Linked Array Notation

[LKAN]

Version: 0.2 (New Section)

The Prologue

The notation called the "Linked Notation" was coined by **NO!.** Shorten by LKAN, is one of the longest notations about several super-ordinal-levels.

It is a little bit complicated and more complicated as you go through it. These notations are split into several units.

The History

In **Early 2021**, the linked notation was created by **NO!.** In the same season where the linked notation is created, It was expanded to the end of the bird's array notation. In the **Middle of 2021**, the linked notation was furthermore expanded up to the new numbers plus. In the **End of 2021**, It was already stopped and the notation was the number and it was **publicized** in the Numbers -Infinity to Infinity Series. However in **Spring of 2022**, the notation was expanded and the first stage of the notation was replaced with the new stage of the notation (The Functions stays the same). **May 2022** is an expansion to prove I am a googologist when an argument was found. **August 2022** which is the major change of the notation. The Notation is changed to prevent the copy of the birds array notation and make the notation more unique.

Unit 1 - Pre-Hyperia-Separators

Chapter 1 - Introduction for the Linked Notation

Numbers 0 to L(10)-1-{1}-1

The Pre-Stage of the Hypi-Separators is known as the beginning of the notation. The notation was started at **L(10)-10 [10-10 in old notation].** All the linked notation functions must start with the **L** on the beginning of the number. Then the base power or the first entry of the function must be exponents.

U1C1S1 - Rules

The chapter rules for the beginner (Including the advanced and late-notation):

1. First entry must have a parenthesis and a L to be considered a linked notation.
    - Without the **L(base)**, This is not a linked notation
2. The Beginning of the notations must have **L** on the beginning of the numbers, otherwise the function will do this: **(20)-3 = 20-3 = 17, (10)-20-10 = 10-20-10 = -10-10 = -20**
3. When doing negative numbers, Use the **Brackets [].**

Section 1 - The Base power [1.1]

Question: So, How does it even work in the first place?
Answer: Well, This is used by this: **L(10)-0**, But why do these zeros come in. Well here is the TIP when using the Linked Notation with powers of exponents. Because if we use the zeros, you will need an **extra number** on the powers when combining **numbers.**

([The Base Power])-[The Exponential Power-10]

Examples:
L(10)-0 = 10 = 10^1
L(10)-1 = 100 = 10^2
L(10)-2 = 1,000 = 10^3
L(10)-3 = 10,000 = 10^4
L(10)-4 = 100,000 = 10^5
L(10)-5 = 1,000,000 = 10^6
Etc..

Function: L(a)-b = a^b+1

Q: What if the base is different like **L(7)-4** as the linked notation?
A: Using the base function, it is just simple. If you take **L(3)-3** using the function above, you will get the number **3^4** which equals **81**. Using the functions makes it simpler with different bases.

Q: Now how to make **10^10^10** as the linked notation?
A: So if you want to make 10^10^10 as a linked notation, use the function **L(10)-<L(10)-9>**. This shows that **L(10)-9 = 10,000,000,000**. Take **L(10)-10,000,000,000.** This is equal to **10^10,000,000,001.** To make it compressed, make sure you have the arrows of **<>** so they can be compressed and stack up to the next ordinal level. This is partially equal to **10^10^10**

L(10)-<L(10)-9> -> L(10)-100,000,000,000 = 10^100,000,000,001
Because of L(10)-0 = 10, L(10)-1 = 100, You will need to add 1 extra number.

