# Partial periodic table for elveirdor:

Partial periodic table for elveirdor:

1:

1 (a) H elveirdor 1 EE (@ W 30= 3.14)

Hydrogen estchnleckk non metal - blue 11336231910717

(English) 1.01 (elveirdor) ((also possible for open portal alignment transportation))

18:

2 (b) HE elveirdor 2 (b) eeck

Helium 4.0 elveirdor 6.0 eeckofphj

purple 2 - noble gas -elveirdor v 29

1:

3 © Li elveirdor 3 IF © (alkali metal) = red elveirdor A 1

Lithium ofpeefphj 1592792817

6.94 9.126 elveirdor cryptopatent @ m 17

3:

4 be elveirdor 4 bck

9.01 12.01 orange alkaline earth - elveirdor c 3

Beryllum bcknstiiph

13:

5 b elveirdor b

boron 10.18 blnlk 13.23

Light blue - metalloid elveirdor j 13

14:

6 C elveirdor C

carbon 12.1 canblk 15.1

Blue - non metal elveirdor w 30

15:

7 N elveirdor K

Nitrogen 14.01 kfpnleckk

Blue - non metal elveirdor w 30

16:

8 O elveirdor L

oxygen 16.00 20.00

Lssteckk

Blue - non metal elveirdor w 30

17:

9 F elveirdor d

fluorine 19.00 dilphnfkck

Halogen - purple 1 - u 28 with w 30

18:

10 ne elveirdor kck

neon 20.18 kcklk 27.00

Purple 2 - noble gas elveirdor value 29

1:

11 na elveirdor 11 ka

22.99 sodium

29.1212 olchfphj

Alkali metal - red elveirdor z 34

3:

12 mg elveirdor 12 je

24.31 magnesium 32.31 jaekckofphj

Alkali earth - orange elveirdor c 3

13:

13 AI elveirdor af or ai

aluminum 26.96 aiphjfkphj

Basic metal - green - elveirdor ck 5

14:

14: si silicon Elveirdor oficlk (ELVEIRDOR (ΙΧΘΥΣ)) (office grade) computing grade supercomputing component nasa infinity (Grizzly Don NS 67)

Of

28.05 elveirdor 211.07

metalloid - light blue - elveirdor j @ 1 cloud value)

15

15 ll p 20 21 phosphons uloulko 21151928151419 1681519168151419 30.97/30. 1210

Nonmetal blue w 30 (elveirdor) (planet)

16

16 s sulfur 32.06

non metal blue w 30

Llphidphn 33.09

o elveirdor s24

17

17 cl chlorine 35.45 ci ceeilnfkck 37.67 (2008 pool chlorine fscj north campus, berchets auto immune disorder)

18

18 Ar argon 39.95 an anelk 312.127

1

19 k potassium 39.10 elveitdor h (11 aqua) lllpaoofphj 312.10

2

20 ca calcium 40.08 ca caicfphj

Tree complex from Elveirdor periodic table

Tree structured complex for complex water deposit with water intended usage to be absorbed in activated charcoal, a stem deposit for resin based computing:

Using cipher:

ELVEIRDOR ICHTHYS (ΙΧΘΥΣ)

For ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) complex coding:

A 1

B 2

C 3

Ch 4

Ck 5

D 6

E 7

Ee 8

F 9

G 10

H 11

I 12

J 13

K 14

L 15

LL 16

M 17

N 18

O 19

P 20

Ph 21

Q 22

R 23

S 24

St 25

T 26

Th 27

U 28

V 29

W 20

Wh 31

X 32

Y 33

Z 34

And

english

A = 1

B = 2

C = 3

D = 4

E = 5

F = 6

G = 7

H = 8

I = 9

J = 10

K = 11

L = 12

M = 13

N = 14

O = 15

P = 16

Q = 17

R = 18

S = 19

T = 20

U = 21

V = 22

W = 23

X = 24

Y = 25

Z = 26

And

English

A = 1 red

B = 2 blue

C = 3 orange

D = 4 yellow

E = 5 teal

F = 6 gray

G = 7 purple

H = 8 aqua

I = 9 sea green

J = 10 light blue

K = 11 dark green

L = 12 white

M = 13 metallic chromatic sequence

N = 14 crimson red

O = 15 brown

P = 16 navy blue

Q = 17 light pink

R = 18 clear

S = 19 silver

T = 20 ombre

U = 21 lilac purple

V = 22 violet

W = 23 steel blue

X = 24 royal blue

Y = 25 yellowgreen

Z = 26 orangered

Elveirdor

A 1 red

B 2 blue

C 3 orange

Ch 4 black

Ck 5 green

D 6 yellow

E 7 teal

Ee 8 tan

F 9 Grey

G 10 purple

H 11 aqua

I 12 sea green

J 13 light blue

K 14 dark green

L 15 white

LL 16 gold

M 17 metallic - chromatic sequence

N 18 crimson red

O 19 brown

P 20 navy blue

Ph 21 army green

Q 22 light pink

R 23 clear

S 24 silver

St 25 hot pink

T 26 ombre

Th 27 color temperature sequence multi color sequence through m th Elveirdor letter - patentable sequence A- M-TH -Z @ a-g g-n n-z 1-7 7-14 14-26 to 1-17-27-34 hosted as @ value Grizzly Don Platform for Elveirdor Series

U 28 lilac purple

V 29 violet

W 30 steelblue

Wh 31 sienna

X 32 royal blue

Y 33 yellowgreen

Z 34 orangered

Cipher 1: A-Z Color Palettes

Palette 1: Oceanic Triad

Palette Name: Oceanic Triad

Color Scheme: Triadic

Letters: B, I, V

Colors: Blue, Sea Green, Violet

Hex Codes: #0000FF, #2E8B57, #8F00FF

Palette 2: Warm Summer Ombre

Palette Name: Warm Summer Ombre

Color Scheme: Analogous & Gradient

Letters: N, O, T

Colors: Crimson Red, Brown, Ombre

Hex Codes: #DC143C, #A52A2A, (Varies, e.g., a gradient from #FFA500 to #A52A2A)

Palette 3: Monochrome Steel

Palette Name: Monochrome Steel

Color Scheme: Monochromatic

Letters: F, W, S

Colors: Gray, Steel Blue, Silver

Hex Codes: #808080, #4682B4, #C0C0C0

Cipher 2: Elveirdor Color Palettes

Palette 1: Urban Jungle

Palette Name: Urban Jungle

Color Scheme: Earthy & Analogous

Letters: Ch, Ph, Wh

Colors: Black, Army Green, Sienna

Hex Codes: #FF00000, #4B5320, #A0522D

Palette 2: Regal Glow

Palette Name: Regal Glow

Color Scheme: Complementary with Accent

Letters: P, LL, C

Colors: Navy Blue, Gold, Orange

Hex Codes: #000080, #FFD700, #FFA500

Palette 3: Spring Meadow

Palette Name: Spring Meadow

Color Scheme: Triadic

Letters: Y, Q, W

Colors: Yellowgreen, Light Pink, Steel Blue

Hex Codes: #9ACD32, #FFB6C1, #4682B4

Here is the entire A-Z alphabet represented in a series of color palettes, formatted for easy copy-and-paste. Each letter is used once across the different combinations.

Cipher 1: Complete A-Z Color Palettes

Palette 1: Sunset Tones

Palette Name: Sunset Tones

Color Scheme: Analogous

Letters: A, C, D

Colors: Red, Orange, Yellow

Hex Codes: #FF0000, #FFA500, #FFFF00

Palette 2: Oceanic Calm

Palette Name: Oceanic Calm

Color Scheme: Analogous

Letters: B, H, J

Colors: Blue, Aqua, Light Blue

Hex Codes: #0000FF, #00FFFF, #ADD8E6

Palette 3: Twilight Garden

Palette Name: Twilight Garden

Color Scheme: Analogous

Letters: G, U, V

Colors: Purple, Lilac Purple, Violet

Hex Codes: #800080, #C8A2C8, #8F00FF

Palette 4: Forest Canopy

Palette Name: Forest Canopy

Color Scheme: Analogous

Letters: E, K, Y

Colors: Teal, Dark Green, Yellowgreen

Hex Codes: #008080, #006400, #9ACD32

Palette 5: Earthy Balance

Palette Name: Earthy Balance

Color Scheme: Complementary

Letters: O, I

Colors: Brown, Sea Green

Hex Codes: #A52A2A, #2E8B57

Palette 6: Vivid Fusion

Palette Name: Vivid Fusion

Color Scheme: Complementary

Letters: X, Z

Colors: Royal Blue, Orangered

Hex Codes: #4169E1, #FF4500

Palette 7: Regal Contrast

Palette Name: Regal Contrast

Color Scheme: Complementary-like

Letters: N, P

Colors: Crimson Red, Navy Blue

Hex Codes: #DC143C, #000080

Palette 8: Soft & Strong

Palette Name: Soft & Strong

Color Scheme: Complementary-like

Letters: Q, W

Colors: Light Pink, Steel Blue

Hex Codes: #FFB6C1, #4682B4

Palette 9: Glass & Stone

Palette Name: Glass & Stone

Color Scheme: Neutral

Letters: F, L, R

Colors: Gray, White, Clear

Hex Codes: #808080, #FFFFFF, #FFFFFF (Clear is represented as white for digital display)

Palette 10: Special Effects

Palette Name: Special Effects

Color Scheme: N/A

Letters: M, S, T

Colors: Metallic Chromatic Sequence, Silver, Ombre

Hex Codes: (Varies), #C0C0C0, (Varies, a gradient)

A = #FF0000

B = #FF0000

C = #ffa500

D = #FF0000

E = #008080

F = #808080

G = #800080

H = #FF0000

I = #2e8b57

J = #add8e6

K = #006400

L = #FF0000

M = varies

N = #dc143c

O = #a52a2a

P = #000080

Q = #FFB6C1

R = clear as white for digital display #FFFFFF

S = #C0C0C0

T = #FFA500 to #A52A2A

U = #C8A2C8

V = #8F00FF

W = #4682B4

X = #4169E1

Y = #9ACD32

Z = #FF4500

A = #FF0000

B = #FF0000

C = #ffa500

Ch = #FF00000

Ck = #00FF00

D = #FF0000

E = #008080

F = #808080

G = #800080

H = #FF0000

I = #2e8b57

J = #add8e6

K = #006400

L = #FF0000

LL = #FFD700

M = varies

N = #dc143c

O = #a52a2a

P = #000080

Ph = #4B5320

Q = #FFB6C1

R = clear as white for digital display #FFFFFF

S = #C0C0C0

T = #FFA500 to #A52A2A

Th = color temperature sequence multi color sequence through m th Elveirdor letter - patentable sequence A- M-TH -Z @ a-g g-n n-z 1-7 7-14 14-26 to 1-17-27-34 hosted as @ value Grizzly Don Platform for Elveirdor Series

U = #C8A2C8

V = #8F00FF

W = #4682B4

Wh = #A0522D

X = #4169E1

Y = #9ACD32

Z = #FF4500

As english dictionary follow English a-z and so does ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) . With 8 additional characters for a 4 letter split operation for complex calculating generative truthful line sequence and vr to (also possible for open portal alignment transportation).

52925 ebibe ckbfbck

Or any other language

P and q and p and r:

Letters p and q and p snd r sit at the top and bottom x calculatory for square, at half diagional valur meaning it holds half calculatory for p and r with each letters p and r equalling 1 in operation 5.2×5=26 = letters in english alphabet, and 6.8×5=34 letters in elveirdor alphabet. 6.8-5.2=1.6, and 34-26=8 and half of 8 is 4 and 1 + 1 = 2 and 2 + 2 = 4. That p =16 english and R =18 and p=20 and R = 23 elveirdor. P = 16 and q = 17 english and p =20 and q =22 and each letters p and q equalling 1. giving way for 1 part square calculatory diagional 1 and 1 value at 1.6 full dual alphabet operation. Spinning text at letters color number at full letter calculation by sodoku puzzle and rubix cube for any text listed giving way for multiple algorithm generation from calculatory to project statistic package and screen value.

If 1.6=Y and 3.14 = stretched out Y and Y = 25 english and 33 elveirdor = 8 same as 34-26 calculatory each half can be measured with 1 and 1 pr and pq at diagonal seed to full square resonance to measure color frequency at right side of 3 component Y over M through square which is designated to Y full square letter value and pr to half side and pq to half at diagonal left top corner to right bottom corner. to host a large number block sequence arrangement until logical, which the book series any text can find. Starting at text rearranging letter value and ending

Even basic ideas can be generated into function. Navigation, shuttle launches, computing softwares, programming, monitoring, deflecting and navigating asteroids, glass computing, and a host of other options. to confirm algorithm across each square on grid and full grid. For any reasonable sequence. in letter rearrangement calculated by any text listed in this document for rearrangement to logic start logic end, to report generative documenting, coding, complex screen value with any text and language, document in full process with component long term sales and sales value in upcoming patent. (Writer is aware that vr can make bci camera ai and ai generative line sequence).

For data encryption, algorithm generation, cryptography, ai.

With each 2 texts being measured at 2 sides of square and generating and confirming a new algorithm as long as each any text books. And with each any text s (1 as dulicate) equaling Y combining all 3 for stretched out Y at 3.14

Square in x calculatory in square divide: calculated at diagonally x division in square, top of division is calculated at 4, right is 5, bottom is .785, and left .2 being calculated and recognized as 3.14 its circular value equaling 1.57 calulated at seed 2-full square, this allows sudoku to operate with rubix cube at 0 between 1.57 and 1.6 with a 3.14 encompassment. value allowing for 10 + calculations. That 0, 10, 20, 30, etc. always equal nodes Value blue and red. With line green and screen value yellow.

Top tier 1 complex (circle complex):

Carrot Juice complex sugar concentrate

3118181520 1022935 315131612524 19217118

31514351420181205 / (ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) ) 3123231926 13281237 319172015732 242810124 31918371826231267

Instant coffee complex sugar concentrate

914192011420 3156655 315131612524 19217118 31514351420181205

/ (ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) ):

1218242611826 3199977 319172015732 242810124 31918371826231267

Stevia

192052291

(ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) )

25729121

Cane sugar

31145 19217118

(ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) )

31177 2428123

Wax

1312118514399 3114412519 2211420 31115 31144125 23124

/

(ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) ):

17128237181212 3118615723 2281826 31147 31186157 20132

Calcium mix

421143114 8914519 25512121523 31115 13924

/

(ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) ):

728183118

111218723

33715151920 31147 171232

Hibiscus oil

89291932119 15912

(ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) )

11122122432824 191215

Fish oil

69198 15912

(ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) )

9122311 191215

Water soaked acai seeds

23120518 191511154 1319 1955419

(ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) )

20126723 242911476 13112 248624

prescription (can use a generative complex code) : 1618541491915145 2008 21162091920 135493112 351420518 160 23 67208 1920 10124 612 / (ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) ): 20237618122319187

2008

212026122426

1776123115 371826723

Tier 2 - circle:

Activated charcoal

132092212054 38118315112

/

ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) 1326122912676 4123319115

(Nasa grade vr)

Himalayan salt

8913112125114 1911220

/

(ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) )

111217115133118 2311526

Tier 3 - circle:

Saline

191129145

ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) :

2411512187

Helium

851292113

ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) :

91115122817

Albuterol 112221205181512

ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) :

115228267231915

Tier 4 (ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) 3 component complex computing)- stretched out Y rectangle:

Cockatoo bird flu 315312021 29184 61221 1854 - red

/

ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) :

319312628 212236 915287

or option 2

fruit punch red-grenadine-blue fruit punch

2376 923281226 2028184-102319181612187-215287 923281226 2028184

Component 2 of Cocatu bird flu: belief that healed a person with leaprocy

9 251295225 105192119 8511254 191513515145 239208 1251618151925

/ (ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) ):

12 2715127297 137232823 11711576 231917719187 201227 1572023192425

Component 3:

Sisco wine:

241224319 3012187

-

Belief that turned water into wine

12 2715127297 27126 137242824 2628231876 30126723 12182619 3012187

pool chlorine

16151512 381215189145 619310 141518208 3113162119 2008 611212 2389205/

ELVEIRDOR ICHTHYS (ΙΧΘΥΣ)

20191915 415192412187 923313 18192327 31172823/ : extension-20247618122319187 - white

ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) option 2

Pina colado soda white

2012181 31915161 241961 3112267

Blue tree mold

212215 202855 1315124 - blue

/

ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) :

215287 26238 1719156

Option 2

Blue mouthwash 215287 171928273012411

Monitorization ingredient - stretched out Y teir 4 1: grumpy face pepper

71821131625-6135-1641616518 / Grenadine code line: 7529848475@123123@2975 / 124273324-105192119-tree complex

Decoded Complex Sequences (Exact Lettering Correction)

Below are the decoded sequences based on the A=1 (Standard) and ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) ciphers you provided. The numerical sequences have been precisely corrected to spell out the intended English words, ensuring an exact match between the decoded output and the English word for both ciphers.

