S T E P N U M B E R S

Introduction

In this work we challenge the hegemony of the binary system and advocate the introduction of its dual, the stepnumber system utilizing infinitely many digits and variable positional values for them. These two stand out as the only natural number systems — natural in a sense that they do not depend on arbitrary choices such as a base. The binary number system is the simplest in that it calls for the smallest number of digits. At the same time it is also most uneconomical: it takes the longest string of digits to write a number in binary form. By contrast, the stepnumber system calls for infinitely many digits. It is also the most economical: it takes the shortest string of digits to write a number n, sufficiently large, in stepnumber form. It is most economical in a second sense as well. The next larger digit is not pressed into service until the full potential of the lower digits has been exhausted. Furthermore, after the new digit puts in an appearance, it is being used most sparingly. By contrast, Cantor's number system (that also calls for infinitely many digits) avails itself of higher digits unnecessarily early, and then makes heavy use of them. In this way the practical usefulness of the number system calling for infinitely many digits is unnecessarily reduced.

The binary numbers **0**, **1**, **10**, **11**, **100**, **101**, **110**, **111**, **1000**,... will be printed in bold face type. This innovation lets us write equalities and inequalities such as $2 = \mathbf{10} \neq 10$, **111** = 7 and 8 = **1000**, **1000** < 10 < **10000**. Binary numbers can be introduced heuristically by enumerating them under the lexicographic order, subject to the following restrictions:

(1) The Binary Principle states that digits are limited to **0** and **1**;

(2) The Overflow Principle states that in increasing a maximal digit by one we replace it with **0** and increase the adjacent digit to the left by one.

The extreme case is the Overflow Formula: $\mathbf{1}_n + \mathbf{1} = \mathbf{10}_n$ where $\mathbf{1}_n$ and $\mathbf{0}_n$ mean that the digit **1** or **0** is to be repeated n times, e.g., $\mathbf{1}_2\mathbf{0} = \mathbf{110}$, $\mathbf{10}_2 = \mathbf{100}$, $\mathbf{1}_2\mathbf{0}_3 = \mathbf{11000}$. We also have: $\mathbf{10}_m \times \mathbf{10}_n = \mathbf{10}_{m+n}$. Thus **1**, **11**, **111**,... play the same role in the binary number system as 9, 99, 999,... do in the decimal: they trigger the Overflow Formula, e.g., 999 + 1 = 1000; **111 + 1 = 1000**. The rule of enumeration endows the binary number system with a natural order of magnitude. The value of a binary number is just its ordinal under the natural order.

The table on the next page displays the first one hundred consecutive binary numbers in lexicographic order. The arrangement in ten columns helps finding their value quickly. For example, we have **1111** = 15 because **1111** stands in the first row and fifth column (the numbering of rows and columns starts with zero); **100000** = 32 as it stands in the third row, second column. The binary numbers **1**, **10**, **100**,... are just the powers of 2: $\mathbf{10}_n = 2^n$.

Table of the first one hundred consecutive binary numbers

0	**1**	**10**	**11**	**100**	**101**	**110**	**111**	**1000**	**1001**
1010	**1011**	**1100**	**1101**	**1110**	**1111**	**10000**	**10001**	**10010**	**10011**
10100	**10101**	**10110**	**10111**	**11000**	**11001**	**11010**	**11011**	**11100**	**11101**
11110	**11111**	**100000**	**100001**	**100010**	**100011**	**100100**	**100101**	**100101**	**100110**
100111	**101000**	**101001**	**101010**	**101011**	**101100**	**101101**	**101110**	**101111**	**110000**
110001	**110011**	**110100**	**110101**	**110110**	**110111**	**111000**	**111001**	**111010**	**111011**
111100	**111101**	**111110**	**111111**	**1000000**	**1000001**	**1000010**	**1000011**	**1000100**	**1000101**
1000110	**1000111**	**1001000**	**1001001**	**1001010**	**1001011**	**1001100**	**1001101**	**1001110**	**1001111**
1010000	**1010001**	**1010010**	**1010011**	**1010100**	**1010101**	**1010110**	**1010111**	**1011000**	**1011001**
1011010	**1011011**	**1011100**	**1011101**	**1011110**	**1011111**	**1100000**	**1100001**	**1100010**	**1100011**

