1. Positional Number Systems

"How much is 12897 times 13863?" "729"
answered Schweik without moving an eyelash.
Jaroslav Hasek, *The Good Soldier Schweik*.

Introduction

This chapter will be primarily concerned with positional number systems including our familiar decimal system as well as the binary, octal and hexadecimal systems which occur frequently in dealing with computers. A few applications of number systems, some of them recreational, will be given, and finally the rather prosaic topic of decimal multiplication will be discussed in some detail. First of all, though, we will consider a few of the number systems which preceded the adoption of our Hindu-Arabic positional notation, the vestiges of which may be still seen today.

Additive systems

In additive systems numbers are formed by putting together in a row several single characters in order of descending value with each character being repeated as many times as required. One example is the hieroglyphic system used in ancient Egypt as early as 3000 B.C. Here 1 was designated by a vertical line representing a staff, 10 by an inverted u-shaped character representing a heel bone or yoke, 100 by a spiral representing a scroll or a coil of rope, 1000 by the common lotus flower, and 10000 by a finger pointing (possibly at the countless stars). For example, the number 34 would be represented by a sequence resembling ∩∩∩|||||.

The Attic numerals used in ancient Greece in about the 3rd century B.C. are another example of an additive system. The symbols used were |, Δ, H, X and M for 1, 10, 100, 1000 and 10000, respectively. The symbols other than that for 1 were from the first letters of the words *deka*, *hekaton*, *chiloi* and *myrioi* for the quantities represented. Also the symbol ⌈ which was the old form of the first letter of *pente* was used for 5 and in combination with other symbols to shorten the representation of numbers.

The best known form of additive notation is the Roman system which was similar to the ancient Greek system using letter symbols for powers of 10 and for the intermediate numbers 5, 50 and 500. The symbols used were I for 1, V for 5, X for 10, L for 50, C for 100, D for 500 and M for 1000. Thus 1969 would be written as MDCCCCLXVIIII. A subtractive notation was also used so that, for example, 4 could be written as IV as well as IIII, and 1949 as MDCCCCXLVIIII. Since the numeral X for 10 resembled for some persons the ends of a sawhorse, a ten-dollar bill was called a "sawbuck" or simply a

"saw" in early 20th-century America, and a five-dollar bill less commonly a "half-saw". A one-dollar bill is still referred to as a "buck" although this usage has also been attributed to an abbreviated form of "buckskin", a unit of trade with the American native peoples. Also a hundred-dollar bill is sometimes called a "C" or a "C-note". The designation "grand" for a thousand-dollar bill is derived not from the Roman system of numeration but from the Latin word for "grand" or "full-grown".

Multiplicative systems

In multiplicative systems there are two kinds of symbols with the symbols of one kind modifying multiplicatively the values of the second kind of symbols. An example is the traditional Chinese national numerals which originated in the Han dynasty (200 B.C. – 200 A.D.) and were later introduced into Japan and Korea where they are used today. There are symbols for the numbers 1 through 9 and for 10, 100, 1000 and 10000 which in Japanese are expressed as follows:

一	二	三	四	五	六	七	八	九
1	2	3	4	5	6	7	8	9

十	百	千	万
10	100	1000	1000

In this system the decimal number 1969 would be written as 一 千 九 百 六 十 九 and intrepeted as

(1 * 1000) + (9 * 100) + (6 * 10) + 9 .

Arithmetic tables

The *American Heritage Dictionary* defines a "scribbler" as either "one who scribbles" or "a minor or very disreputable author". The *Collins Gage Canadian Paperback Dictionary* gives these two definitions and also the definition "a notebook" which is marked as a "Canadianism". Any person who was in elementary school in Canada in the first half of the twentieth century, or possibly later, will remember these notebooks with the tables of useful information on the back cover. The tables began with addition and multiplication tables for a range of arguments from 1 to 12, inclusive. Below the Addition Tables was the following note: "By reversing the above Table Subtraction is learnt, thus: instead of saying 1 and 1 are 2, say 1 from 2 and 1 remains: 1 from 3 and 2 remains." There was a similar note on division below the Multiplication Tables. Below these tables were tables giving such information as Arabic numerals and the equivalent Roman numerals, troy weight, avoirdupois weight, days in the month ("30 days hath September, April June and November; … "), etc. Prominently displayed on the front cover of the scribbler from which this information was obtained was a Union Jack with a Boy Scout and Girl Guide standing at attention and saluting, and followed by the following advertisement: PENMAN'S / KNITTED / UNDERWEAR, HOSIERY / SWEATERS, SWIM SUITS / SPORTS GARMENTS. Scribblers are difficult to find

