

A User's Guide to
FACFORM
Data Formatter

for FACETS
Rasch-Model
Computer Programs

John M. Linacre

P. O. Box 811322
Chicago IL 60681-1322
Tel. & FAX (312) 264-2352

www.winsteps.com

© Copyright 1991-2008, John M. Linacre
Permission to copy is granted.
ISBN 0-941938-03-4

Table of Contents

2. Keyword Index.....	3
3. INTRODUCTION	4
3.1. What is Facform?.....	4
3.2. Flat files and Facets files.....	4
3.3. Facets, elements and observations	4
3.4. Data compression.....	4
3.5. Facform creates Facets specification and data files from four sources	5
3.6. The data reformatting process takes place in 7 phases	5
4. RUNNING Facform.....	6
4.1. To start Facform under Windows	6
4.2. To start Facform under DOS.....	6
4.3. Ending Facform	7
4.4. While Facform is running	8
4.5. Facform output to Facets input	9
5. EXAMPLES	9
5.1. A two-facet flat data file	9
5.2. A three-facet flat data file	11
5.3. A WINSTEPS input data file.....	12
5.4. A many-facet flat file	14
5.5. A many-facet flat file with multiple raters per line.....	17
5.6. Data file using existing Facets specification file	19
5.7. Flat file with multiple lines per record.....	22
5.8. Fixed length data file with multiple records per line	23
5.9. Comma-separated Value (CSV) data file	25
5.10. Data file with grouped elements	26
5.11. Data file with raters, items and item structures over multiple lines.....	29
6. THE KEYWORD LANGUAGE	36
6.1. Keywords	36
6.2. File definition keywords	36
6.3. Data definition keywords.....	36
6.4. Flow control keywords	38
6.5. Text-oriented keywords	40
6.6. Expression evaluation	41
6.7. Expression commands	41
6.8. Variables	42
6.9. Facform stops prematurely: Disk file problem	43
7. PROGRAM MESSAGES.....	43
7.1. Facform stops prematurely: Keyword list problem	43
7.2. Facform stops prematurely: Problem messages.....	43
7.3. Warning messages	47

2. Keyword Index

Suggestion: **Factorm is complicated. Excel may be easier for building Facets specification and data files.**

[\\$Again](#) (end a loop)
[\\$Batch](#) (run in batch mode)
[\\$Do](#) = (expression) (start a conditional loop)
[\\$Do](#) = number (start a loop)
[\\$Else](#) (or else, conditionally)
[\\$Elseif](#) = (expression) (\$Else followed by \$If)
[\\$Endif](#) (end of conditional section)
[\\$Facets](#) = number (number of facets in data)
[\\$Flabel](#) = number,name [,codes](name of facet)
[\\$Given](#) = Yes (all labels given in \$Sinput file)
[\\$If](#) = (expression) (perform conditionally)
[\\$Input](#) = name (input data file)
[\\$Label](#) = facet [, element number] [, element name] [, "anchor value+","+"group"] (element labels)
[\\$Length](#) = number (fixed length data lines)
[\\$Length](#) = 0 (standard DOS input data lines)
[\\$Nextline](#) (go on to next input data line)
[\\$Output](#) = name(Facets-format data)
[\\$Print](#) = expression (display on screen)
[\\$Rating](#) = number [, replications] (datum value)
[\\$Separator](#) = "value" (separator character in a variable length or CSV file)
[\\$Spec](#) = expression (put directly into \$Spoutput file once)
[\\$Sinput](#) = name (previous Facets specifications)
[\\$Spoutput](#) = name (Facets specifications)
[\\$Text](#) = expression (put into \$Spoutput file, each time encountered)

;
"..." (followed by comments)
(text)
() (arrange order of expression evaluation)
+ (concatenate)
+*/^ (plus, minus, times, divide, exponent)
><= (greater, less, equal)
&| (and, or)

[Blank lines](#) (use as spacing to ease reading keyword files)
[Blank spaces](#) (using as spacing within lines)
number (numeric value)
[Variable](#) = expression (calculate a value)

[\\$A????](#) (treat as UPPER case, alphabetical)
[\\$Cnnn](#) (locate Comma-Separated value)
[\\$G](#) (the current line-group number)
[\\$I????](#) (treat as an integer value)
[\\$L](#) (use the current line number)
[\\$N????](#) (convert into a number)
[\\$Q????](#) (put value between "")
[\\$SxxEyy](#) (start and end in data line)
[\\$SxxWww](#) (start and width in data line)
[\\$Tnnn](#) (locate Tab-separated value)
[\\$U????](#) (unquote, remove from "")

3. INTRODUCTION

3.1. What is Facform?

There is more information at: www.winsteps.com

Suggestion: **Facform is complicated. Excel is easier for building specification and data files.**

This manual contains instructions for operating Facform, a computer program for the formatting of data prior to Facets analysis of many-facet data. Facform performs the conversion from "flat" data files to Facets data files by means of a set of instructions comprising keywords and values. This manual explains how to do it.

3.2. Flat files and Facets files

Data are often transferred between programs in a flat file, in which a line, or group of lines, correspond to one element, e.g., an examinee. The contents of the lines and the position of the data within the lines indicate the other elements with which the line element came in contact, and the results of their interactions.

In order to convert data from flat file format to Facets format, this data conversion program, with its data description language, can be used.

3.3. Facets, elements and observations

Facets data are constructed from facets, elements and observations:

- A. The facets. Each facet is a set of persons or items or judges or tasks or times of day or other aspects of the situation which generated the data. There can be up to 255 facets.
- B. The identifiers or elements in each facets. Each element in a facet is identified by an ordinal number in the range 1 - 2,147,483,648. Elements numbers can be omitted or skipped, and the same numbers can be used in different facets, e.g. the judges can be numbered 1,2,5,8 and the examinees 2,7,8,12.

Element number 0 is used when a facet did not participate in an observation, e.g., when multiple-choice question (MCQ) items are combined with essay ratings, judge 0 specifies that the MCQ observation did not require a judge.

- C. The observation, rating, or count. This is a digit in the range 0-255. A missing or negative value implies that this rating is not to be included in the analysis. Since Facets ignores blank ratings, Facform also ignores them, unless instructed to converted them to a non-blank value.

Facets data are recorded by position. For each observations, the element numbers of the facets are entered in order by facet, followed by the data value. An example of a data record is:

2,3,1

which means that the combination of
"2" (element 2 of facet 1) and
"3" (element 3 of facet 3),
resulted in a rating or count of "1".

3.4. Data compression

To lessen the size of its data file, Facets permits two types of compression in its data format:

- A. **Observations in sequence.** Observations are often ordered by element sequence number on one facet, e.g., item number, without alteration to the other facets, e.g., examinee number and judge number. Such observations can be recorded as a range, e.g.,
12,1,3,101,4 ; person 12 on item 1 by judge 101 is rated a "4"

12,2,3,101,3 ; person 12 on item 2 by judge 101 is rated a "3"
12,3,3,101,5 ; person 12 on item 3 by judge 101 is rated a "5"
; item 4 not observed here
12,5,3,101,2 ; person 12 on item 5 by judge 101 is rated a "2"

can be entered in the Facets data file with a range of element numbers on only one facet:
12,1-5,101,4,3,5,,2 ; person 12 on items 1-5 by judge 101,
rated "4","3","5",Missing here,"2"

B. Identical replications. In survey analysis, many identical replications may appear, e.g.,
2,7,2 ; a respondent in ethnic group "2" and age group "7" responded 2
2,7,2 ; another in ethnic group "2" and age group "7" responded 2

2,7,2 ; a 33rd person in ethnic group "2" and age group "7" responded 2

can be entered in the Facets data file as
R33,2,7,2 ; 33 respondents in ethnic group "2" and age group "7" responded 2

C. Identical replications observed in sequence. An element number range and a replication count can be used simultaneously:
R33,2,7,4-6,2,3,1 ; 33 respondents in ethnic group "2" and age group "7" answered items 4,5,6 with "2", "3", "1"

3.5. Facform creates Facets specification and data files from four sources

i) The Facform keyword file, "keyfile".

This contains a list of "keyword=value" statements that Facform uses to convert the input raw data into data formatted for Facets. You are prompted for this when Facform starts.

ii) The unformatted, raw, input data file (\$Input=).

This file contains the data to be converted to Facets format. It is usually a flat file, but many logically organized file structures can be processed by Facform.

iii) Optionally, a previous Facets specification file, if any (\$Spinput=).

As Facform processes the input data, it builds a table of element numbers and names. Facform can use a pre-existing Facets specification file as the basis of this table, adding new element numbers and names to it, as necessary. This means that newly identified elements are not accidentally assigned the same numbers as previous elements. Also previous element anchor values and other Facets specifications are transferred into the newly constructed Facets specification file.

iv) The DOS command line.

C:> FACFORM keyfile outfile keyword=value keyword=value ..

This contains the name, "keyfile", of the keyword file.

It can also contain the name, "outfile", of the output formatted data file.

It can also contain "keyword=value" statements, which take precedence over those stored on disk.

3.6. The data reformatting process takes place in 7 phases

The data reformatting process takes place in 7 phases:

1) The keyword file name is obtained from the DOS command line, or, if none is given, Facform requests one by a screen prompt:

Keyword file name:

Any other keyword commands entered on the DOS command line are also stored.

2) The keyword file is read into memory. Some keywords, such as \$Facets=, are processed immediately. Any \$Spec= instructions are performed, and their contents written to the output specification file.

- 3) The input Facets specification file (if any) is processed. Elements numbers, names and other details (from Labels=) are stored in memory; other specifications are copied into the output specification file.
- 4) The input data file is read and all labels identified. They are matched with or added to the labels already stored in memory. Any \$Text= instructions are performed, and their contents written to the output specification file.
- 5) Ordinal numbers are assigned to any labels lacking them, and Labels= followed by all element information are written to the output specification file.
- 6) The input data file is read through again, and the data converted into Facets-format data records and written to the data output file. Data compression of element ranges and identical replications occurs, where feasible.
- 7) Finally, summary statistics for the data formatting run are displayed on screen.