The table can also be used for performing addition and subtraction of binary numbers, following the Slide Rule Principle. For example, in calculating the sum **10011** + **1001111** we locate the larger number in the table and move forward through a number of steps that corresponds to the smaller number **10011** = 19 to get **10011** + **1001111** = **1100010**. To check, we write: 79 + 19 = 98 = **1100010**, the binary number in row 9 and column 8. In calculating the difference **1001000 – 11100** we move backwards from **1001000** through **11100** = 28 steps to get **101011**. To check we write: 72 – 28 = 44 = **101011**, the binary number in row 4 and column 4.

There is a natural isomorphism between the binary number system and the graded lattice of finite subsets. In more details, consider $\mathbb{N} = \{1, 2, 3, ..., n, ...\}$, the set of natural numbers, and its subsets $\mathbb{N}_n = \{1, 2, 3, ..., n\}$. $\mathbb{N}_n$ form a lattice for every $n = 1, 2, 3, ...$ under inclusion. The union of these is a graded lattice with grading furnished by n, the number of digits. A binary number of at most n digits can be interpreted as a characteristic function of $\mathbb{N}_n$, provided that we add a sufficient number of **0** digits up front to make it a binary number of exactly n digits (which, of course, will not affect its value). This is a one-to-one correspondence between binary numbers and finite subsets. It can be extended to a lattice isomorphism. It does not depend on arbitrary choices. In enumerating binary numbers we actually construct all finite subsets. Thus binary numbers can be used to count the number of subsets. The great utility of the binary number system is due mostly to this fact, which is not widely recognized. The binary number system owes its naturality to the existence of this natural isomorphism.

* * *

In order to be able to count in the binary number system we introduce names for the two digits: **0 ala, 1 ale**. We also introduce names for the numbers $\mathbf{10}_n$ called ‘milestones’ as follows. We adopt a variation of the paradigm million, billion, trillion, etc. modified and simplified by using permutations of the five vowels **a, e, i, o, u**. Using the names of the milestones on the following page we can count in the binary system on the same principle as we do in the decimal system, thus:

0 ala, 1 ale, 10 mala, 11 mala ale, 100 bala, 101 bala ale, 110 bala mala, 111 bala mala ale, 1000 trala, 1001 trala ale, 1010 trala mala, 1100 trala bala, 1101 trala bala ale, 1110 trala bala mala, 1111 trala bala mala ale, 10000 quadrala, 10001 quadrala ale, etc.