today but they do exist with useful information on the back cover including a multiplication table and various tables of weights and measures and metric conversion tables.

As we noted in the dice-rolling example at the end of the previous chapter arithmetic tables may be constructed very simply in **J** with the dyadic adverb / *table*. For example, the first four rows and columns of an addition table are given by the expression

```
1 2 3 4  +/ 1 2 3 4
```

which has the value

```
2 3 4 5
3 4 5 6
4 5 6 7
5 6 7 8 .
```

A bordered 12-by-12 multiplication table

```
+--+------------------------------------------+
|  | 1  2  3  4  5  6  7  8   9  10  11  12|
+--+------------------------------------------+
| 1| 1  2  3  4  5  6  7  8   9  10  11  12|
| 2| 2  4  6  8 10 12 14 16  18  20  22  24|
| 3| 3  6  9 12 15 18 21 24  27  30  33  36|
| 4| 4  8 12 16 20 24 28 32  36  40  44  48|
| 5| 5 10 15 20 25 30 35 40  45  50  55  60|
| 6| 6 12 18 24 30 36 42 48  54  60  66  72|
| 7| 7 14 21 28 35 42 49 56  63  70  77  84|
| 8| 8 16 24 32 40 48 56 64  72  80  88  96|
| 9| 9 18 27 36 45 54 63 72  81  90  99 108|
|10|10 20 30 40 50 60 70 80  90 100 110 120|
|11|11 22 33 44 55 66 77 88  99 110 121 132|
|12|12 24 36 48 60 72 84 96 108 120 132 144|
+--+------------------------------------------+
```

is given by the expression

```
x by x over x */x
```

where

```
x=: 1 2 3 4 5 6 7 8 9 10 11 12
```

and `over` and `by` are utility functions the details of which need not concern us.

Lists of integers may often be conveniently constructed by the monadic verb `i.` *integer*, and, for example, the expression `i. 5` is the list `0 1 2 3 4` of the first five non-negative integers. A list of the first five positive integers `1 2 3 4 5` is given by either of the expressions `1 + i. 5` or `>: i. 5` (but not by the expression `i. 5 + 1` which gives the list `0 1 2 3 4 5)`. Therefore the list `x` of the first twelve positive integers used in the last paragraph is given more simply by the expression `>:i. 12`.

Positional systems

In our familiar decimal or base-10 number system a number consists of a sequence of the digits 0, 1, 2, …, 9 where the value assigned to each digit depends not only on the digit itself but on its position in the sequence. For example, the four-digit number 1969 is a shorthand method of representing

$(1 \times 10^3) + (9 \times 10^2) + (6 \times 10^1) + 9 \times 10^0$

or

$(1 \times 1000) + (9 \times 100) + (6 \times 10) + 9 \times 1$

which is equal to 1969. Any integer greater than 1 may serve as a base, and in a base-b system there are b digits represented conventionally by the digits 0, 1, 2, ..., b-1. (If the base is greater than 10, then the digits greater than 9 are usually represented by the letters A, B, … .) We shall consider binary, octal, and hexadecimal systems with bases 2, 8 and 16, respectively, and give examples of the conversion between these systems and the decimal system. Finally we shall discuss the use of **J** for the representation of number systems to any base and the conversion between bases.