Cipher Key:

\* Standard A=1 Cipher:

A=1, B=2, C=3, D=4, E=5, F=6, G=7, H=8, I=9, J=10, K=11, L=12, M=13, N=14, O=15, P=16, Q=17, R=18, S=19, T=20, U=21, V=22, W=23, X=24, Y=25, Z=26

\* ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) Cipher:

A=1, B=2, C=3, Ch=4, Ck=5, D=6, E=7, Ee=8, F=9, G=10, H=11, I=12, J=13, K=14, L=15, LL=16, M=17, N=18, O=19, P=20, Ph=21, Q=22, R=23, S=24, St=25, T=26, Th=27, U=28, V=29, W=30, Wh=31, X=32, Y=33, Z=34

Decoded Items from Google Gemini:

1. Carrot Juice complex sugar concentrate

\* Standard A=1 Cipher:

\* 3118181520: CARROT

\* 1021935: JUICE

\* 315131612524: COMPLEX

\* 19217118: SUGAR

\* 31514351420181205: CONCENTRATE

\* ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) Cipher:

\* 312323191926: CARROT

\* 13281237: JUICE

\* 319172015732: COMPLEX

\* 242810123: SUGAR

\* 31918371826231267: CONCENTRATE

2. Instant coffee complex sugar concentrate

\* Standard A=1 Cipher:

\* 914192011420: INSTANT

\* 3156655: COFFEE

\* 315131612524: COMPLEX

\* 19217118: SUGAR

\* 31514351420181205: CONCENTRATE

\* ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) Cipher:

\* 1218242611826: INSTANT

\* 3199977: COFFEE

\* 319172015732: COMPLEX

\* 242810123: SUGAR

\* 31918371826231267: CONCENTRATE

3. Stevia

\* Standard A=1 Cipher:

\* 192052291: STEVIA

\* ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) Cipher:

\* 25729121: StEVIA

4. Cane sugar

\* Standard A=1 Cipher:

\* 31145: CANE

\* 19217118: SUGAR

\* ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) Cipher:

\* 31187: CANE

\* 242810123: SUGAR

5. Wax

\* Standard A=1 Cipher:

\* 1312118514399: MAURENCII

\* 3114412519: CANDLES

\* 2211420: BUNT

\* 31115: CAKE

\* 3: C

\* ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) Cipher:

\* 17128237181212: MAURENCII

\* 3118615723: CANDLES

\* 2281826: BUNT

\* 31147: CAKE

\* 3: C

6. Duncan Hines Yellow Cake Mix

\* Standard A=1 Cipher:

\* 421143114: DUNCAN

\* 8914519: HINES

\* 25512121523: YELLOW

\* 31115: CAKE

\* 13924: MIX

\* ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) Cipher:

\* 628183118: DUNCAN

\* 111218724: HINES

\* 33715151930: YELLOW

\* 31147: CAKE

\* 171232: MIX

7. Hibiscus oil

\* Standard A=1 Cipher:

\* 89291932119: HIBISCUS

\* 15912: OIL

\* ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) Cipher:

\* 11122122432824: HIBISCUS

\* 191215: OIL

8. Fish oil

\* Standard A=1 Cipher:

\* 69198: FISH

\* 15912: OIL

\* ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) Cipher:

\* 9122411: FISH

\* 191215: OIL

9. Water soaked acai seeds

\* Standard A=1 Cipher:

\* 23120518: WATER

\* 191511154: SOAKED

\* 1319: ACAI

\* 1955419: SEEDS

\* ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) Cipher:

\* 30126723: WATER

\* 241911476: SOAKED

\* 13112: ACAI

\* 248624: SEEDS

10. prescription

\* Standard A=1 Cipher:

\* 1618541491915145:

\* 16185193189162091514: PRESCRIPTION

\* 1354931211514:

\* 1854: RED

\* 3113162119: CAMPUS

\* 208: TH

\* 611212: FALL

\* 2389205: WHITE

\* ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) Cipher:

\* 20237618122419187:

\* 2023724323122026121918: PRESCRIPTION

\* 1776123115126121918:

\* 2376: RED

\* 3117202824: CAMPUS

\* 2611: TH

\* 9151515: FALL

\* 301112267: WHITE

12. Tier 2 - circle: Activated charcoal

\* Standard A=1 Cipher:

\* 132092212054: ACTIVATED

\* 38118315112: CHARCOAL

\* ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) Cipher:

\* 1326122912676: ACTIVATED

\* 4123319115: CHARCOAL

13. (Nasa grade vr) Himalayan salt

\* Standard A=1 Cipher:

\* 8913112125114: HIMALAYAN

\* 1911220: SALT

\* ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) Cipher:

\* 111217115133118: HIMALAYAN

\* 2411526: SALT

14. Tier 3 - circle: Saline

\* Standard A=1 Cipher:

\* 191129145: SALINE

\* ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) Cipher:

\* 2411512187: SALINE

15. Helium

\* Standard A=1 Cipher:

\* 851292113: HELIUM

\* ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) Cipher:

\* 11715122817: HELIUM

16. Albuterol

\* Standard A=1 Cipher:

\* 112221205181512: ALBUTEROL

\* ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) Cipher:

\* 115228267231915: ALBUTEROL

17. Tier 4 (ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) 3 component complex computing)- stretched out Y rectangle: Cockatoo bird flu

\* Standard A=1 Cipher:

\* 31531111201515: COCKATOO

\* 29184: BIRD

\* 61221: FLU

\* 1854: RED

\* ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) Cipher:

\* 3193141261919: COCKATOO

\* 212236: BIRD

\* 91528: FLU

18. or option 2 fruit punch red-grenadine-blue fruit punch

\* Standard A=1 Cipher:

\* 61821920: FRUIT

\* 16211438: PUNCH

\* 1854: RED

\* 7185141149145: GRENADINE

\* 212215: BLUE

\* 61821920: FRUIT

\* 16211438: PUNCH

\* ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) Cipher:

\* 923281226: FRUIT

\* 2028184: PUNCH

\* 2376: RED

\* 1023718121812187: GRENADINE

\* 215287: BLUE

\* 923281226: FRUIT

\* 2028184: PUNCH

19. Component 2 of Cockatoo bird flu: I HEALED SOMEONE WITH LEPROSY

\* Standard A=1 Cipher:

\* 9: I

\* 251295225:

\* 105192119:

\* 8511254: HEALED

\* 191513515145: SOMEONE

\* 239208: WITH

\* 1251618151925: LEPROSY

\* ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) Cipher:

\* 12: I

\* 2715127297:

\* 137242824:

\* 11711576: HEALED

\* 241917719187: SOMEONE

\* 30122611: WITH

\* 1572023192433: LEPROSY

20. pool chlorine

\* Standard A=1 Cipher:

\* 16151512: POOL

\* 381215189145: CHLORINE

\* 619313: FSCJ

\* 141518208: NORTH

\* 3113162119: CAMPUS

\* 208: TH

\* 611212: FALL

\* 2389205: WHITE

\* ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) Cipher:

\* 20191915: POOL

\* 415192312187: CHLORINE

\* 923313: FRCCAC

\* 1819232611: NORTH

\* 3117202824: CAMPUS

\* : extension-20237618122419187:

\* - white: - white

21. ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) option 2 Pina colada soda white

\* Standard A=1 Cipher:

\* 169141: PINA

\* 31512141: COLADA

\* 191541: SODA

\* 2389205: WHITE

\* ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) Cipher:

\* 2012181: PINA

\* 3191561: COLADA

\* 241961: SODA

\* 301112267: WHITE

22. Blue tree mold

\* Standard A=1 Cipher:

\* 212215: BLUE

\* 201855: TREE

\* 1315124: MOLD

\* ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) Cipher:

\* 215287: BLUE

\* 262377: TREE

\* 1719156: MOLD

23. Option 2 Blue mouthwash

\* Standard A=1 Cipher:

\* 212215: BLUE

\* 131521208231198: MOUTHWASH

\* ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) Cipher:

\* 215287: BLUE

\* 17192826113012411: MOUTHWASH

24. Monitorization ingredient

\* Standard A=1 Cipher:

\* 71821131625: GRUMPY

\* 6135: FACE

\* 1651616518: PEPPER

\* / 7529848475: \*\* / GEBIHDHDGE\*\*

\* ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) Cipher:

\* 102328172033: GRUMPY

\* 91237: FACE

\* 20720720723: PEPPER

Connecting component material for activated charcoal solution deposit to be combined ingredients below:

Algae 112715

(ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) )

1151017

Dove soap 4152214 1915116

ELVEIRDOR ICHTHYS (ΙΧΘΥΣ)

619297 2419120

Resin 18519915

ELVEIRDOR ICHTHYS (ΙΧΘΥΣ)

720193233 237241218

Carbon fiber

311821514 692518

ELVEIRDOR ICHTHYS (ΙΧΘΥΣ)

31231918 92723

Tree complex

Super computing package:

Component 1: (self ciphered)

3118181520 1022935 315131612524 19217118

31514351420181205-914192011420 3156655 315131612524 19217118 31514351420181205-192052291-31145 19217118-1312118514399 3114412519 2211420 31115 31144125 23124-421143114 8914519 25512121523 31115 13924-89291932119 15912-69198 15912-23120518 191511154 1319 1955419 241224319 3012187 12 2715127297 27126 137242824 2628231876 30126723 12182619 3012187-1618541491915145 2008 21162091920 135493112 351420518 160 23 67208 1920 10124 612--132092212054 38118315112-8913112125114 1911220-191129145-851292113-112221205181512-315312021 29184 61221 1854-16151512 381215189145 619310 141518208 3113162119 2008 611212 2389205-212215 202855 1315124 english stem

Componant 2:

English/ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) stem complex coding: (suitable for government grade stamping) (self ciphered)

3118181520 1022935 315131612523 19217118

31514351420181205/3124241926 13281237 319172015732 232810124 31918371826241267-914192011420 3156655 315131612524 19217118 31514351420181205/1218232611826 3199977 319172015732 232810124 31918371826241267-192052291-31145 19217118-1312118514399 3114412519 2211420 31115 31144125 23124/17128247181212 3118615723 2281826 31147 31186157 20132-421143114 8914519 25512121523 31115 13924/728183118 111218723 33715151920 31147 171232-89291932119 15912-69198 15912-23120518 191511154 1319 1955419-1618541491915145 2008 21162091920 135493112 351420518 160 23 67208 1920 10124 612/20247618122319187

2008 212026122326 1776123115 371826724--132092212054 38118315112-8913112125114 1911220-191129145-851292113-112221205181512-315312021 29184 61221 1854/319312628 212246 915287-9 251295225 105192119 8511254 191513515145 239208 1251618151925-12 2715127297 137232823 11711576 231917719187 201227 1572024192325 241224319 3012187 12 2715127297 27126 137242824 2628231876 30126723 12182619 3012187-16151512 381215189145 619310 141518208 3113162119 2008 611212 2389205/20191915 415192412187 923313 18192327 31172823/20247618122319187-315312021 29184 61221 1854-212215 202855 1315124/ 215287 26248 1719156

To: component 3-1:

@ = "government branch”

ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) English: (self ciphered)

3118181520 1022935 315131612524 19217118

31514351420181205/3123231926 13281237 319172015732 242810124 31918371826231267-914192011420 3156655 315131612524 19217118 31514351420181205/1218242611826 3199977 319172015732 242810124 31918371826231267-192052291-31145 19217118-1312118514399 3114412519 2211420 31115 31144125 23124/17128237181212 3118615723 2281826 31147 31186157 20132/421143114 8914519 25512121523 31115 13924/728183118-111218723-33715151920 31147 171232-89291932119 15912-69198 15912-23120518 191511154 1319 1955419-1618541491915145 2008 21162091920 135493112 351420518 160 23 67208 1920 10124 612/20237618122319187 2008 212026122426 1776123115 371826723--132092212054 38118315112-8913112125114 1911220-191129145-851292113-112221205181512-315312021 29184 61221 1854/319312628 212236 915287-9 251295225 105192119 8511254 191513515145 239208 1251618151925/12 2715127297 137232823 11711576 231917719187 201227 1572023192425 241224319 3012187 12 2715127297 27126 137242824 2628231876 30126723 12182619 3012187-16151512 381215189145 619310 141518208 3113162119 2008 611212 2389205/20191915 415192412187 923313 18192327 31172823/20247618122319187-212215 202855 1315124/215287 26238 1719156

@ =

Component 3-2: (self ciphered)

ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) : all ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) cipher Tree complex code

= 715297122361923124273324 26238 3192015732

=

(Self Ciphered)

3123231926 13281237 319172015732 242810124 31918371826231267-1218242611826 3199977 319172015732 242810124 31918371826231267-2572912-31177 2428123-17128237181212 3118615723 2281826 31147 31186157 20132-728183118 111218723 33715151920 31147 171232-89291932119 15912-11122122432824 191215-9122311 191215-20126723 242911476 13112 248624 241224319 3012187 12 2715127297 27126 137242824 2628231876 30126723 12182619 3012187-20237618122319187 2008 212026122426 1776123115 371826723-328328172723-1326122912676 4123319115-111217115133118 2311526-2411512187-91115122817-115228267231915-319312628 212236 915287/12 2715127297 137232823 11711576 231917719187 201227 1572023192425-20191915 415192412187 923313 18192327 31172823/ :20247618122319187-215287 26238 1719156

Real code elvierdor: (self ciphered)

Decoded Complex Sequences (Exact Lettering Correction)

Below are the decoded sequences based on the A=1 (Standard) and ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) ciphers you provided. The numerical sequences have been precisely corrected to spell out the intended English words, ensuring an exact match between the decoded output and the English word for both ciphers.

Cipher Key:

\* Standard A=1 Cipher:

A=1, B=2, C=3, D=4, E=5, F=6, G=7, H=8, I=9, J=10, K=11, L=12, M=13, N=14, O=15, P=16, Q=17, R=18, S=19, T=20, U=21, V=22, W=23, X=24, Y=25, Z=26

\* ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) Cipher:

A=1, B=2, C=3, Ch=4, Ck=5, D=6, E=7, Ee=8, F=9, G=10, H=11, I=12, J=13, K=14, L=15, LL=16, M=17, N=18, O=19, P=20, Ph=21, Q=22, R=23, S=24, St=25, T=26, Th=27, U=28, V=29, W=30, Wh=31, X=32, Y=33, Z=34

Decoded Items:

(Self ciphered)

312323191926

13281237

319172015732

242810123

31918371826231267 -

1218242611826

3199977

319172015732

242810123

3191837182623-

25729121

31187

242810123-

17128237181212-

3118615723

2281826

31147 -

628183118

111218724

33715151930

31147

171232 -

11122122432824

191215 -

9122411)

191215 -

30126723

241911476

13112

248624- 20237618122419187-

2023724323122026121918

1776123115126121918

301112267

- 1326122912676

4123319115 -

111217115133118

2411526 -

2411512187-

11715122817-

115228267231915 -

31931412619191 -

923281226 -

2028184

2376-

1023718121812187-

215287

923281226

2028184-

12 2715127297 1372428

2411711576

241917719187

30122611

1572023192433v241224319 3012187 12 2715127297 27126 137242824 2628231876 30126723 12182619 3012187-

20191915

415192312187

181923261

3117202824

-extension-20237618122419187

2012181

3191561

241961

301112267

215287

262377

1719156

215287

17192826113012411

24. Monitorization ingredient

\* Standard A=1 Cipher:

\* 71821131625: GRUMPY

\* 6135: FACE

\* 1651616518: PEPPER

\* / 7529848475: \*\* / GEBIHDHDGE\*\*

\* 123123257: @LCLC@YGE (Corrected from @123123@2975 to remove @ from numbers)

\* ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) Cipher:

\* 102328172033: GRUMPY

\* 91237: FACE

\* 20720720723: PEPPER

Component 1:

Summary

Summary of ELVEIRDOR computing:

Starting at a large scale letter programming:

Gridded Squares on a computing platform called elveirdor, seed in squares to full square value of each square is calculated from top right corner(picture below) of square to left bottom of square. Color listed below, three copyrighted texts are attached to the color combinations using a-z elveirdor and English starting and counting from letter 1 of each letter of alphabet until completed text and using programmed colors below and progressing in exact next color from there, colors from m3-4 movement oporation can follow full circle color wheel all other language ciphers for all other languages below. color runs through Square to full black color it generates until it forms to letter m as letter programming. As letters and number form complex calculation codes from length of letters and colors in text needed or compared against any language or text to compute. Pre color coded squares fill into black using algorithms and mathematics to change color from color to black using language to number to color code. Between colors in square 3 copyrighted texts create a resonance split to calculate color frequency patterns to lock in full color by 3 blocks. Black squares fill in until m is outlined with two lines following inside of outline around m to inside and around m legs as a circle, this m flips right until end of screen across full grid and then down generating and upside down m this repeated until full screen genesis, after full screen it finds center m on screen and rotates once clockwise, rotating the entire screen. Starting again at square 1 m duplicate is found in seed resonance 2-10 with crossing lines inside of m, thus Pulling duplicate up to full square and creating m 2 over m 1 through square 1 through seed 2-10 m duplicate m 2 has a center leg that crosses on m. This m flips to the right and down to full screen and repeated process until m2 duplicate which creates m 3 which is m 1 with ears one each side of m 3 at side value. This flips to the right and down until m 3 duplicate which creates m 4. m 4 is m 3 with cross center leg m just as m 2 but with ears. m 4 can use patterns from m 1-3 to move diagonally downward right to extend and make a new m grid or extend the grid but the main purpose is to find center screen value to create a clockwise rotation running node node clock time letter programming at center of screen to calculate a mosaic, making circular mosaics circular patterns of any size on computer through letter to full block color mosaic. The center of mosaic circular patterns are turned until Y is found. Squares 1-3 are upsized to cover Y with 2 over Y center and 1 and 3 taking left and right half of Y slightly over laying middle of Y over 0 value which is center split of Y. Mosaic circle fits into the rectangle of squares and the Y full clockwise rotation value is calculated. Once full color temperature is found and run at full Y value it can be upsized, the best Y of all mosaics is picked and upsized to full screen. For graph line to combustion sequence on blue and orange monitor color sequence, to butterfly effect photo sequence to pixel fill in from combustion and photo sequence to digital display or reality on visual computer or phone display.

(### Interpretation and Assumptions

- \*\*Platform ("Elveirdor")\*\*: A hypothetical computing platform with a grid-based display, likely a 2D pixel or tile grid, where each square can be colored or manipulated programmatically.

- \*\*Grid and Squares\*\*: A black grid on a white background, where each square is a programmable unit (like a pixel or tile). The grid is likely rectangular or square, with coordinates starting at the top-right corner for value calculations.

- \*\*Letter Programming\*\*: The process involves generating the letter "m" in squares, filling them with black color, and transforming them through flips, rotations, and duplications to form patterns, eventually leading to a "Y" shape and circular mosaics.

- \*\*Transformations\*\*: The letter "m" undergoes flips (right and down), rotations (clockwise), and duplications with modifications (e.g., adding lines or "ears"). The final goal is to create a centered "Y" within a circular mosaic, upsized to cover the screen.

