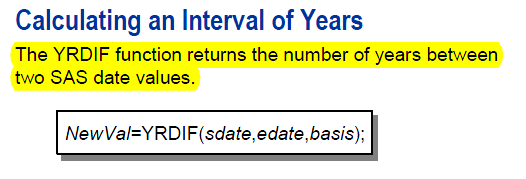
**WEEKDAY( *date* )**

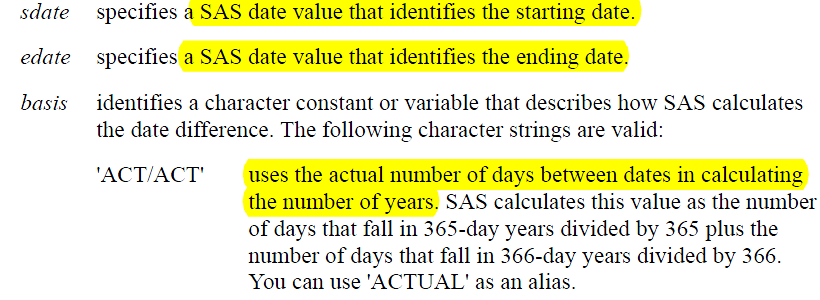
returns the day of the week from a SAS date value. For example **WEEKDAY=WEEKDAY(’17OCT1991’D);** returns http://support.sas.com/documentation/cdl/en/etsug/60372/HTML/default/images/etsug_intervals0011.png, the numerical value for Thursday.











**data** mo;

y=yrdif(**'01nov1984'd**,today(),'act/act');

x=mdy(**06**,**07**,**2012**);

**run**;

**proc** **print**;**run**;

|  |
| --- |
| The SAS System |

| **Obs** | **y** | **x** |
| --- | --- | --- |
| **1** | 27.5984 | 19151 |

# 58 Proc format; value name (no period)

58.The following program is submitted:

proc format;

value salfmt.

0 -< 50000 = 'Less than 50K'

50000 - high = '50K or Greater';

options fmterr nodate pageno=1;

title 'Employee Report';

proc print data=work.employees noobs;

var fullname salary hiredate;

format

salary salfmt.

hiredate date9.;

label

fullname='Name of Employee'

salary='Annual Salary'

hiredate='Date of Hire';

run;

Why does the program fail?

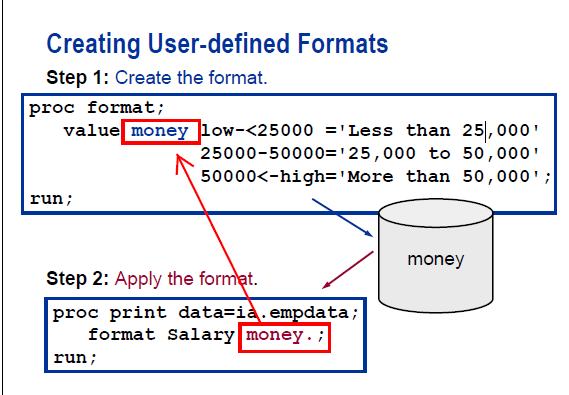
A. The PAGENO option is invalid in the OPTIONS statement.

B. The RUN statement is missing after the FORMAT procedure.

C. The format name contains a period in the VALUE statement.

D. The LABEL option is missing from the PROC PRINT statement.

错误就在PROC FORMAT语句中，value的命名salfmt. 不应搞包括DOT（.）。  
这个点号是SAS用来区别一个变量名以及一个format格式名称的差别。  
看看英文一个说明：  
The period (or “dot”) in the Format  
distinguishes it from a Variable Name. When SAS “sees” the “dot,” it “knows” to associate the instructions in the  given Format to the Variable. Since SAS Variable Names can only contain letters, numbers and the underscore  
symbol, the period symbol is how SAS detects the difference between a Variable and a Format.



Note that the first money is not followed by a period, but the second is.