Examples:

L(10)-<L(10)-0> = L(10)-10 = 10^11 = 100,000,000,000 = {10, 11}
L(10)-<L(10)-1> = L(10)-100 = 10^101 = {10, {10, 2}}
L(10)-<L(10)-2> = L(10)-1000 = 10^1001 = {10, {10, 3}}

L(10)-<L(10)-<L(10)-1>> = L(10)-<L-(10)-100> = 10^(10^101)+1) = {10, {10, 101}}

Functions:

- L(a)-<L(b)-c> = L(a)-b^c+1 = a^b^c+1 or a^((b^c+1)+1)
- L(a)-<L(b)-c> = {a , { b , c+1 }
- L(a)-<L(b)-<L(c)-d> = a^(b^(c^d+1)+1)+1)

Section 2 - Using Multi Arrows as a linked notation [1.2]

Q: Now how to make **10^^10** as the linked notation?
A: To make the **10^^10** in linked notation, this is equal to **L(10)-1-9**. Anything that is **L(a)-1-0 = a.** If you get the 0 on the 3rd entry, that 3rd entry will not count and count the 2nd entry.

Because our linked notation entry is 1 = 2 in birds array notation, Subtract from BEAF in linked notation by 1 in after the 2nd entry. The first entry is a base power so it does not subtract by one from arrow notation

L([1st entry])-[3rd entry-1]-[2nd entry-1]
L([a])-[b]-[c] = a^.. c+1 ..^b+1 = {a, b+1, c+1}

Examples:

L(10)-1-10 = 10^^11 with a **10** to have converting to same number but added extra one.

Q: How to convert **L(10)-5-1** to a linked notation?
A: The converting sounds complex to you but converting it results like this:

Formula:

L(a)-1-b = L(a)-<L(a)-<L(a)-....>>

Which **b** adds a **<L(a)-....>** but needed **2b** in order to stack the 1st entry.
If **b = 0**, then it is equal to **a** instead.

L(10)-1-1 = L(10)-10 = 10^11
L(10)-1-2 = L(10)-<L(10)-10>
L(10)-1-3 = L(10)-<L(10)-<L(10)-10>>
Etc…
WARNING: L(a)-0-1 = a

Different base powers will result in a different number in all separate entries.
L(9)-1-3 = L(9)-<L(9)-<L(9)-9>>
L(6)-1-2 = L(6)-<L(6)-6>

L(a)-1-2 = L(a)-<L(a)-a>

Q: How does the **L(a)-c-b** works ?

Warning: L(a)-0-b = a

Using the formula of **L(a)-c+1-b = L(a)-c-<L(a)-c-<...>> - which b is the amount of "L(a)-c-<"**

Examples:

L(10)-2-1 = L(10)-1-10

L(a)-2-1 = L(a)-1-a

L(a)-2-2 = L(a)-1-<L(a)-1-a>

L(a)-2-3 = L(a)-1-<L(a)-1-<L(a)-1-a>>

L(a)-2-4 = L(a)-1-<L(a)-1-<L(a)-1-<L(a)-1-a>>>

L(a)-3-1 = L(a)-2-a

L(a)-0-1 = a

L(a)-3-2 = L(a)-2-<L(a)-2-a>

L(a)-b-2 = L(a)-b-1-<L(a)-b-1-a>

L(a)-c+1-b = L(a)-c-<L(a)-c-<...>> - where **b** is the amount of "**L(a)-c-<**"

Finally, we can assume that we could get a **L(10)-<L(10)-2-1>-1**, what will happen next.

Section 3 - Multi-Entry Notation [1.3]

To make a multi entry. Let **L(a)-c-e-.. d ..-b**

L(a)-c-.. d ..-b = L(a)-1-1-.. d+1 ..-a

Which **a** is the base number. **B** represents the incrementing number. **C, D** are random entries. **d** is the amount of entries.