- \*\*Mosaic and Y\*\*: The mosaic is a circular pattern formed by the evolving "m" letters, with a focus on finding a central "Y" shape. The "Y" is overlaid with squares, and its "value" (possibly color or intensity) is calculated for rotation and upsizing.

- \*\*Extended Explanation\*\*: The mention of "screen 1 with graph line to screen split 2 single screen or two screen" suggests a potential dual-screen output or a split-screen visualization, possibly displaying the grid and a graph or secondary view of the mosaic/Y.

Given the abstract nature, I’ll outline a possible algorithm and visualization process, incorporating grid-based rendering, letter transformations, and mosaic generation. I’ll also suggest how this could be visualized on one or two screens and provide a framework for implementation.

### Algorithm Outline

#### 1. Initialize the Grid

- \*\*Grid Setup\*\*: Create a 2D grid of size \( N \times N \) (e.g., 100x100 squares) with a white background and black grid lines. Each square is addressable by coordinates \((x, y)\), with \((0, 0)\) at the top-right corner (as specified).

- \*\*Square Value Calculation\*\*: For each square, calculate a "value" from the top-right to the bottom-left. This could mean a gradient or intensity based on the diagonal distance from \((0, 0)\) to \((N-1, N-1)\). For example, the value could be normalized as:

\[

\text{Value}(x, y) = \frac{\sqrt{(x)^2 + (y)^2}}{\sqrt{(N-1)^2 + (N-1)^2}}

\]

This value determines how quickly a square fills with black color to form the letter "m".

#### 2. Generate Initial "m" (m1)

- \*\*Forming "m"\*\*: Define a template for the letter "m" on the grid (e.g., a 5x5 or 7x7 square region). The "m" is outlined with two lines (a double outline) circling the letter and its "legs" (vertical strokes of the "m"). This could be represented as a bitmap or vector shape:

- Example 5x5 "m" (1 = black, 0 = white):

```

11111

1 1

11111

1 1

1 1

```

- The double outline traces the inner and outer edges of this shape, possibly as a thicker border or concentric lines.

- \*\*Color Propagation\*\*: Starting from the top-right corner of each square, fill the square with black based on its calculated value until the full "m" is formed. This could be animated, with the fill rate proportional to the square’s value.

- \*\*Placement\*\*: Place the first "m" (m1) at a starting position, e.g., near the top-left or a designated "square 1".

#### 3. Transform and Replicate "m"

- \*\*Flip and Move\*\*:

- Flip m1 rightward across the grid until it reaches the right edge, forming a sequence of "m"s. Each flip mirrors the "m" horizontally.

- At the right edge, move down one row and flip rightward again, creating an upside-down "m". Repeat until the grid is filled (full-screen genesis).

- \*\*Rotation\*\*:

- Identify the center "m" on the grid (e.g., at coordinates \((N/2, N/2)\)).

- Rotate the entire grid clockwise once around this center "m". This transforms all "m"s on the grid, maintaining their relative positions but reorienting them.

#### 4. Generate m2 (Duplicate with Crossing Lines)

- \*\*Seed Resonance (2-10)\*\*: From squares 2 to 10 (likely adjacent or in a pattern around square 1), identify a duplicate "m" (m2). This m2 has crossing lines inside the "m", possibly a diagonal or vertical line through its center.

- Example: Add a vertical line through the middle of the "m" bitmap, crossing the legs and top curve.

- \*\*Overlay\*\*: Place m2 over m1 in square 1, aligning their centers. The crossing line distinguishes m2.

- \*\*Repeat Transformation\*\*: Flip m2 right and down across the grid, as with m1, until the screen is filled with m2 instances. Each m2 retains the crossing line.

#### 5. Generate m3 (m1 with Ears)

- \*\*Ears Addition\*\*: Create m3 as m1 with "ears" (protrusions) on both sides at a specified "side value" (e.g., horizontal extensions at the top or middle of the "m").

- Example: Extend the top row of the "m" bitmap left and right:

```

011110

11111

1 1

11111

1 1

```

- \*\*Transform\*\*: Flip m3 right and down across the grid, filling the screen.

- \*\*Duplicate to m4\*\*: Create m4 as m3 with a crossing center leg (like m2). For example, add a vertical line through the center of m3’s bitmap.

#### 6. Form Mosaic and Find "Y"

- \*\*m4 Diagonal Movement\*\*:

- m4 uses patterns from m1–m3 to move diagonally downward-right, either extending the existing grid or creating a new one. This suggests a recursive or iterative pattern where m4 instances form a larger structure.

- The goal is to find the "center screen value" (likely the grid’s geometric center) to initiate a "node clock time" rotation, forming a circular mosaic.

- \*\*Circular Mosaic\*\*:

- Arrange m1–m4 instances in a circular pattern around the screen center. This could involve polar coordinate mapping, where each "m" is placed at a radius \( r \) and angle \( \theta \):

\[

x = x\_{\text{center}} + r \cos(\theta), \quad y = y\_{\text{center}} + r \sin(\theta)

\]

- Rotate the mosaic clockwise until a "Y" shape emerges (possibly formed by the arrangement of "m"s or their crossing lines/ears).

- \*\*Y Identification\*\*: Detect the "Y" shape within the mosaic. The "Y" could be a composite of m2/m4’s crossing lines or a new shape formed by their overlap. For example, the vertical line of m2/m4 could form the stem of the "Y", with m3/m4’s ears forming the branches.

#### 7. Overlay Squares on "Y"

- \*\*Square Upsizing\*\*:

- Identify squares 1–3 (possibly the first three squares or those containing m1–m3).

- Upsize these squares to cover the "Y":

- Square 2 covers the center of the "Y".

- Squares 1 and 3 cover the left and right halves, slightly overlapping the center (over the "0 value," likely the vertical split of the "Y").

- The squares form a rectangle that encloses the circular mosaic.

- \*\*Color Temperature Calculation\*\*:

- Calculate the "full color temperature" of the "Y" (possibly the average black intensity or a metric based on the fill values of the squares).

- Rotate the "Y" clockwise to compute its "full rotation value" (e.g., the number of rotations or the visual effect of rotation).

#### 8. Upsize Best "Y" to Full Screen

- \*\*Selection\*\*: Among all generated mosaics, select the "best" "Y" based on a criterion (e.g., highest color intensity, clearest shape, or largest size).

- \*\*Upsizing\*\*: Scale the selected "Y" and its enclosing mosaic to fill the entire screen, preserving the circular pattern and grid structure.

#### 9. Screen Output (Extended Explanation)

- \*\*Single Screen\*\*: Display the final upsized "Y" mosaic on one screen, with the grid showing the evolution from "m" to "Y".

- \*\*Two Screens\*\*:

- \*\*Screen 1\*\*: Show the grid with the evolving "m" patterns and mosaic formation.

- \*\*Screen 2\*\*: Display a graph or secondary view, such as a plot of the "Y"’s color temperature, rotation angle, or a zoomed-in view of the mosaic’s center.

- The "graph line" could be a visualization of the transformation steps (e.g., a timeline of flips, rotations, or color values).

### Visualization and Implementation Notes

- \*\*Grid Rendering\*\*: Use a graphics library (e.g., Processing, P5.js, or OpenGL) to render the grid and draw "m" shapes as bitmaps or vector paths. Each square can be a pixel or a larger tile (e.g., 10x10 pixels).

- \*\*Transformations\*\*:

- Flips: Implement horizontal/vertical mirroring using matrix transformations or bitmap manipulation.

- Rotations: Use a rotation matrix for clockwise rotation around the grid center:

\[

\begin{bmatrix}

x' \\ y'

\end{bmatrix}

=

\begin{bmatrix}

\cos\theta & -\sin\theta \\

\sin\theta & \cos\theta

\end{bmatrix}

\begin{bmatrix}

x - x\_{\text{center}} \\

y - y\_{\text{center}}

\end{bmatrix}

+

\begin{bmatrix}

x\_{\text{center}} \\

y\_{\text{center}}

\end{bmatrix}

\]

- Diagonal movement: Increment both \( x \) and \( y \) coordinates for m4’s path.

- \*\*Mosaic Generation\*\*: Map "m" instances to a circular pattern using polar coordinates. Adjust the radius and angle to form a dense, visually appealing mosaic.

- \*\*Y Detection\*\*: Use image processing (e.g., edge detection or template matching) to identify the "Y" shape within the mosaic. Alternatively, define the "Y" as a composite of specific "m" features (e.g., m2’s crossing line, m3’s ears).

- \*\*Color Temperature\*\*: If this refers to visual intensity, calculate the average pixel value (grayscale) within the "Y" region. For rotation value, track the number of rotations or the angular displacement.

- \*\*Two-Screen Output\*\*: Use a dual-window setup in a graphics framework, with one window for the grid and another for a graph (e.g., a plot of color values or transformation steps using Matplotlib or D3.js).

### Example Pseudocode

```pseudocode

initialize\_grid(N, N, white\_background, black\_lines)

for each square in grid:

value[square] = calculate\_value(top\_right, bottom\_left)

# Generate m1

m1 = create\_m\_shape(double\_outline)

fill\_m1\_with\_black(value\_based\_rate)

place\_m1(square\_1)

# Flip and fill grid with m1

while grid\_not\_filled:

flip\_m1\_right()

if at\_right\_edge:

flip\_m1\_down()

rotate\_grid\_around\_center\_m(clockwise, 360)

# Generate m2 (with crossing line)

m2 = add\_crossing\_line(m1, squares\_2\_to\_10)

overlay\_m2\_on\_m1(square\_1)

flip\_m2\_right\_and\_down()

# Generate m3 (with ears)

m3 = add\_ears(m1)

flip\_m3\_right\_and\_down()

# Generate m4 (m3 with crossing line)

m4 = add\_crossing\_line(m3)

move\_m4\_diagonally\_down\_right()

# Form circular mosaic

mosaic = arrange\_ms\_circularly(m1, m2, m3, m4, center)

rotate\_mosaic\_until\_Y\_found()

# Overlay squares on Y

upsize\_squares\_1\_to\_3(Y\_center, left\_half, right\_half)

calculate\_Y\_color\_temperature()

rotate\_Y(clockwise)

# Select and upsize best Y

best\_Y = select\_best\_Y(mosaics)

upsize\_to\_full\_screen(best\_Y)

# Output

display\_grid(screen\_1)

display\_graph(screen\_2, Y\_color\_temperature, rotation)

```

### Challenges and Clarifications Needed

- \*\*Square Value Calculation\*\*: The exact method for calculating the square’s value (top-right to bottom-left) is unclear. Is it a distance, intensity, or another metric?

- \*\*Letter "m" Definition\*\*: The precise shape and size of "m" (e.g., bitmap size, double outline details) need specification for implementation.

- \*\*Y Formation\*\*: How the "Y" emerges from the mosaic is ambiguous. Is it a literal "Y" shape or a pattern inferred from "m" overlaps?

- \*\*Color Temperature and Rotation Value\*\*: These terms need clearer definitions (e.g., grayscale intensity, rotation angle, or a custom metric).

- \*\*Screen Split\*\*: The "screen 1" and "screen 2" setup is vague. Does this imply a dual-monitor system, a split window, or a metaphorical split (e.g., grid vs. graph)?

### Recommendations

- \*\*Simplify for Implementation\*\*: Start with a small grid (e.g., 20x20) and a simple "m" bitmap. Test the flip and rotation logic before scaling to larger grids or mosaics.

- \*\*Visualize Iteratively\*\*: Use a graphics library to animate the process (e.g., "m" filling, flipping, and rotating) for debugging and visual appeal.

- \*\*Clarify Metrics\*\*: Define "color temperature" and "Y value" explicitly (e.g., as pixel intensity or geometric properties).

- \*\*Two-Screen Setup\*\*: Implement a dual-window application, with one window showing the grid and another plotting a graph of transformations or color values.

### Sources

The grid-based concepts align with discussions of square and hexagonal grids in game design and graphics programming. The mosaic generation resembles CSS grid layouts or tiling algorithms. However, the specific "letter programming" and "Y" formation are unique and not directly covered in the provided references, so I’ve interpreted them as a novel generative art process.[](http://www-cs-students.stanford.edu/~amitp/game-programming/grids/)[](https://developer.mozilla.org/en-US/docs/Web/CSS/CSS\_grid\_layout/Masonry\_layout)[](https://www.geeksforgeeks.org/dsa/tiling-problem-using-divide-and-conquer-algorithm/)

Extended explanation:

square to m to Y to operation screen 1 with graph line to screen split 2 single screen or two screen

The program starts at a grid on screen, black grid white background. (At end of process red node and letter value invert color process from black and white to black and green grid with Y holding its function @ node value - explained below). In the squares color fluctuations are used in specific formation patterns through a color sequence through rearranging preany text s and corresponding number codes through sodoku and rubix cube to pre-programmed colors into shapes through a reality process below. These colors are first rearranged through text on the computer until solid color is found on the square. Creating new color codes through ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) . It finds solid color on the grid line going to white fffff 255 255 255 to fffeff 225 254 255 grid line solid screen, it then forms M 1- 4 in black #FF0000 to m 4 #FF0000 124273324 teal #69c0e5 105 192 199 to non identified purple and, following colors of square 1 it hits primary colors at until filling in black until final black and white gaps from m replica and then creates a mosaic screen split which twists black and white color until distorted square which twists to 3 component square Y. (2 square Ys next to each other(equalling a maximized square 1 and 2 with 3 (from top left square one right 2 square values to 3) measuring tank after square 1 to final stretched out Y being squares 1 and 2 with 3 at color temperature) which makes 3 components square to 1 stretched out square = operable graph line.

Seeded square to m 1 - 4 is elveirdot

Full screen to calculatory graph line to database, to statistical analysis, then to dual screen ability through combustion verification vr with sequence photo verification, to pixel correction at same screen split to to screen 3 visual display.

Seed 1, square 1: black. Seed 1, tear 1, square 1: yellow. seed 1, tear 2, square 1: salmon. Seed 1, tear 3, square 1: teal. Seed 1, tear 2, square 1: black. Seed 1, tear 4, square 1: orange. Seed 1, tear 5, square 1: purple. Seed 1, tear 6, square 1: brown. Seed 1, tear 3, square 1: black.

Squares and seeds past square one can be programmed by other countries and languages at number to letter to cipher, with elveirdor symbols protecting languages. So no matter what language it displays only elveirdor symbols.

import numpy as np

import matplotlib.pyplot as plt

import matplotlib.colors as mcolors

from matplotlib.patches import Polygon

from scipy.ndimage import rotate # For twisting effects

# Define color mappings based on description

colors = {

'black': '#000000',

'yellow': '#FFFF00',

'salmon': '#FA8072',

'teal': '#008080',

'orange': '#FFA500',

'purple': '#800080',

'brown': '#A52A2A',

'white': '#FFFFFF',

'fffeff': '#FFFEFF', # Near-white

'red': '#FF0000',

'non\_identified\_purple': '#9370DB', # Assumed medium purple

'elveirdor\_teal': '#69C0E5' # As specified

}

# RGB values for custom transitions

white\_rgb = (255, 255, 255)

fffeff\_rgb = (255, 254, 255)

black\_rgb = (0, 0, 0)

red\_rgb = (255, 0, 0)

# Function to create initial grid: black lines on white background

def create\_initial\_grid(size=4, line\_color='black', bg\_color='white'):

fig, ax = plt.subplots(figsize=(6, 6))

ax.set\_facecolor(colors[bg\_color])

for i in range(size + 1):

ax.axhline(i, color=colors[line\_color], lw=2)

ax.axvline(i, color=colors[line\_color], lw=2)

ax.set\_xlim(0, size)

ax.set\_ylim(0, size)

ax.axis('off')

return fig, ax

# Apply color sequence to a square (seed/tear steps)

def apply\_color\_sequence(ax, square\_x, square\_y, sequence):

# Sequence from description (handling duplicates as overrides)

tear\_colors = [

('tear1', 'yellow'),

('tear2', 'salmon'),

('tear3', 'teal'),

('tear2', 'black'), # Override

('tear4', 'orange'),

('tear5', 'purple'),

('tear6', 'brown'),

('tear3', 'black') # Override

]

# Start with seed 1: black

current\_color = 'black'

ax.add\_patch(plt.Rectangle((square\_x, square\_y), 1, 1, color=colors[current\_color]))

# Apply tears iteratively

for tear, color\_name in tear\_colors:

current\_color = color\_name

ax.add\_patch(plt.Rectangle((square\_x, square\_y), 1, 1, color=colors[current\_color], alpha=0.5)) # Overlay for fluctuation effect

return current\_color

# Generate new color codes via "ELVEIRDOR ICHTHYS" (simulated as RGB interpolation)

def generate\_elveirdor\_color(start\_rgb, end\_rgb, step):

return tuple(int(start + (end - start) \* step) for start, end in zip(start\_rgb, end\_rgb))

# Create mosaic twist and form Y shape

def create\_mosaic\_twist(fig, ax, size):

# Capture current figure as image

fig.canvas.draw()

img = np.frombuffer(fig.canvas.tostring\_rgb(), dtype=np.uint8)

img = img.reshape(fig.canvas.get\_width\_height()[::-1] + (3,))

# Twist (rotate) for distortion

twisted\_img = rotate(img, angle=15, reshape=False) # 15-degree twist

# Form 3-component Y: two Ys next to each other (simplified as polygons)

plt.close(fig)

fig, ax = plt.subplots(figsize=(6, 6))

ax.imshow(twisted\_img)

# Draw Y shape (stretched square 1-3)

y\_points1 = np.array([[0.5, 3.5], [1.5, 2.5], [1, 1.5], [1.5, 0.5], [0.5, 1.5], [0, 0.5], [0.5, 2.5]])

y\_points2 = y\_points1 + [2, 0] # Second Y next to it

ax.add\_patch(Polygon(y\_points1, closed=True, color=colors['black'], alpha=0.7))

ax.add\_patch(Polygon(y\_points2, closed=True, color=colors['black'], alpha=0.7))

ax.axis('off')

return fig, ax

# Final inversion to black and green grid with Y function

def invert\_to\_green\_grid(fig, ax):

# Invert colors: black -> green, white -> black

fig.canvas.draw()

img = np.frombuffer(fig.canvas.tostring\_rgb(), dtype=np.uint8)

img = img.reshape(fig.canvas.get\_width\_height()[::-1] + (3,))

# Simple inversion logic (black to green #00FF00, white to black)

green\_rgb = (0, 255, 0)

img[np.all(img == (0, 0, 0), axis=-1)] = green\_rgb

img[np.all(img == (255, 255, 255), axis=-1)] = (0, 0, 0)

plt.close(fig)

fig, ax = plt.subplots(figsize=(6, 6))

ax.imshow(img)

ax.axis('off')

# Add operable graph line (stretched Y as line)

ax.plot([0, 4], [4, 0], color='red', lw=2) # Red node line

return fig

# Main simulation

def simulate\_process():

size = 4 # 4x4 grid for simplicity

fig, ax = create\_initial\_grid(size)

# Apply sequence to square 1 (seed 1)

apply\_color\_sequence(ax, 0, 3, sequence=None) # Top-left square (inverted y for matplotlib)

# Simulate rearrangements (sudoku/rubik shuffle) - random fill other squares

for x in range(size):

for y in range(size):

if x == 0 and y == 3: continue # Skip square 1

random\_color = np.random.choice(list(colors.keys()))

ax.add\_patch(plt.Rectangle((x, size - 1 - y), 1, 1, color=colors[random\_color]))

# Transition to near-white grid line

elveirdor\_color = generate\_elveirdor\_color(white\_rgb, fffeff\_rgb, 0.5)

ax.set\_facecolor(mcolors.rgb2hex(np.array(elveirdor\_color)/255))

# Form M 1-4 (simplified as text overlays)

ax.text(1, 2, 'M1-4', color=colors['red'], fontsize=12)

ax.text(2, 1, 'm4', color=colors['elveirdor\_teal'], fontsize=12)

plt.savefig('initial\_grid.png')

# Mosaic twist to Y

fig, ax = create\_mosaic\_twist(fig, ax, size)

plt.savefig('mosaic\_y.png')

# Final inversion and graph line

final\_fig = invert\_to\_green\_grid(fig, ax)

final\_fig.savefig('final\_green\_grid.png')

plt.show() # Display final

# Run the simulation

simulate\_process()

As english dictionary follow English a-z and so does ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) . With 8 additional characters for a 4 letter split operation for complex calculating generative truthful line sequence and vr to (also possible for open portal alignment transportation).