# 59 Delimited raw data file and truncover

59.Given the contents of the raw data file TYPECOLOR.DAT:

----+----10---+----20---+----30

daisyyellow

The following SAS program is submitted:

data FLOWERS;

infile 'TYPECOLOR.DAT' truncover;

length

Type $ 5

Color $ 11;

input

Type $

Color $;

run;

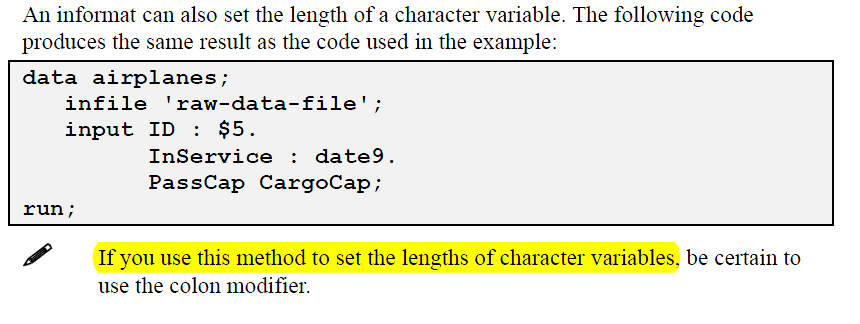
What are the values of the variables Type and Color?

A. Type=daisy, Color=yellow

B. Type=daisy, Color=w

C. Type=daisy, Color=daisyyellow

**D. Type=daisy, Color=**



**data** FLOWERS;

infile 'TYPECOLOR.DAT' truncover;

length

Type $ **5**

Color $ **11**;

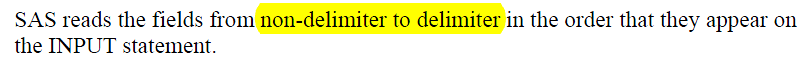
input

Type $

Color $;

**run**;

Note that the above code does not specify the delimiter, the delimiter is by default blank.



Since there is no blank in the original string, SAS simply stops after reading daisy.

|  |
| --- |
| The SAS System |

| **Obs** | **Type** | **Color** |
| --- | --- | --- |
| **1** | daisy |  |

In order to solve the problem, we can just have a blank between daisy and yellow.

**data** FLOWERS;

infile 'TYPECOLOR1.DAT' missover;

length

Type $ **5**

Color $ **11**;

input

Type $

Color $;

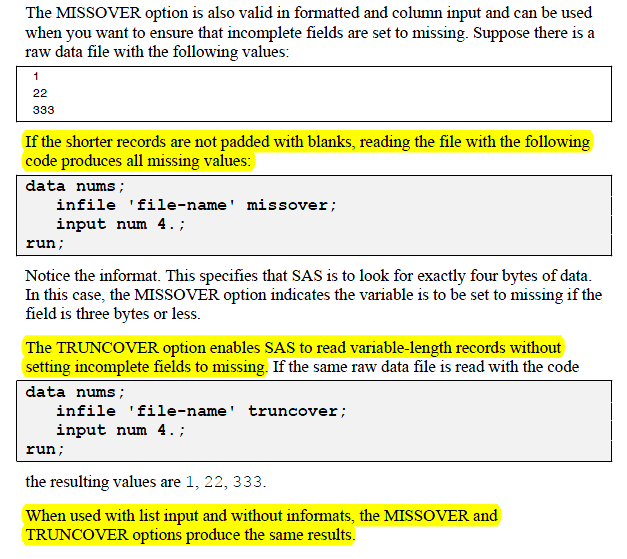
**run**;

**proc** **print**;**run**;

|  |
| --- |
| The SAS System |

| **Obs** | **Type** | **Color** |
| --- | --- | --- |
| **1** | daisy | yellow |

By the way, in this case missover and truncover are equivalent.