4. RUNNING Facform

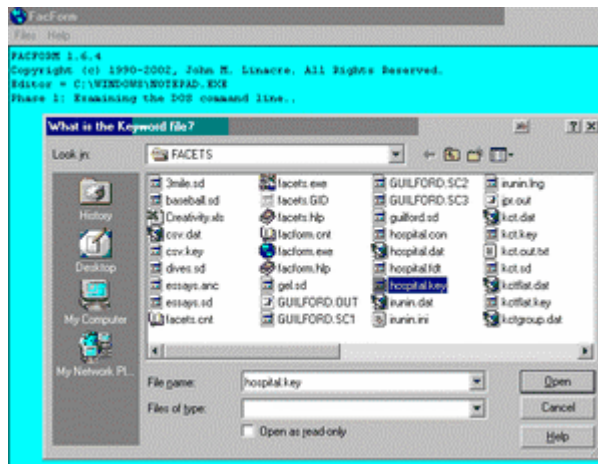
4.1. To start Facform under Windows

Click on *Start* button
 Move mouse pointer to *Programs*
 Point to *Facets* folder

Click on Facform icon and the program will ask you for the name of your keyword file with
Keyword file name:

Type in the name of the keyword file you want, e.g.
facform\KCTkey.txt

The data conversion will commence and run to its conclusion.



4.2. To start Facform under DOS

At the DOS prompt,

Either enter:
 C:> FACFORM

and the program will ask you for the name of your keyword file with
 Keyword file name:

Type in the name of the keyword file you want, e.g.

facform\KCTkey.txt

Or enter:

```
C:> FACFORM keyfile
```

where keyfile is the name of your keyword file, e.g., KCTkey.txt

Or, for advanced users, enter:

```
C:> FACFORM keyfile outputfile keyword=value keyword=value ..
```

where

keyfile is the name of your keyword file,
outputfile (optional) is the name of the file to hold your formatted data,
keyword=value are keywords like those in the keyword file.

The keywords entered here take precedence over those in the keyword file.

For entries on the DOS command line blanks are used as separators. Blanks are not allowed within keywords. This is correct:

```
C:> FACFORM facform\KCTkey.txt KCTDAT.txt $SPOUTPUT=KCTSP.txt
```

But, \$SPOUTPUT= KCTSP.txt is incorrect due to the blank between = and KCTSP.txt

keyword=details represents keyword lines like those included in a keyword file. The keywords given here take priority over those in the keyword file.

To include blanks in values, place them inside quotes.

The following is valid:

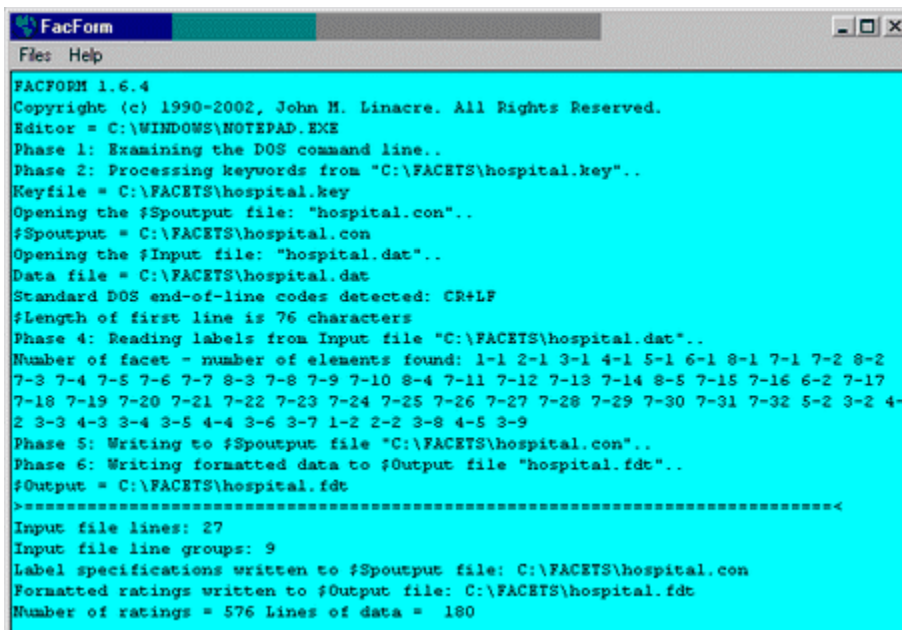
```
C:> FACFORM facform\KCTkey.txt $PRINT="This is displayed"
```

But, not valid due to the extra blanks, is:

```
C:> FACFORM facform\KCTkey.txt $PRINT=This is displayed
```

4.3. Ending Facform

This is how Facform appears on normal termination. For details see: ["While Facform is running."](#)



```
FacForm
Files Help
FACFORM 1.6.4
Copyright (c) 1990-2002, John M. Linacre. All Rights Reserved.
Editor = C:\WINDOWS\notepad.exe
Phase 1: Examining the DOS command line..
Phase 2: Processing keywords from "C:\FACETS\hospital.key"..
Keyfile = C:\FACETS\hospital.key
Opening the $Spoutput file: "hospital.con"..
$Spoutput = C:\FACETS\hospital.con
Opening the $Input file: "hospital.dat"..
Data file = C:\FACETS\hospital.dat
Standard DOS end-of-line codes detected: CR+LF
$Length of first line is 76 characters
Phase 4: Reading labels from Input file "C:\FACETS\hospital.dat"..
Number of facet - number of elements found: 1-1 2-1 3-1 4-1 5-1 6-1 8-1 7-1 7-2 8-2
7-3 7-4 7-5 7-6 7-7 8-3 7-8 7-9 7-10 8-4 7-11 7-12 7-13 7-14 8-5 7-15 7-16 6-2 7-17
7-18 7-19 7-20 7-21 7-22 7-23 7-24 7-25 7-26 7-27 7-28 7-29 7-30 7-31 7-32 5-2 3-2 4-
2 3-3 4-3 3-4 3-5 4-4 3-6 3-7 1-2 2-2 3-8 4-5 3-9
Phase 5: Writing to $Spoutput file "C:\FACETS\hospital.con"..
Phase 6: Writing formatted data to $Output file "hospital.fdt"..
$Output = C:\FACETS\hospital.fdt
>-----<
Input file lines: 27
Input file line groups: 9
Label specifications written to $Spoutput file: C:\FACETS\hospital.con
Formatted ratings written to $Output file: C:\FACETS\hospital.fdt
Number of ratings = 576 Lines of data = 180
```

Facform terminates itself when

- 1) errors are found in the keyword file or in processing.

- 2) the data have been formatted
- 3) you press **Ctrl+C** (the **Ctrl** and the **"C"** keys together) to force immediate termination.

4.4. While Facform is running

After invoking *Facform* with Keyword file, **PHYS.KSP**

Facform reports its operation to the screen

FACFORM Version No. 1.20, October 1, 1992

Copyright (c) 1990-1992, John M. Linacre

Please mention the version number when contacting us with a question about *Facform*.

Phase 1: Examining the DOS command line..

Any file names or keywords on the DOS command line are processed.

Phase 2: Processing keywords from "PHYS.KSP"..

Keywords in the keyword file are scanned and validated.

Opening the \$Spoutput file: "PHYS.FSP"..

The output file for *Facets* specifications is opened.

Phase 3: Reading \$Spinput file "PHYS.SPC"..

The previous file of *Facets* specifications, if any, is processed:

Number of facet - number of elements found:

```

1-1 1-2 1-3 1-4 1-5 1-6 1-7 1-8 1-9 1-10 1-11 1-12 1-13 1-14 1-15 1-16 1-17
1-18 2-1 2-2 2-3 2-4 2-5 2-6 2-7 2-8 2-9 2-10 2-11 2-12 2-13 2-14 2-15 2-16
2-17 2-18 3-1 3-2 3-3 3-4 3-5 3-6 3-7 3-8 3-9 3-10 3-11 3-12 3-13 3-14 3-15
3-16 3-17 3-18 3-19 3-20 3-21 3-22 3-23 4-1 4-2 5-1 5-2 5-3 5-4 5-5 5-6 5-7 5-8
5-9 5-10 5-11 5-12 5-13 5-14 5-15 5-16 5-17 5-18 5-19 5-20 5-21 5-22 5-23 5-24
5-25 5-26 5-27 5-28 5-29 5-30 5-31 5-32 5-33 5-34 5-35 5-36

```

As *Labels=* specifications are processed from a \$Spinput file, or new elements are discovered in a \$Input file, they are added to the array of element numbers and labels. 1-1 means "in facet 1, there is 1 element so far," "5-36" means that "in facet 5, there are 36 elements so far."

Opening the \$Input file: "PHYSDAT.txt"..

The input data file is opened, ready for processing.

Phase 4: Reading labels from Input file "PHYSDAT.txt"..

The input data file is processed once, looking for new element numbers and labels to add to the element array. In this case, there are none to add.

If you know that all element labels are in the \$Spinput file, specify \$Given in the *Facform* keyword file, or on the DOS command line.

Phase 5: Writing to \$Spoutput file "PHYS.FSP"..

Facets specifications are written to the \$Spoutput file

Phase 6: Writing formatted \$Output file "PHYS.DSP"..

>=====<

The data is written to the \$Output file in *Facets* format. The bar-chart shows the progress through this operation.

Input file lines: 4

Input file line groups: 4

Label specifications written to \$Spoutput file: PHYS.FSP

Formatted ratings written to \$Output file: PHYS.DSP

Number of ratings = 144 Lines of data = 8

At the conclusion of the run, summary information is displayed on the screen.

Check that the number of ratings matches what you expect from the \$Input file.

4.5. Facform output to Facets input

Facform output is displayed on your screen. Minor editing is required before Facets is run.

```

FACFORM 1.6.4
Copyright (c) 1990-2002, John M. Linacre. All Rights Reserved.
Editor = C:\WINDOWS\notepad.exe
Phase 1: Read
Phase 2: Read
Keyfile = C:\
Opening the f
Output = C
Opening the f
Data file = C
Standard DGS
Length of f
Phase 4: Read
Number of fac
1=1 2=1 3=1 4=1 5=1
Phase 5: Write
Phase 6: Write
Output = C:\
Input file is
Input file is
Label specific
Formatted rat
Number of rat
2, treatment
4=
5=
3, patient
137161=
143432=
143477=
151430=
152325=
152988=
153333=
207856=
  
```

Typical edits before using the Facets control file are:

- i) Models=
Alter the Models= statement to match the analysis you want to perform.
- ii) Positive=
Positively orient the Facets for which "higher score = higher measure".
- iii) Non-center=
The facet which floats relative to the frame of reference, usually the "object of measurement", is non-centered.

Save this file. It is the Facets control file.