The binary or base-2 number 101001is equal to

$(1 \times 2^5) + (0 \times 2^4) + (1 \times 2^3) + (0 \times 2^2) + (0 \times 2^1) + (1 \times 2^0)$

or

$(1 \times 32) + (0 \times 16) + (1 \times 8) + (0 \times 4) + (0 \times 2) + (1 \times 1)$

which is equal to 32 + 8 + 1 or 41. Likewise the octal or base-8 number 7123 is equal to

$(7 \times 8^3) + (1 \times 8^2) + (2 \times 8^1) + (3 \times 8^0)$

or

$(7 \times 512) + (1 \times 64) + (2 \times 8) + (3 \times 1)$

which is 3584 + 64 + 16 + 3 or 3667. Finally the hexadecimal or base-16 number 5EA28 would be interpreted as

$(5 \times 16^4) + (14 \times 16^3) + (10 \times 16^2) + (2 \times 16^1) + (8 \times 16^0)$

where A would be interpreted as 10, B as 11, etc., which when evaluated gives the decimal number 387624.

Conversion of a decimal number to an arbitrary base is done simply by repeated integer division of the decimal number by the base and recording the integer remainder at each division. The required number is simply the remainders written down in the reverse order in which they were obtained. We will illustrate the procedure with the one example given above of the decimal number 41 and its binary equivalent 101001:

41 divided by 2 is 20 with a remainder of **1**.

20 divided by 2 is 10 with a remainder of **0**.

10 divided by 2 is 5 with a remainder of **0**.

5 divided by 2 is 2 with a remainder of **1**.

2 divided by 2 is 1 with a remainder of **0**.

1 divided by 2 is 0 with a remainder of **1**.

Therefore the decimal number is equivalent to the binary number 101001.

Working with numbers to various bases is facilitated in **J** by the two primitive verbs `#.` *base* and `#:` *antibase*. We should note that in **J** it is often convenient to represent integers by a list of their digits rather than as individual scalar numbers so that, for example, the decimal number `41` would be represented by the list `4 1` and its binary equivalent as `1 0 1 0 0 1`. The following dialogue gives some examples of the use of the base and antibase verbs:

```
   2 #. 1 0 1 0 0 1      NB. Binary to decimal conversion
41
   8 #. 7 1 2 3          NB. Octal to decimal conversion
3667
   16 #. 5 14 10 2 8     NB. Hexadecimal to decimal conversion
387624
   2 2 2 2 2 2 #: 41     NB. Decimal to binary conversion
1 0 1 0 0 1
   6$2                   NB. Shape
2 2 2 2 2 2
   (6$2) #: 41
1 0 1 0 0 1
   8 8 8 8 #: 3667       NB. Decimal to octal conversion
7 1 2 3
   (5$16) #: 387624      NB. Decimal to hexadecimal conversion
5 14 10 2 8
   24 60 60 #. 5 10 25   NB. Mixed base: Number of seconds in
18625                    NB.   5 hours, 10 minutes and 25 seconds
```

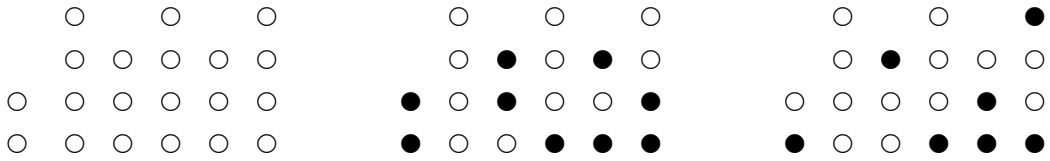
Guessing numbers

A simple game which might help introduce schoolchildren to the decimal number system and provide a little practice in arithmetic, is to ask the boy or girl to select two numbers between 1 and 9. Then have him or her perform the following simple arithmetic operations: Choose either number and multiply it by 5, add 7, double this result, and finally add the other number. The two numbers originally chosen may be found by subtracting 14 from the final result. For example, if the two numbers chosen

were 8 and 4, then the calculations could be represented as **4** + (2 × (7 + 5 × **8**)) which when rearranged is ((**8** × 10) + **4**) + 14 which is equal to **84** + 14 or 98.

