-

**THE EXCITING UNIVERSE**

**OF SUPER LARGE NUMBERS!!!**

**A BRIEF REVIEW**

David Galilei Natale

May 2024

The purpose of what you are about to read is to review some of the mathematical notations for the expression of very large numbers, and then expand on them to discover phenomenally greater numbers! In 1995, Clifford Pickover defined the superfactorial as follows:

n!

.

. (n! terms)

n! .

n!

n$ = n!

In this definition, n$ “associates to the right.”

This means for example:

3!)))))

(3!

(3!

(3!

(3!

3$ = (3!

I will now define ‘exploding superfactorial’ as follows:

(n$)$. . . $

. ^ ^ ^

| | |

1 2 (Z$)$

(Z$)$ . . . $ # of Towers

^ ^ ^

. | | |

1 2 (Z$)$ . . . $

^ ^ ^

| | |

1 2 (Z$)$ . . . $

^ ^ ^

| | |

1 2 (Z$)$ . . . $

^ ^ ^

| | |

. 1 2 (Z$)$ . . . $

(n$)$. . . $ ^ ^ ^

(n$)$. . . $ ^ ^ ^ | | |

nEx$ = (n$)$. . . $ ^ ^ ^ | | | 1 2 Z$$$

^ ^ ^ || |1 2 (Z$)$

| | | 1 2(Z$)$

1 2 (Z$)$

Where Z is defined on the following 2 pages:

Z = (n$)$. . . $

^ ^ ^

| | |

1 2 (n$)$ . . . $ = 1 a = 1 y

^ ^ ^

| | |

1 2 (n$)$ . . . $ = 2 a

^ ^ ^

.

.

.

(n$)$ . . . $ = (n$)$ . . . $ a

^ ^ ^ ^ ^ ^

| | | | | |

1 2 (n$)$ 1 2 (n$)$. . . $ = 1 b = 2 y

^ ^ ^

| | |

1 2 (n$)$ . . .

Keep on going and eventually you’ll reach:

(n$)$ . . . $ = (n$)$ . . . $ b

^ ^ ^ ^ ^ ^

| | | | | |

1 2 (n$)$ 1 2 (n$)$ . . . $ = 1 c = 3 y

^ ^ ^

| | |

1 2 (n$)$ . . .

Keep on going and eventually you’ll reach:

(n$)$ . . . $ = (n$)$ . . . $ G

^ ^ ^ ^ ^ ^

| | | | | |

1 2 (n$)$ 1 2 (n$)$ . . . $ = 1 q = (n$)$ . . . $ y

^ ^ ^ ^ ^ ^

| | | | | |

1 2 (n$)$ 1 2 (n$)$ . . . $ = 1 a (A1) = 1 y (A1)

^ ^ ^

| | |

1 2 (n$)$ . . .

Keep on going and eventually you’ll reach:

(n$)$ . . . $ = (n$)$ . . . $ G (A1)

^ ^ ^ ^ ^ ^

| | | | | |

1 2 (n$)$ 1 2 (n$)$ . . . $ = 1 q (A1) = (n$)$ . . . $ y (A1)

^ ^ ^ ^ ^ ^

| | | | | |

1 2 (n$)$ 1 2 (n$)$ . . . $ = 1 a (A2) = 1 y (A2)

^ ^ ^

| | |

1 2 (n$)$ . . .

Keep on going and eventually you’ll reach:

(n$)$ . . . $ = (n$)$ . . . $ G (A1) (A2)

^ ^ ^ ^ ^ ^

| | | | | |

1 2 (n$)$ 1 2 (n$)$ . . . $ = 1 q (A1) (A2) = (n$)$ . . . $ y (A1) (A2)

^ ^ ^ ^ ^ ^

| | | | | |

1 2 (n$)$ 1 2 (n$)$ . . . $ = 1 a (A3) = 1 y (A3)

^ ^ ^

| | |

1 2 (n$)$ . . .

