

# books Group

John D. Baker

<https://github.com/bakerjd99/jackshacks/blob/main/books.ijs>

SHA-256: b39da3628062899655bddc311828bd690bcc7b14176279c09445409680cdf4dc

November 14, 2024

## Contents

<b>books Overview</b>	<b>2</b>
books Interface . . . . .	2
books Notes . . . . .	3
<b>books Source Code</b>	<b>4</b>
<b>=: Index</b>	<b>17</b>

## books Overview

`books.ijs` is a J script that reads a TAB delimited text file of books read and calculates some summary statistics.

`books` is distributed as an auxiliary J addon. Auxillary addons are hosted in private GitHub repositories. `books` can be installed in the local J folder `~addons/jacks` with the standard J pacman utility:

```
load 'pacman'
```

```
NB. files from https://github.com/bakerjd99/jackshacks  
install 'github:bakerjd99/jackshacks'
```

```
NB. installed files  
dir '~addons/jacks'
```

```
NB. sample data files  
dir '~addons/jacks/testdata'
```

## books Interface

<code>bookctgstats</code>	<code>[6]</code>	<i>book category statistics</i>
<code>booksperyear2</code>	<code>[6]</code>	<i>books per year from standard btcl books table</i>
<code>manyauthors</code>	<code>[9]</code>	<i>authors read more than once</i>
<code>manyreads</code>	<code>[10]</code>	<i>books read more than once</i>
<code>stdbookstab</code>	<code>[14]</code>	<i>standard books table</i>

## **books Notes**

Books data originates in an Excel spreadsheet `BOOKS.xlsx`.

To uses `books.ijs` do:

1. Open `BOOKS.xlsx` and save as a TAB delimited text file. A sample TAB delimited file `books_sample.txt` is in `~addons/jacks/testdata/`
2. Define a J configured folder `~BOOKS` pointing to the directory containing the file saved in step one.
3. Load `books.ijs` and use the interface words. ‘

## books Source Code

```
NB.*books s-- summarize books read.
NB.
NB. verbatim:
NB.
NB. interface word(s):
NB. -----
NB.  bookctgstats  - book category statistics
NB.  booksperyear2 - books per year from standard btcl books table
NB.  manyauthors   - authors read more than once
NB.  manyreads     - books read more than once
NB.  stdbookstab   - standard books table
NB.
NB. created: 2024nov03
NB. -----
NB. 24nov05 (bookctgstats) added
NB. 24nov11 (fmtbooks) added
NB. 24nov14 (manyauthors) editors, translators removed, multiple authors split

coclass 'books'

NB.*end-header

NB. carriage return line feed character pair
CRLF=: 13 10{a.

NB. interface words (IFACEWORDSbooks) group
```

```
IFACEWORDSbooks=: <;._1 ' bookctgstats booksperyear2 manyauthors manyreads stdbookstab'
```

*NB. line feed character*

```
LF=: 10{a.
```

*NB. read more than once - must satisfy 2 <: READCNT*

```
READCNT=: 2
```

*NB. root words (ROOTWORDSbooks) group*

```
ROOTWORDSbooks=: <;._1 ' IFACEWORDSbooks ROOTWORDSbooks VMDbooks bookctgstats booksperyear2 dstat manyautho  
>..>rs manyreads ofreqlist portchars stdbookstab'
```

*NB. version, make count and date*

```
VMDbooks=: '0.5.1';6;'14 Nov 2024 17:07:28'
```

*NB. trims all leading and trailing white space*

```
allwhitetrिम=: ] #~ [: -. [: (*./\ . +. */.\) ] e. (9 10 13 32{a.)"_
```

```
antimode=: 3 : 0
```

*NB.\*antimode v-- finds the least frequently occurring item(s) in*

*NB. a list.*

*NB.*

*NB. monad: ul =. antimode ul*

*NB.*

```
NB.  antimode ?.500#100
NB.  antimode ;:'blah blah blah yada yada wisdom'

if. 0 < # y =. ,y do.    NB. no antimodes for null lists
  f =. #/.~ y            NB. nub frequency
  (~. y) #~ f e. <./ f   NB. lowest frequency items
else. y
end.
)

NB. retains string before first occurrence of (x)
beforestr=: ] {~ 1&(i.~)@([ E. ])

bookctgstats=: 3 : 0

NB.*bookctgstats v-- book category statistics.
NB.
NB. monad:  ct =. bookctgstats btclBtab
NB.
NB.  bookctgstats stdbookstab '~BOOKS/books.txt'

'ctg cnt'=: ofreqlist }. tolower&.> y {"1~ (tolower&.> 0{y) i. <'type'
ctg ,.' ',.":0.001 round cnt,.(100*cnt%t),.s,.t %~ s=. +/\cnt [ t=. +/\cnt
)

booksperyear2=: 3 : 0
```