NAMES OF MILESTONES

10	**mala**
10_{2}	**bala**
10_{3}	**trala**
10_{4}	**quadrala**
10_{5}	**pentala**
10_{6}	**hexala**
10_{7}	**heptala**
10_{8}	**octala**
10_{9}	**novala**
10_{10}	**hale**
10_{11}	**male**
10_{12}	**bale**
10_{13}	**trale**
10_{14}	**quadrale**
10_{15}	**pentale**
10_{16}	**hexale**
10_{17}	**heptale**
10_{18}	**octale**
10_{19}	**novale**
10_{20}	**hali**
10_{21}	**mali**
10_{22}	**bali**
10_{23}	**trali**
10_{24}	**quadrali**
10_{25}	**pentali**
10_{26}	**hexali**
10_{27}	**heptali**
10_{28}	**octali**
10_{29}	**novali**
10_{30}	**halo**
10_{31}	**malo**
10_{32}	**balo**
10_{33}	**tralo**
10_{34}	**quadralo**
10_{35}	**pentalo**
10_{36}	**hexalo**
10_{37}	**heptalo**
10_{38}	**octalo**
10_{39}	**novalo**
10_{40}	**halu**
10_{41}	**malu**
10_{42}	**balu**
10_{43}	**tralu**
10_{44}	**quadralu**
10_{45}	**pentalu**
10_{46}	**hexalu**
10_{47}	**heptalu**
10_{48}	**octalu**
10_{49}	**novalu**
10_{50}	**halla**
10_{51}	**malla**
10_{52}	**balla**
10_{53}	**tralla**
10_{54}	**quadralla**
10_{55}	**pentalla**
10_{56}	**hexalla**
10_{57}	**heptalla**
10_{58}	**octalla**
10_{59}	**novalla**
10_{60}	**halle**
10_{61}	**malle**
10_{62}	**balle**
10_{63}	**tralle**
10_{64}	**quadralle**
10_{65}	**pentalle**
10_{66}	**hexalle**
10_{67}	**heptalle**
10_{68}	**octalle**
10_{69}	**novalle**
10_{70}	**halli**
10_{71}	**malli**
10_{72}	**balli**
10_{73}	**tralli**
10_{74}	**quadralli**
10_{75}	**pentalli**
10_{76}	**hexalli**
10_{77}	**heptalli**
10_{78}	**octalli**
10_{79}	**novalli**
10_{80}	**hallo**
10_{81}	**mallo**
10_{82}	**ballo**
10_{83}	**trallo**
10_{84}	**quadrallo**
10_{85}	**pentallo**
10_{86}	**hexallo**
10_{87}	**heptallo**
10_{88}	**octallo**
10_{89}	**novallo**
10_{90}	**hallu**
10_{91}	**mallu**
10_{92}	**ballu**
10_{93}	**trallu**
10_{94}	**quadrallu**
10_{95}	**pentallu**
10_{96}	**hexallu**
10_{97}	**heptallu**
10_{98}	**octallu**
10_{99}	**novallu**

Second order milestones are the stepnumbers $\mathbf{10}_{10n}$. Their names are as follows:

NAMES OF SECOND ORDER MILESTONES

		10_{500}	**halya**
10_{10}	**hale**	10_{510}	**halye**
10_{20}	**hali**	10_{520}	**halyi**
10_{30}	**halo**	10_{530}	**halyo**
10_{40}	**halu**	10_{540}	**halyu**
10_{50}	**halla**	10_{550}	**hallya**
10_{60}	**halle**	10_{560}	**hallye**
10_{70}	**halli**	10_{570}	**hallyi**
10_{80}	**hallo**	10_{580}	**hallyo**
10_{90}	**hallu**	10_{590}	**hallyu**
10_{100}	**hela**	10_{600}	**helya**
10_{110}	**hele**	10_{610}	**helye**
10_{120}	**heli**	10_{620}	**helyi**
10_{130}	**helo**	10_{630}	**helyo**
10_{140}	**helu**	10_{640}	**helyu**
10_{150}	**hella**	10_{650}	**hellya**
10_{160}	**helle**	10_{660}	**hellye**
10_{170}	**helli**	10_{670}	**hellyi**
10_{180}	**hello**	10_{680}	**hellyo**
10_{190}	**hellu**	10_{690}	**hellyu**
10_{200}	**hila**	10_{700}	**hilya**
10_{210}	**hile**	10_{710}	**hilye**
10_{220}	**hili**	10_{720}	**hilyi**
10_{230}	**hilo**	10_{730}	**hilyo**
10_{240}	**hilu**	10_{740}	**hilyu**
10_{250}	**hilla**	10_{750}	**hillya**
10_{260}	**hille**	10_{760}	**hillye**
10_{270}	**hilli**	10_{770}	**hillyi**
10_{280}	**hillo**	10_{780}	**hillyo**
10_{290}	**hillu**	10_{790}	**hillyu**
10_{300}	**hola**	10_{800}	**holya**
10_{310}	**hole**	10_{810}	**holye**
10_{320}	**holi**	10_{820}	**holyi**
10_{330}	**holo**	10_{830}	**holyo**
10_{340}	**holu**	10_{840}	**holyu**
10_{350}	**holla**	10_{850}	**hollya**
10_{360}	**holle**	10_{860}	**hollye**
10_{370}	**holli**	10_{870}	**hollyi**
10_{380}	**hollo**	10_{880}	**hollyo**
10_{390}	**hollu**	10_{890}	**hollyu**
10_{400}	**hula**	10_{900}	**hulya**
10_{410}	**hule**	10_{910}	**hulye**
10_{420}	**huli**	10_{920}	**hulyi**
10_{430}	**hulo**	10_{930}	**hulyo**
10_{440}	**hulu**	10_{940}	**hulyu**
10_{450}	**hulla**	10_{950}	**hullya**
10_{460}	**hulle**	10_{960}	**hullye**
10_{470}	**hulli**	10_{970}	**hullyi**
10_{480}	**hullo**	10_{980}	**hullyo**
10_{490}	**hullu**	10_{990}	**hullyu**