Answer: D

这里考察的还是INFILE语句以及参数TRUNCOVER。  
TRUNCOVER这个参数类似于MISSOVER这个参数，是组织INPUT语句从下一数据行读取数据，如果本行数据读完还有未赋值的语句，那么就默认为缺失值。  
这里daisyyellow这一行数据，[input](http://crackman.net/?tag=input)读入前5个字符到边了TYPE中，那么这一行就读取完，但是TURNCOVER阻止读下一数据行，所以COLOR为缺失值。  
这里可以思考一下，如果变成 dais yyellow 那么结果会如何呢？

# 60 Drop option in the data step

60.Given the SAS data set WORK.PRODUCTS:

ProdId Price ProductType Sales Returns

------ ----- ----------- ----- -------

K12S 95.50 OUTDOOR 15 2

B132S 2.99 CLOTHING 300 10

R18KY2 51.99 EQUIPMENT 25 5

3KL8BY 6.39 OUTDOOR 125 15

DY65DW 5.60 OUTDOOR 45 5

DGTY23 34.55 EQUIPMENT 67 2

The following SAS program is submitted:

data WORK.REVENUE(drop=Sales Returns Price);

set WORK.PRODUCTS(keep=ProdId Price Sales Returns);

Revenue=Price\*(Sales-Returns);

run;

How many variables does the WORK.REVENUE data set contain?

A. 2

B. 3

C. 4

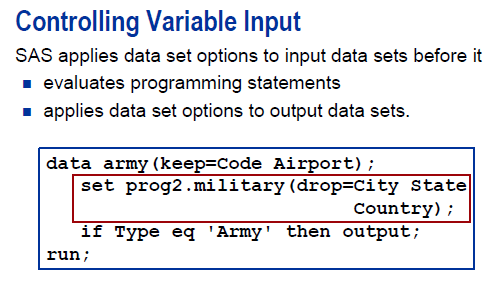
D. 6

Answer: A

答案是A，只有两个，一个是prodid，一个是revenue。

SET WORK.PRODUCTS(keep=ProdId Price Sales Returns)这条语句，在建立PDV时，就已经对ProdId Price Sales Returns做了KEEP操作，所以当你在：

  data WORK.REVENUE(drop=Sales Returns Price);  
     set WORK.PRODUCTS(keep=ProdId Price Sales Returns);  
     Revenue=Price\*(Sales-Returns);  
     PUT  ProductType；  
  run;  
   
 ProductType是缺失值，等于是在DATA 步中再次建立一个变量，名字叫 ProductType。



City state and country will not even enter into the army dataset.

# 61 if Age not in(15,16) then

61.Consider the data step:

data WORK.TEST;

infile 'c:\class1.csv' dsd;

input Name $ Sex $ Age Height Weight;

if Age NE 16 and Age NE 15 then Group=1;

else Group=2;

run;

Which statement produces a functionally equivalent result for assigning Group a value?

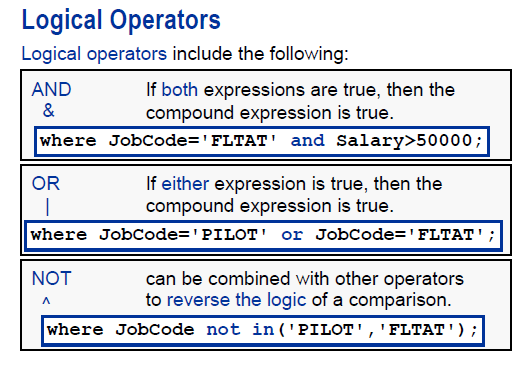
A. if Age not in(15,16) then Group=1; else Group=2;

B. if (Age NE 16) or (Age NE 15) then Group=1; else Group=2;

C. where Age not between 15 and 16 then Group=1; else Group=2;

D. both A or C will work.

Answer: A  
本题考察IF ELSE语句以及筛选条件。  
C是错误，WHERE不能这样与THEN连用，THEN只能与IF 配对使用。



The above example is used in the proc print. I paste it here because it shows how to use not in.

# 62 ods html file='sales.html'

62.The following SAS program is submitted:

<\_insert\_ods\_code\_>

proc means data=SASUSER.SHOES;

where Product in ('Sandal' , 'Slipper' , 'Boot');

run;

<\_insert\_ods\_code\_>

Which ODS statements, inserted in the two locations above, create a report stored in an html file?