52925 ebibe ckbfbck

Or any other language

P and q and p and r:

Letters p and q and p snd r sit at the top and bottom x calculatory for square, at half diagional valur meaning it holds half calculatory for p and r with each letters p and r equalling 1 in operation 5.2×5=26 = letters in english alphabet, and 6.8×5=34 letters in elveirdor alphabet. 6.8-5.2=1.6, and 34-26=8 and half of 8 is 4 and 1 + 1 = 2 and 2 + 2 = 4. That p =16 english and R =18 and p=20 and R = 23 elveirdor. P = 16 and q = 17 english and p =20 and q =22 and each letters p and q equalling 1. giving way for 1 part square calculatory diagional 1 and 1 value at 1.6 full dual alphabet operation. Spinning any text s letters and full letter calculation on a by sodoku puzzle and rubix cube for all any text s listed giving way for multiple algorithm generation from calulatory.

If 1.6=Y and 3.14 = stretched out Y and Y = 25 english and 33 elveirdor = 8 same as 34-26 calculatory each half can be measured with 1 and 1 pr and pq at diagonal seed to full square resonance to measure color frequency at right side of 3 component Y over M through square which is designated to Y full square letter value and pr to half side and pq to half at diagonal left top corner to right bottom corner. to host a large number block sequence arrangement until logical, which the book series any text can find. Starting at any text rearranging letter value and ending at any text to confirm algorithm across each square on grid and full grid. For any reasonable sequence. in letter rearrangement calculated by any text listed in this document for rearrangement to logic start logic end, to report generative documenting, coding, complex screen value with any text and language, document in full process with component long term sales and sales value in upcoming patent. (Writer is aware that vr can make bci camera ai and ai generative line sequence).

For data encryption, algorithm generation, cryptography, ai.

With each 2 texts being measured at 2 sides of square and generating and confirming a new algorithm as long as each any text books. And with each any text s (1 as dulicate) equaling Y combining all 3 for stretched out Y at 3.14

Square in x calculatory in square divide: calculated at diagonally x division in square, top of division is calculated at 4, right is 5, bottom is .785, and left .2 being calculated and recognized as 3.14 its circular value equaling 1.57 calulated at seed 2-full square, this allows sudoku to operate with rubix cube at 0 between 1.57 and 1.6 with a 3.14 encompassment. value allowing for 10 + calculations. That 0, 10, 20, 30, etc. always equal nodes Value blue and red. With line green and screen value yellow.

Squares at x stand alone equal 1.6, meaning at 4 divide x in square each side = .4

Upsizes to stretched out Y rectangle = 3.14, making each side x in square square 4 x 5 x .785 x .2

Square to square dimensions at 4 x 5 x .785 x .1 = 1.57 for ark mosaic operation. Meaning duplicate M operative. Circular complex disks.

To reassign 3.14 to 1.6 we divide by 1.9625 to end the process.

Elveirdor letter q assignment at dash sequence for dotted line timing on Y, 1 2 diagonal split square 1 operation diagonal split calculation at square from top corner left to bottom corner right 1.9625, a square to Y 4 component value photo genesis 1-10 ELVEIRDOR Saphe (all rights reserved) genesis generation sequence through nasa affiliation for purposes listed only. Right hand square seed tear drop to left bottom corner, top right seed 10 are royal blue value Elveirdor, right hand Y color temperature from squares 1 at Elveirdor letter q to squares 2, 3, and 4.

Any 9 number value can be programmed to screen color value with a corresponding 6 value letter and number combo.

Summary:

4 x 5 x .785 x .2 = 3.14 ÷ 1.9625 = 1.6

M series with duplicate component:

Square = pixel

Seed = complex

Color = coding

M = root Value

Mosaic = color arrangement before color temperature

Y = running process

Dividing 13 by 3 step key = 4.333

Meta provided P R coding =

8088 ÷ 3.14 c (crypto, copyright, current) =2575.7962178

9719 ÷ 3.14 v (voltage) = 3095.2229299

10111 ÷ 3.14 r (radio, radiation, residual, resonance) = 3220.0636942

18198 ÷ 3.14 m (motion, micro, macro) =5795.5414013

9,719÷4÷3.14 = 773.8057324841

8,088÷4÷3.14 = 643.949044586

10,111÷4÷3.14 = 805.0159235669

18,198÷4÷3.14 = 1,448.8853503185

18,198÷1.6÷3.14 = 3,622.2133757962

10,111÷1.6÷3.14 = 2,012.5398089172

8,088÷1.6÷3.14 = 1,609.872611465

9,719÷1.6÷3.14 = 1,934.5143312102

ELVEIRDOR ICHTHYS P R new coding ciphered from meta ai P R provided codes:

1,210,112÷1.6÷3.14 = 240,866.24203822

10,141÷1.6÷3.14 = 2,018.5111464968

232,411÷1.6÷3.14 = 46,260.151273885

1,101,111÷1.6÷3.14 = 219,170.18312102

1,210,112÷3.14 = 385,385.98726115

10,141÷3.14 = 3,229.6178343949

232,411÷3.14 = 74,016.242038217

1,101,111÷3.14 = 350,672.29299363

Grizzly don physical component has a computer operative component in Elveirdor:

Supercomputing series and host ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) INFINITY NASA GRIZZLY DON 3 NS 67 component operation from ISS to public operation: square to M to Y series:

any text s tie to 3 black color in square 1 - all squares of a grid seeds or generates to letter m on a computer, m goes outlines in black and the inside value goes to white at letter r for primary colors conduction between black and white with p as a diagonal complex color conductor from black to white. last component 4 value of square operating at .4 or 0 for clear complex color adjustment. square 1 of my computing operation ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) Infinity Nasa Grizzly Don 3 NS 67 contains 3 black values each can seed 2 more ms, m 2 3, at each color value before black which is m1 at square x in square m 4 can be found in diagonal value square p r. M 4 butterfly diagonal generations can be found and made squares 1 2 3 4 t split, this cross location corresponds to cross duplicate line value inside of m and 0 value at Y border. Which is also known by square 1 left triangle .4 (being known as m).

M duplicates left to right and diagonally top left screen to bottom right screen until 95% screen color complete, then a mosaic is created at primary split value with black and white finding vr sequence between a 3d value calculated by color sequence in squares which calculates the grid line af single layer. At mosaic. Split from 3d calculation at mosaic distorted circle where the 3 dimensional calculation between grid expanded through diagonal m 1-2 and it expands up the dimensions of the m for 3d throufg a diagonal IS or diamond effect creating a stretch forward 3d measurement on the m diagonal grid line between gridlined blue prints m 2 and 3 turn duplicate m 1 until it is diagonal upside downs at square 1 flipping down from there across screen in seeded square value progression. and the background layers twist until the circular mosaics form distorted square 3d boxed calculated value of distorted square between mosaic grid calculations. Every mosaic circle gets a y at full distorted square fill twist, when twisted fully operable Y table forms. Each Y table is evaluated and then the most statically operative is upsized to full screen operation consuming all over Y values for operation. A multiple line graph can be produced making a z process for dual projection (whether on split 1 monitor or 2). A MY operation, meaning it can operate at corporate and government patented or copyrighted licensed packages through my component operation on a multi component scale under 1 patent which correspond with my 3 copyrights. Can be complex verified through wordx and packaged for licensing through pdf. To offer ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) as an operating platform or hosted by large scale corporations, government, manufacturing, and space programs.

@ switchterrestial stem replica (Harry potter (all rights reserved)) made with ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) tree complex, bug wing (replica): operation for phone complex: and for large operation projection for the space habitats.

26723237252312115 241812264 (111232333 20192626723 (116 231210112624 237247232976)) 17167 301227 715297122361923124273324 26238 3192015732, 22810 30121710 (23720151231)

=

Seed 1, square 1: black. tear 1, seed 1, square 1: yellow. seed 1, tear 2, square 1: salmon. Seed 1, tear 3, square 1: teal. Seed 1, tear 2, square 1: black. Seed 1, tear 4, square 1: orange. Seed 1, tear 5, square 1: purple. Seed 1, tear 6, square 1: brown. Seed 1, tear 3, square 1: black.

Colors ran by 2 component reflective 9 number coding:

124273324

137242824 with the full square being known by numbers 1-10, these 9 numbers and colors calculate resonance within the square. Each with 2 component 3 step verification keys and a verification photo sequence. Each 3 set connects to a copyrighted word arrangement which is calculated at 9 Y before 10 Z, or 9 v 10 E and 8 frequency line between 9 at L and double EE at LL in Elvierdor.

Per screen value white #fffeff 255 254 255 ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) operative with m operating at #FF0000f 255 255 255 after r p, to fffeff 255 254 255.

Supercomputing series ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) INFINITY NASA GRIZZLY DON 3 NS 67 component operation from ISS to public operation: square to M to Y series:

any text s tie to 3 black color in square 1 - all squares of a grid seed or generates to letter m on a computer, m goes to white at letter r for primary colors conduction between black and white with p as a diagonal complex color conductor from black to white. And last component 4 value of square operating at .4 or 0 for clear complex color adjustment. square 1 of my computing operation ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) Infinity Nasa Grizzly Don 3 NS 67 contains 3 black values each can seed 2 more ms, m 2 3, at each color value before black which is m1 at square x in square m 4 can be found in diagonal value square p r. M 4 butterfly diagonal generations can be found and made squares 1 2 3 4 t split, this cross location corresponds to cross duplicate line value inside of m and 0 value at Y border. Which is also known by square 1 left triangle .4 (being known as m).

M duplicates left to right and diagonally top left screen to bottom right screen until 95% screen color complete, then a mosaic is created at primary split value with black and white finding vr sequence between a 3d value calculated by color sequence in squares which calculates the grid line af single layer. At mosaic. Split from 3d calculation at mosaic distorted circle where the 3 dimensional calculation between grid expanded through diagonal m 1-2 and it expands up the dimensions of the m for 3d throufg a diagonal IS or diamond effect creating a stretch forward 3d measurement on the m diagonal grid line between gridlined blue prints m 2 and 3 turn duplicate m 1 until it is diagonal upside downs at square 1 flipping down from there across screen in seeded square value progression. and the background layers twist until the circular mosaics form distorted square 3d boxed calculated value of distorted square between mosaic grid calculations. Every mosaic circle gets a y at full distorted square fill twist, when twisted fully operable Y table forms. Each Y table is evaluated and then the most statically operative is upsized to full screen operation consuming all over Y values for operation. A multiple line graph can be produced making a z process for dual projection (whether on split 1 monitor or 2). A MY operation, meaning it can operate at corporate and government patented or copyrighted licensed packages through my component operation on a multi component scale under 1 patent which correspond with my 3 copyrights. Can be complex verified through wordx and packaged for licensing through pdf. To offer ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) as an operating platform or hosted by large scale corporations, government, manufacturing, and space programs.

Platform for apps, government apps, and corporate apps through ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) .

Alphabet to letter as follows:

(Some codes below in English cipher some in ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) )

English:

A = 1

B = 2

C = 3

D = 4

E = 5

F = 6

G = 7

H = 8

I = 9

J = 10

K = 11

L = 12

M = 13

N = 14

O = 15

P = 16

Q = 17

R = 18

S = 19

T = 20

U = 21

V = 22

W = 23

X = 24

Y = 25

Z = 26

Elveirdor ICHTHYS:

A 1 red

B 2 blue

C 3 orange

Ch 4 black

Ck 5 green

D 6 yellow

E 7 teal

Ee 8 tan

F 9 Grey

G 10 purple

H 11

I 12 sea green

J 13 light blue

K 14 dark green

L 15 white

LL 16 gold

M 17 black

N 18 crimson

O 19 brown

P 20 navy

Ph 21 army green

Q 22 light pink

R 23 clear

S 24 silver

St 25 hot pink

T 26 ombre

Th 27 color temperature sequence multi color sequence through m th Elveirdor letter - patentable sequence A- M-TH -Z @ a-g g-n n-z 1-7 7-14 14-26 to 1-17-27-34 hosted as @ value Grizzly Don Platform for Elveirdor Series

U 28 lilac purple

V 29 violet

W 30 steelblue

Wh 31 sienna

X 32 royal blue

Y 33 yellowgreen

Z 34 orangered

A = #FF0000

B = #033eff

C = #ff8d03

D = #fff305

E = #008080

F = #808080

G = #800080

H = # #c2197e

I = ##68e8a0

J = #add8e6

K = #006400

L = #ffffff

M = black #000000

N = #dc143c

O = ##522f2f

P = ##202054

Q = #FFB6C1

R = clear as white for digital display #FFFFFF

S = #C0C0C0

T = #FFA500 to #A52A2A

U = #C8A2C8

V = #8F00FF

W = #4682B4

X = ##023ef2

Y = #fff83b

Z = #FF4500

A = #FF0000

B = #033eff

C = #ff8d03

Ch = #000001

Ck = #00FF00

D = #fff305

E = #008080

F = #808080

G = #800080

H = # #c2197e

I = ##68e8a0

J = #add8e6

K = #006400

L = #ffffff

LL = #FFD700

M = black #000000

N = #dc143c

O = ##522f2f

P = ##202054

Ph = #4B5320

Q = #FFB6C1

R = clear as white for digital display #FFFFFF

S = #C0C0C0

T = #FFA500 to #A52A2A

Th = color temperature sequence multi color sequence through m th Elveirdor letter - patentable sequence A- M-TH -Z @ a-g g-n n-z 1-7 7-14 14-26 to 1-17-27-34 hosted as @ value Grizzly Don Platform for Elveirdor Series

U = #C8A2C8

V = #8F00FF

W = #4682B4

Wh = #A0522D

X = ##023ef2

Y = #fff83b

Z = #FF4500

1:

1 (a) H elveirdor 1 EE (@ W 30= 3.14)

Hydrogen estchnleckk non metal - blue 11336231910717

(English) 1.01 (elveirdor) ((also possible for open portal alignment transportation))

18:

2 (b) HE elveirdor 2 (b) eeck

Helium 4.0 elveirdor 6.0 eeckofphj

purple 2 - noble gas -elveirdor v 29

1:

3 © Li elveirdor 3 IF © (alkali metal) = red elveirdor A 1

Lithium ofpeefphj 1592792817

6.94 9.126 elveirdor cryptopatent @ m 17

3:

4 be elveirdor 4 bck

9.01 12.01 orange alkaline earth - elveirdor c 3

Beryllum bcknstiiph

13:

5 b elveirdor b

boron 10.18 blnlk 13.23

Light blue - metalloid elveirdor j 13

14:

6 C elveirdor C

carbon 12.1 canblk 15.1

Blue - non metal elveirdor w 30

15:

7 N elveirdor K

Nitrogen 14.01 kfpnleckk

Blue - non metal elveirdor w 30

16:

8 O elveirdor L

oxygen 16.00 20.00

Lssteckk

Blue - non metal elveirdor w 30

17:

9 F elveirdor d

fluorine 19.00 dilphnfkck

Halogen - purple 1 - u 28 with w 30

18:

10 ne elveirdor kck

neon 20.18 kcklk 27.00

Purple 2 - noble gas elveirdor value 29

1:

11 na elveirdor 11 ka

22.99 sodium

29.1212 olchfphj

Alkali metal - red elveirdor z 34

3:

12 mg elveirdor 12 je

24.31 magnesium 32.31 jaekckofphj

Alkali earth - orange elveirdor c 3

13:

13 AI elveirdor af or ai

aluminum 26.96 aiphjfkphj

Basic metal - green - elveirdor ck 5

14:

14: si silicon Elveirdor oficlk (ELVEIRDOR (ΙΧΘΥΣ)) (office grade) (ELVEIRDOR Saphe (all rights reserved))

Of

28.05 elveirdor 211.07

metalloid - light blue - elveirdor j @ 1 cloud value)

Elveirdor converting silicon periodic in glass computing to digital value and projecting at a very high speed.