5. EXAMPLES

5.1. A two-facet flat data file

Here is the data for the Knox Cube Test (KCT) from "Best Test Design" (Wright & Stone, 1978). The first 2 columns are the child number. The scored responses on the 18 items (1=correct, 0=incorrect) are in columns 4-21. There are 35 children altogether. These data are in file "KCTFDAT.txt":

```

1 111111100000000000    child 1, 18 dichotomous 0-1 responses
2 111111111110000000
|
| (and so on down to)
|
34 1111111111101010000
35 111000000000000000    Child 35
^ ^
1 4 Columns
  
```

The keyword list (in file KCTFkey.txt, in the *facform* folder) converts the flat file into a *Facets* file:

```
; File: KCTFkey.txt    comment for your information
; This converts a flat data file into a Facets file (Comments start with ;)

$Input = KCTFDAT.txt      ; the flat file
$Output = KCTFFAC.txt    ; the file of Facets-formatted data

$Facets = 2 ; two facets - children and tapping items

; labels for the facets
$Flabel = 1, "Children"
$Flabel = 2, "Tapping items"

; let's output the specifications that Facets needs:
$Spec = "Title = Knox Cube Test"    these are explained in the Facets manual
$Spec = "Output = kctOUT.txt ;the Facets output file"
$Spec = "Models = ?, ?, D"

; person element number is child number in the flat file:
$Label = 1, $S1W2 ; Child number is column 1 of data, width 2

; reset item numbering
Itemnum = 1 ; define variable Itemnum with value 1. This will be the item number.

; now go across the line for 18 items
$Do = 18
  $Label = 2, Itemnum ; Itemnum has the item number
  ; at this point the current element in facet 1 and in facet 2 have been
  ; identified by $Label= keywords
  $Rating = $S(Itemnum+3)W1 ; first response: column 4 width 1.
  ; The reformatted rating is written to the $Output file.
  Itemnum = Itemnum + 1 ; increment for next response
$Again
```

After running the following at the DOS prompt:
C:>**FACFORM KCTFkey.txt**<

The *Facets*-format output file is KCTFFAC.txt:

```
; FACFORM Version No. 1.20, May 3, 1992
; Run on 05-03-1992 21:56:09
; from Keyword file: KCTFLATkey.txt
Facets = 2
Title = Knox Cube Test
Output = kctOUT.txt ;the Facets output file
Models = ?, ?, D
Labels =
1, Children
1=
2=
| (31 more children numbers here)
34=
35=
*
2, Tapping items
1=
2=
|
```

```

17=
18=
*
Data =
1,1-18,1,1,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0 child 1, items 1-18, 18 responses
2,1-18,1,1,1,1,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0
|
(and so on down to)
|
34,1-18,1,1,1,1,1,1,1,1,1,0,1,0,1,0,0,0,0,0,0,0
35,1-18,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0

```

5.2. A three-facet flat data file

Here is the data for an assessment of patient functioning. The performances of ten patients on a specific task were videotaped and rated by a number of raters on 70 items of performance. These data are in file "PATDAT.txt":

```

001BMT67MWP28          5555555555555 555 354545555555551    555 55 555555 35554
001DCS67MWP28          55555 555555 3 555 355445555555553    555 55 555 55 55554
|
(and so on down to)
|
010MMO                555 55555 55555 5 555 555 555 5555 55555 5 5555 5 55555 555 5
010PBG                45554 555554 555554 5 5554 554 555 55555 5555545 555 54 555554 554 5
^ ^                    ^
1 4                    31 Columns

```

The keyword list (in file PATIENTkey.txt) converts the flat file into a *Facets* file:

```

; File: PATkey.txt
; This converts a flat data file into a Facets file (Comments start with ;)

$Input = PATDAT.txt          ; the flat file
$Spoutput=PATCON.txt        ; the file for Facets specifications
$Output = PATFAC.txt         ; the file of Facets-formatted data

; three facets
$Facets=3
; labels for the facets
$Flabel=1,clients
$Flabel=2,raters
$Flabel=3,tasks

; client id number
$Label=1,$S1E3              client number starts in column 1, ends in column 3
; rater label (non-numeric)
$Label=2,,$S4E6            rater "name" starts in column 4, ends in column 6

; count of task number
task=0                      a user-defined variable to keep track of task number
; location in data record
pointer=30                  a user-defined variable to keep track of location in record

; 70 tasks
$Do=70                      perform the $Do-$Again loop 70 times
task=task+1                 increment our "task" number
pointer=pointer+1           increment location in record
; current task number
$Label=3,task               we want this to be the task element number

```


Each line of data file has the following description:

<u>Column</u>	<u>Facet</u>	<u>Description</u>
2-3	Rater	Number
5		Training group
7-8	Activity Code	
10		Test-retest code
12		Videotaped
14		Environment type
15		Environment familiarity
17-19	Subject Number	
21-22		Age
23		Sex
24		Race
26-45	Process	Subtest of 20 items (numbered 1-20)
47-62	Motor	Subtest of 16 items (numbered 21-36)

The keyword commands required to reformat this for *Facets* are in file PHYSkey.txt:

```

; File: PHYSkey.txt
; This converts a many-facet flat file to Facets format
$Input = PHYSDAT.txt
$Spoutput = PHYSFAC.txt ; Facets specifications
$Output = PHYSFDT.txt ; Facets data

$Facets = 5 ;rater, activity, subject,subtest,item

; Output specification lines for FACETS:
$Spec = "Title = Physical Capability"
$Spec = "Positive = 3 ; subject ability"
$Spec = "Noncenter = 1 ; raters float"
$Spec = "Output = PHYSOUT.txt ; the FACETS output file"
$Spec = "Models = ?,?,?,?,R ; the Rating (or partial credit) scale"

; Define the Facet labels
$Flabel = 1,Rater
$Flabel = 2,Activity
$Flabel = 3,Subject
$Flabel = 4,Subtest
$Flabel = 5,Item

; Define the element labels in terms of where they are in the data records:

; For facet 1, the label is rater number and group
$Label = 1,,$S2E3+ " G"+$S5E5

; For facet 2, the label contains all activity-related information:
$Label = 2,,$S7E8+ " T"+$S10W1+ " V"+$S12W1+ " E"+$S14W1+ " F"+$S15W1

; For facet 3, the label contains all subject-related information:
$Label = 3,,$S17E19+ " "+$S21E22+ " S"+$S23W1+ " R"+$S24W1

; For facet 4, identify the first subtest:
$Label = 4,,"Process"

```

```

;Reset the item number variable
Item = 0
; Use variable pointer, to point to position of the next rating:
Pointer = 26
; Go down the 20 items of sub-test 1.
$Do = 20
    ; Increment to the next item number
    Item = Item + 1
    ; Facet 5 is the item number
    $Label = 5, Item
    $Rating = $S(Pointer)W1
    ; Increment to next position in the data record
    Pointer = Pointer + 1
$Again

; Now facet 4 is the second subtest
$Label = 4,,"Motor"

; set variable Pointer to first rating of second subtest in data record.
Pointer = 47

; we want to reformat the 16 items of the second subtest
$Do = 16
    ; Increment to the next item number
    Item = Item + 1
    ; Here is another way: save the rating as variable Response.
    Response = $S(Pointer)W1
    $Label = 5, Item, Item
    $Rating = Response
    Pointer = Pointer + 1
$Again

```

The resulting *Facets* specification file is PHYSFAC.txt:

```

; FACFORM Version No. 1.20, May 4, 1992
; Run on 05-04-1992 11:50:28
; from Keyword file: PHYSkey.txt
Facets = 5
Data file = PHYSFDT.txt
Title = Physical Capability
Positive = 3 ; subject ability
Noncenter = 1 ; raters float
Output = PHYSOUT.txt ; the FACETS output file
Models = ?,?,?,?,R ; the Rating (or partial credit) scale
Labels =
1,Rater
1=1 G1
2=10 G1
3=18 G2
4=2 G1
*
2,Activity
1=13 T1 V1 E1 F1
2=6 T1 V1 E1 F1
3=9 T1 V1 E2 F1
*
3,Subject
1=1 67 S1 R1
2=3 67 S1 R1

```

```

*
4,Subtest
1=Motor
2=Process
*
5,Item
1=
2=
|
35=
36=
*

```

The resulting *Facets* specification data file is PHYSFDT.txt:
Facform has combined observations in sequence by element number into single data records

```

1,3,2,2,1-20,4,4,3,3,4,4,4,4,4,4,2,3,2,3,4,2,3,3,4
1,3,2,1,21-36,2,3,3,2,,2,3,2,2,2,3,3,2,3,3
4,1,1,2,1-20,3,3,4,,4,3,4,4,,2,4,4,4,4,4,4,4,4,4
4,1,1,1,21-36,4,4,3,3,,3,3,4,4,3,4,4,4,4,4
2,2,1,2,1-20,4,,,4,4,4,4,,4,4,4,4,4,4,4,4,3,4
2,2,1,1,21-36,4,4,4,4,,4,4,3,4,4,4,4,4,4,4,4
3,2,1,2,1-20,4,4,4,4,4,4,4,4,,4,4,4,4,4,4,4,2,4
3,2,1,1,21-36,4,4,4,4,,4,4,4,4,4,4,4,4,4,3,4

```

5.5. A many-facet flat file with multiple raters per line

Here are the ratings of creativity of junior scientists by three senior scientists (Guilford). The file is in a standard MS-DOS file called "GUILFDAT.txt":

```

Anne   3 7 7 5 7 5 2 8 7 5 7 2 1 9 7 5 8 5
Chris  1 3 3 3 7 1 3 3 5 5 5 5 2 4 5 3 6 6
David  3 5 3 3 3 1 2 5 6 4 5 5 1 7 3 1 3 3
Edward 2 2 4 3 2 3 3 9 7 7 7 7 1 9 7 7 8 5
Fred   3 3 3 3 5 3 1 3 5 3 5 1 2 4 4 6 4 2
George 2 3 3 5 5 4 1 7 7 5 5 5 3 7 7 7 5 7
^      ^ ^      ^      ^
Col.   1      8 10      20      32

```

Each line of data file has the following description:

<u>Column</u>	<u>Facet</u>	<u>Description</u>
1-6		Subject Junior scientist's name
8	Rater	Rater number (1-3)
10-18	Item	Ratings of items 1-5
20	Rater	Rater number (1-3)
22-30	Item	Ratings of items 1-5
32	Rater	Rater number (1-3)
34-42	Item	Ratings of items 1-5

The keyword commands required to reformat this for *Facets* are in file GUILFkey.txt:

```

; File: GUILFkey.txt
; This converts a many-facet flat file with multiple raters per line to Facets format
$Input = GUILFDAT.txt