Keep on going and eventually you’ll reach:

(n$)$ . . . $ = (n$)$ . . . $ G (A1) (A2) . . . (A(10 ^ googolecettaplex))

^ ^ ^ ^ ^ ^

| | | | | |

1 2 (n$)$ 1 2 (n$)$ . . . $

^ ^ ^

| | |

1 2 (n$)$

Another notation for expressing huge numbers is the up-arrow notation published by Donald Knuth in 1976. Like exponentiation, the arrows ‘associate to the right’. Thus, w^x^y^z means w^(x^(y^z)), etc. This association yields the biggest number.

The rules for the up-arrow notation are as follows:

1) a^b^c = a^(b^c) For example, 2^4^5 = 21024

1. a^^b = a^a^. . .^a (b terms)
2. a^^^b = a^^a^^a^^. . .^^a (b terms)

4) a^^^^b = a^^^a^^^a^^^ . . .^^^a (b terms)

.

.

.

etc.

One very large number is the famous Graham's number. Using the up-arrow notation, this number is constructed as follows:

G1 = 3^^^^3

G2 = 3^^^. . .^^^3 (G1 number of up-arrows)

G3= 3^^^. . .^^^3 (G2 number of up-arrows)

.

.

.

G64 = 3^^^. . .^^^3 (G63 number of up-arrows) This is the famous Graham's number.

Continuing on with this sequence . . . . .

G65 = 3^^^. . . ^^^3 (G64 number of up-arrows)

G66 = 3^^^. . . ^^^3 (G65 number of up-arrows)

.

.

.

etc.

Knuth's notation is equivalent to the hyper operator. This operator allows for a more succinct

representation of extremely large numbers that would otherwise take an enormous number of arrows.

For example, Graham's number is expressed as:

G64 = hy(3,G63,3)

Now consider: hy(G64,G64,G64)

Then consider: hy(G(100000000000000Ex$Ex$), G(100000000000000Ex$Ex$), G(100000000000000Ex$Ex$))

John Conway developed the chained arrow notation, which is based on Knuth's up-arrow notation, but much more powerful. In this notation:

a^b = a-->b-->1

a^^b = a-->b-->2

a^^^b = a-->b-->3

a^^...^^b (c up arrows) = a-->b-->c

The following relation holds using this notation: 3-->3-->64-->2 < G64 < 3-->3-->65-->2 < 3-->3-->3-->3

Now imagine: G(10000000Ex$Ex$)-->G(10000000Ex$Ex$)-->G(10000000Ex$Ex$)-->G(10000000Ex$Ex$)

Then try imagining:

G(10000000Ex$Ex$)-->G(10000000Ex$Ex$)-->G(10000000Ex$Ex$)-->. . . -->G(10000000Ex$Ex$)

G(10000000Ex$Ex$) terms

Jonathan Bowers starting in 2004, defined another set of notations, Bowers’ Exploding Array Function, to also represent extremely large numbers. Bowers claims that the number represented by the 5-element array [n,n,n,n,n] is larger than n-->n-->n. . . -->n (n terms) in Conway’s notation. Robert Munafo, a famous number guru, says he is well convinced of the accuracy of this claim. Thus the number above is less than [G(10000000Ex$Ex$),G(10000000Ex$Ex$),G(10000000Ex$Ex$),G(10000000Ex$Ex$),G(10000000Ex$Ex$)].

In Bowers’ “Infinity Scrapers” web page, he defines numerous gargantuan numbers which go far beyond where

anyone in the human race has currently ventured!!! This isn’t an exaggeration! I’ve combed the web and nobody

to date has surpassed Bowers in this arena.

I’ll outline some of his very impressive work:

{10,3,1} = 10^3 = {10,3,1,1}

{10,10} = 10,000,000,000 = {10,2,1 (1) 2}

decker = {10,10,2} = 10^^10 = 10^10^ . . . ^10 (10 terms) Knuth’s up-arrow notation rules apply.

giggol = {10,100,2} = 10^^100 = 10^10^ . . . ^10 (100 terms)

giggolplex = {10,giggol,2} = 10^^giggol = 10^10^ . . . ^10 (giggol terms)

giggolplexplex = 10^^(10^^giggol) = 10^^(10^10^. . .^10) (giggol terms) = 10^10^ . . . ^10 (giggolplex terms)

tritri = {3,3,3} = 3^^^3 = 3^^(3^^3) = 3^^(3^3^3) = 3^^(3^27) = 3^^(7,625,597,484,987) =