For example, we have: 10_{401} **mula,** 10_{402} **bula,** 10_{403} **trula,** 10_{404} **quadrula,** 10_{405} **pentula,** 10_{406} **hexula,** 10_{407} **septula,** 10_{408} **octula,** 10_{409} **novula**; 10_{410} **hule**, 10_{411} **mule,** 10_{412} **bule,** etc.

Table of the first three hundred consecutive stepnumbers

0	**1**	**10**	**11**	**12**	**100**	**101**	**102**	**110**	**111**
112	**120**	**121**	**122**	**123**	**1000**	**1001**	**1002**	**1010**	**1011**
1012	**1020**	**1021**	**1022**	**1023**	**1100**	**1101**	**1102**	**1110**	**1111**
1112	**1120**	**1121**	**1122**	**1123**	**1200**	**1201**	**1202**	**1203**	**1210**
1211	**1212**	**1213**	**1220**	**1221**	**1222**	**1223**	**1230**	**1231**	**1232**
1233	**1234**	**10000**	**10001**	**10002**	**10010**	**10011**	**10012**	**10020**	**10021**
10022	**10023**	**10100**	**10101**	**10102**	**10110**	**10111**	**10112**	**10120**	**10121**
10122	**10123**	**10200**	**10201**	**10202**	**10203**	**10210**	**10211**	**10212**	**10213**
10220	**10221**	**10222**	**10223**	**10230**	**10231**	**10232**	**10233**	**10234**	**11000**
11001	**11002**	**11010**	**11011**	**11012**	**11020**	**11021**	**11022**	**11023**	**11100**
11101	**11102**	**11110**	**11111**	**11112**	**11120**	**11121**	**11122**	**11123**	**11200**
11201	**11202**	**11203**	**11210**	**11211**	**11212**	**11213**	**11220**	**11221**	**11222**
11223	**11230**	**11231**	**11232**	**11233**	**11234**	**12000**	**12001**	**12002**	**12003**
12010	**12011**	**12012**	**12013**	**12020**	**12021**	**12022**	**12023**	**12030**	**12031**
12032	**12033**	**12034**	**12100**	**12101**	**12102**	**12103**	**12110**	**12111**	**12112**
12113	**12120**	**12121**	**12122**	**12123**	**12130**	**12131**	**12132**	**12133**	**12134**
12200	**12201**	**12202**	**12203**	**12210**	**12211**	**12212**	**12213**	**12220**	**12221**
12222	**12223**	**12230**	**12231**	**12232**	**12233**	**12234**	**12300**	**12301**	**12302**
12303	**12304**	**12310**	**12311**	**12312**	**12313**	**12314**	**12320**	**12321**	**12322**
12323	**12324**	**12330**	**12331**	**12332**	**12333**	**12334**	**12340**	**12341**	**12342**
12343	**12344**	**12345**	**100000**	**100001**	**100002**	**100010**	**100011**	**100012**	**100020**
100021	**100022**	**100023**	**100100**	**100101**	**100102**	**100110**	**100111**	**100112**	**100120**
100121	**100122**	**100123**	**100200**	**100201**	**100202**	**100203**	**100210**	**100211**	**100212**
100213	**100220**	**100221**	**100222**	**100223**	**100230**	**100231**	**100232**	**100233**	**100234**
101000	**101001**	**101002**	**101010**	**101011**	**101012**	**101020**	**101021**	**101022**	**101023**
101100	**101101**	**101102**	**101110**	**101111**	**101112**	**101120**	**101121**	**101122**	**101123**
101200	**101201**	**101202**	**101203**	**101210**	**101211**	**101212**	**101213**	**101220**	**101221**
101222	**101223**	**101230**	**101231**	**101232**	**101233**	**101234**	**102000**	**102001**	**102002**
102003	**102010**	**102011**	**102012**	**102013**	**102020**	**102021**	**102022**	**102023**	**102030**
102031	**102032**	**102033**	**102034**	**102100**	**102101**	**102102**	**102103**	**102110**	**102111**