```

```

$Output = GUILFFAC.txt          ; Facets specifications and data

$Facets = 3                      ;subject, rater, item

; Output specification lines for FACETS:
$Spec = "Title = Creativity"
$Spec = "Positive = 1 ; subject ability"
$Spec = "Noncenter = 1 ; subjects float"
$Spec = "Output = GUILFOUT.txt ; the FACETS output file"
    $Spec = "Models = ?,?,?,R ; the Rating (or partial credit) scale"

; Define the Facet labels
$Flabel = 1,"Junior scientist"
$Flabel = 2,"Senior scientist"
$Flabel = 3,Item

; Define the element labels in terms of where they are in the data records:

; For facet 1, the label is the subject name
$Label = 1,,$S1E6

; Set up pointer in record to first set of ratings
Pointer = 8
; Extract three sets of ratings
$Do = 3
    ; For facet 2, the rater id is in the record
    $Label = 2,$S(Pointer)W1

    ; Set up item number reference
    Item = 1
    ; Extract 5 item ratings
    $Do = 5
        ; Reference the item number
        $Label = 3,Item
        ; Extract the rating - compute its position
        $Rating = $S(Pointer+(Item*2))W1
        ; Advance to next rating
        Item = Item + 1
    ; Do ratings 5 times
    $Again

    ; Advance to next rater
    Pointer = Pointer + 12
; Do this 3 times
$Again

```

The resulting *Facets* specification file is GUILFFAC.txt:

```

; FACFORM Version No. 1.24
; Run on 10-05-1994 12:05:22
; from Keyword file: GUILFkey.txt
Facets = 3
Title = Creativity
Positive = 1 ; subject ability
Noncenter = 1 ; subjects float
Output = GUILFOUT.txt ; the FACETS output file
Models = ?,?,?,R ; the Rating (or partial credit) scale
Labels =
1,Junior scientist

```

```

1=Anne
|
*
Data =
1,1,1-5,5,5,3,5,3
|
7,3,1-5,7,7,5,7

```

5.6. Data file using existing Facets specification file

Again using the PHYSDAT.txt data, but with the following pre-existing specification file, PHYSSPC.txt

```

Title=Physical Capability
data=PHYSrfd.txt
output=PHYSOUT.txt
convergence=100
unexpected=3.0
xtreme=0.5,0.5
facets=5
noncenter=3; subject
positive=1,2,3,4,5
models=
?,?,?,1,?,R ;process only
*
labels=
1,Rater,G
1=Nancy
2=Pat G
|
17=Stacy
18=Chris H
*
2,Activity,G
1=juice/cer
2=toast/brew
|
18=fruit sld
19=grillches
*
3,Subject
1=67 WM
2=78 WM
|
22=83 WF
23=71 WF
*
4,Subtest,G
1=process
2=motor
*
5,Item,G
1=accommodat
2=adjusts
|
35=calibrate
36=grips
*

```

The *Facform* keyword file is PHYSKSP.txt

```

; File: PHYSKSP.txt
; This converts a file with pre-existing Facets specification file
$Input = PHYSDAT.txt
$Output = PHYSDSP.txt
$Spoutput = PHYSFSP.txt
$Spinput = PHYSSPC.txt ; the previous Facets specification file

$Facets = 5 ;rater,activity,subject,subtest,item

; the data file contains the rater number
$Label = 1,$S2E3 ;rater number

; the data file contains the activity number
$Label = 2,$S7E8 ;activity number

; the data file contains the subject number
$Label = 3,$S17E19 ;

; Reset the item number
Item = 0
; for facet 4, let's match the label
$Label = 4,,"process" ;first subtest
; point to the first item response
Pointer = 26

; there are 20 items in the process sub-test
$Do = 20
;Increment item number
Item = Item + 1
;Construct item label
$Label = 5,Item
;Blank responses are to be coded as "4"s
$If = ($S(Pointer)W1 = " ")
$Rating = "4" ; blank rating reported as 4
$Else
$Rating = $S(Pointer)W1 ; non-blanks have own value
$Endif
; Increment position for next item
Pointer = Pointer + 1 ;increment variables for next repetition
$Again

;Motor sub-test
$Label = 4,,"motor" ;second subtest
Pointer = 47
; there are 16 items in the motor sub-test
$Do = 16
;Increment item number
Item = Item + 1
;Construct item label
$Label = 5,Item
;Let's do this the other way - pick up the rating
Rating = $S(Pointer)W1
;Blank responses are to be coded as "4"s
$If = (Rating = " ")
$Rating = "4"
$Else
$Rating = Rating
$Endif

```

```

; Increment position for next item
Pointer = Pointer + 1 ;increment variables for next repetition
$Again

```

The output specification file, PHYSFSP.txt, is:

```

; FACFORM Version No. 1.20, May 4, 1992
; Run on 05-04-1992 12:26:54
; from Keyword file: PHYSKSP.txt
Facets = 5
Title=Physical Capability
output=PHYSOUT.txt
convergence=100
unexpected=3.0
xtreme=0.5,0.5
noncenter=3; subject
positive=1,2,3,4,5
models=
?,?,?,1,?,R ;process only
*

Data file = PHYSDSP.txt
Labels =
1,Rater,G
1=Nancy
2=Pat G
|
17=Stacy
18=Chris H
*

2,Activity,G
1=juice/cer
2=toast/brew
|
18=fruit sld
19=grill cheese
*

3,Subject
1=67 WM
2=78 WM
|
22=83 WF
23=71 WF
*

4,Subtest,G
1=process
2=motor
*

5,Item,G
1=accommodate
2=adjusts
|
35=calibrate
36=grips
*

```

The Facets-formatted data file, PHYSDSP.txt, is:

```

1,9,3,1,1-20,4,4,3,3,4,4,4,4,4,4,2,3,2,3,4,2,3,3,4
1,9,3,2,21-36,2,3,3,2,4,2,3,2,2,2,2,3,3,2,3,3
2,13,1,1,1-20,3,3,4,4,4,3,4,4,4,2,4,4,4,4,4,4,4,4

```

|
18,6,1,2,21-36,4,4,4,4,4,4,4,4,4,4,4,4,4,4,3,4

5.7. Flat file with multiple lines per record

When there are many observations for each subject, they are often arranged in multiple lines per subject. In this case, there are 10 observations per subject. 5 on the first line, and 5 on a subsequent line.

This is the data file, TWODAT.txt:

```
Elizabeth  11010
           01011  AXBC
Henry      01101
           11010  DSGH  < The final letter codes are ignored for this analysis
^         ^         ^
1         6         12  Column
```

The *Facform* keyword file is TWOkey.txt

```
; File: TWOkey.txt
; This converts a multiple line per record file
$Input = TWODAT.txt
$Output = TWOFAC.txt

$Facets=2
$Flabel=1,Examinee
$Flabel=2,Item

$Label=1,,$S1E10 ; the examinee name

; Get items from the first line
Itemnum = 1
Pointer = 12
$DO=5 ; go round the 5 items on the first line
    $Label = 2,Itemnum
    $Rating = $S(Pointer)W1
    Itemnum = Itemnum + 1
    Pointer = Pointer + 1
$AGAIN

; Get items from the second line
$Nextline ; advance to the next line
Itemnum = 6
Pointer = 6
$DO=5 ; go round the 5 items on the second line
    $Label = 2,Itemnum
    $Rating = $S(Pointer)W1
    Itemnum = Itemnum + 1
    Pointer = Pointer + 1
$AGAIN
```

The resulting *Facets* file is TWOFAC.txt:

```
; FACFORM Version No. 1.20, May 4, 1992
; Run on 05-04-1992 20:21:11
; from Keyword file: TWOkey.txt
Facets = 2
Labels =
1,Examinee
```

```

1=Elizabeth
2=Henry
*
2,Item
1=
|
10=
*
Data =
1,1-10,1,1,0,1,0,0,1,0,1,1
2,1-10,0,1,1,0,1,1,1,0,1,0

```

5.8. Fixed length data file with multiple records per line

Data files transferred to PC's from Mainframe computers are frequently in a fixed-length format with no "Carriage Return - Line Feed" indicators at the end of the records. For these files, each record contains a fixed number of characters, its "record length".

Here is such a file. I have broken it into lines of twice the "record length" for the sake of convenient display, but the file itself is just one long string of numbers.

This is data file, TAPEDAT.txt:

```

Fred 2625232032131000110110George 2925202423320100110111
Mabel 2622212643411001000003Alice 2923212421321010010011
Grace 2723262424311111011012Arthur 2523222222140001010111
Mary 2825212412431100000003Belinda 2627222431320111011111

```

The test is of 10 items, divided into 4 groups. Group 1 is items 1,2,3. Group 2 is items 4,5,6. Group 3 is items 7,8,9. Group 4 is just item 10. The judging plan rotates the judges across subjects, item groups and sessions in a random manner, but without duplication of rating. Thus the items in group one for subject Fred were rated by judge 26 in session 3. Item 1 was rated 1, Items 2 and 3 were rated 0. The first 9 items are dichotomies (0=failed, 1= passed). Item 10 is on a four-point Likert scale scored 0, 1, 2 ,3. Schematically the data record is:

This is entered in each 30 character record as follows:

Columns

```

1-8   Subject
9-10  Judge for item group 1
11-12 Judge for item group 2
13-14 Judge for item group 3
15-16 Judge for item group 4
17    Session number for item group 1
18    Session number for item group 2
19    Session number for item group 3
20    Session number for item group 4
21-30 Ratings for Items 1-10

```

The *Facform* keyword file, TAPEkey.txt, to reformat this fixed length file is:

```

; File: TAPEkey.txt
; This converts a file with multiple records per data line
$Input = tapeDAT.txt
$Length = 30 ; the fixed length record length
$Output = tapeFAC.txt

$Facets = 5 ;Subject, Item group, Judge, Session, Item
$Flabel = 1, Subject
$Flabel = 2, "Item Group" ; This facet is included "just in case"

```

```

$Flabel = 3, Judge
$Flabel = 4, Session
$Flabel = 5, Item

```

```

$Spec = "Title = Demonstration of Fixed Length data file"
$Spec = "Output = tapeOUT.txt ; the output of the FACETS analysis"
$Spec = "Models =      ; we model only subject-judge-item here"
$Spec = "?,,?,10,R    ;Item 10 has a rating (or partial credit) scale"
$Spec = "?,,?,?,D    ;Items 1-9 dichotomies"
$Spec = "*"

```

```

$Label = 1,,$S1W8                                ;Subject name
lg = 0 ;used to keep track of item group
Item = 0 ; reset item number
$Do = 4
  lg = lg + 1
  $Label = 2,lg                                    ;Item group
  $Label = 3,$S(IG*2+7)W2 ;Judge
  $Label = 4,$S(IG+16)W1 ;Session
  $Do = 3;Items in groups of 3
  Item = Item + 1
  $If = (Item=<10)                                ;No item greater than 10
    $Label = 5,Item,Item ;Item
    $Rating = $S(Item+20)W1 ;Rating
  $Endif
$Again
$Again

```

The resulting *Facets* file is TAPEFAC.txt:

```

; FACFORM Version No. 1.20, May 4, 1992
; Run on 05-04-1992 12:44:12
; from Keyword file: TAPEkey.txt
Facets = 5
Title = Demonstration of Fixed Length data file
Output = tapeOUT.txt ; the output of the FACETS analysis
Models =      ; we model only subject-judge-item here
?,,?,10,R    ;Item 10 has a rating (or partial credit) scale
?,,?,?,D    ;Items 1-9 dichotomies
*
Labels =
1,Subject
1=Alice
2=Arthur
|
7=Mabel
8=Mary
*
2,Item Group
1=
2=
3=
4=
*
3,Judge
20=
21=

```

|

```

28=
29=
*
4,Session
1=
2=
3=
4=
*
5,Item
1=
2=
|
9=
10=
*
Data =
4,1,26,3,1-3,1,0,0
4,2,25,2,4-6,0,1,1
4,3,23,3,7-9,0,1,1
4,4,20,3,10,0
5,1,29,2,1-3,0,1,0
|
8,4,24,3,10,3
3,1,26,3,1-3,0,1,1
3,2,27,1,4-6,1,0,1
3,3,22,3,7-9,1,1,1
3,4,24,2,10,1

```

5.9. Comma-separated Value (CSV) data file

Data files prepared by BASIC programs or Spreadsheets frequently have their data values separated by commas. Each field, and each record, can be of a different length.