3^3^ . . . ^3 (7,625,597,484,987 terms)

gaggol = {10,100,3} = 10^^^100

gaggolplex = {10,gaggol,3} = 10^^^gaggol

tritet = {4,4,4} = 4^^^^4 = 4^^^4^^^4^^^4

tripent = {5,5,5} = 5^^^^^5 = 5^^^^5^^^^5^^^^5^^^^5

geegol = {10,100,4} = 10^^^^100

gigol = {10,100,5}

goggol = {10,100,6}

gagol = {10,100,7}

tridecal (TRI-da-cal) = {10,10,10} = {10,3,1 (1) 2} = 10^^^^^^^^^^10

Graham’s Number = {3,65,1,2}

Tridecalplex = {10,3,1,2} = 10[^]3 = 10[](10 [] 10) = 10[](10^^^^^^^^^^10) = 10[]tridecal

= 10^^. . . ^10 (tridecal # of ^’s)

{10,3,2,2} = 10[^^]3 = 10[^](10[^]10) = 10[^](10[]10[]10[]10[]10[]10[]10[]10[]10[]10)

Conway’s = 3 🡪 3 🡪 3 🡪 3

{3,4,2,2}

corporal = {10,100,1,2}

grand tridecal = {10,10,10,2}

biggol = {10,10,100,2} (this is less than {3,3,1,3})

{10,3,1,3} = 10[[^]]3 = 10[[]](10[[]]10) = 10[[]](10[^^^^^^^^^^]10)

{10,3,2,3} = 10[[^^]]3 = 10[[^]](10[[^]]10)

tetratri (teh-TRA-tree) = {3,3,3,3}

baggol = {10,10,100,3}

baggolplex = {10,10,baggol,3}

beegol = {10,10,100,4}

general = {10,10,10,10}

generalplex = {10,10,10,general}

iteral = {10,10, . . . ,10} (10 terms) = {10,10 (1) 2} = {10,10,1 (1) 2}

goobol = {10,10, . . . ,10} (100 terms) = {10,100 (1) 2 } = {10,100,1 (1) 2}

iteralplex = {10,10, . . . ,10} (iteral terms) = {10,iteral (1) 2}

gibbol = {10,100,2 (1) 2} is described as:

10 1st level

iteral 2nd level

iteralplex 3rd level

iteralplexplex 4th level

.

.

.

iteralplexplex . . . plex (98 times) 100th level - this last line is the gibbol

gabbol = {10,100,3 (1) 2} is described as:

tower 1: 10

tower 2: 10 1st level

iteral 2nd level

iteralplex 3rd level

iteralplexplex 4th level

.

.

.

iteralplexplex . . . plex (8 times) 10th level

tower 3: 10 1st level

iteral 2nd level

iteralplex 3rd level

.

.

.

{10,10,. . . ,10} iteralplexplex . . . plexth (8 times) level

tower 4: 10 1st level

iteral 2nd level

iteralplex 3rd level

.

.

.

{10,10, . . . ,10} “result of tower 3”th level

.

.

.

tower 100 10 1st level

iteral 2nd level

iteralplex 3rd level

.

.

.

{10,10, . . . ,10} “result of tower 99”th level

FINALLY! This last line is the gabbol.

geebol = {10,100,4 (1) 2} is described as:

city 1: 10

city 2: “result of tower 9”th level (because you are going to the 10th tower)

city 3: “result of city 2”th tower

city 4: “result of city 3”th tower

.

.

.

city 100: “result of city 99”th tower

This last line is the geebol.

gibol = {10,100,5 (1) 2} is described as:

state 1: 10

state 2: “result of city 9”th tower (because you are going to the

10th city)

state 3: “result of state 2”th city

state 4: “result of state 3”th city

.

.

.

state 100: “result of state 99”th city

Again, this last line is the gibol.

boobol = {10,10,100 (1) 2} = {10,10,100,1 (1) 2} is described as:

object 1: 10

object 2: {10,10} = {10,2,1 (1) 2}

object 3: {10,10,10} = {10,3,1 (1) 2}

.