A.

ods html open='sales.html';

ods html close;

B.

ods file='sales.html' / html;

ods file close;

C.

ods html file='sales.html';

ods html close;

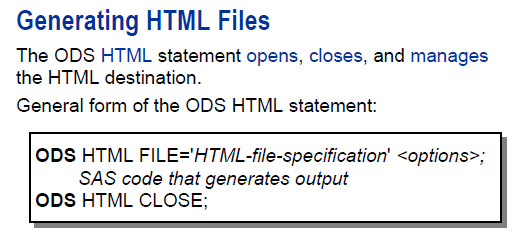
D.

ods file html='sales.html';

ods file close;

Answer: C

ODS里面，没有ODS FILE这一个语句。  
ODS HTML FILE=”DESTINATION”；  
具体可以看看HELP文档。





# 63 Do until

63.The following SAS program is submitted:

data WORK.OUTDS;

do until(Prod GT 6);

Prod + 1;

end;

run;

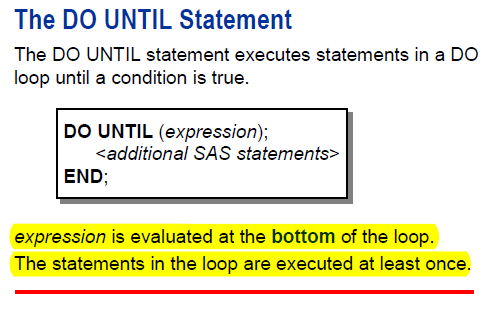
What is the value of the variable Prod in the output data set?

A. . (missing)

B. 6

C. 7

D. Undetermined, infinite loop.



If the initial statement is true, then SAS goes through the data step once.

**data** WORK.OUTDS;

do until(Prod GT **6**);

Prod + **1**;

end;

**run**;

**proc** **print**;**run**;

|  |
| --- |
| The SAS System |

| **Obs** | **Prod** |
| --- | --- |
| **1** | 7 |

# 65 Permanently assign labels in the data step

65.The following SAS program is submitted:

data WORK.ACCOUNTING;

set WORK.DEPARTMENT;

label Jobcode='Job Description';

run;

Which statement is true about the output dataset?

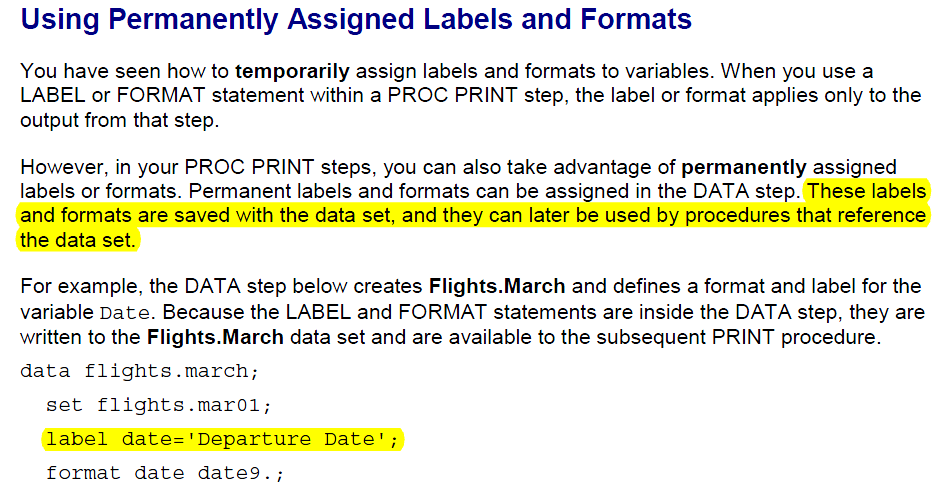
A. The label of the variable Jobcode is Job (only the first word).