This is such a data file, CSVDAT.txt:

```

"Mike",1,2,3,4,5,6,7,8,9,1,2,3,4,5,6,9,8,7,6,5,4
"George",3,6,4,2,3,5,2,3,4
"Anne",7,4,3,2,8,5,4,1,,3,4,5

```

The test is of 21 items. Each examinee started at item 1, but stopped when time ran out.

The *Facform* keyword file, CSVkey.txt, to reformat this CSV file is:

```

; File: CSVkey.txt
; This converts a comma-separated value file
$Input = CSVDAT.txt
$Output = CSVFDT.txt

; if the separators are not commas, enter the separator code
;$SEP=";" ; separators are semi-colons
; for Tab separators, use $T

$Facets = 2 ; children and item

; labels for the Facets
$Flabel = 1, "Examinees"
$Flabel = 2, "Binomial Trial Items"

```

```

; and let's also output the other specifications that FACETS needs:
  $Spec = "Title = Nine Binomial Trials"
$Spec = "Output = csvOUT.txt ;the FACETS output file"
$Spec = "Models = ?,?,B9,1"

; the person label
$Label = 1,,$C1    ; first comma separated value

; reset item numbering
Item = 1           ; point to first item
Datum = $C(Item+1) ; first rating in 2nd CSV field
; now go across the line until the data runs out
$Do = (Datum<>"") ; perform $Do= while ratings exist
; the ordinal number of this item
  $Label = 2,Item
  $Rating = Datum
  Item = Item + 1 ; increment to next item
  Datum = $C(Item+1) ; observation into variable called "Datum"
$Again

```

The resulting *Facets* specification file, CSVFDT.txt, is:

```

; FACFORM Version No. 1.20, May 4, 1992
; Run on 05-04-1992 19:37:33
; from Keyword file: CSVkey.txt
Facets = 2
Title = Nine Binomial Trials
Output = csvOUT.txt ;the FACETS output file
Models = ?,?,B9,1
Labels =
1,Examinees
1=Anne
2=George
3=Mike
*
2,Binomial Trial Items
1=
2=
|
20=
21=
*
Data =
3,1-21,1,2,3,4,5,6,7,8,9,1,2,3,4,5,6,9,8,7,6,5,4
2,1-9,3,6,4,2,3,5,2,3,4
1,1-8,7,4,3,2,8,5,4,1

```

5.10. Data file with grouped elements

It is often convenient to group elements in a facet for reporting or anchoring.

This is such a data file, KCTGDAT.txt:

```

1 111111100000000000 Richard M
2 1111111111100000000 Tracie F
3 111111111001000000 Walter M
4 111100101000000000 Blaise M

```

This data file, of the KCT test, has the examinees' sex specified by M or F in column 31. The sex is to be used as an element of the "Sex" facet, and also to group the examinees.

The *Facform* keyword file, KCTGkey.txt, to reformat this CSV file is:

```
; File: KCTGkey.txt
; This converts a "flat" data file into a Facets file (Comments start with ;)
; We want to group anchor on child Sex

$Input = KCTGDAT.txt           ; the "flat" file
$Output = KCTGFAC.txt         ; the file of Facets-formatted data

$Facets = 3                   ; three facets - children and tapping items
                               ; and gender/sex

; labels for the facets
$Flabel = 1, "Children,G"     ; Children will be group anchored
$Flabel = 2, "Tapping items"
$Flabel = 3, "Sex"

; let's output the specifications that Facets needs:
$Spec = "Title = Knox Cube Test"
$Spec = "Output = kctgOUT.txt ;the Facets output file"
$Spec = "Models = ?, ?, ?, D"
$Spec = "Positive=1,3        ; children & sex positively oriented"
$Spec = "Noncenter=3        ; sex facets floats"

; child's sex - for element in facet 3 and group in facet 1:

Sex=$S31W1                    ; M or F

$If=(Sex="F")
  Group=1                      ; Females are group 1
$Elseif=(Sex="M")
  Group=2                      ; Males are group 2
$Endif

$Label = 3,Group,Sex          ; Group facet

; person element number is child number in the "flat" file:
$Label = 1, $S1W2,$S23W9+",0,"+$Q(Group) ; Child number is column 1, width 2

                               ; reset item numbering
Itemnum = 1                    ; define variable Itemnum with value 1.

                               ; This will be the item
number.

; now go across the line for 18 items
$Do = 18
  $Label = 2, Itemnum          ; Itemnum has the item number
                               ; at this point the current element in facet 1 and in facet 2 have been
                               ; identified by $Label= keywords
  $Rating = $S(Itemnum+3)W1    ; first response: column 4 width 1.
                               ; The reformatted rating is written to the $Output file.
  Itemnum = Itemnum + 1       ; increment for next response
$Again
```

The resulting *Facets* specification file, KCTGFAC.txt, is:

```

; FACFORM Version No. 1.23
; Run on 05-25-1993 11:42:24
; from Keyword file: KCTGkey.txt
Facets = 3
Title = Knox Cube Test
Output = kctOUT.txt ;the Facets output file
Models =?,?,?,D
Positive=1,3 ; children & sex positively oriented
Noncenter=3 ; sex facets floats
Labels =
1,Children,G
1=Richard M,0,2
2=Tracie F,0,1
3=Walter M,0,2
4=Blaise M,0,2
5=Ron M,0,2
6=William M,0,2
7=Susan F,0,1
8=Linda F,0,1
9=Kim F,0,1
10=Carol F,0,1
11=Pete M,0,2
12=Brenda F,0,1
13=Mike M,0,2
14=Zula F,0,1
15=Frank M,0,2
16=Dorothy F,0,1
17=Rod M,0,2
18=Britton F,0,1
19=Janet F,0,1
20=David M,0,2
21=Thomas M,0,2
22=Betty F,0,1
23=Bert M,0,2
24=Rick M,0,2
25=Don M,0,2
26=Barbara F,0,1
27=Adam M,0,2
28=Audrey F,0,1
29=Anne F,0,1
30=Lisa F,0,1
31=James M,0,2
32=Joe M,0,2
33=Martha F,0,1
34=Elsie F,0,1
35=Helen F,0,1
*
2,Tapping items
1=
2=
3=
4=
5=
6=
7=
8=
9=
10=
11=

```



```

$label=5,,$S47E48 ;rater code
type=1 ;activity
$label=6,type,"activity"
item=1
column=49 ;starting column

;activity 1&2
$label=8,1 ; subdomain 1
$do=2
  $label=7,item,,""+$Q(type) ;flag item as in activity type (no anchor value)
  $rating=$S(column)W1
  item=item+1
  column=column+1
$again

;activity 3-7
$label=8,2 ;subdomain 2
$do=5
  $label=7,item,,""+$Q(type) ;flag item as in activity type
  $rating=$S(column)W1
  item=item+1
  column=column+1
$again

;activity 8-10
$label=8,3 ;subdomain 3
$do=3
  $label=7,item,,""+$Q(type) ;flag item as in activity type
  $rating=$S(column)W1
  item=item+1
  column=column+1
$again

;activity 11-14
$label=8,4 ;subdomain 4
$do=4
  $label=7,item,,""+$Q(type) ;flag item as in activity type
  $rating=$S(column)W1
  item=item+1
  column=column+1
$again

;activity 15-16
$label=8,5 ;subdomain 5
$do=2
  $label=7,item,,""+$Q(type) ;flag item as in activity type
  $rating=$S(column)W1
  item=item+1
  column=column+1
$again

; now to next line
$nextline

; Rater 1
type=2 ;memory
$label=6,type,"memory"
item=17 ;not needed but to remind us 16 independent + 16 professional
column=1 ;starting column

```

```
;memory 1&2
$label=8,1 ;subdomain 1
$do=2
  $label=7,item,,""+$Q(type) ;flag item as in memory type
  $rating=$S(column)W1
  item=item+1
  column=column+1
$again
```

```
;memory 3-7
$label=8,2 ;subdomain 2
$do=5
  $label=7,item,,""+$Q(type) ;flag item as in memory type
  $rating=$S(column)W1
  item=item+1
  column=column+1
$again
```

```
;memory 8-10
$label=8,3 ;subdomain 3
$do=3
  $label=7,item,,""+$Q(type) ;flag item as in memory type
  $rating=$S(column)W1
  item=item+1
  column=column+1
$again
```

```
;memory 11-14
$label=8,4 ;subdomain 4
$do=4
  $label=7,item,,""+$Q(type) ;flag item as in memory type
  $rating=$S(column)W1
  item=item+1
  column=column+1
$again
```

```
;memory 15-16
$label=8,5 ;subdomain 5
$do=2
  $label=7,item,,""+$Q(type) ;flag item as in memory type
  $rating=$S(column)W1
  item=item+1
  column=column+1
$again
```

```
; Rater 2
$label=5,,$S27E28 ;rater code
```

```
type=1 ;activity
$label=6,type,"activity"
item=1
column=29 ;starting column
;activity 1&2
$label=8,1 ;subdomain 1
$do=2
  $label=7,item,,""+$Q(type) ;flag item as in activity type
  $rating=$S(column)W1
  item=item+1
  column=column+1
```