*NB.\*booksperyear2 v-- books per year from standard btcl books table.*

*NB.*

*NB. monad: it =. booksperyear2 btclBtab*

*NB.*

*NB. btab=. stdbookstab '~BOOKS/books.txt'*

*NB. d=. booksperyear2 btab*

*NB. 0.01 dstat 1{d*

*h=. tolower&.> 0{y*

*d=. }. y*

*d=. freqlist (h i. <'year') {"1 d*

*d=. (\_1&".&> 0{d) ,: ;1{d*

*NB. merge in missing zero years*

*d=. d ,. 0 ,:~ (0{d) -.~ ({.0{d) + i. >:(>./ - <./) 0{d*

*(/: 0{d) {"1 d*

*)*

*charsub=: 4 : 0*

*NB.\*charsub v-- single character pair replacements.*

*NB.*

*NB. dyad: clPairs charsub cu*

*NB.*

*NB. '-\_\$ ' charsub '\$123 -456 -789'*

*'f t'=. ((#x)\$0 1)<@,&a./.x*

```
t {~ f i. y
)

NB. deviation about mean
dev=: -"_1 _ mean

dstat=: 3 : 0

NB.*dstat v-- descriptive statistics
NB.
NB. monad: ct =. dstal nl
NB.
NB.    dstat  ?.1000#100
NB.
NB. dyad:  ct =.  faRound dstat nl
NB.
NB.    0.1 dstat  ?.1000#100

0.0001 dstat y
:
t=. '/sample size/minimum/maximum/1st quartile/2nd quartile/3rd quartile/first mode'
t=. t , '/first antimode/mean/std devn/skewness/kurtosis'
min=. <./
max=. >./
t=. ,&' : ' ;._1 t
v=. $,min,max,q1,median,q3,({.@mode2),({.@antimode),mean,stddev,skewness,kurtosis
t,. ": x round ,. v , y
)
```



```
fmtbooks=: 4 : 0

NB. *fmtbooks v-- format book counts and authors/titles as bt
NB.
NB. dyad: btCntWtxt =. (ia;il) fmtbooks blcl

'width cnts'=. x

NB. partition by count - sort on first word
s=. ,&' ; '@(' _ '&rebu)@(' _ '&charsub)&.> y
s=. (width&wrapwords@;)&.> /:~&.> (b=. ~:cnts) <;.1 s

NB. format as bt of counts and word wrapped text
(~.&.> b <;.1 cnts) ,. ] ; _2@tlf@(' _ '&charsub)&.> s
)

NB. frequency distribution of boxed list items
freqlist=: ~. ,: [: <"0 #/.~

NB. REFERENCE - standard z locale verb: jpath '~temp/'
jpath=: jpath_j_

NB. kurtosis
kurtosis=: # * +/@(^&4)@dev % *: @ssdev

manyauthors=: 4 : 0
```

```
NB.*manyauthors v-- authors read more than once.
NB.
NB. dyad:  btCntAuthors =. iaWidth manyauthors btclBtab
NB.
NB. 70 manyauthors stdbookstab '~BOOKS/books.txt'

NB. remove editors and translators
authors=. }. y {"1~ (tolower&.> 0{y) i. <'author'
authors=. ':tr:'&beforestr@(':ed:'&beforestr)&.> authors

NB. split multiple authors to singles
authors=. rebc@allwhitetrim&.> <;._2 ;' '&tlc&.> authors

NB. authors by decreasing read counts
'authors cnts'=. ofreq s: authors

NB. read more than once
authors=. b#authors [ cnts=. b#cnts [ b=. READCNT <: cnts

NB. format as bt cnts and authors
(x;cnts) fmtbooks 5 s: authors
)

manyreads=: 4 : 0

NB.*manyreads v-- books read more than once.
NB.
NB. dyad:  btCntBooks =. iaWidth manyreads btclBtab
```

*NB.*

*NB.* 70 manyreads stdbookstab '~BOOKS/books.txt'

*NB. titles by decreasing read counts*

```
'titles cnts'=. ofreq s: }. y {"1~ (tolower&.> 0{y) i. <'title'
```

*NB. read more than once*

```
titles=. b#titles [ cnts=. b#cnts [ b=. READCNT <: cnts
```

*NB. format counts and wrapped titles*

```
(x;cnts) fmtbooks 5 s: titles  
)
```

*NB. mean value of a list*

```
mean=: +/ % #
```

*NB. median value of a list*

```
median=: -:@(+/@)((<. , >.)@midpt { /:~) ::_:
```

*NB. mid-point*

```
midpt=: -:@<:@#
```

```
mode2=: 3 : 0
```

*NB.\*mode2 v-- finds the most frequently occurring item(s) in a  
NB. list.*

*NB.*

```
NB. monad:  ul =. mode2 ul
NB.
NB.  mode2 ?.500#100
NB.  mode2 ;:'I do what I do because I am what I am'

if. 0 < # y =. ,y do.      NB. null lists have no modes
  f =. #/.~ y              NB. nub frequency
  (~. y) #~ f e. >./ f     NB. highest frequency items
else. y
end.
)

NB. like (freq) but results in descending frequency
ofreq=: [: (([: < [: \: [: ; 1 { ]) { &.> ]) ~. ; #/.~

NB. ordered boxed list frequency distribution - see long document
ofreqlist=: [: (([: \: [: ; 1 { ]) { "1 ]) ~. ,: [: <"0 #/.~

NB. parse TAB delimited table text after removing (x) chars - see long document
parsetdwc=: [: <;._2&> (a.{~9) ,&.>~ [: <;._2 [: (] , ((10{a.)"_ = {:) }. (10{a.)"_ ) (13{a.) -.-.~

NB. portable box drawing characters
portchars=: [: 9!:7 '+++++++|-'_" [ ]

NB. first quartile
q1=: median@((median > ]) # ]) ::_:
```