.

.

object 10: iteral = {10,10,1 (1) 2}

level 1: 10

level 2: iteral

level 3: object iteral (“result of level 2”th object

which is {10,3,2 (1) 2})

.

.

.

level 10: “result of level 9”th object = {10,10,2 (1) 2}

tower 1: 10

tower 2: “result of level 9”th object = {10,10,2 (1) 2}

tower 3: “result of tower 2”th level = {10,3,3 (1) 2}

.

.

.

tower 10: “result of tower 9”th level = {10,10,3 (1) 2}

city 1: 10

city 2: “result of tower 9”th level = {10,10,3 (1) 2}

city 3: “result of city 2”th tower = {10,3,4 (1) 2}

.

.

.

city 10: “result of city 9”th tower = {10,10,4 (1) 2}

state 1: 10

state 2: “result of city 9”th tower = {10,10,4 (1) 2}

state 3: “result of state 2”th city = {10,3,5 (1) 2}

.

.

.

state 10: “result of state 9”th city = {10,10,5 (1) 2}

Noticed how we started with objects where object 10 = {10,10,1 (1) 2},

then went to levels where level 10 = {10,10,2 (1) 2},

then went to towers where tower 10 = {10,10,3 (1) 2},

then went to cities where city 10 = {10,10,4 (1) 2},

then went to states where state 10 = {10,10,5 (1) 2}.

We would need to continue until we reached the 100th type of “entity”.

entity 1 = objects

entity 2 = levels

entity 3 = towers

entity 4 = cities

entity 5 = states

.

.

.

entity 99 = solar systems

entity 100 = galaxies

Thus, galaxy 1: 10

.

.

.

galaxy 10: “result of galaxy 9”th solar system = {10,10,100 (1) 2} = {10,10,100,1 (1) 2}

AFTER ALL THIS! This last line is the boobol.

bibbol = {10,10,100,2 (1) 2} is described as:

10 = {10,1,1,2 (1) 2} super entity 1

entity 1

.

.

.

entity 10 = {10,2,1,2 (1) 2} = {10,10, {10,1,1,2 (1) 2}, 1 (1) 2} = {10,10,10,1 (1) 2} super entity 2

“result of super entity 2” i.e. {10,10,. . .,10} (entity 10 terms) = {10,3,1,2 (1) 2}

= {10,10, {10,2,1,2 (1) 2}, 1 (1) 2} = {10,10, super entity 2,1 (1) 2}

= entity(entity 10) super entity 3

.

.

.

{10,10,1,2 (1) 2} = {10,10, {10,9,1,2 (1) 2}, 1 (1) 2}

= entity(entity(entity(entity(entity(entity(entity(entity(entity 10)))))))) super entity 10

“result of super entity 99” = {10,100,1,2 (1) 2}

= entity(entity(. . .(entity 10)). . .) 99 times super entity 100

{10,10,2,2 (1) 2} = {10, {10,9,2,2 (1) 2}, 1,2 (1) 2} = super entity (s.e.) {10,9,2,2 (1) 2}

= s.e.(s.e.(s.e.(s.e.(s.e.(s.e.(s.e.(s.e.(s.e. 10))))))))

= duper entity (d.e.) 10

{10,10,3,2 (1) 2} = d.e.(d.e.(d.e.(d.e.(d.e.(d.e.(d.e.(d.e.(d.e. 10))))))))

= truper entity 10

{10,10,4,2 (1) 2} = quadruper entity 10

{10,10,5,2 (1) 2} = quintuper entity 10

{10,10,10,2 (1) 2} = decuper entity 10

{10,10,100,2 (1) 2} = hectuper entity 10 = bibbol

babbol = {10,10,100,3 (1) 2} is described as:

{10,1,1,3 (1) 2}: 10 = super uper entity 1

{10,2,1,3 (1) 2}: 10 1st level

duper entity 10 2nd level

truper entity 10 3rd level

.

.

.

decuper entity 10 10th level = super uper entity 2

{10,3,1,3 (1) 2}: 10 1st level

duper entity 10 2nd level

.