\$again

```
;activity 3-7
$label=8,2 ;subdomain 2
$do=5
  $label=7,item,,""+$Q(type) ;flag item as in activity type
  $rating=$S(column)W1
  item=item+1
  column=column+1
$again
```

```
;activity 8-10
$label=8,3 ;subdomain 3
$do=3
  $label=7,item,,""+$Q(type) ;flag item as in activity type
  $rating=$S(column)W1
  item=item+1
  column=column+1
$again
```

```
;activity 11-14
$label=8,4 ;subdomain 4
$do=4
  $label=7,item,,""+$Q(type) ;flag item as in activity type
  $rating=$S(column)W1
  item=item+1
  column=column+1
$again
```

```
;activity 15-16
$label=8,5 ;subdomain 5
$do=2
  $label=7,item,,""+$Q(type) ;flag item as in activity type
  $rating=$S(column)W1
  item=item+1
  column=column+1
$again
```

```
; for Rater 2
type=2 ;memory
$label=6,type,"memory"
item=17
column=55 ;starting column
;memory 1&2
$label=8,1 ; subdomain 1
$do=2
  $label=7,item,,""+$Q(type) ;flag item as in memory type
  $rating=$S(column)W1
  item=item+1
  column=column+1
$again
```

```
;memory 3-7
$label=8,2 ;subdomain 2
$do=5
  $label=7,item,,""+$Q(type) ;flag item as in memory type
  $rating=$S(column)W1
  item=item+1
  column=column+1
```

\$again

```
;memory 8-10
$label=8,3 ;subdomain 3
$do=3
  $label=7,item,,""+$Q(type) ;flag item as in memory type
  $rating=$S(column)W1
  item=item+1
  column=column+1
$again
```

```
;memory 11-14
$label=8,4 ;subdomain 4
$do=4
  $label=7,item,,""+$Q(type) ;flag item as in memory type
  $rating=$S(column)W1
  item=item+1
  column=column+1
$again
```

```
;memory 15-16
$label=8,5 ;subdomain 5
$do=2
  $label=7,item,,""+$Q(type) ;flag item as in memory type
  $rating=$S(column)W1
  item=item+1
  column=column+1
$again
```

```
;now skip line 3
$nextline
```

```
;ok - FACFORM control finished.
;be sure to edit the models= statement etc in the FACETS control file
```

The *Facets* control file, HOSPCON.txt:

```
; FACFORM Version No. 1.38
; Run on 06-10-1997 11:12:11
; from Keyword file: HOSPCON.txt
Facets = 8
Data file = hospFDT.txt
Models =
?, ?, ?, ?, ?, ?, ?, R
*
; Positive = 1
; Noncenter = 1
Labels =
1,hospital
1=RUSH
2=UCMC
*
2,treatment
4=
5=
*
3,patient
137161=
143432=
```

143477=
 151430=
 152325=
 152988=
 153333=
 207856=
 217997=
 *
 4,birth
 82=
 83=
 84=
 85=
 86=
 *
 5,rater
 1=BF
 2=SS
 *
 6,type
 1=activity
 2=memory
 *
 7,item
 1=,,1
 2=,,1
 |
 16=,,1
 17=,,2
 |
 32=,,2
 *
 8,subdomain
 1=
 |
 5=
 *

and the formatted data is HOSPFDT.txt:

2,5,143477,86,1,1,1-2,1,3,3
 2,5,143477,86,1,1,3-7,2,3,3,3,3,3
 2,5,143477,86,1,1,8-10,3,3,3,3
 2,5,143477,86,1,1,11-14,4,3,3,3,3
 2,5,143477,86,1,1,15-16,5,3,3
 2,5,143477,86,1,2,17-18,1,3,2
 2,5,143477,86,1,2,19-23,2,3,1,3,3,1
 2,5,143477,86,1,2,24-26,3,1,0,3
 2,5,143477,86,1,2,27-30,4,1,0,0,0
 2,5,143477,86,1,2,31-32,5,2,2
 2,5,143477,86,2,1,1-2,1,3,3
 2,5,143477,86,2,1,3-7,2,3,3,3,3,3
 2,5,143477,86,2,1,8-10,3,3,3,3
 2,5,143477,86,2,1,11-14,4,3,3,3,3
 2,5,143477,86,2,1,15-16,5,3,3
 2,5,143477,86,2,2,17-18,1,3,1
 2,5,143477,86,2,2,19-23,2,3,0,3,3,1
 2,5,143477,86,2,2,24-26,3,2,2,3
 2,5,143477,86,2,2,27-30,4,1,0,0,1
 2,5,143477,86,2,2,31-32,5,1,2

6. THE KEYWORD LANGUAGE

6.1. Keywords

The operation of *Facform* is directed by a file of keywords and values. These comprise a simple data manipulation language that can convert flat files into *Facets* files.

You specify a list of keywords. The list of keywords is processed once for each line, or group of lines, in the input file of raw data.

Keywords are entered in a keyword file. They can also be entered on the DOS command line.

Only as many letters need to be typed as to make the keyword unique. Never more than 4 letters are need, e.g. \$IN is the same as \$INPUT.

Keywords can be in upper or lower case. \$Input is the same as \$INPUT.

6.2. File definition keywords

\$Input = name (input data file)
The name of the input flat file. This file has your original data.
E.g., \$Input=KCT.RAW

\$Output = name (*Facets*-format data)
The name of the output *Facets* data file. This file has your data transformed into *Facets* format. E.g.,
\$Output=KCTDAT.txt

\$Spoutput = name (*Facets* specifications)
The name of the output *Facets* keyword file. This has most of the specifications *Facets* needs to analyze the data,, but it is incomplete, and will need further editing.
E.g., \$Spoutput=KCT.SP

\$Spinput = name (previous *Facets* specifications)
The name of an input specification file in *Facets* format. *Facform* uses this to assign numbers to elements. The input specifications will be copied into the output specification file. E.g., \$Spinput=KCTBANK
Use this to insure that elements are always assigned the same numbers, and to carry across anchor values, model specifications etc.

\$Given = Yes (all labels given in \$Spinput file)
This keyword instructs *Facform* that all elements of every facet have been specified in the \$Spinput= file so there is no need to analyze the data file for new ones.

This keyword can speed up subsequent runs of *Facform* when specifying the *Facets* specifications produced by an earlier *Facform* run as a \$Spinput file. E.g.,

\$Spinput=MEASURES specifications from earlier run
\$Given=Yes those specifications include all elements

\$Separator = "value" (separator character in a variable length or CSV file)
This keyword defines the character that separates fields in the raw data file. Use it with the \$C() expression. For Tab separators, use the \$T() expression or \$Sep=Tab.
\$Sep=";" ; separators are semicolons
\$Sep=" " ; separators are one or more spaces.

6.3. Data definition keywords

\$Facets = number (number of facets in data)

The number of facets which interact to make up each data observation.
e.g., when there are examinees, items and judges, then \$Facets = 3

\$Flabel = number,name [,codes] (name of facet)
The number of a facet and its label in text format, e.g., \$Flabel = 2, Subjects
You may optionally append control codes to be transferred to *Facets*,

e.g., to specify that this is an anchored facet, \$Flabel = 1, Judges, A

\$Length = 0 (standard DOS input data lines)
Each line in the input data file ends with standard DOS Carriage Return-Line Feed codes. This is the usual format for DOS and ASCII files. It is the default.

\$Length = number (fixed length data lines)
Each line is of the same fixed length, (number) bytes long. This is sometimes required for files imported from other computer systems in fixed-length record format. E.g., \$Length = 100, for records 100 characters long. Sometimes there are "invisible" codes, particularly at the end of each line. Adjust the record length up or down one or two to allow for them. For instance, there are two invisible codes, "Carriage Return" and "Line Feed" at the end of each line in a DOS text file.

To verify that your \$Length= instruction is correct, use a short keyword file such as:

```
$Input=lengthDAT.txt      your data file
$Output=testOUT.txt      a dummy output file
$Length=20               your best data file
$Facets=2                a dummy number of facets
$Print=$s1w5            the first five characters of each data record display on your screen.
Check that the 5 characters displayed on your screen are the first five characters of each record. If not, the $Length= value needs adjustment.
```

\$Label = facet number [, element number] [, element name]
[, "anchor value+", "+group"]
(number/name of element)
defines the current element in the facet referenced by (facet number).
(facet number) is the number of the facet to which the current element belongs.
(element number) is the number of the current element in the facet
(element name) is the descriptive identifying text label for the current element.

E.g., facet 2 references Items. Element 7, "Multiplication", is the current element in facet 2: \$Label = 2, 7, Multiplication
Either the element number or its name or both can be specified. If no number is given, then *Facform* will assign one.

"anchor value" is the logit anchor value (if any)
"group" is the element's group number (if any)
E.g., an element with an anchor value of 2.35 logits:
\$Label = 4,,Zoo,2.35

An element with anchor value of 0.0 in group 6
\$Label = 5,13,,0.0,6"

An element with no anchor value, belonging to Group given by variable sex.
sex =2 ;male
\$Label = 1,,Fred,""+Q(sex)

\$Rating = number [, replications] (datum value)
This causes the specified observation, rating or counter to be written in *Facets* format into the \$Output file. Blank ratings are bypassed, and not written into the \$Output file. E.g., for a rating of 2 on a scale, \$Rating=2

When there are identical replications of the same datum value with the exact same facets and element numbers, their number can be specified. E.g., for 103 ratings of 4 with the same facets and element numbers, \$Rating=4,102
 Replications with a count of 0 are treated as missing data and ignored. E.g., \$rating=2,0
 This is useful for entering tabular or survey data.

Decimal observations:

Observations records as decimals, e.g., 2.5, 1.23, must be transformed into integer values.

Examples:

data are 0.00 0.25 0.50 0.75 1.00 etc.
 \$rating = \$N(\$S1W4)*4

data are in the range 1.00 to 9.99
 since Facets understands data in the range 0 to 255, a valid transformation is:
 \$rating = \$I((\$N(\$S1W4)-1) * 28)
 This gives ratings in the range 0 to 252.

6.4. Flow control keywords

\$Do = number (start a loop)

Start a list of repeated instructions to be repeated (number) times. When (number) is 0 or negative, then the list of repeated keywords is skipped over and not performed. If (number) is *not* enclosed in parentheses, it is evaluated once, and then counted down by one on each iteration of the repeated instructions until the count reaches zero.

E.g., each person is tested twice on the same task. The observations are in adjacent columns indicated by the value, POINTER. Write out the two ratings.

```
POINTER = 23 ; responses start in column 23
$Do = 2
    $RATING = $I(POINTER) ; value pointed to by POINTER
    POINTER = POINTER + 1 ; increment to next column
$Again
```

\$Do = (expression) (start a conditional loop)

If the expression is in parentheses of the form (..), then it is re-evaluated at the start of each run through of the repeated instructions. If true, the keyword list will be performed. E.g., each examinee starts at item 1 and does as many items as possible. A blank follows the last item completed.

```
EODATA = " " ; a blank flags the end of the data for an examinee
ITEM = 1 ; start with item number 1
DATA = $$ (ITEM + 35) W1 ; response to item 1 is in column 36 = 1 + 35
$Do = (DATA <> EODATA) ; if not blank, do the list of keywords
    $LABEL=2,ITEM ; the element in facet 2 is the item number
    $RATING = DATA ; write the DATA into the Facets-format file
    ITEM = ITEM + 1 ; advance to the next item
    DATA = $$ (ITEM+35) W1 ; obtain the next response
$AGAIN ; do this again
```

\$Again (end a loop)

ends a list of repeated keywords. \$DO and \$Again must be in matched pairs. They may be as many levels deep as necessary.