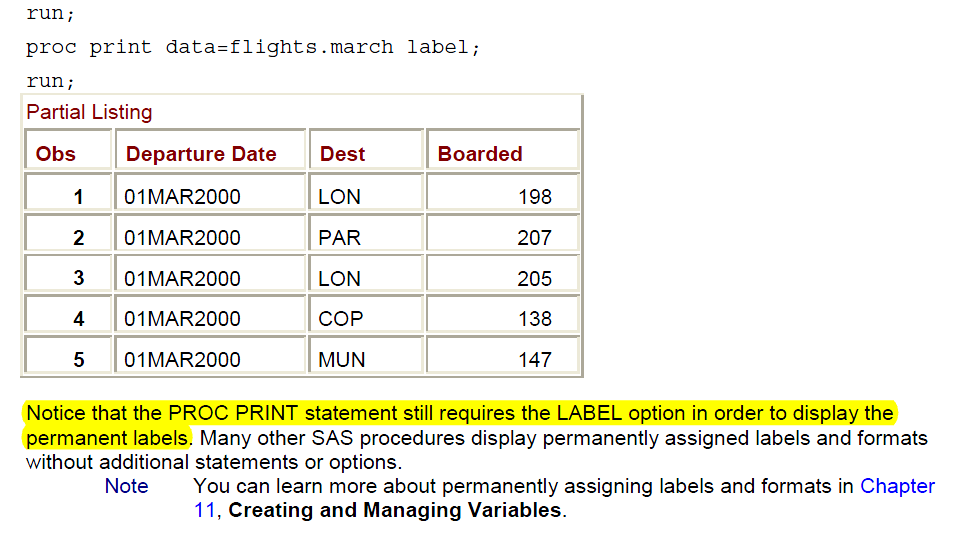
B. The label of the variable Jobcode is Job Desc (only the first 8 characters).

**C. The label of the variable Jobcode is Job Description.**

D. The program fails to execute due to errors. Labels must be defined in a PROC step.

I forgot this.





|  |
| --- |
| Details |

Using a LABEL statement in a DATA step permanently associates labels with variables by affecting the descriptor information of the SAS data set that contains the variables. You can associate any number of variables with labels in a single LABEL statement.

You can use a LABEL statement in a PROC step, but the rules are different. See the Base SAS Procedures Guide for more information.

|  |
| --- |
|  |
| Comparisons |

Both the ATTRIB and LABEL statements can associate labels with variables and change a label that is associated with a variable.

|  |
| --- |
|  |
| Examples |

Example 1: Specifying Labels

Here are several LABEL statements:

* label compound=Type of Drug;
* label date="Today's Date";
* label n='Mark''s Experiment Number';
* label score1="Grade on April 1 Test"
* score2="Grade on May 1 Test";

Example 2: Removing a Label

This example removes an existing label:

data rtest;

set rtest;

label x=' ';

run;

|  |
| --- |
|  |

# 66 do loop

66.The following SAS program is submitted:

data WORK.SALES;

do Year=1 to 5;

do Month=1 to 12;

X + 1;

end;

end;

run;

How many observations are written to the WORK.SALES data set?

A. 0

**B. 1**

C. 5

D. 60

Answer: B

这里主要考虑的是循环结束之后才输出到数据集WORK.SALES中。  
X + 1;  
等价于  
[retain](http://crackman.net/?tag=retain) x 0;  
x=x+1;  
如下：  
year=1 month 1 x=1  
       month 2 x=上一次X的值+1=2  
       month 3 x=month为2时X的值，也就是2，所以此时X=3。  
注意这一语句执行一直在DO 循环中，X是不断的被改变的，所以当循环结束之后，X才输出到数据集中，所以只有一个观测对象。  
如何把每一步的X输出呢？  
可以思考一下。

Note the differences in the positions of output lead to differences in the numbers of observations in the output.

