A brief summary of J

```
   3 + 5           NB. Plus
8
   2 * 3           NB. Times
6
   3 - 5           NB. Minus
_2
   15 % 6          NB. Divided by
2.5
   % 8             NB. Reciprocal
0.125
   2 + 3 * 4       NB. Precedence
14
   2 * 3 + 4
14
   (2 * 3) + 4
10
   4 + 2 * 3
10
   % 15 % 6        NB. Ambivalence
0.4
   2 | 0 1 2 3 4 5 NB. Residue
0 1 0 1 0 1
   6.5 <. 3        NB. Lesser of
3
   4 >. 10         NB. Larger of
10
   <: 8            NB. Decrement
7
   >: 3.14         NB. Increment
4.14
   2.3 + 5 + 3.5 + 6  NB. Sum
16.8
   +/2.3 5 3.5 6
16.8
   w=: 2.3 5 3.5 6
   +/w
16.8
   <./w            NB. Minimum
2.3
   >./w            NB. Maximum
6
   (<./,>./)w      NB. Min. and max.
2.3 6
   #w              NB. Tally
4
   (+/w) % #w      NB. Arithmetic mean
4.2
   (+/ % #) w
4.2
   am=: +/ % #
   am w
4.2
   am 2.3 5 3.5 6
4.2
   dev=: - am      NB. Deviations from
   dev w           NB.  mean
_1.9 0.8 _0.7 1.8
   i. 6            NB. Integers
0 1 2 3 4 5
   >: i. 6         NB. Positive
1 2 3 4 5 6        NB.  integers
   pos=: >: @ i.
   pos 6
1 2 3 4 5 6
   ei=: i. @ >:    NB. Extended
   ei 6            NB.  integers
0 1 2 3 4 5 6
   pos=: [: >: i.  NB. Pos. integers
   pos 6           NB.  using [:
1 2 3 4 5 6
   i. 3 4
0 1  2  3
4 5  6  7
8 9 10 11
   +/i. 3 4        NB. Col. sums
12 15 18 21
   +/"1 i. 3 4     NB. Row sums
6 22 38
```

- The standard ASCII character set is used.
- The terminology of English grammar is used rather than that of programming languages. Functions are referred to as *verbs*. Their arguments are considered as *nouns* and *pronouns* instead of constants and variables, although the use of these latter terms is quite common. Verbs may be modified by *adverbs* and joined by *conjunctions* to give additional verbs. For example, the verb `+/` is derived from the verb `+` *plus* by use of the adverb `/` *insert* to give the sum of the items of a list, and the conjunction *rank*, represented by `"`, in the expression `+/"1` gives the row sums of a two-dimensional array.
- Primitives, i.e., verbs, adverbs and conjunctions, are represented by a single character or a single character followed by either a period or a colon. For example, `>` is the verb *larger than*, and `6 > 3.5` is `1` and `2 > 7` is `0` indicating that the first relationship is true and the second false. The verb `>.` is *larger of* and gives the larger of its two arguments so that `6 >. 5` is `6`, while `>:` is *larger or equal* and `6 >: 5` is `1` as is `6 >: 6` but `2 >: 7` is `0`. In addition, the verbs `<./` and `>./` are similar to the verb `+/` and give the minimum and maximum, respectively, of their list arguments.
- Most verb symbols represent one function when used with one argument on the right and another function when used with arguments on the right and left. For example, the verbs *reciprocal* and *divided by* are represented by the symbol `%`, so that `% 8` is `0.125` and `15 % 6` is `2.5`. Both forms may be used in the same expression so that `% 15 % 6` is `0.4` and is interpreted as "the reciprocal of `15` divided by `6`". Functions with a single argument are termed *monadic*, and those with two *dyadic*. As another example, the verb `>:` with a single argument represents *increment* so that `>: 3.5` is `4.5`, but with two arguments represents *larger or equal*.
- Precedence amongst verbs is determined by parentheses, and in their absence the right argument is the entire expression on the right and the left argument is the noun immediately on the left. For example, the expression `% 15 % 6` in the previous paragraph is "the reciprocal of (`15` divided by `6`)" rather than "(the reciprocal of `15`) divided by `6`". Likewise, `2 + 3 * 4` is `14` as is `(3 * 4) + 2` but `3 * 4 + 2` is `18`.
- Negative numbers are indicated by a preceding underbar `_` which is considered to be part of the number as is, for example, the decimal point. Also the decimal point is necessarily preceded by at least one digit so that, for example, two-fifths as a decimal fraction is represented as `0.4`.
- Nouns may be single items or *atoms*, one-dimensional arrays or *lis*ts, two-dimensional arrays or *tables*, or arrays of higher dimension or *reports*. Thus the expression `a + b` is a valid sum as long as `a` and `b` are compatible arrays.
- Verbs may be defined in a *functional* or *tacit* manner without explicit arguments appearing in their definition. For example, the monadic verb

  ```
  am=: +/ % #
  ```

  gives the arithmetic mean However, *explicit* verbs may be defined where the arguments are specified in the definition which may extend over several lines and involve control structures similar to those in conventional programming languages.
- Finally, we mention a construct of considerable usefulness known as a *fork*, an uninterrupted sequence of three verbs. The definition of the arithmetic mean given in the last paragraph is an almost mandatory example of a fork. A similar construct involving a sequence of two verbs is called a *hook*, an example being

  ```
  dev=: - am
  ```

  where `am` is the arithmetic mean of the last paragraph, which gives deviations from the mean. The monadic verb *cap*, represented by `[:`, which caps the left branch of a fork, may often be used instead of the conjunction `@` *atop*, or informally *after*, and, for example, `pos=: >: @ i.` and `pos=: [: >: i.` are alternative definitions of a verb for positive integers.