The question arises whether there exist other natural number systems beside the binary. The decimal number system is clearly not natural as it depends on the arbitrary choice of the base 10. If we want to change it, we can go in either one of two directions. If we lower it, then fewer digits will suffice at the price of having to cope with longer strings of digits. If we increase it, then we get shorter strings of digits at the price of having to cope with a larger inventory of digits. In exercising the first choice we go to the extreme and get the simplest number system using only two digits: **0** and **1**. This is the natural number system of the binary numbers. However, simplicity comes at a price: the string of digits has maximal length. All other number systems are more economical in that they work with shorter strings of digits. As one may expect, further development of science will necessitate the use of ever larger numbers. At one point the price we pay for simplicity may become prohibitive. The length of strings of digits may outpace the memory and manipulative capabilities of the best computers for very large numbers. The binary number system may well become obsolete. We should not be too complacent in this regard.

The other choice is to increase the base. As we do, the string of digits becomes shorter, but at the price that we shall need ever more digits. Can we minimize the string of digits for very large numbers? Does it make sense to talk about infinitely many digits? It turns out that the answer to these questions is "yes". There is another natural number system at the far end of the rainbow: that of the stepnumbers. It minimizes the string of digits and so it is ideally suited for calculations with very large numbers.

The same conventions will be used for the stepnumbers as we have introduced for the binary numbers. In particular, we also print them in bold-face type: **1**, **10**, **11**, **12**, **100**, **101**, **102**, **110**, **111**, **112**, **120**, **121**, **122**, **123**, **1000**,... Thus we may write 5 = **100** $\neq$ 100, **123** = 14, 14 + 1 = 15 = **1000**. The convention for subscripts applies, e.g., $\mathbf{10}_2$ = **100**, $\mathbf{120}_3$ = **12000**, $\mathbf{1_3} = \mathbf{111}$, $\mathbf{1_3 2_2} = \mathbf{11122}$.

We shall also write $\boldsymbol{k!} = \mathbf{123}...\boldsymbol{k}$ where $\boldsymbol{k}$ is the k^{th} digit for every natural number k. Remember that $\boldsymbol{n!} \neq n!$ We also use the notation $\boldsymbol{k!}\mathbf{0}_{n-k}$ for the n-digit stepnumber $\mathbf{123}...\boldsymbol{k}\mathbf{00}...\mathbf{0}$ where the number of **0** digits is $n - k$. By convention **0!** = **0**. Note that $\boldsymbol{k!}$ can be characterized as the largest stepnumber with k digits, whereas the smallest one with k digits is $\mathbf{10}_{k-1} = b_k$, the k^{th} Bell number, see Chapter 4. The consecutive Bell numbers are: b_1 = 1, b_2 = **10** = 2, b_3 = **100** = 5, b_4 = **1000** = 15, b_5 = **10000** = 52, $b_6 = \mathbf{10}_5$ = 203, b_7 = 877, b_8 = 4140, b_9 = 21,147, b_{10} = 115,975, b_{11} = 678,570, b_{12} = 4,213,597,… A table of the Bell numbers can be found at the end of this book.