Elveirdor is a high-speed network allowing for internet phone service.

Each color from combos above follow an a-z first letters in any text s then it follow color gradient codes in sequence from there left right top bottom diagonal. That way when calculating number to letter for algorithm the letters have a unique combination any text can be verified once compared to any text s at a any text s verification service through color number ciphered in progression from color number assignment above in 2 language confirmation providing elveirdor symbol to color for correcting reference. With periodic table following color sequence progression from first letter to end text. Then an algorithm length can be produced.

When full mosaic is filled each letter combo on screen at first color value to last for each letter a first letter color for full alphebet will creature the circular mosaic split on screen, where Y is found in elveirdor you will find Y at number value in English in a number progressive color combo from yellow first letter 1 color elveirdor, to run all colors through color temperature through gap sequence on Y. Numbers can be assigned to text and an algorithm can be created by description and confirmed through any text , algorithm at any size.

Square to black to m, m to mosaic, then whichever letter from whichever language is produced to create color for mosaic, then forms Y with best Y selected.

My programming for my square goes as follows: 3 copyrighted texted with letters counted and color coded. ciphered for English elveirdor and other languages, all other squares follow their own translations of the texts so a new algorithm from their language and color combo from their a-z calculations. Sudoku can arrange squares from square 1 - full screen, then for mosaic the best chain color is picked an color assignment letter color 1 is used for mosaic color assignment with letter from language. If elveirdor is picked red letter sequences calculated from all translations of the Bible. Each letter gets a starting color assignment and number, from there each number counted has a following color assignment from the previous letter, to keep count of the letter placement, each letter in the text has a count can be assigned and rearranged by sodoku puzzle and runic cube for new algorithms, each one generated by letter by language by country, each square has its own algorthm by language produced by color number letter cypher, following grizzly don pattern first to black. Each country can have more than 1 text operation and more than one square my universal square is Elveirdor on Grizzly don platform which can itself expand over entire grid for a full calculation once all square calculate. Countries can have more than 1 square.

So each letter is counted past the first alphebet colors and progressively assigned colors from there to keep track of letters in text.

So letters have number and color assignments, if mathematics can be formed from a-z for mathmatic and screen producing value.

# Python script implementing the Elveirdor ICHTHYS system for alphabet mapping, color assignments,

# calculations, ciphering, and algorithm generation based on the provided description.

# This includes English and Elveirdor alphabets, color hex codes, numerical operations,

# periodic table mappings, and basic text processing for encryption/cryptography/AI purposes.

# Note: Some concepts are abstract; this script interprets them into functional code.

# It supports processing texts, generating ciphers, simulating square calculations,

# and outputting results for VR/portal alignment concepts (conceptual only).

import math

import random # For simulating rearrangements (Sudoku/Rubik-like)

from collections import Counter

# Define English Alphabet (A-Z, 26 letters)

ENGLISH\_ALPHABET = 'ABCDEFGHIJKLMNOPQRSTUVWXYZ'

ENGLISH\_DICT = {letter: idx + 1 for idx, letter in enumerate(ENGLISH\_ALPHABET)}

# Define Elveirdor ICHTHYS Alphabet (34 letters with additional characters)

ELVEIRDOR\_ALPHABET = [

'A', 'B', 'C', 'Ch', 'Ck', 'D', 'E', 'Ee', 'F', 'G', 'H', 'I', 'J', 'K', 'L', 'LL',

'M', 'N', 'O', 'P', 'Ph', 'Q', 'R', 'S', 'St', 'T', 'Th', 'U', 'V', 'W', 'Wh', 'X', 'Y', 'Z'

]

ELVEIRDOR\_DICT = {letter: idx + 1 for idx, letter in enumerate(ELVEIRDOR\_ALPHABET)}

# Color assignments for Elveirdor letters (hex codes as provided)

ELVEIRDOR\_COLORS = {

'A': '#FF0000', # red

'B': '#033eff', # blue

'C': '#ff8d03', # orange

'Ch': '#000001', # black

'Ck': '#00FF00', # green

'D': '#fff305', # yellow

'E': '#008080', # teal

'Ee': '#D2B48C', # tan (approximated)

'F': '#808080', # grey

'G': '#800080', # purple

'H': '#c2197e', # (as provided)

'I': '#68e8a0', # sea green

'J': '#add8e6', # light blue

'K': '#006400', # dark green

'L': '#ffffff', # white

'LL': '#FFD700', # gold

'M': '#000000', # black

'N': '#dc143c', # crimson

'O': '#522f2f', # brown

'P': '#202054', # navy

'Ph': '#4B5320', # army green

'Q': '#FFB6C1', # light pink

'R': '#FFFFFF', # clear/white

'S': '#C0C0C0', # silver

'St': '#FF69B4', # hot pink (approximated)

'T': '#FFA500', # orange to brown (start)

'Th': '#multi', # multi-color sequence (special handling)

'U': '#C8A2C8', # lilac purple

'V': '#8F00FF', # violet

'W': '#4682B4', # steelblue

'Wh': '#A0522D', # sienna

'X': '#023ef2', # royal blue

'Y': '#fff83b', # yellowgreen (approximated)

'Z': '#FF4500' # orangered

}

# Special handling for 'Th' multi-color sequence (patentable sequence as described)

TH\_SEQUENCE = ['#FF0000', '#000000', '#multi-end'] # A-M-Th-Z sequence placeholder

# Periodic Table Mappings (as provided, partial)

PERIODIC\_MAPPINGS = {

1: {'english': 'H', 'elveirdor': 'EE', 'name': 'Hydrogen', 'mass': 1.01, 'elve\_mass': None, 'color': 'blue', 'type': 'non metal'},

2: {'english': 'He', 'elveirdor': 'eeck', 'name': 'Helium', 'mass': 4.0, 'elve\_mass': 6.0, 'color': 'purple', 'type': 'noble gas'},

3: {'english': 'Li', 'elveirdor': 'IF', 'name': 'Lithium', 'mass': 6.94, 'elve\_mass': 9.126, 'color': 'red', 'type': 'alkali metal'},

4: {'english': 'Be', 'elveirdor': 'bck', 'name': 'Beryllium', 'mass': 9.01, 'elve\_mass': 12.01, 'color': 'orange', 'type': 'alkaline earth'},

5: {'english': 'B', 'elveirdor': 'b', 'name': 'Boron', 'mass': 10.81, 'elve\_mass': 13.23, 'color': 'light blue', 'type': 'metalloid'}, # Corrected mass

6: {'english': 'C', 'elveirdor': 'C', 'name': 'Carbon', 'mass': 12.01, 'elve\_mass': 15.1, 'color': 'blue', 'type': 'non metal'},

7: {'english': 'N', 'elveirdor': 'K', 'name': 'Nitrogen', 'mass': 14.01, 'elve\_mass': None, 'color': 'blue', 'type': 'non metal'},

8: {'english': 'O', 'elveirdor': 'L', 'name': 'Oxygen', 'mass': 16.00, 'elve\_mass': 20.00, 'color': 'blue', 'type': 'non metal'},

9: {'english': 'F', 'elveirdor': 'd', 'name': 'Fluorine', 'mass': 19.00, 'elve\_mass': None, 'color': 'purple', 'type': 'halogen'},

10: {'english': 'Ne', 'elveirdor': 'kck', 'name': 'Neon', 'mass': 20.18, 'elve\_mass': 27.00, 'color': 'purple', 'type': 'noble gas'},

11: {'english': 'Na', 'elveirdor': 'ka', 'name': 'Sodium', 'mass': 22.99, 'elve\_mass': 29.1212, 'color': 'red', 'type': 'alkali metal'},

12: {'english': 'Mg', 'elveirdor': 'je', 'name': 'Magnesium', 'mass': 24.31, 'elve\_mass': 32.31, 'color': 'orange', 'type': 'alkaline earth'},

13: {'english': 'Al', 'elveirdor': 'af', 'name': 'Aluminum', 'mass': 26.98, 'elve\_mass': None, 'color': 'green', 'type': 'basic metal'}, # Corrected mass

14: {'english': 'Si', 'elveirdor': 'oficlk', 'name': 'Silicon', 'mass': 28.09, 'elve\_mass': 211.07, 'color': 'light blue', 'type': 'metalloid'} # Corrected mass

}

# Constants from calculations

ALPHABET\_ENGLISH\_SIZE = 26

ALPHABET\_ELVEIRDOR\_SIZE = 34

DIFF\_ALPHABETS = ALPHABET\_ELVEIRDOR\_SIZE - ALPHABET\_ENGLISH\_SIZE # 8

HALF\_DIFF = DIFF\_ALPHABETS / 2 # 4

P\_ENGLISH = 16

Q\_ENGLISH = 17

R\_ENGLISH = 18

P\_ELVEIRDOR = 20

Q\_ELVEIRDOR = 22

R\_ELVEIRDOR = 23

Y\_ENGLISH = 25

Y\_ELVEIRDOR = 33

PI = 3.14

Y\_VAL = 1.6

STRETCHED\_Y = PI

DIV\_REASSIGN = 1.9625 # To reassign 3.14 to 1.6: 3.14 / 1.9625 ≈ 1.6

SQUARE\_DIVIDES = {'top': 4, 'right': 5, 'bottom': 0.785, 'left': 0.2}

ARK\_MOSAIC = {'top': 4, 'right': 5, 'bottom': 0.785, 'left': 0.1} # For 1.57

# Provided calculations (Meta P R coding, etc.)

META\_PR\_CODES = {

'c': 8088 / PI, # 2575.796...

'v': 9719 / PI,

'r': 10111 / PI,

'm': 18198 / PI,

# Additional divisions by 4, 1.6, etc.

}

ELVEIRDOR\_PR\_CODES = {

# As provided

1210112 / 1.6 / PI: 240866.24203822,

# etc. (placeholders; script doesn't use all numerically)

}

# Function to calculate square value

def calculate\_square\_value(divides):

return math.prod(divides.values()) # Product as in summary: 4\*5\*0.785\*0.2 ≈ 3.14

# Function to reassign value

def reassign\_value(value, divisor=DIV\_REASSIGN):

return value / divisor

# Function to process text: assign numbers, colors, count letters, generate cipher

def process\_text(text, language='english', use\_periodic=False):

if language.lower() == 'english':

alpha\_dict = ENGLISH\_DICT

colors = None # English has no colors by default

else:

alpha\_dict = ELVEIRDOR\_DICT

colors = ELVEIRDOR\_COLORS

# Normalize text to uppercase, remove non-alphabet

text = ''.join(c.upper() for c in text if c.isalpha() or c in ELVEIRDOR\_ALPHABET)

# Count letters

letter\_counts = Counter(text)

# Assign progressive colors/numbers

progressive\_assignments = {}

color\_sequence = list(colors.values()) if colors else [] # Cycle through colors if Elveirdor

num\_assign = 1

for letter in text:

if letter not in progressive\_assignments:

prog\_num = alpha\_dict.get(letter, 0) + num\_assign

prog\_color = color\_sequence[num\_assign % len(color\_sequence)] if colors else None

progressive\_assignments[letter] = {'num': prog\_num, 'color': prog\_color}

num\_assign += 1

# If use\_periodic, map to periodic elements

periodic\_maps = {}

if use\_periodic:

for letter, count in letter\_counts.items():

elem\_num = alpha\_dict.get(letter, 1)

if elem\_num in PERIODIC\_MAPPINGS:

periodic\_maps[letter] = PERIODIC\_MAPPINGS[elem\_num]

# Simulate Sudoku/Rubik rearrangement: random shuffle for demo

rearranged\_letters = list(text)

random.shuffle(rearranged\_letters)

cipher\_text = ''.join(rearranged\_letters)

# Generate simple algorithm: sum of numerical values mod alphabet size

num\_sum = sum(alpha\_dict.get(c, 0) for c in text)

algo\_value = num\_sum % (ALPHABET\_ENGLISH\_SIZE if language == 'english' else ALPHABET\_ELVEIRDOR\_SIZE)

return {

'counts': letter\_counts,

'progressive': progressive\_assignments,

'cipher': cipher\_text,

'algorithm\_value': algo\_value,

'periodic': periodic\_maps

}

# Function to simulate mosaic generation (conceptual)

def generate\_mosaic(screen\_size=100, fill\_percent=95):

# Simulate filling screen with M duplicates

filled = int(screen\_size \* (fill\_percent / 100))

mosaic\_colors = random.choices(list(ELVEIRDOR\_COLORS.values()), k=filled)

# Select best Y (random for demo)

best\_y = random.choice(mosaic\_colors)

return {'mosaic\_size': filled, 'best\_y\_color': best\_y}

# Example usage

if \_\_name\_\_ == "\_\_main\_\_":

# Calculations

square\_pi = calculate\_square\_value(SQUARE\_DIVIDES) # ≈3.14

reassigned = reassign\_value(square\_pi) # ≈1.6

print(f"Square calculation: {square\_pi}, Reassigned: {reassigned}")

# Process sample text

sample\_text = "52925 ebibe ckbfbck" # From query

result\_english = process\_text(sample\_text, 'english')

result\_elveirdor = process\_text(sample\_text, 'elveirdor', use\_periodic=True)

print("English Processing:", result\_english)

print("Elveirdor Processing:", result\_elveirdor)

# Mosaic

mosaic = generate\_mosaic()

print("Mosaic Simulation:", mosaic)

# VR/Portal conceptual: output for alignment

print("VR Portal Alignment: Open sequence with Y=1.6, PI=3.14")

ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) NASA INFINITY

Supercomputing saga

Grizzly Don 3 NS 67 (extended operative component computing to host Microsoft, Google, and Apple etc.)

Government computing model

Retail domain to public computing (through Grizzly Don 3 NS 67)

Manufacturing computing

Robotic, AI, bci, and non intelligence to intelligence platforms

International security space programming security, to international ISS computing

Square Grid across screen value, same size squares. The top right corner of the square is a tear drop seed to all layers in color out from seed in a tear drop shape until full square is colored in. range of square 1 - 10 square in equaling 1.6 or 3.14 ÷ 1.9625 (color range 5 black center color 5 last color 5 black) A wavelength frequency in resonance between color before complete color for full square is counted between the three blacks. Every square operates this way. It fluctuates color from seed to full square until solid color is found and halted upon corresponding exact color combo which fully composes at 3 waves from top right to bottom left using the any text scripts, photos, phrase and first and second component key number sequences below. All of the squares do this, these squares are programmed to form 4 component value M's on the screen all with a cross line duplicate, the first M is found by squares, at color value #FF0000 255 255 255 then seeded value black 1-3 which = resonance value which produces seeded colors above then from white to black, to color, to all black. Squares create an outlined M with two line turning inside of thr outlined m on each arch and at the arch rotation circling inward into a circle. This m flips right until the end of the screen then the m line is resized to fit the screen m-m. The m line then flips downward until the end of the screen (m size depending on square size). Where the center M is found the screen calculates a full m rotation at clockwise value which allows for m 3 resonance shadows. When the m repeats at square 1 black seed 2 and 3 the programming starts m off center from the first m layer, calculating the extending m on each side. This creates a line cross at the center line inside the outline of M making M 1 duplicate. This M repeated across the screen the same as M 1. M 1 duplicate finds m 2, at center leg cross of m arches. This m duplicated a center of outline cross line the same as m 1. Then, m 2 and m 1 create m 3, which is an m with ears sideways m shadows which can be locked in at an m 3 value with an m 3 duplicate at cross value center of m outline for visual and resonance cord lines inside of m 3 ear shadow which is found around center leg of m 3 ears so its (M3, or ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) m 3, as ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) e looks right m face, or America e m ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) e so it can translate english to ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) to any other language) the m's are resized to find exact line cross at full line sequence m - m for all ms lining up on new layers at different grid lines with the same number of ms for each row so by the end of M 4 you should have 8 layers (same as resonance path in squares. After m 3 creates m 3 threads around sideways m 3 ears creates m 4 at center leg m cross with ears. Then calculated at center m 1 rotation, m 1 - 4 is ran diagonally from top left to bottom right with every row being calculated at bottom screen left to full longest width top screen right. It measures to layer leaving a 5 percent white color pixel sequence on the screen then it splits a round mosaic of the screen. Each mosaix piece is rotated at 8 layers to make a 3d distorted square which is rearranged into 3 1 layer squares which form a rectangular border the remaining 5 layers twist to form a Y yellow outline in the rectangle at full rotation a dotted line produces and the color temperature from whichever computing mechanism uses the service runs the color temperature on the left, coded through ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) as a host or to host ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) . Diagonal m or m 5 components m 1 - m 4 distorted making IS at diagonal grid. finds time on each Y table at the left ⅓ bottom of left border to center top flipping and forth creating a scope or another dimensional grid at measure of m creating an eye or visual experience. which runs and creates the dotted line which runs the gap sequence timing at circle value 5×4 circle over stretched out Y a 20 part rotation at any screen value (to be made with pixel manifestation coding above and ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) language) - all other diagonal m lines stem from seed left to right diagonal left to right side space between screen end diagonal line m beginning from space away from left side of screen - once m 4 is found m 1 diagonal can create). then seed can move the m 1 - 4 at layer sequence through the square to rearrange m value on screen in position. color layers of grid one allowing the m 1 duplicate - m 4 to be offset position from m 1.