.

.

{10,10,. . .,10} decuper entity 10th level = super uper entity 3

(“result of {10,2,1,3 (1) 2}”th level)

.

.

.

{10,100,1,3 (1) 2}: 10 1st level

duper entity 10 2nd level

.

.

.

{10,10,. . .,10} “result of {10,99,1,3 (1) 2}”th level = super uper entity 100

{10,10,2,3 (1) 2} = {10, {10,9,2,3 (1) 2}, 1,3 (1) 2} = super uper entity (s.u.e.) {10,9,2,3 (1) 2}

= s.u.e.(s.u.e.(s.u.e.(s.u.e.(s.u.e.(s.u.e.(s.u.e.(s.u.e.(s.u.e. 10))))))))

= duper uper entity (d.u.e) 10

{10,10,3,3 (1) 2} = d.u.e.(d.u.e.(d.u.e.(d.u.e.(d.u.e.(d.u.e.(d.u.e.(d.u.e.(d.u.e. 10))))))))

= truper uper entity 10

{10,10,4,3 (1) 2} = quadruper uper entity 10

{10,10,5,3 (1) 2} = quintuper uper entity 10

{10,10,10,3 (1) 2} = decuper uper entity 10

{10,10,100,3 (1) 2} = hectuper uper entity 10 = babbol

beebol = {10,10,100,4 (1) 2} is described as:

{10,1,1,4 (1) 2}: 10 = super uper uper entity 1

{10,2,1,4 (1) 2}: 10 1st level

duper uper entity 10 2nd level

truper uper entity 10 3rd level

.

.

.

decuper uper entity 10 10th level = super uper uper entity 2

{10,3,1,4 (1) 2}: 10 1st level

duper uper entity 10 2nd level

.

.

.

{10,10,. . .,10} decuper uper entity 10th level = super uper uper entity 3

(“result of {10,2,1,4 (1) 2}”th level)

.

.

.

{10,100,1,4 (1) 2}: 10 1st level

duper uper entity 10 2nd level

.

.

.

{10,10,. . .,10} “result of {10,99,1,4 (1) 2}”th level = super uper uper entity 100

{10,10,2,4 (1) 2} = {10, {10,9,2,4 (1) 2}, 1,4 (1) 2} = super uper uper entity (s.u.u.e.) {10,9,2,4 (1) 2}

= s.u.u.e.(s.u.u.e.(s.u.u.e.(s.u.u.e.(s.u.u.e.(s.u.u.e.(s.u.u.e.(s.u.u.e.(s.u.u.e. 10))))))))

= duper uper uper entity (d.u.u.e.) 10

{10,10,3,4 (1) 2} = d.u.u.e.(d.u.u.e.(d.u.u.e.(d.u.u.e.(d.u.u.e.(d.u.u.e.(d.u.u.e.(d.u.u.e.(d.u.u.e. 10))))))))

= truper uper uper entity 10

{10,10,4,4 (1) 2} = quadruper uper uper entity 10

{10,10,5,4 (1) 2} = quintuper uper uper entity 10

{10,10,10,4 (1) 2} = decuper uper uper entity 10

{10,10,100,4 (1) 2} = hectuper uper uper entity 10 = beebol

troobol = {10,10,10,100 (1) 2} = {10,10,10,100,1 (1) 2} is described as:

a @^ b = {a,a,a, . . . ,a} : b # of a’s

so: 10 @^ 10 = iteral

10 @^ 10 @^10 = 10 @^ iteral = iteralplex

{10,b,2,1 (1) 2} = 10 @^ 10 @^ . . . @^ 10 (b times) = 10 @^^ b

{10,10,2,1 (1) 2} = 10 @^ 10 @^ . . . @^ 10 (10 times) = 10 @^^ 10 = 10 @^^^ 2

{10,3,3,1 (1) 2} = 10 @^^ 10 @^^ 10 = 10 @^^^ 3

{10,10,3,1 (1) 2} = 10 @^^ 10 @^^ . . . @^^ 10 (10 times) = 10 @^^^ 10

{10,10,4,1 (1) 2} = 10 @^^^^ 10

Let a @(c) b = a @^^ . . . ^ b where there are c # of ^’s

This makes a @(c) b = {a,b,c,1 (1) 2}

Thus {10,10,100,1 (1) 2} = 10 @(100) 10 = boobol

Now let a @((1)) 1 = a

a @((1)) 2 = a @(a @((1)) 1) a = a @(a) a = a @^^ . . . ^ a (a # of ^’s)