The binary clock

Digital clocks which display the time in decimal digits are of course not new but recently digital clocks with a binary display have become available. One model called "Classic Powers of 2" is, according to the accompanying brochure, "based on the binary number system, the language of computers". The display consists of the rectangular array of diodes as represented in the following diagrams:



The second row of diodes – we shall come back to the first row at the end of the paragraph - gives the binary represenatation of the hour, the third row the binary representation of the minutes, and the fourth the binary represenation of the seconds. For example, the middle diagram shows a time of 10 hours, 41 minutes and 39 seconds where the black circles represent lit diodes. The first row of diodes is used for an alternate representation of the time as a binary-coded-decimal number where each decimal digit is repesented as a binary number with the first two columns representing the hour, the third and fourth columns the minutes and the fifth and sixth column the seconds. The last diagram gives the same time of 10 hours, 41 minutes and 39 seconds as a binary-coded-decimal number.

Down the rabbit-hole

An interesting example of the representation of numbers to different bases has been suggested by the experience Alice had just after she had fallen down the rabbit-hole when she was trying to determine who she was. In Chapter II of *Alice in Wonderland* we read the following:

> "I'm sure I'm not Ada," she said, "for her hair goes in such long ringlets, and mine doesn't go in ringlets at all; and I'm sure I can't be Mabel, for I know all sorts of things, and she, oh, she knows such a very little! Besides *she's* she, and *I'm* I, and – oh dear, how puzzling it all is! I'll try if I know all the things I used to know. Let me see: four times five is twelve, and four times six is thirteen, and four times seven is – oh dear! I shall never get to twenty at that rate! However, the Multiplication Table doesn't signify: ...".

One explanation of Alice's arithmetic has been quoted in Martin Gardner's *The Annotated Alice* from *The White Knight* by A. L. Taylor:

> Four times 5 actually is 12 in a number system using a base of 18. Four times 6 is 13 in a system with a base of 21. If we continue this progression, always increasing the base by 3,

our products keep increasing by one until we reach 20, where for the first time the scheme breaks down. Four times 13 is not 20 (in a number system with a base of 42), but "1" followed by whatever symbol is adopted for "10".

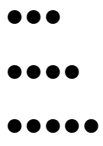
However if we do the arithmetic in **J** using the base and antibase verbs we see that Alice can actually reach 20 as seen by the following dialogue:

```
 18 18 #: 4 * 5
1 2
   10 #. 18 18 #: 4 * 5
12
   10 #. 21 21 #: 4 * 6
13
   10 #. 24 24 #: 4 * 7
14
   10 #. 27 27 #: 4 * 8
15
   10 #. 30 30 #: 4 * 9
16
   10 #. 33 33 #: 4 * 10
17
   10 #. 36 36 #: 4 * 11
18
   10 #. 39 39 #: 4 * 12
19
   10 #. 42 42 #: 4 * 13
20
```

If this last product is also expressed as `42 42 #: 4 * 13`, which has the value `1 10`, we see that Alice is correct in saying that she "shall never get to twenty" and so A. L. Taylor is correct in giving this second representation of the product.

The game of Nim

In the simplest form of the game of Nim, an ancient game thought to be of Chinese origin, twelve coins are arranged in three rows as follows:



Two players take turns alternately removing one or more coins from any one row. The winner is the person who removes the last coin. Various winning strategies have been proposed for this game, but very early in the twentieth century a method based on the binary represenation of the numbers of coins in the rows was given. It was proven to be valid for an arbitrary number of rows and an arbitrary number of coins in each row. We shall give a brief discussion of this method and illustrate it with the simple example given above.