Examples:

L(a)-1-1-1 = L(a)-a-a ~ {a,2,1,2}

L(a)-1-1-2 = L(a)-<L(a)-a-a>-a ~ {a,3,1,2}

L(a)-1-1-3 = L(a)-<L(a)-<L(a)-a-a>-a>-a ~ {a,4,1,2}

L(a)-1-1-1-1 = L(a)-a-a-a ~ {a,1,1,1,2}

L(a)-2-1-1-1 = L(a)-<L(a)-a-a-a>-a-a ~ {a,1,1,1,3}

L(a)-3-1-1-1 = L(a)-<L(a)-<L(a)-a-a-a>-a-a>-a-a ~ {a,1,1,1,4}

L(a)-1-1-1-1-1 = L(a)-<L(a)-a-a-a-a>-a-a-a ~ {a,1,1,1,1,2}

Increasing the existing entries. Now we have covered the amount of entries. Let Increase any entries.

L(a)-e-...-c+1-1-....-f-b =

L(a)-e-...-c-<L(a)-e-...-c-<..b..>-....-a-a>-....-a-a

Which equaled all previous entries replaced by a. E is the last entry, C is the increasing entry. D is stacked before the increasing entry.

Examples:

L(a)-1-1-2-1 = L(a)-1-1-1-a

L(a)-1-1-2-2 = L(a)-1-1-1-<L(a)-1-1-1-a>

L(a)-1-1-3-2 = L(a)-1-1-2-<L(a)-1-1-2-a>

L(a)-1-2-1-1 = L(a)-1-1-a-a

L(a)-1-2-1-2 = L(a)-1-1-a-<L(a)-1-1-a-a>

L(a)-e-d-c+1-b = L(a)-e-d-c-<..b..<L(a)-1-1-1-a>..b..>

L(a)-e-d+1-c-b = L(a)-e-d-<..b..<L(a)-e-d-a-a>..b..>-a

L(a)-e+1-d-c-b = L(a)-e-<..<L(a)-e-a-a-a>..>-a-a

After this, There will be more entries and more entries…

Chapter 2 - Linked- Separator Arrays

Numbers L(10)-1-{1}-2 to L(10)-1-{1-1}-1

You are now in the second stage of the Linked notation. As you continue the progress, things start getting more different. First, Linked notation looks like a reverse BEAF & BAN Notation. This is still a bit of a copy of the Birds array notation. However, the more you go further, the more different the notation is.

Section 1 - First Row [2.1]

Q: What happens if you reach **many entries on a linked notation**?

A: There is no way to compress it into a "--". Instead, we have gotten the "-{}-" seperators!

L(a)-c-{1}-b = L(a)-a-a-a-a- … b …-a-a-a-a-

Which **b** is the amount of entries

DON'T INCLUDE THE BASE INCREMENT AS A ENTRY "L(A)"

Examples:

L(a)-1-{1}-0 = L(a) = a -

L(a)-1-{1}-1 = L(a)-a - {a, a}

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

L(a)-1-{1}-3 = L(a)-a-a-a - {a, a, a, a} / {a, 4 [2] 2}

L(a)-1-{1}-4 = L(a)-a-a-a-a - {a, a, a, a, a} / {a, 5 [2] 2}

/ / / COMING SOON / / /

The future of the linked array notation

The future of the linked array notation.

Note that they are not well defined yet. Some changes may occur.

They are in order

L(10)-1-{1}-1 - Linked Separator Arrays ~ Chapter 2

L(10)-2-{1}-1

L(10)-1-{1}-1-{1}-1 - Doubled-Linked Separator Arrays

L(10)-1-{1}-1-{1}-1-{1}-1

L(10)-1-{2}-1

L(10)-1-{1-1}-1 - Entries-Separators ~ Chapter 3

L(10)-1-{2-1}-1

L(10)-1-{1-1-1}-1

L(10)-1-{1-{1}-1}-1 - Nested-Separators ~ Chapter 4

L(10)-1-[1]-1 - Superseperators ~ Unit 2

L(10)-1-{1}[1]-2 ~ {10, 10 [3 \ 2] 2}

L(10)-1-{1}{1}[1]-2 ~ {10, 10 [1 \ 3] 2} (stacking is possible)