a @((1)) 3 = a @(a @((1)) 2) a

Example: 10 @((1)) 3 = a @(10 @((1)) 2) a

= 10 @(10 @(10) 10) 10

Work what’s inside ( ) first, then work your way out.

a @((1)) 4 = a @(a @((1)) 3) a

Example: 10 @((1)) 4 = 10 @(10 @((1)) 3) 10

= 10 @(10 @(10 @((1)) 2) 10) 10

NOTE: 10@((1)) 2 = entity 10 = 10 @^^^^^^^^^^ 10 = {10,10,10,1 (1) 2}

= 10 @(10 @(entity 10) 10) 10

= 10 @(10 @^^^ . . . ^ (entity 10 times)  10) 10

a @((1)) b = a @(a @((1)) ‘b-1’) a = {a,b,1,2 (1) 2}

a @((2)) b = a @((1)) [a @((2)) ‘b-1’] = a ‘@((1)) to itself’ b times

Example: a @((2)) 4 = a @((1)) [a @((2)) 3]

a @((2)) 3 = a @((1)) [a @((2)) 2]

a @((2)) 2 = a @((1)) [a @((2)) 1]

a @((2)) 1 = a @((1)) [a @((2)) 0] = a @((1)) 1 = a

Thus: a @((2)) 4 = a @((1)) a @((1)) a @((1)) a = {a,4,2,2 (1) 2}

a @((3)) b = a @((2)) [a @((3)) ‘b-1’] = a ‘@((2)) to itself’ b times = {a,b,3,2 (1) 2}

10 @((100)) 10 = {10,10,100,2 (1) 2} = bibbol

a @(((1))) 1 = a = {a,1,1,3 (1) 2}

a @(((1))) 2 = a @((a)) a = {a,2,1,3 (1) 2}

a @(((1))) 3 = a @((a @((a)) a)) a = {a,3,1,3 (1) 2}

a @(((1))) b = a @((a @(((1))) ‘b-1’)) a = {a,b,1,3 (1) 2}

a @(((2))) b = a @(((1))) a @(((1))) a . . . a @(((1))) a

^ ^ ^ ^ ^

| | | | |

1 2 3 b-1 b

10 @(((100))) 10 = {10,10,100,3 (1) 2} = babbol

10 @(( . . . (10)) . . . ) 10 = {10,10,10,100 (1) 2} = {10,10,10,100,1 (1) 2} = troobol

^ ^ ^ ^ ^ ^

| | | | | |

1 2 100 1 2 100

tribbol = {10,10,10,100,2 (1) 2} is described as:

Let a[1] = a

a[2] = a @((. . . ((a)). . .)) a (a = a[1] # of parentheses pairs) = {a,a,a,a = a[1] (1) 2} NOTE: If a = 10,

beebol < 10[2] < troobol

a[3] = a @((. . . ((a)). . .)) a (a[2] # of parentheses pairs) = {a,a,a,a[2] (1) 2} If a = 10, MUCH >> troobol

a[n] = a @((. . . ((a)). . .)) a (a[n-1] # of parentheses pairs) = {a,a,a,a[n-1] (1) 2} = {a,a,a,a[n-1],1 (1) 2}

= {a,n,1,1,2 (1) 2}

Now lets go for {a,n,2,1,2 (1) 2}:

{a,2,2,1,2 (1) 2} = {a,{a,1,2,1,2 (1) 2},1,1,2 (1) 2} = {a,a,1,1,2 (1) 2} = a[a]

Let a @[1] b = a[b]

a @[1] (a @[1] a) = a @[1] (a[a]) = a[a[a]] = {a,a[a],1,1,2 (1) 2} = {a,{a,a,1,1,2 (1) 2},1,1,2 (1) 2}