A position, i.e., the numbers of coins in the rows, is considered "safe" after a player's move if it guarantees a win if the player continues to play judiciously; otherwise the position is considered "unsafe". Any unsafe position may be converted to a safe position, and a safe position is changed to an unsafe position by any move. A position is safe if the sum of the digits in each column of the binary representations of the numbers in the rows is either 0 or 2; otherwise the position is unsafe. In our simple example the intial position of 3, 4 and 5 coins, the binary representations and the corresponding column sums is given in the following table:

```
3 0 1 1
4 1 0 0
5 1 0 1
0 2 1 2  .
```

As the column sums given in the last row are not all 0 or 2, the position is unsafe. It may be converted to a safe position by removing two coins from the first row giving a safe position which may be represented as

```
1 0 0 1
4 1 0 0
5 1 0 1
0 2 0 2  .
```

It should be apparent that the next move must convert this position to one which is unsafe.

The above representations may be found using the **J** defined verb `Nim` given at the end of this section, and for example, `Nim 3 4 5` is

```
3 0 1 1
4 1 0 0
5 1 0 1
0 1 2 0  .
```

The following is a dialogue giving one complete play of the simple Nim game with 3, 4 and 5 coins:

```
   Nim 3 4 5   NB. Initial position
3 0 1 1
4 1 0 0
5 1 0 1
0 2 1 2
   Nim 1 4 5   NB. Player 1 takes 2 coins from 1st row
1 0 0 1
4 1 0 0
5 1 0 1
0 2 0 2
   Nim 1 4 2   NB. Player 2 takes 3 coins from 3rd row
1 0 0 1
4 1 0 0
2 0 1 0
0 1 1 1
   Nim 1 3 2   NB. Player 1 takes 1 coin from 2nd row
1 0 0 1
3 0 1 1
2 0 1 0
0 0 2 2
   Nim 1 1 2   NB. Player 2 takes 2 coins from 2nd row
1 0 0 1
1 0 0 1
2 0 1 0
0 0 1 2
   Nim 1 1 0   NB. Player 1 takes 2 coins from 3rd row
1 0 0 1
1 0 0 1
0 0 0 0
0 0 0 2
```

```
   Nim 0 1 0   NB. Player 2 takes 1 coin from 1st row
0 0 0 0
1 0 0 1
0 0 0 0
0 0 0 1
   Nim 0 0 0   NB. Player 1 takes 1 coin from 2nd row
0 0 0 0
0 0 0 0
0 0 0 0
0 0 0 0
               NB. Player 1 wins
```

The verb `Nim` used in the above example is defined as

```
Nim=:(P ,. T), 0: , [: +/ T=:2 2 2&#:@P=:] .
```

Although the details need not concern us, we might mention that `P` is the decimal number of coins in the three piles and `T` is their binary representation.

A genealogical problem

A problem that arises occasionally in genealogical work is to find the date of birth given a person's date of death and age at death. One method of calculating the date of birth is known as Formula 8870 (since the number 8870 is used in the calculation) or as the Tombstone Formula. The calculation gives only the approximate date of birth since it is assumed that all months have 30 days. In this section we shall give a couple of examples of the use of this formula and then show how the method can be expressed very simply if the date and age are expressed in a number system with the mixed base 10000 12 30.

As a first example consider that a person died on July 11, 2008 at the age of 75 years, 10 months and 25 days. According to Formula 8870 the date of birth is found as follows: Write the date of death as the eight-digit number 20080711, and the age at death similarly as 751025; subtract the second number from the first, and then subtract 8870: 20080712 – 751025 – 8870 = 19320817. Thus the date of birth is August 17, 1932. With some dates an adjustment must be made in the calculated date of birth. For example, with the same date of death and with an age at death of 75 years, 10 months and 5 days, the calculation becomes 20080712 – 751005 – 8870 = 19320837. If we interpret this date as August 37, 1932, we may arrive at the correct date by subtracting 30 days from the day number and adding 1 to the month number to obtain 19320907 which represents a date of September 7, 1932. In some calculations adjustments must be made to the month number or to both the day number and the month number.