L(10)-1-{1}{1}{1}[1]-2 ~ {10, 10 [1 \ 1 \ 2] 2}

L(10)-1-[1][1]-2 ~ {10, 10 [1 [2 ¬ 2] 2] 2}

L(10)-1-[1][1][1]-2 ~ {10, 10 [1 \ 2 [2 ¬ 2] 2] 2}

L(10)-1-[2]-1 - Superseperators ~ {10, 10 [1 [2 ¬ 2] 3] 2}

L(10)-1-[3]-2 ~ {10, 10 [1 [2 ¬ 2] 4] 2}

L(10)-1-[1-1]-1 - Superseperators ~ {10, 10 [1 [2 ¬ 2] 1, 2] 2}

L(10)-1-[1-{1}-1]-1 - Nested Superseperators ~ {10, 10 [1 [2 ¬ 2] 1 [2] 2] 2}

L(10)-1-[1-[1]-1]-1 - Super Nested Superseperators ~ {10, 10 [1 [2 ¬ 2] 1 [1 \ 2] 2] 2}

L(10)-1-[1,1]-1 - Super-Entry Seperators ~ {10, 10 [1 [2 ¬ 2] 1 [2 ¬ 2] 2] 2}

L(10)-1-[1,1-[1,1]-1]-1 - {10, 10 [1 [2 ¬ 2] 1 [2 ¬ 2] 1 [2 ¬ 2] 2] 2}

L(10)-1-[2,1]-1 - {10, 10 [1 [3 ¬ 2] 2] 2}

L(10)-1-[1[1,2]1,1]-2 - {10, 10 [1 [1 [2 ¬ 2] 2 ¬ 2] 2] 2}

L(10)-1-[1,1,1]-2 - {10, 10 [1 [1 ¬ 3] 2] 2}

L(10)-1-[1,1,1,1]-2 - {10, 10 [1 [1 ¬ 4] 2] 2}

L(10)-1-[1~1]-2 - Connected-Function - {10, 10 [1 [1 ¬ 1, 2] 2] 2}

Between {10, 10 [1 [2 - 2] 2] 2} to {10, 10 [1 [1 - 3] 2] 2}

L(10)-1-[1,1~1]-2 - {10, 10 [1 [1 ¬ 1 ¬ 2] 2] 2}

L(10)-1-[1,1,1~1]-2 - {10, 10 [1 [1 ¬ 1 ¬ 1 ¬ 2] 2] 2}

L(10)-1-[1-[1~1]-1~1]-2 - {10, 10 [1 [1 [1 $\backslash_3$ 2] 2] 2] 2}

L(10)-1-[1~,1]-2 - {10, 10 [1 [1 [2 $\backslash_3$ 2] 2] 2] 2}

L(10)-1-[1~,1,1]-2

L(10)-1-[1~,1~1]-2

L(10)-1-[1~,1~,1]-2

L(10)-1-[1~,1~,1~,1]-2

L(10)-1-[1~~1]-2 - {10, 10 [1 [1 [1 [2 $\backslash_4$ 2] 2] 2] 2] 2}

L(10)-1-[1~~~1]-2

L(10)-1-[1~~~~1]-2

At this point, This is no longer a “Linked Notation”, It is now called “Complex Linked Notation”.
Birds array notations end at Complex+ Linked Notation

Complex Linked Array Notation

L(10)-1-[1~[1]-1]-2 - {10, 10 [1[2 $\backslash_{1,2}$ 2] 2] 2}
L(10)-1-[1~~[1]-1]-2
L(10)-1-[1~[2]-1]-2
L(10)-1-[1~[1~1]-1]-2 - {10, 10 [1[2 $\backslash_{1 \backslash 2}$ 2] 2] 2}
L(10)-1-[1~[2~1]-1]-2
L(10)-1-[1~[1~[1]-1]-1]-2