= {a,{a,2,2,1,2 (1) 2},1,1,2 (1) 2} = {a,3,2,1,2 (1) 2}

So . . . a @[1] a = {a,2,2,1,2 (1) 2}

and a @[1] (a @[1] a) = {a,3,2,1,2 (1) 2}

a @[1] (a @[1] (a @[1] a)) = a @[1] (a[a[a]]) = a[a[a[a]]] = {a,a[a[a]],1,1,2 (1) 2}

= {a,{a,a[a],1,1,2 (1) 2},1,1,2 (1) 2} = {a,{a,3,2,1,2 (1) 2},1,1,2 (1) 2}

= {a,4,2,1,2 (1) 2}

Let a @[2] b = a @[1] [a @[2] ‘b-1’] = a ‘@[1] to itself’ b times = {a,b,2,1,2 (1) 2}

Examples: a @[2] 2 = a @[1] [a @[2] 1] = a @[1] a = a[a] = {a,2,2,1,2 (1) 2}

a @[2] 3 = a @[1] [a @[2] 2] = a @[1] [a @[1] a] = {a,3,2,1,2 (1) 2}

a @[3] b = a ‘@[2] to itself’ b times = {a,b,3,1,2 (1) 2}

Thus a @[n] b = {a,b,n,1,2 (1) 2}

Now let a @[[1]] b = {a,b,1,2,2 (1) 2}

Examples: a @[[1]] 2 = a @[a] a = {a,a,a,1,2 (1) 2} = {a,2,1,2,2 (1) 2}

a @[[1]] 3 = a @[a @[a] a] a = {a,a,{a,2,1,2,2 (1) 2},1,2 (1) 2} = {a,3,1,2,2 (1) 2}

a @[[1]] 4 = a @[a @[a @[a] a] a] a = a nested in @[] 4 times

So . . . a @[[1]] b = a nested in @[] b times

Continuing on . . . a @[[2]] b = a ‘@[[1]] to itself’ b times = {a,b,2,2,2 (1) 2}

a @[[3]] b = a ‘@[[2]] to itself’ b times = {a,b,3,2,2 (1) 2}

a @[[n]] b = {a,b,n,2,2 (1) 2}

Let a @[[[1]]] b = a nested in @[[]] b times

Example: a @[[[1]]] 3 = a @[[a @[[a]] a]] a

a @[[[2]]] b = a ‘@[[[1]]] to itself’ b times = {a,b,2,3,2 (1) 2}

a @[[[n]]] b = {a,b,n,3,2 (1) 2}

a @[[[[n]]]] b = {a,b,n,4,2 (1) 2}

10 @[[ . . . [10]] . . . ] 10 = {10,10,10,100,2 (1) 2} = tribbol

^ ^ ^ ^ ^ ^

| | | | | |

1 2 100 1 2 100

trabbol = {10,10,10,100,3 (1) 2} is described as:

Let a{1} = a

a{2} = a @[[. . . [[a]]. . .]] a (a = a{1} # of bracketed pairs) = {a,a,a,a = a{1} (1) 2} NOTE: If a = 10,

troobol < 10{2} < tribbol

a{3} = a @[[. . . [[a]]. . .]] a (a{2} # of bracketed pairs) = {a,a,a,a{2} (1) 2} If a = 10, MUCH >> tribbol

Thus,

10{1} = 10

10{2} = 10 @[[[[[[[[[[10]]]]]]]]]] 10 = {a,a,a,a = a{1} (1) 2} which is > troobol but < tribbol

10{3} = 10 @[[. . . [[10]]. . .]] 10 (10{2} # of bracketed pairs) which is >> tribbol

Now lets go for “a,n,2,1,3 (1) 2” :

“a,2,2,1,3 (1) 2” = “a,”a,1,2,1,3 (1) 2”,1,1,3 (1) 2” = “a,a,1,1,3 (1) 2” = a{a}

Let a @{1} b = a{b}

a @{1} [a @{1} a] = a @{1} [a{a}] = a{a{a}} = “a,a{a},1,1,3 (1) 2” = “a,”a,a,1,1,3 (1) 2”,1,1,3 (1) 2”