The above calculations may be expressed very simply and the use of the constant 8870 avoided if the dates are considered as numbers to the mixed base 10000 12 30 as shown by the following calculations for the first example given in the last paragraph:

```
   a=: 10000 12 30 #. 2008 7 12  NB. Date of death
   a
723102
   b=: 10000 12 30 #. 75 10 25   NB. Age at death
   b
27325
   10000 12 30 #: a - b          NB. Date of birth
1932 8 17
```

The calculations may be simplified by the use of two defined functions as illustrated in the following examples:

```
   DateNum=: 10000 12 30&#.
   BirthDate=: [: 10000 12 30&#: (DateNum@[) - DateNum@]
   2008 7 12 BirthDate 75 10 25
1932 8 17
   2008 7 12 BirthDate 75 10 5
1932 9 7
```

In some instances an adjustment must be made in the calculated date of birth because of the 0-origin indexing used in **J** and in positional number systems. For example, an age at death of 75 years, 6 months and 15 days with the same date of death as used in the previous examples gives the expression

```
   2008 7 12 BirthDate 75 6 15
```

which has the value `1933 0 27` which is interpreted as December 27, 1932 which agrees with the date given by Formula 8870.

A closer look at multiplication

The customary pencil-and-paper method of multiplying numbers is shown in the following example which gives the product of 472 and 1963 which is equal to 926536:

```
      1 9 6 3
        4 7 2
      --------
      3 9 2 6
  1 3 7 4 1
  7 8 5 2
 ------------
  9 2 6 5 3 6
```

This method of multiplication tends to be an error-prone operation as carry digits may be required in each digit-by-digit multiplication in each of the three rows of partial products and again in the summation of these rows.

One very popular method of simplifying multiplication which probably originated in India and was introduced into Europe at the end of the fifteenth century was known as *galosia*. In this method the products of all pairs of digits were displayed in a rectangular array and then summed diagonally as indicated in the following diagram:

```
        1     9     6     3
     ----------------------
  0  0     3     2     1    |   4
     4     6     4     2    |
     ----------------------|
  9  0     6     4     2    |   7
     7     3     2     1    |
     ----------------------|
  2  0     1     1     0    |   2
     2     8     2     6    |
     ----------------------|
        6     5     3     6
```

The diagonal sums starting at the lower right corner may be calculated as follows:

6 = 6

3 = 1 + 0 + 2

5 = 2 + 2 + 2 + 1 + 8 = 5, and a carry of 1

6 = 1 + 1 + 4 + 4 + 3 + 1 + 2, and a carry of 1

2 = 1 + 2 + 6 + 6 + 7 + 0, and a carry of 2

9 = 2 + 3 + 4

0 = 0

The name comes from the similarity of the format to a blind or shutter with adjustable horizontal slats for regulating the passage of air and light, and the English *jalousie* with the same meaning is derived from it.

For another look at this example let us represent the first number by the list `a=: 4 7 2` with the list `p1=: 100 10 1` representing the associated powers of 10, and the second number similarly by the lists `b=: 1 9 6 3` and `p2=: 1000 100 10 1`. We now construct the following two tables of

`a */ b` where the table on the left has borders corresponding to the pairs of digits being multiplied and the one on the right has borders of the corresponding powers of ten:

```
a by b over a */ b       p1 by p2 over a */ b
+-+----------+           +---+--------------+
| |1  9  6  3|           |   |1000 100 10  1|
+-+----------+           +---+--------------+
|4|4 36 24 12|           |100|   4  36 24 12|
|7|7 63 42 21|           | 10|   7  63 42 21|
|2|2 18 12  6|           |  1|   2  18 12  6|
+-+----------+           +---+--------------+
```

To see the relationship of the table `a */ b` with the familiar method of multiplication given by the example at the beginning of this section consider the following step-by-step normalization of its rows:

```
4 36 24 12      7 63 42 21     2 18 12 6

4 36 25 2       7 63 44 1      2 19 2 6

4 38 5 2        7 67 4 1       3 9 2 6

7 8 5 2         13 7 4 1

              1 3 7 4 1
```

The last rows are the partial products which must be added, again with appropriate carry digits, to get the desired product. These products may be found simply by use of the base function `#.` and we have `10 #. a */ b` is equal to `7852 13741 3926`.