*NB. third quartile*

```
q3=: median@((median < ]) # ]) ::_:
```

*NB. reads a file as a list of bytes*

```
read=: 1!:1&([`<@.(32&>@ (3!:0)))
```

```
reb=: 3 : 0
```

*NB.\*reb v-- removes redundant blanks - leading, trailing multiple*

*NB.*

*NB. monad: reb cl*

*NB. dyad: ua reb ul*

```
' ' reb y
```

```
:
```

```
y=. x , y
```

```
b=. x = y
```

```
}.(b*: 1|.b)#y
```

```
)
```

*NB. removes multiple blanks (char only)*

```
rebc=: ] #~ [: -. ' ' &E.
```

*NB. generalization of (rebc) (x) argument is any atom*

```
rebu=: ] #~ [: -. (2: # [) E. ]
```

*NB. round (y) to nearest (x) (e.g. 1000 round 12345)*

```
round=: [ * [: (<.) 0.5 + %~
```

*NB. skewness*

```
skewness=: %:@# * +/@(~&3)@dev % ^&1.5@ssdev
```

*NB. sum of square deviations (2)*

```
ssdev=: +/@:~:@dev
```

```
stdbookstab=: 3 : 0
```

*NB.\*stdbookstab v-- standard books table.*

*NB.*

*NB. monad: btcl =. stdbookstab clBooksfile*

*NB.*

*NB. btab=. stdbookstab '~BOOKS/books.txt'*

*NB. btab=. stdbookstab '~JACKSHACKS/testdata/books\_sample.txt'*

```
t=. '' -~ utf8 read jpath y
allwhitetrim@rebc.> '' parsetdwc t
)
```

*NB. standard deviation (alternate spelling)*

```
stddev=: %:@:var
```

*NB. terminate with character if not present: '&' tlc 'end it'*

```
tlc=: ] , [ }.~ [ = [: {: ]
```

*NB. appends trailing line feed character if necessary*

```
tlf=: ] , ((10{a.)"_ = {:) }. (10{a.)"_
```

*NB. convert to lower case*

```
tolower=: 0&(3!:12)
```

*NB. transitive closure*

```
tranclose2=: # (i.~ {. ]) [: }. (, #) { ~^:a: 0:
```

*NB. character list to UTF-8*

```
utf8=: 8&u:
```

*NB. var*

```
var=: ssdev % <:@#
```

```
wrapwords=: 4 : 0
```

*NB.\*wrapwords v-- wrap words into lines of length (x).*

*NB.*

*NB. This algorithm: due to Roger Hui. Wraps words (nonblank) runs*

*NB. into lines of length (x) without breaking words. Words cannot*

*NB. be longer than (x). Transitive closure is used to compute*

*NB. where appropriate newline (LF) characters replace blanks.*

*NB.*

```
NB. dyad: cl =. iaWidth wrapwords clWords
```

*NB.*

```
NB. 27 wrapwords 7770$'go ahead make my day and surprise me'
```

*NB. remove extra blanks and CRLF*

```
y=. reb y -. CRLF
```

```
e=. (' ' I.@:= y),#y
LF (e {~ <: tranclose2 e I. (x+2)+}:_1,e)} y
)
```

*NB.POST\_books post processor.*

```
smoutput IFACE_books=: (0 : 0)
NB. (books) interface word(s): 20241114j170728
NB. -----
NB. bookctgstats    NB. book category statistics
NB. booksperyear2  NB. books per year from standard btcl books table
NB. manyauthors    NB. authors read more than once
NB. manyreads      NB. books read more than once
NB. stdbookstab    NB. standard books table
)
```

```
cocurrent 'base'
coinsert  'books'
```



# Index

' , 6

allwhitetrim, 5

antimode, 5

beforestr, 6

bookctgstats, 6

booksperyear2, 6

charsub, 7

CRLF, 4

dev, 8

dstat, 8

fmtbooks, 9

freqlist, 9

IFACE\_books, 16

IFACEWORDSbooks, 5

jpath, 9

kurtosis, 9

LF, 5

manyauthors, 9

manyreads, 10

mean, 11

median, 11

midpt, 11

mode2, 11

ofreq, 12

ofreqlist, 12

parsetdwc, 12

portchars, 12

q1, 12

q3, 13

read, 13

READCNT, 5

reb, 13

rebc, 13

rebu, 13

ROOTWORDSbooks, 5

round, 13

skewness, 14

ssdev, 14

stdbookstab, 14

stddev, 14

tlc, 14

tlf, 14

tolower, 15

tranclose2, 15

utf8, 15

var, 15

VMDbooks, 5

wrapwords, 15