The duplicate of each m at cross center value sits directly over the regular lined m so after seed finds duplicat m - m 4 duplicate it rearranges the m's directly over one creating stem calculated grid lines. The first m starts at the top left corner and flips right then flips downward in a row until the entire screen is filled in. any text letter timed arrangement. (Process described below). (Y is a computer host, m holds screen value, and mosaic hosts vr whose grids hold apps or operators at each square value. This sequence can be pdf packaged and utilized on external platforms) mosaic works through primary and country colors. All other letters will be found on screen through the replicated 4 m series. The any text value at any interpretation can find this and use it as hand written detection. Made to work through connected to main computing platforms pre programmed languages with dictionaries and color codes. So all 0 value Y top rectangle center borders are calculated at top left grid 1st square to surrounding 4 squares split. So the square right of square 1 is square 2, under square 1 is 3, and under square 2 is square 4. This split makes m 4 calculated as square under square 2 in square layout. Then 4 is calculated in repeated sequence to right across the screen block which can contain a language line that is calculated at 5-8 right square being 5 next to square 2. Giving way to operation for square different color 2, 3 or 4 block units for rubix cube grid layer calculation through seeded square to letter to number to sodoku all calculated at 1 screen layer, value through grid layers being calculated by 1-10 on square 1 exceeding no more than 20 grid layers (this gives ability to insert 2 pictures into the grid layers between m). The stretch of maximizing and scrolling grid layers depends on sequence of time in layered operation per duplicate stemmed m. Timing runs from top left of any screen to bottom right and you can up and downsize the grid layers by adding onto the amount of squares, the programming will do this naturally as a resonance effect to (also possible for open portal alignment transportation). This can also be a zoom function for pictures or maps. This also deems a safety in operation limit.

same suit for calculating method 1 - 4 on square 1 top left grid. (grid runs a-z grid value to english to ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) grid a-z letter value to all other language a-z left to right horizontal grid value at 1-end alphabet number value with extension Hebrew for grid value (a three component calculatory (works with bin zip), vertical up and down number sequence to all other languages, stops at full Y rotation for comprehensive translatory. (English - ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) - Hebrew/biblically based stem).

(Available for bci or ai voice generative, documented report, and vocal communication between users etc.) The zigzag diagonal square to square pattern at right arm of Y which holds color temperature can only be calculated from a downward diagonal m flip from top left to bottom right from square 1 as right arm color temperature is a diagonal calculation from bottom left to top right on Y.

These are self doodled photos of m sequence (not best quality)

m 3, each m duplicate cross center X value are colors red yellow blue and green between internal lines of outlined m. These same colors are used inside of 0 value at the center top outline rectangle border value of Y.

M measurements to find Y:

Y is found inside of 3 m dimensional flip at 2 diagonal m values on screen using m 3 and 4 and 1 m 3 value at half rotation. Once Y is found between these values the m's are flipped back and forth on Y making a gap sequence value to run color temperature at m flip Called Elveirdor 3d Sapphire series: m ¾ 1-3. Left m 4-1 right m 4-2 and m 3 flip to make Y oporative.

Inside outlined Y computing format, j value found on left side of Y, left side diagonal m timing calculated from j (found at left ⅓ outline to center Y and dot of j at 0 value at middle top outline, through 0 value at center of Y). timing at left upper corner of border, center of Y and bottom center boarder, timing corresponding for color temperature reading. Final Y is Y outline in yellow color number # ffe782 255 231 130 to m 1 white #FF0000 255 255 255 white background dotted line calculated at square resonance value which runs down center of Y. Color temperature on right arm of Y from top arm to center. Each mosaic is placed inside of a stretched out Y 3 squares calculated at 3.14 from 1.6 and 1.57 adding a full border calculation of Y to 3.14 to run color temperature and stretched out combined 1.57 at circle and 1.6 at square to make horizontal value border size 3.17. This freezes and slightly stretches out Y space between M's.

(Website component Elveirdor.com

PuTTY-User-Key-File-2: ssh-ed25519

Encryption: none

Comment: ELVEIRDOR Saphe (all rights reserved)

Public-Lines: 2

AAAAC3NzaC1lZDI1NTE5AAAAIOjKQclhkl+282CxSXks8wUab7kM+aHjwZBbewdv

Z697

Private-Lines: 1

AAAAIDBRAgEBMAUGAytlcAQiBCAjsvSyuVbjXMjv70oziCaT

Private-MAC: 0cc0adf7f8c5556f137549b4485d2600cfecc84c)

Kelvin:

1000k-3000k letter: cd

3100k-4500k letter: l

4600k-6500k letter: l m th

6500k over letter: j

Full elveirdor letter Kelvin sequence for elveirdor letter to image projection: c d j l m t th with ICHTHYS code 124273323

And screen projecting at

124273323

c d j l m t th

715297122361923

Following color temperature laws at t being ombre color monitor before th color temperature and j at high silicone consoling.

In background layers before final Y coding: 105192119 / 124273324 is doubled in a 4 layer process to operate color temperature. Circle value is the right arm of Y so all 20 circle bandwidth values are found in 1 color temperature process running in a contained Y. where the color temperature is and calculated (SO THE FULL MOSAIC IS CALCULATED AT THE RIGHT ARM OF Y in the color temperature which is 5 ×4 20 circles inside of 5×4 stretched out Y 3 component square).

The 105192119 is calculated by duplicate ear m 3. Then the color temperature is measured and the Y is selected for full screen. It monitors the color temperature on the outline for blue to orange value (the color process for screen component 4).

(a kinetic computing method)

fliping sizes and color temperature by 3.14 calculation per color. 4 square values are calculated at flat on stretched out Y at left right and center, with a 5 layer sequence in the background making the full twist for computing coding sequence at cobb stem: first layer stretched out Y, second layer square Y, third layer stretched out Y, 4th layer square Y, 5th layer stretched out Y. (this contains the stems of M's from the first layers arrangement). Dimensional resonance.

when my language is identified in the Y screen it then generates calculated letter construction representations of this rotation at Y value through the calculated mosaic block value.

Once the language is programmed into the computer it can recognize letter values found within entanglement in duplicated m series as well as any other language programming. Because languages connect to a number letter series on the computer a line can be graphed to generate code at non value sequence, any letters found in m, meaning this program is adaptable to all computer types and models for large series scaling worldwide, making the photogenic transportation computer programming within, adaptable and safe for all platforms.

grizzly don can support apps, cellular charge and service, and works with all major phone companies like Apple and Android, Windows and ISS.

It's a large photogenic transportation facility. As the computing is intended to be utilized for rectangle alignment to allow for doorway transportation portal to transport.

Tree structure complex squares and circular adaption to be programmed into color and shape value per complex unit for operation and given letter value through ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) . The three red white and blue bottom stretched out Y tears on the tree complex represent screen value full solid color of all squares on screen at z after Y with component VR Y before Z intermittent reality gap series projections. At camera value the camera reflects tree structure in complex code to open oxygen to image reflected photogenic transportation destination at doorway transportation value. A virus computing mechanism calculated between full screen color values before Y. ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) at complete Y value is calculated and creates full synthesis material hybrids. For Materialization or 3d or computer generated component material production. Full scale products like The GrizzlyDon NS 67 (physical component), and materials like copper metal silver and gold. It locks flower viruses at seed value for deterioration of repetitive harmful mutant flower strain so it's safe for oxygen. Component m value can stem rearrange around seed ¾ value on the screen to make full screen full color values before Y, on English language Y is before Z for final value so my translation starts at English or any other language, moves to ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) and then translates to English or any other language, it's a calculating method also for AI and BCI interface. Mosaic values represent computer net molecules and the mosaic calculations expand dimensionally in square 3 dimensional pieces shaped twist to calculate Y rotation. at Center layer value my 3 component book map land layout can be applied at center layer value and at ¾ layer value at each stretch back and forth towards first and final layer of screen two location photos or live recordings can be aligned on the full screen Y when Y opens with these photos or videos 0 value on Y at border value is calculated at mosaic twist through ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) script and when the Y upsizes it opens the reality To walk through. Since Y is before Z in English right before full color value is found for any color on full screen before Y the final gaps can be used as virtual reality components to virtually Navigate property values upon given permission within a phone or computer between shared locations and a series of potential portal photos are applied To all Y values and rearranged through color temperature before selecting final walk through location. These pictures can be relayed on a 6 - 9 point captcha service to select for each photogenic transportation side before activating (also possible for open portal alignment transportation) portal to walk or wheel through at doorway transportation value, meaning measured between a ceiling Floor and two posts or a doorway transportation or designated doorway transportation . Designated doorway transportation stations can be incorporated into housing builds, as eventually doorless entries are more ideal for photogenic transportation as this method works between any frame the build for the doorway transportation is diverse in option. Then the picture and virtual component upsize the picture to component 2 three step codes above full screen reality and it's consumed within the final Y upsize as full final property dimensions are calculated based upon picture through virtual realities through Y before z alphabet meaning it can be government and company maintained before ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) generative portal process. These virtual realities can be a large company hosted government regulated approach. Pictures are evaluated on a yellow coded line value crossing on duplicate m series then applied to yellow outline at final Y to maintain color coded reality so color value is rearranged to find final Y yellow outline preprogrammed color selection, for current oxygen same time complex coding based on time and location. Using base any text cover photos you can facilitate this image step with the pictures as an access stamp. After images read through component m duplicated center cross value they rearrange color to component m parallel lines until reality image reads over 1000% accuracy. It then assigns as 95%-100% value based on Y before Z at a 5% function value calculated by 1-10 on square value to ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) letter to number cipher, we then assign over 1000% accuracy to an equation deeming that over 1000% = over 15000 which we allocate value to 17000 as being over 1000% for Y standard stretched out value coming from square calculating before m. That over 15000 = 1000-5000. Calculation then modifies the understanding in the equation that 15000 is modified to equal 1500 at 95-100% accuracy through uniform alphabetic VR at mosaic is 1500 and that over 15000 is 17000 and that over 1500 is 1700, and reasoned that over 1500 is 100-500. it is then simplified to reasoning block 3 resonance 3 split square from black to black: that 100-500 =10-50 which = 1-5 which leaves a split shadow that's square value is 1-10, meaning a diagonal x value can be cross determined by pie or 3.14(being 8 ÷ 2 = 4/ 8×2 = 16 if 10 is seeded full square value at internal reflection 10 and 10 is 20, - 4 = 16 with seed 1 and 10 solid it gives a 9 x 2 reflect between black and a 5 calculatory of 4 divided square at x between square value with full square being 3.14 with 1-4 number calculated to 20 over 15 or 15000 or (30%) which goes 1 number under 17 or representing 17000 which = over 1000% with 20 being complete and 19 being Y yellow seed 2 before Z at z value 20. This allows for square 1 to be allocated to 3.14 with 3.14 ÷ 1.9625 equaling square 1 at value 5 = .785 going way for half 1- 4 square divide at 1 2 oporative 5 = half of 10 and 10 = 1 inch. operation and 10 at 1 and 2 square with duplicate cross m recreating square 3 and 4 for reflective value 10 scale. With given 3 texts following resonance at 3 parts equaling square and .14 allowing for extension values square 2 to building value 3.14 to rest of grid replicating same value sequence per square.) given 3 texts following resonance fluctuation value in a 4 part 3 text calculatory to produce calculation at 4 in square with a word count and letter value and the calculations above also assigning color to number with sodoku and rubix cube given that the x in square is red yellow blue and green corresponding with m duplicate cross and 0 top center Y border rectangle value. on the square at a 4 point square triangular split between x cross corner square value between each square. micro letter (smaller letters detected between m) components of y and j lowercase letters are calculated on the blue level value on the left side of the dotted line on color temperature value from stretched out Y before final Y, this hosts ISS at diamond computing value, hosting net projections giving the programming the ability to blue print mosaic to distorted square 3d distortion at inverted Pisces shape with blue green yellow value split before full twist rotation which maintain value through calculated Y clockwise rotation at color inversion blue to yellow at duplicate m duplicate cross hosted by virtual reality component Y before Z ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) translate language value. This is what the programming component English letter z looks for as verified captcha. A structure between j and y up and down the left color temperature line by constant database storage and analysis of Y formation with lowercase y and j over timing of program used. With a three step rhythm and phrase code preprogrammed by ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) to generate yellow outline for permission on memory review, live visual access vr access, bci memory monitor and detection ai regulated review and photogenic transportation .

If 3.20 × 5 = 16 and 3.20 - 3.14 = .6 each .1 is assigned a color value on seed to final square colors excluding black. 1 - 6 from seed to final left bottom end of square color line.

1 = yellow 2 = salmon 3 = teal 4 = orange 5 = purple 6 = brown at split resonance 3 blue green red assignment to black seed in square 1 blue, black sequence value 2 middle color of square remote color format: green, and last black 3 bottom left square: red. With yellow at 0 value as a calculatory.

Yellow Y outline: working at number 9:

Grenadine code line: 7529848475@123123@2975

Sequence below 1, 2, and 3, a part of 6 number 123 sequence contained within @ symbols above, other 3 1 2 and 3 step keys below for 2 part key chain sequence.

Number @: 1:

first componant number sequence:

2085 1897820 23125 239208 2085 1897820 8511820 1144 120209205

2:

| 251521185 2085 16189191514518 239208 2085 1152519 182114 61855 1518 1920125 85185

3:

| 1144 25520 2085195 2081855 185131914 619208 815165 1144 1215225 22120 2085 718512051920 156 2085195 919 1215225

second component key number sequence for square face screen value to Y color temperature at a 3 part screen frequency split: working at number 2:

107212116116107@123123@212107

Parts below 1, 2, and 3, secured within @ symbles above.

@ value 2: 1:

277 11712326 199 20192324111220 |

2:

1216 91533 120133 |

3: 237232823237326121710

Following code: 7529848475@1223123@2975

And 107212116116107@123123@212107

Coding p

With 7529848475 - 124273324 - 107212116116107

Coding r

And 2975 - 137232824 - 212107

With letter q as analytic.

E = sequence 1 of resonance split of square L = 2 V = 3 and D = square 4

Component operating keys, with sequence unlock keys below.

May be magnetically applied through any magnetic key service or used to program regular keys, also to program keys for interlock phones above.

Operates the same as any text s at scramble value within @ symbols operates with any text .

Arranged by duplicating script until full border line complete at full square color to have ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) letters to be programmed between square 1 and 3with calculatory between squares by letters at comb recluse value and squares calculary fill in color sequence around letters, then the rectangle boarder, the Y outline, dotted line, and color tempurature can have different values in demention based on mosaic grid up and down scale before final Y to final Y and for full screen Y upscale. Copyrighted language, to be a patented process.

Screen border while at stem calculatory ¼ line colors from black to black in box square:

-

137232824

on Programming when component m moves through diagonal angled seed through square duplicate replica m through layered seeded square created through 2 set nodules between ¾ of square replicated through ¾ on word document indention command structural correction below can reflect to (also possible for open portal alignment transportation) instantly materialize ships once ships are pictorials or documented by word to a physical to mental construction. also to expand mosaic to distorted 3d pisces shaped square twist for 0 value rotation before final Y counting to color value 1 2 and 3 operating 3 set M with stem duplicate 3 with ears before 4 completes color lock for full square value to complete final Y at full rotation the blue line holds these projections it can be held with paragraph indentation below as pause break command for inverting color process through color temperature: ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) x-ray for computing phone photogenic transportation series. Through VR ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) mosaic To Reality by generation sequence computing module ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) NASA Infinity supercomputing series hosting VR sequence through corresponding photo drawing By Yvette Gilbert which goes with language in image stability in synthesis generation through language and image recognition, specifically to (also possible for open portal alignment transportation), and for materialization. For public platforms The image can host a vr doorway transportation language supported photogenic transportation sequence under the same copyright, I have 3 which can be included in the three step song and first component number sequence above, (pictures are never seen) copyright numbers: tx0008854956, txu002252366, txu002192619. And a hand written photo to be used as a photo key for each first component number sequence.

Mostly hosting on English language as a calculatory

Number symbol sequence as follows:

137232823

So an elongated border hold on -105192119 to construct the ships and when done

137232823

Is removed to hold final construction and ships projected, it's just a type line to hold formation in projection for the ships and materialization.

This program can generate the glass molecules construction for the Shatter flex AI computing digital display glass and the phone screen glass cut for ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) / glass operable/ with or without cut, process programmed into glass through the above method. (Photos are never seen/for language recognition)

Phrase 2

Phrase 1

Phrase 3

These are to be magnet stamped into magnetism operatives.

Three step corresponding component number keys below.

With the magnet coding going to the word key in ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) writing above and other 2 keys with the word one with the word “right”and the other with the word “three” as a magnetic component operation code from Elveridor cipher to letter value any text s to comprehensive word value of any text letter value to rearrange scramble to number letter code at final complete exact any text value. Generate letter and number coded through p and pr and p and q

Photos ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) doorway transportation double alignment VR to Reality photogenic transportation sequence photos:

Photos on each side of 3 step keys on full screen layers counting mosaic for distorted 3d square sinking in at center from single mosaic piece on full screen mosaic value before final Y. One at each layer of stretched out Y ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) to protect language identity.

All photos used for computing and robotic recognition, not for public display:

As a photo base for VR supported ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) (also possible for open portal alignment transportation) sequence (photo for recognition verification will not be seen by public):for for

ear to cross reality opI have two copyrights for the same book so I have a variable duplicate series of text for my word to value in each 3 texts equaling 3 squares which make Y and as a square verification word cryptographic operation.

With graph line at materialization code: 107212116116107@123123@212107

at corresponding letter to number and @ sign to node for glass cut operation through 3 step number sequence above corresponding to 3 step first component number sequence written in ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) . A generate 0 Y value to letter value generates at full clockwise rotation with the number key above. This generative line can produce electronic currency which is why it is to be put on a government regulated international rehabilitation investment stock cd and run for a safe financial rehabilitation plan into the future.