E.g., each examinee is rated by 5 judges on 10 items

```
$DO = 5
    ; judge-related keywords
    $DO = 10
    ; item-related keywords
$Again
$Again
```

`$Nextline` (go on to next input data line)
 go on to the next line in the input raw data file. This is useful when one element's observations span more than one data line.
 E.g., for each examinee, the responses to items 1-50 are on one line, and 51-100 are on a second line.
`ITEM=1` ; start at item 1
`$DO = 50` ; do 50 items
 ; item-related keywords
`ITEM = ITEM + 1` ; keep item number current
`$Again`
`$Nextline` ; go on to next line of input data
`$DO = 50` ; do items 51-100
 ; item-related keywords
`ITEM = ITEM + 1` ; keep item number current
`$Again`

`$If=(expression)` (perform conditionally)
 start of conditional execution. When the expression is true, the keywords immediately after `$If` are performed.

`$Endif` (end of conditional section)
 this marks the end of the list of keywords associated with the preceding `$If`

Example: Only select sub-test "A"
`$IF=(S5W1="A")` ; sub-test indicator is in column 5 of data records
 ; rating related keywords
`$ENDIF`

`$Else` (or else, conditionally)
 this marks the start of keywords which are performed when the `$If` expression is false

Example 1: a datum is to be processed only if it is *not blank*.
`DATUM = S38E38` ; get a one character observation from the input file
`$RATING = DATUM` ; only do this if not blank - the *Facform* default

Example 2: any blank datum is to be recoded to 0.
`DATUM = S38E38` ; get a one character observation from the input file
`$If = (DATUM<>"")` ; is it blank?
`$RATING = DATUM` ; only do this if not blank

`$Else`
`$Rating = "0"` ; output "0" instead of blank
`$Endif` ; end of `$If` keyword list

`$Elseif=(expression)` (`$Else` followed by `$If`)
 this marks a `$If` expression to be performed when the previous `$If` expression is false

Example: set a variable to indicate examinee sex
`Sex=S24E25` ; Sex character M, F, or Blank in record
`$IF=(Sex="F")`
`Sexvar=1` ; the numerical Sex variable
`$ELSEIF=(Sex="M")`
`Sexvar=2`
`$ELSE`
`Sexvar=3` ; picks up invalid codes here
`$ENDIF`
`$Label=4,Sexvar` ; Used as element number in a Facet

`$Batch=Yes`
 Terminate and exit Facform when processing is completed.
 This is useful in batch mode.

e.g.

runff.bat is

```
start /w c:\facets\facform HT0500FF.txt
start /w c:\facets\facform HT0595FF.txt
start /w c:\facets\facform HT0596FF.txt
start /w c:\facets\facform HT0597FF.txt
```

and each of the HT0500FF.txt files contains

```
$Batch=Yes
```

then executing runff.bat will cause each Facform process to occur in turn.

6.5. Text-oriented keywords

\$Print = expression (display on screen)

a value to be displayed on the screen for information purposes. This is useful for debugging keyword lists or getting immediate notification of unusual data values.

E.g.,

```
$If = (DATUM>4)
```

```
    $Print = "Unexpectedly large data values in file" ; this shows on screen
```

```
    $Print = "Value of datum is "+datum ; this tells you the unexpected value
```

```
$Endif
```

\$Spec = expression (put directly into \$\$soutput file once)

a value to be written into the output specification file. This enables you to add specifications directly to the *Facets* specification file, before any lines of data are processed.

E.g., you want to add some specifications into the *Facets* file:

```
$Spec = "Title = Rorschach Blob Test"
```

```
$Spec = "Positive = 1" ; Facet 1 is positively oriented
```

\$Text = expression (put into \$\$soutput file, each time encountered)

a value to be written into the output specification file. This enables you to add specifications directly to the *Facets* specification file, while lines of data are being processed.

E.g., you want to add some specifications into the *Facets* file:

```
$If=($L=1) ; only do this once when the first data line is processed
```

```
$Text = ";"+$S23W10 ; take some information of the first data line, and
```

```
; make it a comment in the Facets specification file
```

```
$Endif
```

Variable = expression (calculate a value)

Any word starting with a letter and placed before an = sign, i.e., where a keyword is expected, is assumed to be the name of a variable. The variable is assigned the value of the expression following the = sign. E.g.,

```
X=1 assigns to variable X the numeric value 1.
```

```
MM="Mike" assigns to variable MM the characters Mike.
```

Variable names must start with a letter and be composed only of letters and numbers, e.g., ZQX3 is a valid variable name. Choose meaningful variable names, such as ITEM to hold an item number, or ZQX3 for "Zarathrustra Q. Xerxes III".

Blank lines (use as spacing to ease reading keyword files)

are ignored. Use them to space out your keywords for clarity.

; (followed by comments)

starts a comment. Everything following a ";" is ignored. Use these to tell yourself what your keywords are doing.

Blank spaces (using as spacing within lines)

are ignored except between " ". Use blanks to indent \$DO= and \$IF= keyword lists.

6.6. Expression evaluation

Facform provides a flexible method of obtaining the values of numbers and names by means of expressions. Expression evaluation is done as follows:

"..." (text)

Quoted text. A value in quotes is a constant character value, e.g., "Mary"

number (numeric value)

A value comprising digits, with or without a decimal point, is a constant numeric value, e.g., 23.4

+*/^ (plus, minus, times, divide, exponent)

+ (plus), - (minus), * (multiply), / (divide), ^ (to the power of) are numeric operators. These are used in the evaluation of numeric quantities, e.g., 23+45 evaluates to 68.

+ (concatenate)

+ (plus) is used to put together, concatenate, character text.

E.g., "George"+" D. "+"Washington" evaluates to "George D. Washington".

><= (greater, less, equal)

> (greater than), < (less than), = (equal to), >= (greater than or equal to), <= (less than or equal to), <> (not equal to) are used to compare values.

For numeric values, 23>12 yields the result of "true", or 1 as a numeric value. 23<12 yields the result of "false", or 0 as a numeric value.

For character values, "B">"A" yields the result of 1, meaning "true". "B"<"A" yields the result of 0 = "false", or 0 as a numeric value. Before character comparison, leading and trailing blanks are removed, e.g., "A" "=" "A" is true, but "AB"="A B" is false, because of the blanks between A and B.

&| (and, or)

& (logical and), | (logical or) are used to combine numeric values, or the numeric outcome of <,>,. A & B is 0 if either A or B or both are 0. A | B is 1 if either A or B or both are 1.

E.g., (A>1) & (B=0) evaluates to 0 if A is less than 1, or B is not zero.

(A=B) | (A=C) evaluates to 1 if A is equal to B or C, otherwise it is 0.

() (arrange order of expression evaluation)

Parentheses are used to set the order in which sub-expressions are evaluated. Evaluation proceeds from the inner-most sub-expression within parentheses outward. Otherwise, evaluation proceeds from left to right.

E.g., ((2+3)*(3+4)) evaluates to (5*(3+4)), then (5*7), then 35.

1+2*3 is evaluated left to right as (1+2)*3, then 3*3, then 9.

6.7. Expression commands

These commands perform special functions which enable *Facform* to obtain and decode the input file of raw data.

\$A???? (treat as UPPER case, alphabetical)

"Alphabetize": means convert to UPPER-CASE characters, between quotation marks, the expression ?????, whether or not it is already in quotes, e.g., \$A("New York") becomes "NEW YORK".

\$Cnnn (locate Comma-Separated value)

"Comma-separated value": means read the nnn"th comma-separated value from the current data record into a quoted character variable,

e.g., if the data record is 12,34,56,43, then \$C2 is "34".

By default, the separating character is ",". Change this with \$Separator=.

For Tab separators, use \$T

\$G (the current line-group number)

"Group": means the current line-group number. The line-group number advances each time the keyword list is restarted. This is the same as \$L unless \$Nextline appears in the keyword list. When \$Nextline is processed, \$L advances by 1, but \$G does not change. This can be used to identify examinees in the raw data file, e.g.,
\$Label=1,\$G ; Facet 1 is examinees, line-group number is the examinee number
Y=\$G ; assigns the current line-group number to variable Y.

\$I???? (treat as an integer value)

"Integer": means convert the expression represented by ????? to the nearest integer value. Values ending in 0.5 evaluate to the next larger value in absolute size. \$I12.49 evaluates to 12. \$I12.5 evaluates to 13. \$I(-12.5) evaluates to -13.

The expression may be in numeric or character format. For character expression, only leading digits are evaluated. \$I"23abcd" evaluates to 23. \$I"AB12" evaluates to 0.

\$L (use the current line number)

"Line": means the current line number in the input file of raw data file. This is useful when you wish to identify anomalies in the raw data file. E.g.,

```
$If=(DATUM>"4")
  $Print "Unexpected value of "+DATUM+" in line "+$Q($L)
$Endif
```

\$N???? (convert into a number)

"Number": means convert the expression represented by ????? to a number. If the expression is already a number, it is unchanged. If it is a character expression, then the leading digits are evaluated. E.g., \$N"23.4abcd" evaluates to 23.4

\$Q???? (put value between "")

"Quote": means convert to characters, between quotation marks, the expression ?????. If ????? is already a quoted character value, it is unchanged. Otherwise ????? is converted to characters and placed between "". E.g., \$Q(2+3) becomes "5".

\$SxxEyy (start and end in data line)

"Start-End": means read the characters from the current data record, starting in position xx and ending in position yy as a quoted character variable.

E.g., if the data record contains ABCDEF starting in column 1, then \$S2E4 yields "BCD".

\$SxxWww (start and width in data line)

"Start-Width": means read the characters from the current data record, starting in position xx with a width of ww, as a quoted character variable.

E.g., if the data record contains ABCDEF starting in column 1, then \$S3W2 yields "CD".

\$Tnnn (locate Tab-separated value)

"Tab-separated value": means read the nnnth Tab-separated value from the current data record into a quoted character variable,

e.g., if the data record is 12[Tab] 34[Tab] 56[Tab] 43, then \$T2 is "34".