The right-hand table above shows that the items of `a */ b` associated with the same powers of ten lie along the diagonals of the table. These diagonals are

```
+-+----+-------+--------+-----+-+
|4|36 7|24 63 2|12 42 18|21 12|6|
+-+----+-------+--------+-----+-+
```

and are given by the expression `</.a*/b`, where `/.` is the adverb *oblique* and `<` is the monadic verb *box*. The expression `+//.a*/b` gives the sums of the diagonals, `4 43 89 72 33 6`, which when normalized has the value `926536`.

Finally in this section we shall mention the “Russian peasant” method of multiplication which requires the successive doubling and halving of the two numbers being multiplied. For example to find the product of 37 and 41 we successively halve 37 discarding the remainders while doubling 41 as shown in the following table

```
37  18    9    4    2    1
41  82   164  328  656  1312
```

and then finding the sum of those numbers in the second row corresponding to odd numbers in the first row, i.,e., 41 + 164 + 1312, which is equal to 1517, the required product. In this method we are finding

the binary representation of the first number, 1 0 0 1 0 1, and then selecting the products of the second number with the appropriate powers of 2:

```
   41 * 37
   +/41 * 1 0 0 1 0 1 * 2^5 4 3 2 1 0
   +/41 * 1 0 0 1 0 1 * 32 16 8 4 2 1
   +/41 * 32 4 1
   +/1312 164 41
   1517
```

Rolling more dice

We shall conclude this chapter, as we did the last, with some remarks on the use of **J** in rolling dice although it is difficult to see any connection between dice and positional number systems. In this section we shall consider dice other than those with just four faces and introduce a primitive **J** verb which will allow us to simulate the rolling of dice.

Dice may be constructed in the shape of any one of the five regular polyhedral, viz., a tetrahedron with 4 faces, a hexahedron with 6 faces which corresponds to our familiar die, an octahedron with 8 faces, a dodecahedron with 12 faces, and an icosahedron with 20 faces. The rolling of dice, with an arbitrary number of faces and not just the five just given, may be simulated using the monadic verb `?` *roll* as shown in the following dialogue:

```
   ?10                NB. Random integer from the list 0, 1, 2,…, 9
4
   ?10
5
   10$6
6 6 6 6 6 6 6 6 6 6
   ?10$6              NB. 10 random integers from the list 0, 1,..., 5
3 4 0 2 0 2 4 3 5 5
   >:?10$6            NB. 10 random integers from the list 1, 2,..., 6
4 1 4 3 5 6 5 2 1 5   NB.    simulating throwing a six-sided die
   >:?10$6
2 4 5 6 3 2 6 5 5 4
```

In order to restrict the faces considered by the defined **J** verb to those corresponding to the regular polyhedra we shall make use of the verb `e.` *member* illustrated in the following dialogue:

```
   8 e. 4 6 8 12 20
1
   20 e. 4 6 8 12 20
1
   15 e. 4 6 8 12 20
0
```

The following defined verb `Dice` simulates the rolling of a die with an arbitrary number of faces an arbitrary number of times. The result is a table with two rows with the face numbers in the first row and the corresponding frequencies in the second row. If the number of faces is not valid, then the result is an empty table with 2 rows and 0 columns.

```
   Dice=: 3 : 0
   :
   Nfaces=: x.
   Nrolls=: y.
   if. Nfaces e. 4 6 8 12 20 do.
      Faces=: >:i.Nfaces
      R=: Faces ,: +/"1 Faces =/ >:?Nrolls$Nfaces
   else.
   R=: i. 2 0
   end.
   )
   6 Dice 25
1 2 3 4 5 6
5 2 3 3 6 6
   6 Dice 25
1 2 3 4 5 6
2 5 5 7 2 4
   T=: 9 Dice 40
   T


   $T
2 0
```

```
   20 Dice 500
 1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20
26 26 22 25 24 27 20 24 29 30 40 26 15 36 27 20 24 17 24 18
   20 Dice 500
 1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20
20 20 37 21 24 24 20 38 28 24 14 24 27 32 20 24 35 25 21 22
```