**data** WORK.SALES;

do Year=**1** to **5**;

do Month=**1** to **12**;

X + **1**;

end;

end;

**run**;

**proc** **print**;**run**;

|  |
| --- |
| The SAS System |

| **Obs** | **Year** | **Month** | **X** |
| --- | --- | --- | --- |
| **1** | 6 | 13 | 60 |

**data** WORK.SALES;

do Year=**1** to **5**;

do Month=**1** to **12**;

X + **1**;

output;

end;

end;

**run**;

**proc** **print**;**run**;

|  |
| --- |
| The SAS System |

| **Obs** | **Year** | **Month** | **X** |
| --- | --- | --- | --- |
| **1** | 1 | 1 | 1 |
| **2** | 1 | 2 | 2 |
| **3** | 1 | 3 | 3 |
| **4** | 1 | 4 | 4 |
| **5** | 1 | 5 | 5 |
| **6** | 1 | 6 | 6 |
| **7** | 1 | 7 | 7 |
| **8** | 1 | 8 | 8 |
| **9** | 1 | 9 | 9 |
| **10** | 1 | 10 | 10 |
| **11** | 1 | 11 | 11 |
| **12** | 1 | 12 | 12 |
| **13** | 2 | 1 | 13 |
| **14** | 2 | 2 | 14 |
| **15** | 2 | 3 | 15 |
| **16** | 2 | 4 | 16 |
| **17** | 2 | 5 | 17 |
| **18** | 2 | 6 | 18 |
| **19** | 2 | 7 | 19 |
| **20** | 2 | 8 | 20 |
| **21** | 2 | 9 | 21 |
| **22** | 2 | 10 | 22 |
| **23** | 2 | 11 | 23 |
| **24** | 2 | 12 | 24 |
| **25** | 3 | 1 | 25 |
| **26** | 3 | 2 | 26 |
| **27** | 3 | 3 | 27 |
| **28** | 3 | 4 | 28 |
| **29** | 3 | 5 | 29 |
| **30** | 3 | 6 | 30 |
| **31** | 3 | 7 | 31 |
| **32** | 3 | 8 | 32 |
| **33** | 3 | 9 | 33 |
| **34** | 3 | 10 | 34 |
| **35** | 3 | 11 | 35 |
| **36** | 3 | 12 | 36 |
| **37** | 4 | 1 | 37 |
| **38** | 4 | 2 | 38 |
| **39** | 4 | 3 | 39 |
| **40** | 4 | 4 | 40 |
| **41** | 4 | 5 | 41 |
| **42** | 4 | 6 | 42 |
| **43** | 4 | 7 | 43 |
| **44** | 4 | 8 | 44 |
| **45** | 4 | 9 | 45 |
| **46** | 4 | 10 | 46 |
| **47** | 4 | 11 | 47 |
| **48** | 4 | 12 | 48 |
| **49** | 5 | 1 | 49 |
| **50** | 5 | 2 | 50 |
| **51** | 5 | 3 | 51 |
| **52** | 5 | 4 | 52 |
| **53** | 5 | 5 | 53 |
| **54** | 5 | 6 | 54 |
| **55** | 5 | 7 | 55 |
| **56** | 5 | 8 | 56 |
| **57** | 5 | 9 | 57 |
| **58** | 5 | 10 | 58 |
| **59** | 5 | 11 | 59 |
| **60** | 5 | 12 | 60 |

**data** WORK.SALES;

do Year=**1** to **5**;

do Month=**1** to **12**;

X + **1**;

end;

output;

end;

**run**;

**proc** **print**;**run**;

|  |
| --- |
| The SAS System |

| **Obs** | **Year** | **Month** | **X** |
| --- | --- | --- | --- |
| **1** | 1 | 13 | 12 |
| **2** | 2 | 13 | 24 |
| **3** | 3 | 13 | 36 |
| **4** | 4 | 13 | 48 |
| **5** | 5 | 13 | 60 |

# 67 See a missing value as negative infinity

67.Consider the following data step:

data WORK.NEW;

set WORK.OLD(keep=X);

if X < 10 then X=1;

else if X >= 10 AND X LT 20 then X=2;

else X=3;

run;

In filtering the values of the variable X in data set WORK.OLD, what value new value would be assigned to X if its original value was a missing value?