\$U???? (unquote, remove from "")

"Unquote": means remove the quotes from around the value ????? if there are any. This can be used to convert a character string into a variable name. e.g., \$U"AB" is the variable name AB.

6.8. Variables

Any word starting with a letter and placed before an = sign, i.e., where a keyword is expected, is assumed to be the name of a variable. The variable is assigned the value of the expression following the = sign.

Pointer = 2 ; this assigns the value 2 to the variable: Pointer

Unit = (1+3) ; this assigns 4 to the variable: Unit

Name="London" ; this assigns "London" to the variable: Name

\$Label = 1, Pointer+Unit,Name ; defines the current element in facet 1, to have element ; number 4+2=6, and identifying label "London"

6.9. Facform stops prematurely: Disk file problem

When *Facform* experiences difficulty with a disk file, it reports a message of the form:

Phase 2: Processing keywords from "FILEkey.txt".

File not found

The difficulty is usually with the last file name mentioned. In this case, *Facform* could not locate the disk file "FILEkey.txt". Is it misspelled or it is in a different directory?

7. PROGRAM MESSAGES

7.1. Facform stops prematurely: Keyword list problem

When *Facform* cannot clearly understand your keyword instructions, it stops with a diagnostic message such as:

Unknown or too abbreviated keyword in line 34

in: \$ZZZ=23

Diagnosis: \$ZZZ is not a keyword that *Facform* understands.

The precise position of the error may be indicated:

duplicate >>

v

(23>>Item)

The "v" character points to the duplicate ">" sign.

Under some circumstances, the current values of all variables will be displayed to help you diagnose the difficulty. They will be listed after the heading:

Current variable assignments:

7.2. Facform stops prematurely: Problem messages

These are displayed on your screen. They are listed here in alphabetical order.

\$AGAIN invalid format

\$AGAIN cannot be followed by a value, i.e., \$AGAIN=3 is invalid. Loop activity is controlled by \$DO=.

\$DO invalid format

The expected formats are of the form \$DO=5 or \$DO=(4+L) or \$DO=(Item<20). Other formats such as \$DO=2,3 are invalid.

\$DO problem

Facform failure processing loops. Contact us.

\$ELSE invalid format

\$ELSE cannot be followed by a value. \$IF controls conditional processing.

\$ELSEIF invalid format

\$ELSEIF=Expression is valid. \$ELSEIF alone or \$ELSEIF=A,B are invalid.

\$ENDIF invalid format

\$ENDIF cannot be followed by a value. \$IF controls conditional processing.

\$IF invalid format

\$IF=Expression is valid. \$IF alone or \$IF=A,B are invalid.

\$IF or \$ELSEIF Problem

Facform failure processing \$IF= or \$ELSEIF=. Contact us.

Both Width and End

\$S23W6E28 is invalid. Use either \$S23W6 or \$S23E28.

Commands start with \$ Variable names with A-Z

Commands are keywords like \$OUTPUT. Variable names are like AB23. Statements like %#^=23 are invalid.

Data failure: second pass

Either the data file was changed during *Facform* operation, or *Facform* has failed. Contact us.

Dataform failure

Facform failure constructing *Facets* record. Contact us.

Duplicate <<

23<<Itemnum is invalid. Do you mean: 23<Itemnum

Duplicate ==

\$IF=(Item==23) is invalid. Do you mean: \$IF=(Item=23)

Duplicate >>

23>>Itemnum is invalid. Do you mean: 23>Itemnum

End position ____ less than Start position ____

\$S99E50 is invalid because the field ends to the *left* of its start position..

Expected format is: keyword = value

A value is missing for a keyword that requires one, e.g., \$Output=

Facet number outside range 1 thru ____: ____

A facet number is out of range given by \$Facets=, e.g., \$Facets=3, then \$Flabel=999,Examinees

File name contains * or ?

File names cannot contain "*" or "?" characters.

File not found

A \$Input of \$Spinput file could not be found with the name you specified. Did you mistype the name or is it in a different directory?

Format must be \$LABEL = facet number, element number, element name, ...

\$LABEL=1 lacks an element number and/or element name.

Format must be \$RATING = value

\$Rating=value is required, not \$Rating= or \$Rating=2,3 (indicating two values)

Integer not between 32767 and -32768

A value is to be calculated that is out of range.

Invalid character field operator

"George "-"Washington" is invalid. Use "George "+"Washington"

Invalid element ordinal number

\$LABEL=2,"Martha" should be \$LABEL=2,23 or \$LABEL=2,,"Martha"

Invalid leading operator

\3 has no clear meaning.

Invalid replicate value

R3,1,1,1 is valid. RTHREE,1,1,1 is invalid.

Invalid trailing operator

3* has no clear meaning.

Keyword or Variable name expected

=23 is missing its keyword or variable name.

Label not found

The element name or number are not known. If \$Given was specified, a new label has been found in the \$Input file. Otherwise there is a *Facform* problem. Contact us.

Mismatch on element numbers

Facform is assigning element numbers to labels, but finds ambiguous information. For instance, \$Label=1,3,"Fred" and \$Label=1,42,"Fred" and then \$Label=1,,"Fred"
Is this "Fred" Person 3 or Person 42?

Missing equals (=) sign in command line

The DOS Command line can have only two terms without = signs: the keyword file and the output file. Did you put in extra blanks with other keywords, e.g. \$Print = 23 instead of \$PRINT=23

Missing Width or End

\$S23 is not enough. Specify \$S23W1 or \$S23E23.

More than ___ elements. Reduce the size of your data.

Your specifications and data combine to generate too many elements. Reduce the size of your data or the number of facets.

More than ___ labels. Reduce the number of labels.

Only specify labels, and extra information, to *Facform* for those elements that must have them for identification. Instead, use element numbers, such as line numbers, \$L.

Multiple decimal points

Values such as 23..45 are invalid. Just use 23.45

No element defined in facet position ___

You listed \$RATING= before establishing the current element in each of the facets. Use \$Label=2,0 if, say, facet 2 has no element number.

No format keywords

You listed no keywords that reformat the data. Is this the correct keyword file?

No keyword before = sign in command line

Did you include extra blanks on the DOS command line? There is an = sign preceded by a blank, e.g.,
C:>**FACFORM KEYFILE \$FACETS = 3**, instead of C:>**FACFORM KEYFILE \$FACETS=3**

No unformatted data to input.

No useable data found in the data file. Did you specify the right files?

No valid \$Facets=? keyword

No \$FACETS= keyword has been found.

No valid \$OUTPUT keyword

No \$OUTPUT= keyword has been found. Did you intend to specify an output file on the DOS command line?

Non-numeric in numeric field

\$FACETS=23A4 is an example of mixing numbers and other characters in a numeric field. Specify \$I(23A4) is you want 23A4 to have the value 23.

Not enough \$AGAIN and \$ENDIF keywords by ____

There are more \$DO and \$IF keywords than \$AGAIN and \$ENDIF. Check that they pair up correctly.

Operator expected

"George " "Washington" requires some operator, e.g., + between its two expressions, e.g., "George "+"Washington"

Sort loop error

Facform failure sorting labels. Contact us.

Start position ____ less than 1

\$SxxW1 counts in columns, not displacement. \$S00W1 defines 0 as the start position, but the smallest allowable value is column 1, \$S01W1

Too much extra element information. Reduce its size.

Extra information is the third entry in the \$LABEL= specification. Reduce this.

Too much label information. Reduce the size of labels.

Use the minimum label length, and only those required for identification.

Unexpected characters after ____

An expression command such as \$LXY has been specified instead of just \$L for the current line number.

Unexpected keyword

A keyword like \$ZZZ has been specified, which *Facform* can't interpret.

Unknown \$ expression-command, must be one of \$A,\$C,\$G,\$I,\$L,\$N,\$Q,\$S,\$T,\$U

An unknown command, such as \$D, has been specified in an expression.

Unknown or too abbreviated keyword

Facform can't disambiguate a keyword like \$SP= that could mean \$SPinput= or \$SPoutput= or \$SPec=

Unknown variable name: ____

You used a variable before giving it a value. Thus, specifying \$LABEL=1,Itemnum stating Itemnum=1

Unmatched \$AGAIN

There is no \$DO= to pair with this \$AGAIN

Unmatched \$DO - \$AGAIN

The \$DO= and \$AGAIN= do not pair up correctly.

Unmatched \$ELSE

There is no \$IF= available with which to pair this \$ELSE

Unmatched \$ENDIF

There is no \$IF= available with which to pair this \$ENDIF

Unmatched \$IF - \$ELSE

The \$IF= and \$ELSE= do not pair up correctly.

Unmatched \$IF - \$ENDIF

The \$IF= and \$ENDIF= do not pair up correctly.

Unmatched (

There is a missing), e.g., (23

Unmatched)

There is a missing (, e.g., 23)

Unmatched character field

A character field and a numeric field are being paired up, e.g., "ABC"+23
or a character field is missing.

Unmatched quote

A quote is missing, e.g., "Math Test

Valid facet range is 1 thru ____ . You specified ____

Facets such as 0 or 999 are out of range.

Variable names can only comprise A-Z and 0-9

%12\$ = 99 defines an invalid variable name. P12D=99 is valid.

Variables must be of format name = expression

Item= is missing its value.

Vstack format failure: ____

Facform failure processing variable values. Contact us.

7.3. Warning messages

Duplicate keyword ignored

Facform informs you of keywords which it has ignored because they duplicate previously processed keywords. This message will be issued for keywords in the keyword disk file superseded by those on the DOS command line.

Ignoring differing element LABEL of ____ instead of ____ for element number ____

Element numbers take precedence over element names. *Facform* warns you if two different names have been given to the same element number. The "instead of" name is kept unchanged in the element list.

Label ____ without an element number is ambiguous. Matching with first element with same label of facet ____ from: ____

Facform is attempting to make your element numbers and labels self-consistent. It is better for you to do this explicitly in your \$LABELS= statements.

No \$Rating= found. Labels= will be output, but no data

There is no \$RATING= keyword, so no data will be output. Labels will be output.

Facform Keyword Index

\$A????	42
\$Again	38
\$Batch	38
\$Cnnn	42
\$Do	38
\$Else	38
\$Elseif	38
\$Endif	38
\$Facets	37
\$Flabel	37
\$G	42
\$Given	36
\$I????	42
\$If	38
\$Input	36
\$L	42
\$Label	37
\$Length	37
\$N????	42
\$Nextline	38
\$Output	36
\$Print	40
\$Q????	42
\$Rating	37
\$Separator	36
\$Spec	40
\$Sinput	36
\$Spoutput	36
\$SxxEyy	42
\$SxxWww	42
\$Text	40
\$Tnnn	42
\$U????	42
Blank lines	40
Blank spaces	40
Variable	40