```
/ http://inferno.bell-labs.com/cm/cs/who/bwk/interps/pap.html

/ do one million increments on initial state 0
\t do[1000000;i+:1]

/ ackermann's
\t {$[x;.z.s[x-1;$[y;.z.s[x;y-1];1]];y+1]}[3;7]

/ array indexed forwards and backwards
\t x(x;reverse x:til 200000)

/ while(x>count string) join chop join ... on "abcdef"
f:{{500000>count x}{(i _ x),(1+i:floor .5*count x)#x:raze("123";x↩
   ;"456";x;"789")}/x}
\t do[10;f"abcdef"]

/ lookup hex strings in decimal strings
\t {sum("0123456789abcdef"16 vs'x)in string x}til 200000

`f 0:(30000?300)#\:"king "; /james

/ write read file
\t `f 0:read0`f

/ (lines;words;chars) file
\t (count;sum sum each" "=;sum count each)@\:read0`f

/ write reverse read file
\t `f 0:reverse read0`f

`f 0:100000#enlist"-123.456"; / some numbers

/ sum float-from-ascii file
\t sum"F"$read0`f

\
/ approximate times on 100MHZ pentium (32MB)
t:( 2   10    .15  2.2  1  3.5  3.2   4   5.7 /q
     .3  1    .8   5   25 80   50   125  15 /java
    3   40   8     1    6  4   15     8  10. /perl
  100 1000 100    20   12 80   15    70  50.  ) /tcl

the 9 tests are loops, text-processing and text file io.
```

```
even though q avoids all these
 loops - rare, e.g. none in kdb+.
 text  - we prefer data. binary is better.
 stdio - we prefer mmap to read/write.

q is faster (sum of times)
 q(32) perl(95) java(300) tcl(1400+)

q is shorter (lines of code)
 q(9) awk(66) perl(96) tcl(105) scheme(170) vb(200) java(350)
```