Complex+ Linked Array Notation

L(10)-1-[1;1]-2 ~ Unit 3
L(10)-1-[1;1,1]-2
L(10)-1-[1;1~1]-2
L(10)-1-[1;1-[1;1]-1]-2
L(10)-1-[1;1-[1;1-[1;1]-2]-1]-2
L(10)-1-[2;1]-2
L(10)-1-[1;1;1]-2
L(10)-1-[1;1;1;1]-2

Complex++ Linked Array Notation

L(10)-1-[1:1]-2

Complex+++ Linked Array Notation

L(10)-1-[1/1]-2

Complex^ Linked Array Notation

L(10)-1-[1-[$\backslash_{4}$]-1]-2
L(10)-1-[1-[$\backslash_{1-2}$]-1]-2
L(10)-1-[1-[$\backslash_{1/2}$]-1]-2
L(10)-1-[1-[$\backslash_{1[\backslash 4]2}$]-1]-2

Hypo-Separators Linked Array Notation

L(10)-1-[#]-2
Scaled and the [#] is more powerful, can put any numbers inside.
L(10)-1-[##]-2
L(10)-1-[###]-2
L(10)-1-[#-#]-2
L(10)-1-[#[\]#]-2
L(10)-1-[#[#]#]-2

Walled linked array Notation

Which the previous function becomes scaled!
Inside the wall, all previous function become very powerful
L(10)-1-|-1 ~ Unit 4
L(10)-1-|-1-2

L(10)-1-|-1-[#[#]#]-2
L(10)-1-|-1-|1|-1
L(10)-1-|-1-|1-|1|-1|-1
L(10)-2-|-1
L(10)-<1-2>-|-1
L(10)-<1-1><1-1>-|-1
L(10)-1-1-|-1
L(10)-1-[1]-|-1
L(10)-1-||-1
L(10)-1-|||-1
L(10)-1-|1-2-1
L(10)-1-|1-|-2-1

The Far Linked array Notation

L(10)-1-[`]-1 - The Far Linked Notation
L(10)-1-[!]-1 - The Farther Linked Notation
L(10)-1-[&]-1 - The Fartherer Linked Notation
L(10)-1-[=]-1 - The Farthest Linked Notation

Mathematical Int-Linked array Notation

L(10)-1-(+)-1 - MILKAN
L(10)-1-(x)-1 - MILKAN+
L(10)-1-(^)-1 - MILKAN++
L(10)-1-(^^)-1 - MILKAN+++

Exponental Linked Notation

The <B> scales the other function as well

2nd Stage of the Linked Notation

L(10)-1-<B[1],[1]>-1 ~ Unit 5
L(10)-1-<B[1],[1],[1]>-1

Future Linked Notation Words

Super Ordinal Levels "L(10)-1-<???>-1" ~ Unit 6

- Multi-replicant Linked Notation
- New Linked Notation+
- Powering Linked Notation
- Cilvizated Linked Notation
- Imperium Linked Notation
- Advanced-Shape Notation
- Dyans Notation
- 10th stage: Functionality linked notation linking to next classes

Mega Ordinal Levels "L(10)-???" ~ Might be the end for Fast-Growing Hierarchy ~ Unit 7

- Unlinked Notation
- Reversed Linked Notation

- Broken Linked Notation
- Sepi-Linked Notation

4th Level of classes marks the end for

10th Level of Classes which resembles the linked notation into this.

X[class]-entry

This becomes incomputable temporarily until all levels of classes are defined. <=> is used.

After the 10th resembles, It begins to complexility

10th Layer of complexility, Untypeable linked notation occur

10th stage of untypable linked notation, the great link circle expands.

The Almost end of the linked notation

After Circle Expanded, Uncomputable Linked Begins which is unknown what's between it. Its possible to go beyond Utter Obilivon and Sams Number without reaching Finity or Infinity.

The end of the linked notation is unknown.
This makes the linked notation the longest notation.

(a)-<NAN>(a)- ~ Infinity

Notation ended in Infinity