A. X would get a value of 1.

B. X would get a value of 3.

C. X would retain its original value of missing.

D. This step does not run because of syntax errors.

Answer: A

如果是缺失值，那么在排序过程中是默认为最小。  
大家可以用[proc sort](http://crackman.net/?tag=proc-sort) 过程来测试一组有缺失值的数据，就知道排在最小的观测的是缺失值。

The instructor told us to see a missing value as negative infinity.

**data** old;

x=**.**;

**run**;

**proc** **print**;**run**;

**data** WORK.NEW;

set WORK.OLD(keep=X);

if X < **10** then X=**1**;

else if X >= **10** AND X LT **20** then X=**2**;

else X=**3**;

**run**;

**proc** **print**;**run**;

|  |
| --- |
| The SAS System |

| **Obs** | **x** |
| --- | --- |
| **1** | . |

|  |
| --- |
| The SAS System |

| **Obs** | **x** |
| --- | --- |
| **1** | 1 |

# 68 Character variable vs. numeric variable.

68.The following SAS program is submitted:

data WORK.ACCOUNTING;

set WORK.DEPARTMENT;

length EmpId $6;

CharEmpid=EmpId;

run;

If data set WORK.DEPARTMENT has a numeric variable EmpId,which statement is true about the output dataset?

A. The type of the variable CharEmpid is numeric.

B. The type of the variable CharEmpid is unknown.

C. The type of the variable CharEmpid is character.

D. The program fails to execute due to errors.

Answer: D   
这里主要是LENGTH定义数据的格式与[WORK.DEPARTMENT](http://crackman.net/?tag=work-department)中的数据格式存在冲突。

# 69 where Code like 'E\_U%';

69.Given the data set WORK.EMPDATA:

Employee\_ Manager\_

ID Job\_Title Department ID

------- ---------------------- ---------------- ------

120101 Director Sales Management 120261

120102 Sales Manager Sales Management 120101

120103 Sales Manager II Sales Management 120101

120104 Administration Manager Administration 120101

120105 Secretary I Administration 120101

Which one of the following where statements would display observations with job titles containing the word 'Manager'?

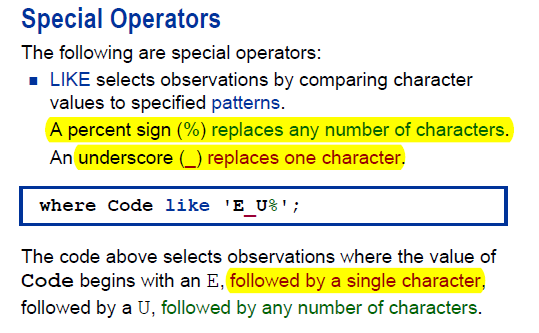
A. where substr(Job\_Title,(length(Job\_Title)-6))='Manager';

B. where upcase(scan(Job\_Title,-1,' '))='MANAGER';

C. where Job\_Title='% Manager ';

D. where Job\_Title like '%Manager%';

答案是D  
WHERE在做筛选数据时的，LIKE是检查数据是否满足某一样式。  
%可以替代任意数量的字符。



# 70 Data step drop option

70.After a SAS program is submitted, the following is written to the SAS log:

105 data WORK.JANUARY;

106 set WORK.ALLYEAR(keep=Product Month Quantity Cost);

107 if Month='JAN' then output WORK.JANUARY;

108 Sales=Cost \* Quantity;

109 drop=Month Quantity Cost;

-----

22

ERROR 22-322: Syntax error, expecting one of the following: !,

!!, &, \*, \*\*, +, -,

, <=, <>, =, >, >=,

AND, EQ, GE, GT, IN, LE, LT, MAX, MIN, NE, NG, NL,

NOTIN, OR, ^=, |, ||, ~=.

110 run;

What data set option could be attached to WORK.JANUARY to replace the DROP statement that generated the error in the log?

A. (drop Month Quantity Cost)

B. (drop Month, Quantity, Cost)

C. (drop=Month, Quantity, Cost)

D. (drop=Month Quantity Cost)

Answer: D

DROP语句，在DATA STATEMENT中，不需要等号  
在SET 语句中，需要括号以及等号。