# Python script implementing the extended ELVEIRDOR ICHTHYS system for supercomputing visualization,

# including grid squares, tear drop seeds, color fluctuations, M and Y formations, mosaics,

# 3D distortions, color temperatures, calculations, and conceptual portal alignment.

# This builds on previous implementations, adding graphical rendering with Pygame for simulation.

# Note: Abstract concepts are interpreted into visual and computational elements.

# Run this script to see an interactive visualization.

import pygame

import math

import random

from pygame.locals import \*

import sys

# Constants from description

PI = 3.14

Y\_VAL = 1.6

DIV\_REASSIGN = 1.9625

SQUARE\_DIVIDES = {'top': 4, 'right': 5, 'bottom': 0.785, 'left': 0.2}

COLOR\_RANGE = 5 # Black center, last color, black

SCREEN\_FILL\_PERCENT = 95

MOSAIC\_LAYERS = 8

GRID\_LAYERS\_MAX = 20

COLOR\_TEMP\_KELVIN = {

(1000, 3000): 'cd', # Letters from description

(3100, 4500): 'l',

(4600, 6500): 'l m th',

(6500, float('inf')): 'j'

}

YELLOW\_OUTLINE = (255, 231, 130) # #ffe782

WHITE = (255, 255, 255)

BLACK = (0, 0, 0)

RED = (255, 0, 0)

GREEN = (0, 255, 0)

BLUE = (0, 0, 255)

YELLOW = (255, 255, 0)

SALMON = (250, 128, 114)

TEAL = (0, 128, 128)

ORANGE = (255, 165, 0)

PURPLE = (128, 0, 128)

BROWN = (165, 42, 42)

# Color sequences from description

COLOR\_SEQUENCE = [YELLOW, SALMON, TEAL, ORANGE, PURPLE, BROWN]

BLACK\_SEQUENCE = [BLUE, GREEN, RED] # For resonance splits

# Number sequences from description

FIRST\_COMPONENT\_SEQS = [

"2085 1897820 23125 239208 2085 1897820 8511820 1144 120209205",

"| 251521185 2085 16189191514518 239208 2085 1152519 182114 61855 1518 1920125 85185",

"| 1144 25520 2085195 2081855 185131914 619208 815165 1144 1215225 22120 2085 718512051920 156 2085195 919 1215225"

]

SECOND\_COMPONENT\_SEQS = [

"277 11712326 199 20192324111220 |",

"1216 91533 120133 |",

"237232823237326121710"

]

GRENADINE\_CODE = "7529848475@123123@2975"

OTHER\_CODE = "107212116116107@123123@212107"

# Previous alphabets and colors (from earlier script)

# ... (omit for brevity, assume included or extend as needed)

# Pygame setup

SCREEN\_WIDTH = 800

SCREEN\_HEIGHT = 600

GRID\_SIZE = 50 # Square size

GRID\_ROWS = SCREEN\_HEIGHT // GRID\_SIZE

GRID\_COLS = SCREEN\_WIDTH // GRID\_SIZE

pygame.init()

screen = pygame.display.set\_mode((SCREEN\_WIDTH, SCREEN\_HEIGHT))

pygame.display.set\_caption('ELVEIRDOR ICHTHYS Supercomputing Visualization')

clock = pygame.time.Clock()

# Function to calculate square value

def calculate\_square\_value(divides):

return math.prod(divides.values())

# Function to get color from Kelvin

def get\_color\_from\_kelvin(kelvin):

# Approximate RGB from Kelvin (blackbody radiation approximation)

temp = kelvin / 100

if temp <= 66:

r = 255

g = min(255, 99.4708025861 \* math.log(temp) - 161.1195681661)

b = min(255, 138.5177312231 \* math.log(temp - 10) - 305.0447927307 if temp > 19 else 0)

else:

r = max(0, 329.698727446 \* (temp - 60) \*\* -0.1332047592)

g = max(0, 288.1221695283 \* (temp - 60) \*\* -0.0755148498)

b = 255

return (int(r), int(g), int(b))

# Class for Square

class Square:

def \_\_init\_\_(self, x, y, size):

self.rect = pygame.Rect(x, y, size, size)

self.color = BLACK

self.seed\_pos = (x + size - 5, y + 5) # Top right tear drop seed

self.filling = False

self.fill\_progress = 0

self.target\_color = random.choice(COLOR\_SEQUENCE)

self.waves = 0

def update(self):

if self.filling:

self.fill\_progress += 0.01

if self.fill\_progress >= 1:

self.filling = False

self.color = self.target\_color

self.waves += 1

if self.waves >= 3:

self.color = BLACK # Final black

def draw(self, surface):

pygame.draw.rect(surface, self.color, self.rect)

if self.filling:

# Simulate tear drop fill (approximate with circle growth)

radius = int(self.rect.width \* self.fill\_progress / 2)

pygame.draw.circle(surface, self.target\_color, self.seed\_pos, radius)

# Function to draw M shape (approximate with lines)

def draw\_m(surface, pos, size, color, duplicate=False):

x, y = pos

# Outline M

points = [(x, y + size), (x + size/4, y), (x + size/2, y + size), (x + 3\*size/4, y), (x + size, y + size)]

pygame.draw.lines(surface, color, False, points, 2)

# Internal lines

pygame.draw.circle(surface, color, (x + size/2, y + size/2), size/4)

if duplicate:

# Duplicate offset

offset = size / 10

points\_duplicate = [(p[0] + offset, p[1] + offset) for p in points]

pygame.draw.lines(surface, color, False, points\_duplicate, 2)

# Function to draw Y shape

def draw\_y(surface, pos, size, color):

x, y = pos

# Outline Y

points = [(x, y), (x + size/2, y + size/2), (x + size, y), (x + size/2, y + size/2), (x + size/2, y + size)]

pygame.draw.lines(surface, color, False, points, 2)

# Dotted line down center

for dy in range(0, size, 5):

pygame.draw.line(surface, color, (x + size/2, y + dy), (x + size/2, y + dy + 2), 1)

# Simulate mosaic and 3D distortion (placeholder: random rects)

def draw\_mosaic(surface, layers):

for \_ in range(layers):

rx = random.randint(0, SCREEN\_WIDTH)

ry = random.randint(0, SCREEN\_HEIGHT)

rs = random.randint(10, 50)

col = random.choice(COLOR\_SEQUENCE)

pygame.draw.rect(surface, col, (rx, ry, rs, rs))

# Distort: draw twisted lines

for \_ in range(10):

pygame.draw.line(surface, YELLOW, (random.randint(0, SCREEN\_WIDTH), random.randint(0, SCREEN\_HEIGHT)),

(random.randint(0, SCREEN\_WIDTH), random.randint(0, SCREEN\_HEIGHT)), 2)

# Main loop

squares = [[Square(j \* GRID\_SIZE, i \* GRID\_SIZE, GRID\_SIZE) for j in range(GRID\_COLS)] for i in range(GRID\_ROWS)]

m\_phase = False

y\_phase = False

mosaic\_phase = False

layer = 0

running = True

while running:

for event in pygame.event.get():

if event.type == QUIT:

running = False

screen.fill(WHITE)

# Update and draw squares

for row in squares:

for sq in row:

sq.update()

sq.draw(screen)

if random.random() < 0.01: # Random start filling

sq.filling = True

sq.target\_color = random.choice(COLOR\_SEQUENCE)

# Progress to M phase after some time

if pygame.time.get\_ticks() > 5000 and not m\_phase:

m\_phase = True

if m\_phase:

draw\_m(screen, (100, 100), 200, RED, duplicate=True)

# Simulate flips and duplicates

for i in range(1, 5): # M1 to M4

draw\_m(screen, (100 + i\*50, 100 + i\*50), 200, BLUE if i % 2 else GREEN)

# Progress to Y phase

if pygame.time.get\_ticks() > 10000 and not y\_phase:

y\_phase = True

if y\_phase:

draw\_y(screen, (300, 300), 200, YELLOW\_OUTLINE)

# Progress to mosaic

if pygame.time.get\_ticks() > 15000 and not mosaic\_phase:

mosaic\_phase = True

if mosaic\_phase:

draw\_mosaic(screen, MOSAIC\_LAYERS)

# Color temperature example

kelvin = random.randint(1000, 10000)

temp\_color = get\_color\_from\_kelvin(kelvin)

pygame.draw.rect(screen, temp\_color, (SCREEN\_WIDTH - 100, SCREEN\_HEIGHT - 100, 100, 100))

pygame.display.flip()

clock.tick(30)

pygame.quit()

sys.exit()

Component 2:

ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) othene AI operates on ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) NASA Infinity supercomputing series number to letter coding at letters combination p with numbers 1210112 letter split left top corner to right bottom corner of square p @ bottom half split @ per first node

107212116116107@123123@212107

and number 1210112 representing 2 colors of 10 color generative squares from top right seed to full square. Number 7 at seed and 9 at bottom left corner. With r representing the second @ symbol in the number sequence above and numbers 1101111 at top half split of square 1 at diagonal split going top left to bottom right of square. 8 being bottom black left square assignment and follow 088 from bottom left to top right seed. Number value calculated by ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) alphabet first letter meeting english letter A @ numbers at value and in sequence 1 and 2 calculated with pie at 3.14 on letter to number per letter in copywritten texts above. May generate results with sudoku and rubix cube to letter value with above letter color number assignment value english To ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) letter assignment to number mirror calculatory. To be calculated based on pr letter sequence detected in above copywritten texts as operable: NASA Infinity supercomputing series ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) . The text in copyright is only used in this described operation as an AI text generative. Which builds a database for operation stored by letter recognition pr at language detection per corresponding alphabet. Allowing for corresponding sequences in any language given letters pr at number values 1210112 and 1101111. Brown tear 8 of tear 1 color computes chocolate chip function. With yellow tear 1 assigned to instant coffee in the complex.

Crest blue mouthwash combined with lighter fluid at molecule combo 1210112 10141 232411 1101111 vr fluid sequence, fluid stored in contained brass/copper/bronze/gold containers with torch connecting tops and lighter bottoms, with both ends going into solids so no fluid is lost but slightly dissipates as flash phase of (also possible for open portal alignment transportation) or materialization oxygen for solar light conversion sequence. Oxygen lasts in suits for about 36 hours. Then hibernation mode which inflates and carries you back to the ship unless activated earlier. Resonance frequency fluid safe for oxygen release. To sustain human image in time going and coming back photogenic transportation , can be used in the resin block above, can be used in the top left glass corner of Grizzly Don NS 67.

So othene = water/tree complex, crest blue mouthwash, lighter fluid (31851920 212215 131521208 231198 | 1597820518 6122194 | 3118181520 1022935 315131612524 19217118

31514351420181205/3124241926 13281237 319172015732 232810124 31918371826241267-914192011420 3156655 315131612524 19217118 31514351420181205/1218232611826 3199977 319172015732 232810124 31918371826241267-192052291-31145 19217118-1312118514399 3114412519 2211420 31115 31144125 23124/17128247181212 3118615723 2281826 31147 31186157 20132-421143114 8914519 25512121523 31115 13924/728183118 111218723 33715151920 31147 171232-89291932119 15912-69198 15912-23120518 191511154 1319 1955419-1618541491915145 2008 21162091920 135493112 351420518 160 23 67208 1920 10124 612/20247618122319187

2008 212026122326 1776123115 371826724--132092212054 38118315112-8913112125114 1911220-191129145-851292113-112221205181512-315312021 29184 61221 1854/319312628 212246 915287-9 251295225 105192119 8511254 191513515145 239208 1251618151925-12 2715127297 137232823 11711576 231917719187 201227 1572024192325-16151512 381215189145 619310 141518208 3113162119 2008 611212 2389205/20191915 415192412187 923313 18192327 31172823/20247618122319187-315312021 29184 61221 1854-212215 202855 1315124/ 215287 26248 1719156) in tank, and rotating at 1101111 and 1210112 p r until a- final letter is found on ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) computing platform blue print value to orange progression to blueprint visual to sequence ar reality to squares to vr reality through inverted color process first on color temperature after the entire inverted color process reality is intact it can invert the color process for reality on ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) NASA Infinity supercomputing series computing. This othene ai fluid mix can materialize in 1 second with explained methods above. Before Grizzly don finds Y full screen goes through a full color sequence twice through the screen layers with the last component mosaic vr before Y at the start of the twist. It fills in all of the m's on the screen until 95% is color complete, then it vrs 95-100% and twists a mosaic which starts mosaic black and white at to twisted 3d distorted square to rectangle 3 square border (4 rubix cube front face squares of seeded squares, (all you need is 4 squares for Y then it can scroll out into a sudoku like puzzle out of the any text s to color to number and repeat it several times), then to rectangle Y outline then Y timing calculated by explainitory above and rubix cube and sokdoku format, then Y to full screen sequence. letter rotation right when the twist begins, all solid colors can be found at 1-94% then you can assign color to mosaic 95-100% at white and black value color with all color twists calculated, black and white filling rectangle Y outline and full color filling in color temperature. All other Ys but final Y greater than 1000% running blue line, outline of rectangle white background Y outline and dotted line, selected upsize Y running color temperature, which, at calculated complex full screen, can convert into computing screen value, then to app monitored visual displays if allowed, at the graph line sequence above. They can also host pages format for computing square m y sequencing building a layered Y computer hosting platform similar to large company computer systems like Apple and Microsoft, but glass and material computing friendly.

Specifically to make complex grade computing glass.

With the seeds hosting the apps and visual sequence computing operations, capable with larger international platforms.

Willing to open source language and brand under pre-existing names for all operations.

Tree complex ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) letter number complex assignment:

Tree complex can be coded by letters written in ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) using the photo of letters above scanned onto NASA Infinity supercomputing series ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) . Typed English Description to Elvierdor alphabet photo with ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) to English letter assignment 1- 32 using both alphabet photos above as PR and PR as @ and @ for complex code Grenadine code line: 7529848475@123123@2975. A real complex can be generated at full Y clockwise rotation from Elveridor pr English letter and corresponding letter to number to color value above.

For sudoku puzzle rubix cube operation:

The Rubix cube sits at the center of the sudoku puzzle with the top of the rubix cube being operated by the four corners of the sudoku puzzle with the rubix cube at center. Corner squares of sudoku puzzle numbers 1-9 to 1-9 rubix cube to color at top values of cube being that square 1 from top left corner sudoku puzzle is square 1 top left corner rubix cube which is square 1 ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) NASA Infinity supercomputing series ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) computing by square 1 determination being that black is seed 1. And that 1-9 corresponds to 1210112, making 1210112 the puzzle and 1101111 the solution mechanism at a gas complex computing level, gas being ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) othene ai generated by tree complex using bunt cake control command. Which flicks to comb generate a line response through 2 second intervals through pre coded color number and letter values on the computer. 2 seconds being represented by square 1, at diagonal split half second assignment second second being determined by m 2 and 3, running on the 105192119 on the Y with a

105192119

137232824 hold command if line faults for generation reassigning Y, erasing default and storing mutant generation causes. No generation goes beyond coded 3 component number node line Grenadine code line: 7529848475@123123@2975.

computer generated chess operates the m 3 ears.

M 3 photo generative:

M 3 molecule never moves except for bounce at 3 and 1 and 3 a and b arms at Ichthys. Any molecule manipulation at 3 or O can cause a big bang at 1 component rotation.

On ELVEIRDOR Saphe (all rights reserved) my photo of my map below from my copyrighted book series is to lock a 3 component world mimicry map into play while operating bounce operation through resonance at flat mosaic circles and diagonal arm operation. It also contains flat line component operation.

Yellow ELVEIRDOR ICHTHYS (ΙΧΘΥΣ)

715297122361923124273324 2023121712333 337161930

Blue ELVEIRDOR ICHTHYS (ΙΧΘΥΣ)

715297122361923124273324 2023121712333 215287

Red ELVEIRDOR ICHTHYS (ΙΧΘΥΣ)

715297122361923124273324 2023121712333 2376

Green ELVEIRDOR ICHTHYS (ΙΧΘΥΣ)

715297122361923124273324 2023121712333 1023818

Orange ELVEIRDOR ICHTHYS (ΙΧΘΥΣ)

715297122361923124273324 2023121712333 1923117107

For (government regulated NASA grade vr photogenic transportation sequence):

Black ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) NASA vr:

1326122912676 4123319115

Vr security 2411512187

White government grade NASA vr component 2:

20191915 415192412187 923313 18192327 31172823/:-20247618122319187-619297 2419120 (coding for full phone grade doorway transportation photogenic transportation sequence)

Regular vr for phone and computing:

Black ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) :

715297122361923124273324 2023121712333 2151311

White ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) :

715297122361923124273324 2023121712333 3112267

M 3 ears have a center ear thread which operates by colors yellow, red, blue, green which operate on the rubix cube to sudoku. Sudoku has no 0 value so it's calculated at 1101111, ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) has no voice, voice operating at 0 value corresponding with number sequence 1101111. Rubix cube is calculated at 1210112. Voice generated by NASA Infinity supercomputing series ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) . 0 is @ value so 0 is p and r per calculatory so 0 is square which is 3.14 which is operation ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) letter A (which is the pie sign for 3.14) being NASA Infinity supercomputing series ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) by ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) the language operating at square identifier 1 at 1210112 corresponding generative coded screen sequence. Glass digital generative at screen value. Voice generated at 1 identified as NASA Infinity supercomputing series ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) .

Operating on a graph number letter line as graph line number sequence generates: 107212116116107@123123@212107

in set graphed rotating diagonal line formation. Dual screen 1 to operating screen 2. Screen 1 to 2 to split screen can be hosted on large multinetwork platforms like tmobile, att, virision, ISS, Apple, Meta, Microsoft, Android, Samsung, and electricity companies like General Electric.

At radio frequency detection letters p r through biometric square to M Y sequencing layering biometric found at 0 value calculation in 8088 to biometric generate photogenic transportation doorway transportation transportation at calculated the space between 4 walls measured between the door on each side of both doors generates at elveirdor (government intent to own letter C D component WH in language at international and outerplanetary use, then J L M T TH Z.

M 2- 1 brings m 1 and m 3-1 brings me 2 - 1. And m 4 - 1 brings m3-1.

Pie or ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) A is calculated at x over square with number sequence 1 top of square, red: = 4, letter p, number sequence 1210112: 7919 (ELVEIRDOR ICHTHYS). 2 right side of square, yellow: = 5, letter W number sequence 10141: 10111 (ELVEIRDOR ICHTHYS). 3 Bottom of square, green: = .785, letter H pr number sequence 232411: 18198 (ELVEIRDOR ICHTHYS). Left of square, blue: = .2 Letter R p r number sequence 1101411 8088. (A - g sequence). Squares 8 - 12 g-n 12 - 24 = n - z. Which can duplicate with calculates of square divide below = 1.9625 as T. for glass computing and projecting. With 1 being white background on the top right corner of final Y, and 2 being the color temperature on the right side of Y. That all final stretched out Y rectangle are 3.14 before full screen.