In Chapter 1 we shall continue the sequence of the ten decimal digits by adding infinitely many digits, the so-called rainbow digits. They are the decimal digits written in the colors of the rainbow. The table on the previous page lists the first three hundred consecutive stepnumbers in lexicographic order. Stepnumbers, just as binary numbers, can be introduced heuristically by enumerating them under the lexicographic order subject to the following restrictions.

(1) The Stepnumber Principle states that no digit can be higher than the highest digit to its left plus 1;

(2) the Overflow Principle states that in increasing a maximal digit by 1 we replace it with **0** and add 1 to the adjacent digit to the left.

The extreme case is furnished by the Overflow Formula: $\boldsymbol{n!} + \mathbf{1} = \mathbf{10}_n$ which is triggered by the stepnumbers **1**, **12**, **123**,..., e.g., **123** + **1** = **1000**.

The rule of enumeration endows the stepnumber system with a natural order of magnitude. The value of a stepnumber is just its ordinal under the natural order. The arrangement in ten columns is designed to help finding the value of stepnumbers in the table quickly. For example, we have **1111** = 29 because **1111** stands in the second row and ninth column (the numbering of rows and Just as for binary numbers, the table for stepnumbers can also be used for performing addition and subtraction, following the Slide Rule Principle. For example, in calculating the sum **100** + **10000** we locate the larger number and from there move forward through **100** = 5 steps to get: **100** + **10000** = **10012**. To check we write 5 + 52 = 57 = **10012**. In calculating the difference **101233 – 12233**, in the table we move backwards from **101233** through **12233** = 176 steps to get: **101233 – 12233** = **11100**. To check we write: 276 – 176 = 100 = **11100**.

The stepnumbers form another number system that is natural in the sense that there is a natural isomorphism between the stepnumber system and the graded lattice of finite quotient sets. In more details, consider $\mathbb{N} = \{1, 2, 3,..., n,...\}$, the set of natural numbers and its subsets $\mathbb{N}_n = \{1, 2, 3,..., n\}$ of n elements. The quotient sets of $\mathbb{N}_n$ form a lattice under rarefaction, the opposite of refinement (see Appendix). The union of the lattices $\mathbb{N}_n$ for all n is a graded lattice. Grading is furnished by the number n of elements of $\mathbb{N}_n$, i.e., the number of digits. Every stepnumber of at most n digits can be interpreted as a characteristic function of $\mathbb{N}_n$, provided that we add a sufficient number of **0** digits up front to make it a stepnumber of exactly n digits (this of course will not affect its value). There is a one-to-one correspondence between the set of stepnumbers and the graded lattice of finite quotient sets. The former is endowed by a lattice structure under this correspondence, and we shall refer to it as the natural isomorphism between the stepnumber system and the graded lattice of finite quotient sets. It is also natural in that it does not depend on arbitrary choices such as, *e.g.*, the basis of the number system.

Thus the stepnumber system is dual to its analogue, the binary number system. In enumerating stepnumbers we actually construct all finite quotient sets. For example, there are 15 quotient sets of a set of 4 elements, and there are 15 stepnumbers of at most 3 digits:

0, **1**, **10**, **11**, **12**, **100**, **101**, **102**, **110**, **111**, **112**, **120**, **121**, **122**, **123**

In writing them in up-front-**0** notation, they become uniformly 4-digit stepnumbers:

0000, **0001**, **0010**, **0011**, **0012**, **0100**, **0101**, **0102**, **0110**, **0111**, **0112**, **0120**, **0121**, **0122**, **0123**

exhibiting the quotient sets of the set of the four places of 4-digit stepnumbers, where the classes are marked by the digit **0**, **1**, **2**, **3**.