= “a,”a,2,2,1,3 (1) 2”,1,1,3 (1) 2” = “a,3,2,1,3 (1) 2”

So . . . a @{1} a = “a,2,2,1,3 (1) 2”

and a @{1} [a @{1} a] = “a,3,2,1,3 (1) 2”

a @{1} [a @{1} [a @{1} a]] = a @{1} [a{a{a}}] = a{a{a{a}}} = “a,a{a{a}},1,1,3 (1) 2”

= “a,“a,a{a},1,1,3 (1) 2”,1,1,3 (1) 2” = “a,”a,3,2,1,3 (1) 2”,1,1,3 (1) 2”

= “a,4,2,1,3 (1) 2”

Let a @{2} b = a @{1} {a @{2} ‘b-1’} = a ‘@{1} to itself’ b times = “a,b,2,1,3 (1) 2”

Examples: a @{2} 2 = a @{1} {a @{2} 1} = a @{1} a = a{a} = “a,2,2,1,3 (1) 2”

a @{2} 3 = a @{1} {a @{2} 2} = a @{1} {a @{1} a} = “a,3,2,1,3 (1) 2”

a @{3} b = a ‘@{2} to itself’ b times = “a,b,3,1,3(1) 2”

Thus a @{n} b = “a,b,n,1,3 (1) 2”

Now let a @{{1}} b = “a,b,1,2,3 (1) 2”

Examples: a @{{1}} 2 = a @{a} a = “a,a,a,1,3 (1) 2” = “a,2,1,2,3 (1) 2”

a @{{1}} 3 = a @{a @{a} a } a = “a,a, “a,2,1,2,3 (1) 2”,1,3 (1) 2” = “a,3,1,2,3 (1) 2”

a @{{1}} 4 = a @{a @{a @{a} a} a} a = a nested in @{} 4 times

So . . . a @{{1}} b = a nested in @{} b times

Continuing on . . . a @{{2}} b = a ‘@{{1}} to itself’ b times = “a,b,2,2,3 (1) 2”

a @{{3}} b = a ‘@{{2}} to itself’ b times = “a,b,3,2,3 (1) 2”

a @{{n}} b = “a,b,n,2,3 (1) 2”

Let a @{{{1}}} b = a nested in @{{}} b times

Example: a @{{{1}}} 3 = a @{{a @{{a}} a}} a

a @{{{2}}} b = a ‘@{{{1}}} to itself’ b times = “a,b,2,3,3 (1) 2”

a @{{{n}}} b = “a,b,n,3,3 (1) 2”

a @{{{{n}}}} b = “a,b,n,4,3 (1) 2”

10 @{{ . . . {10}} . . . } 10 = “10,10,10,100,3 (1) 2” = trabbol

^ ^ ^ ^ ^ ^

| | | | | |

1 2 100 1 2 100

10 @{{ . . . { 10 }} . . . } 10 = GalileiTrabbol

^ ^ ^ ^ ^ ^

| | | | | |

1 2 X Ex$Ex$Ex$Ex$Ex$Ex$Ex$Ex$Ex$Ex$Ex$Ex$Ex$Ex$Ex$Ex$Ex$Ex$Ex$ 1 2 X Ex$Ex$Ex$Ex$Ex$Ex$Ex$Ex$Ex$Ex$Ex$Ex$Ex$Ex$Ex$Ex$Ex$Ex$Ex$

X = trabbolEx$Ex$ . . . Ex$

^ ^ ^

| | |

1 2 trabbolEx$Ex$ . . . Ex$ = 1 a = 1 y (See page 4.)

^ ^ ^

| | |

1 2 trabbolEx$Ex$ . . . Ex$ = 2 a

^ ^ ^

.

.

.

trabbolEx$Ex$ . . . Ex$ = trabbolEx$Ex$ . . . Ex$ y

^ ^ ^ ^ ^ ^

| | | | | |

1 2 trabbolEx$Ex$ 1 2 trabbolEx$Ex$Ex$Ex$Ex$Ex$Ex$Ex$Ex$Ex$Ex$Ex$