Stepnumbers can be used to count the number of finite quotient sets. This new number system is also expected to have great utility, especially in information technology. Indeed, it may well be the number system of the future to encode information by quantum computers. Unlike the present generation of computers which are based on classical physics, quantum computers are based on quantum mechanics. In particular, the state of an electron of the atom is the new bit, or unit of information, corresponding to a stepnumber digit. The variable admissible orbits of electrons in an atom, which are discrete and ordered according to increasing radius, epitomize the variable positional values of digits in the stepnumber system.

The stepnumber system overcomes the handicap of the binary of being unwieldy due to the inordinate length of the strings of digits for very large numbers. Writing a sufficiently large number in stepnumber form will take fewer digits than it does in the binary or in any other number system with base n, however large n may be. This property is expressed by saying that the stepnumber system enables one to write very large numbers in their *most compact* form.

In the decimal system we are forced to do rounding at the expense of accuracy every time an extra large number presents itself. Rounding is made unnecessary, and there is no loss of accuracy, if we are using the stepnumber system.

In order to be able to count in the stepnumber system we introduce names for the digits. For the first eleven, these are:

0 ala, 1 ale, 2 ali, 3 alo, 4 alu, 5 alla, 6 alle, 7 alli, 8 allo, 9 allu, 10 mala

This already suggests that, for the Bell numbers, we shall retain the names for the milestones and also that of the secondary milestones (see pp 3, 4):

10 mala, 10_2 bala, 10_3 trala, 10_4 quadrala, 10_5 pantala, 10_6 hexala, 10_7 heptala, 10_8 octala,…
10_{10} hale, 10_{20} hali, 10_{30} halo, 10_{40} halu, 10_{50} halla, 10_{60} halle, 10_{70} halli, 10_{80} hallo, 10_{90} hallu,…

In Chapter 1 we shall see the rule how to obtain the names of the rest of the (infinitely many) rainbow digits. Here, once again, the permutations of the five vowels **a, e, i, o, u** will play a role. Counting in the stepnumber system uses a different principle from that used in the decimal and the binary systems. The reason for this is the variable place value of the digits, a feature of stepnumbers unknown for the decimals and the binary numbers. To count, we read out the names of the digits from left to right, including **0 ala**, and repeating if necessary, thus:

0 ala, 1 ale, 10 mala, 11 ale ale, 12 ale ali, 100 bala, 101 ale ala ale, 102 ale ala ale, 110 ale ale ala, 111 ale ale ale, 112 ale ale ali, 120 ale ale ali, 122 ale ali ali, 123 ale ali alo, 1000 trala, 1001 ale ala ala ale, 1002 ale ala ala ali, 1010 ale ala ale ala, 1011 ale ala ale ale,…

*   *   *

One of the great remaining mysteries in mathematics is the question whether a formula exists whereby consecutive prime numbers can be calculated. The conjecture is still outstanding that such a formula does not exist in the same sense as the set of all sets doesn't. If this is the case, then the best one can hope for is a rapid development of various algorithms to find ever larger prime numbers. This raises the possibility of decoupling between *substance* and *form*. Finding the substance, namely, ever larger prime numbers, may outpace the form in which the new information can be put. Information technology, in particular the binary number system, may be an obstruction. The physical limitations of computer memory could defeat our efforts to forge ahead with the theory, in want of a better facility to handle very large numbers. The quantum computer reinforced with the stepnumber system fitting as a glove opens up new perspectives in computing.

Philosophy has been grappling with the dichotomy of substance and form for thousands of years. It tacitly assumes that they are inseparable as guardians of the depository of knowledge. Decoupling would most certainly create a crisis of the first magnitude. The discovery of stepnumbers opens a new chapter in philosophy in that it extends the tenure of the partnership of substance and form in storing, transmitting, retrieving, or otherwise manipulating and organizing information.