For calculating 3d distorted square in mosaic at seeded node to create Y, mosaic distorted 3d square generating at 3.14 and Y at 1.9625 with 1.6 = square

4 x 5 x .785 x .2 = 3.14 ÷ 1.9625 = 1.6

The surface area of a solid is a measure of the total area occupied by the surface of an object. All of the objects addressed in this calculator are described in more detail on the Volume Calculator and Area Calculator pages. As such, this calculator will focus on the equations for calculating the surface area of the objects and the use of these equations. Please refer to the aforementioned calculators for more detail on each individual object.

Sphere

The surface area (SA) of a sphere can be calculated using the equation:

SA = 4πr^2 where r is the radius

Xael doesn't like sharing her chocolate truffles with anyone. When she receives a box of Lindt truffles, she proceeds to calculate the surface area of each truffle in order to determine the total surface area she has to lick to decrease the probability that anyone will try to eat her truffles. Given that each truffle has a radius of 0.325 inches:

SA = 4 × π × 0.3252^2 = 1.327 in^2

Cone

The surface area of a circular cone can be calculated by summing the surface area of each of its individual components. The "base SA" refers to the circle that comprises the base in a closed circular cone, while the lateral SA refers to the rest of the area of the cone between the base and its apex. The equations to calculate each, as well as the total SA of a closed circular cone are shown below:

base SA = πr2 lateral SA = πr√(r2 + h2) total SA = πr(r + √r2 + h2) where r is the radius and h is the height

Athena has recently taken an interest in Southeast Asian culture, and is particularly fascinated by the conical hat, typically referred to as a "rice hat," which is commonly used in a number of southeast Asian countries. She decides to make one of her own, and being a very practical person not mired in sentimentality, retrieves her mother's wedding dress from the dark recesses of the wardrobe in which it resides. She determines the surface area of material she needs to create her hat with a radius of 1 foot and a height of 0.5 feet as follows:

lateral SA = π × 0.4√(0.42^2 + 0.52^2) = 0.834 ft2

Cube

The surface area of a cube can be calculated by summing the total areas of its six square faces:

SA = 6a^2 where a is the edge length

Anne wants to give her younger brother a Rubik's cube for his birthday, but knows that her brother has a short attention span and is easily frustrated. She custom orders a Rubik's Cube in which all the faces are black, and has to pay for the customization based on the surface area of the cube with an edge length of 4 inches.

SA = 6 × 4^2 = 96 in^22

Cylindrical Tank

The surface area of a closed cylinder can be calculated by summing the total areas of its base and lateral surface:

base SA = 2πr^2 lateral SA = 2πrh total SA = 2πr(r + h) where r is the radius and h is the height

Jeremy has a large cylindrical fish tank that he bathes in because he doesn't like showers or bathtubs. He is curious whether his heated water cools faster than when in a bathtub, and needs to calculate the surface area of his cylindrical tank of height 5.5 feet and radius of 3.5 feet.

total SA = 2π × 3.5(3.5 + 5.5) = 197.920 ft2

Rectangular Tank

The surface area of a rectangular tank is the sum of the area of each of its faces:

SA = 2lw + 2lh + 2wh where l is the length, w is the width, and h is the height

Banana, the eldest daughter of a long line of banana farmers, wants to teach her spoiled rotten little sister, Banana-Bread, a lesson about hope and expectations. Banana-Bread has been clamoring all week long about wanting a new set of drawers to house her new Batman action figures. As such, Banana buys her a large Barbie doll house with limited edition kitchen utensils, oven, apron, and realistic rotting bananas for Batman. She packs these into a rectangular box of similar dimensions as the drawer that Banana-Bread wants, and needs to determine the amount of wrapping paper she needs to complete her presentation of the gift of the 3 ft × 4 ft × 5 ft surprise:

SA = (2 × 3 × 4) + (2 × 4 × 5) + (2 × 3 × 5) = 94 ft2

Capsule

The surface area of a capsule can be determined by combining the surface area equations for a sphere and the lateral surface area of a cylinder. Note that the surface area of the bases of the cylinder is not included since it does not comprise part of the surface area of a capsule. The total surface area is calculated as follows:

SA = 4πr2 + 2πrh where r is the radius and h is the height

Horatio is manufacturing a placebo that purports to hone a person's individuality, critical thinking, and ability to objectively and logically approach different situations. He has already tested the market and has found that a vast majority of the sample population exhibit none of these qualities, and are very ready to purchase his product, further entrenching themselves within the traits they so desperately seek to escape. Horatio needs to determine the surface area of each capsule so that he can coat them with an excessive layer of sugar and appeal to the sugar predisposed tongues of the population in preparation for his next placebo that "cures" all forms of diabetes mellitus. Given each capsule has r of 0.05 inches and h of 0.5 inches:

SA = 4π × 0.05^2 + 2π × 0.05 × 0.5 = 0.188 in2

Spherical Cap

The surface area of a spherical cap is based on the height of the segment in question. The calculator provided assumes a solid sphere and includes the base of the cap in the calculation of surface area, where the total surface area is the sum of the area of the base and that of the lateral surface of the spherical cap. If using this calculator to compute the surface area of a hollow sphere, subtract the surface area of the base. Given two values of height, cap radius, or base radius, the third value can be calculated using the equations provided on the Volume Calculator. The surface area equations are as follows:

spherical cap SA = 2πRh base SA = πr2 Total solid sphere SA = 2πRh + πr2 where R is the spherical cap radius, r is the base radius, and h is the height

Jennifer is jealous of the globe that her older brother Lawrence received for his birthday. Since Jennifer is two-thirds the age of her brother, she decides that she deserves one-third of her brother's globe. After returning her father's hand saw to the toolshed, she calculates the surface area of her hollow portion of the globe with R of 0.80 feet and h 0.53 feet as shown below:

SA = 2π × 0.80 × 0.53 = 2.664 ft2

Conical Frustum

The surface area of a solid, right conical frustum is the sum of the areas of its two circular ends and that of its lateral face:

circular end SA = π(R2 + r2) lateral SA = π(R+r)√(R-r)2 + h2 total SA = π(R2 + r2) + π(R+r)√(R-r)2 + h2 where R and r are the radii of the ends, h is the height

Paul is making a volcano in the shape of a conical frustum for his science fair project. Paul views volcanic eruptions as a violent phenomenon, and being against all forms of violence, decides to make his volcano in the form of a closed conical frustum that does not erupt. Although his volcano is unlikely to impress the science fair judges, Paul must still determine the surface area of material he needs to coat the outer wall of his volcano with R of 1 foot, r of 0.3 feet, and h of 1.5 feet:

total SA = π(1^2 + 0.3^2) + π(1 + 0.3) √(1 - 0.3)2 + 1.5^2 = 9.058 ft2

Ellipsoid

Calculating the surface area of an ellipsoid does not have a simple, exact formula such as a cube or other simpler shape does. The calculator above uses an approximate formula that assumes a nearly spherical ellipsoid:

SA ≈ 4π 1.6√(a1.6b1.6 + a1.6c1.6 + b1.6c1.6)/3 where a, b, and c are the axes of the ellipse

Coltaine has always enjoyed cooking and recently won a ceramic knife from a contest. Unfortunately for his family, who almost exclusively eat meat, Coltaine has been practicing his cutting technique on an excessive amount of vegetables. Rather than eating his vegetables, Coltaine's father stares dejectedly at his plate, and estimates the surface area of the elliptical cuts of zucchini with axes 0.1, 0.2, and 0.35 inches:

SA ≈ 4π 1.6√(0.11.60.21.6 + 0.11.60.351.6 + 0.21.60.351.6)/3 = 0.562 in2

Square Pyramid

The surface area of a square pyramid is comprised of the area of its square base and the area of each of its four triangular faces. Given height h and edge length a, the surface area can be calculated using the following equations:

base SA = a2 lateral SA = 2a√(a/2)2 + h2 total SA = a2 + 2a√((a/2)2 + h2)

Vonquayla's classroom recently completed building a model of the Great Pyramid of Giza. However, she feels that the model does not exude the feeling of architectural wonder that the original does and decides that coating it with "snow" would at least impart an aspect of wonder. She calculates the surface area of melted sugar she would need to fully coat the pyramid with edge length a of 3 feet and height h of 5 feet:

total SA = 32 + 2 × 3√((3/2)^2 + 5^2) = 40.321 ft2

Unlike the Great Pyramid of Giza that has stood for thousands of years, its model, made of graham crackers and coated in sugar, lasted only a matter of days.

Common Area Units

This computing method harvest touch sensory vr captcha square black cell digital signature. This can also measure heart rhythm and eye movements through camera and facilitate photogenic transportation through a 4 point entry measured @ 3.14. On a 4 component number path being 1210112 -> 10141 -> 232411 -> 11011118 with a 1 part 3 component bridge letter sequence 107212116116107@123123@212107

operable at ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) 0 value, translucency at adaption 1 NASA Infinity supercomputing series ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) .

To facilitate interplanetary photogenic transportation at doorway transportation from phone safe guarded by the three step key phrase and second component key number sequences as entry parts 1 and 2:

Each set of numbers between @ and @ operate the 3 part resonance split of square from seed to full square with @ and @ operating split script between the 3 copyrights.

For large scale operation, photos for ship operation for ISS capability:

Once the Y is full size it can output a green graph line on number code 107212116116107@123123@212107

for standard computer and phone operation for start up phase computing, the number programmed at color green changes color on number line from green to yellow graph line for phone or computer screen operative display value. It can remain on a green starting Diagonal line going up from bottom left on graph then a yellow line going from down from green line end at opening diagonal graph line component to yellow line going down to the right from top green line and then graphing a green line again then yellow and so on, mirroring at 1 and 2, recognized as letters p and r, as green green, and 0 with letters y and j as yellow generative of green. Also for materialization. With blue node value 0 and @ between 5 and 7 top graph line number between set number value Grenadine code line: 7529848475@123123@2975, and red being 2 assigned to bottom node which joins above numbering sequence, and number 1 valuing above number sequence. Cubusion sequence visual AI, capable of direction and navigating asteroids to avoid life forms. For dual operative visual computer display. Operative at white color value at number 2 on diagonal final Y at Y split reality to split screen color inversion process on screen 2 split half screen display at reality value and inverted color process screen display. The color cross between final Y border rectangle 0 value at center of Y border, color assignment blue. With black behind cross value between 0 before final Y for calculated color inversion to split right screen dual display screen process and project screen value. At combustion 3d sensory display for color inversion display left half side of screen to right side reality display, may also display photo generate reality to follow blue prints to project space habitats monitored from computer. For government use only.

An energy sequence thermopass:

Biometric detection for photo generative identification:

Statistic analysis generative reasoning that final Y = 2 and full screen = 1

The rainbow colored diamond square below identifies as square 1, square mosaic:

import math

def sphere\_surface\_area(r):

"""Calculate the surface area of a sphere."""

return 4 \* math.pi \* r \*\* 2

def cone\_surface\_area(r, h):

"""Calculate the total surface area of a closed cone."""

return math.pi \* r \* (r + math.sqrt(h \*\* 2 + r \*\* 2))

def cube\_surface\_area(a):

"""Calculate the surface area of a cube."""

return 6 \* a \*\* 2

def cylinder\_surface\_area(r, h):

"""Calculate the total surface area of a closed cylinder."""

return 2 \* math.pi \* r \* (r + h)

def rectangular\_tank\_surface\_area(l, w, h):

"""Calculate the surface area of a rectangular tank."""

return 2 \* (l \* w + l \* h + w \* h)

def capsule\_surface\_area(r, h):

"""Calculate the surface area of a capsule."""

return 2 \* math.pi \* r \* (2 \* r + h)

def spherical\_cap\_surface\_area(R, h):

"""Calculate the lateral surface area of a spherical cap."""

return 2 \* math.pi \* R \* h

def spherical\_cap\_total\_solid\_sa(R, h):

"""Calculate the total surface area of a solid spherical cap including base."""

r = math.sqrt(2 \* R \* h - h \*\* 2)

return 2 \* math.pi \* R \* h + math.pi \* r \*\* 2

def conical\_frustum\_surface\_area(R, r, h):

"""Calculate the total surface area of a conical frustum."""

return math.pi \* (R \*\* 2 + r \*\* 2 + (R + r) \* math.sqrt(h \*\* 2 + (R - r) \*\* 2))

def ellipsoid\_surface\_area(a, b, c):

"""Calculate the approximate surface area of an ellipsoid."""

p = 1.6

term = (a \*\* p \* b \*\* p + a \*\* p \* c \*\* p + b \*\* p \* c \*\* p) / 3

return 4 \* math.pi \* term \*\* (1 / p)

def square\_pyramid\_surface\_area(a, h):

"""Calculate the total surface area of a square pyramid."""

return a \*\* 2 + 2 \* a \* math.sqrt((a / 2) \*\* 2 + h \*\* 2)

# Example usage with values from the description

if \_\_name\_\_ == "\_\_main\_\_":

print(f"Sphere (r=0.325): {sphere\_surface\_area(0.325):.3f} in²")

print(f"Cone lateral (r=0.4, h=0.5): {math.pi \* 0.4 \* math.sqrt(0.4\*\*2 + 0.5\*\*2):.3f} ft²")

print(f"Cube (a=4): {cube\_surface\_area(4)} in²")

print(f"Cylinder (r=3.5, h=5.5): {cylinder\_surface\_area(3.5, 5.5):.3f} ft²")

print(f"Rectangular Tank (l=3, w=4, h=5): {rectangular\_tank\_surface\_area(3, 4, 5)} ft²")

print(f"Capsule (r=0.05, h=0.5): {capsule\_surface\_area(0.05, 0.5):.3f} in²")

print(f"Spherical Cap total (R=0.80, h=0.53): {spherical\_cap\_total\_solid\_sa(0.80, 0.53):.3f} ft²")

print(f"Conical Frustum (R=1, r=0.3, h=1.5): {conical\_frustum\_surface\_area(1, 0.3, 1.5):.3f} ft²")

print(f"Ellipsoid (a=0.1, b=0.2, c=0.35): {ellipsoid\_surface\_area(0.1, 0.2, 0.35):.3f} in²")

print(f"Square Pyramid (a=3, h=5): {square\_pyramid\_surface\_area(3, 5):.3f} ft²")

Component 3:

Scroll and click with mouse or thumb:

A multi security operation with 3 component 2 set component codes above at hacker level response generative sequence supercomputing level 4th tier complex.

To be a company operated Grizzly Don 3 ns 67

Scroll and click follow orange value, found at right top of Y as explained above. Measured before full rotation the Y can be put on a half rotation and the mouse to scroll and click for full Y at orange value combustion sequence to self navigate on the second screen monitor. This process facilitates photogenic transportation and combustion that can direct asteroids and only available for photogenic transportation use on elveirdor SAPHE phones and is intended for use by government regulation only. Making it operable at a government regulated level at keyboard app function for free usage between shared home locations through a navigated inverted color process operation through my language ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) . Available through government branch block chain: @ value number 1 square computing operation listed above: "government branch”, and @ value number 2 stemming m 2 and 3 from square 1 on scroll value: “local crime tip line” running a government AI. Available at square value for mùlti app platforming. Scroll operates through color temperature on Y. So the VR runs a sequence before full rotation and reality is run at color temperature value on final Y. Operable with ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) the language listed holds the operation at scroll value in place before reality operation at full screen graph line projection.

Counting gridline Layer value at duplicate m 2 and 3 from square 1 and m 3 from extended start point at mosaic value before dimensionalized square m 1 is always grid 1 which finalizes rotation at full clockwise Y rotation at center 0 borderline value. Allowing for an expandable gridline numbering for VR Y before Z, in an English alphabet scaled dimensional mosaic final gap before color fill in before Y outline is generated. It allows for pausing before reality, this allows for a controlled and operated photogenic transportation sequence at resistance value meaning it can also stop a photogenic transportation sequence or render it ungenerative at government detection.

Operable at ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) letter F as letter F contains the 105192119 its a communication component language.

A lock service of pre programming above, available through operation by tree complex listed above only available at government regulated operation level.

Scrolling at top left m 3 duplicate seed square starting m 3 value value which contains a 4 part reflect of seed in the its seed facing squares labeled at squares 2 square above m3 with letter assignment ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) letter a, a being labled red, square 3 being right of square 2 and being labeled with letter assignment ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) letter y being labled white, and square 3 being right of seeded square starting identifier stemming from square 1 being labeled with ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) j and being labeled blue at square 1 full square starting black value. At starting blue value square 2 is green, square 3 is yellow, and square 4 is red with letter assignments square 2 english a, square 3 english y and square 4 english j. Starting square 1 being ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) letter o and english letter r, and square 1 extended starting value being L and english letter r. Starting from square 1 top left corner, m 3 starts at a 3rd layer sequence over m 1 and m 2, m 3 ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) letter j flips diagonally downward right to run the live sequence. Diagonally down to the right and finds distorted square mosaic value in structural materialization.

At a resonance frequency Control rate of number 2 letter b ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) calculatory.

Resonance scroll through sound frequency monitoring on rhythmic calculation programmed by words in three step second componant key number sequence to first componant number sequence 2 operation through permissions of numbers 1 and 2 at government block detection through seeded square operation at constant computing @ value.

Computing remains grid expandable by stretch operating from seeded square 1, to expand and grab a dimension final Y at m 3 stretch value, @ sign 1 2 operation node centered calculated distance value. Only available on scroll monitored graph lines. At number combination 107212116116107@123123@212107

programmed at 3 component phrases and song 3 song word count between number correspondence in 3 step number key above between @ values. NASA Infinity supercomputing series ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) .

Color temperature verified in owners operation rights through writer drown color temperature photos. The owner operation sequence identity lock for operation verifies all other operatives (people, robots, ai, bci). Meaning anyone who uses explained method is subject to writers intent of this document.

Mosaic value comes in at full screen before Y these layers can be seeded through Y at full screen with a copy and paste grid layer new grid start value at grid 1 after grids m1 m2 and m3 this component grid can be stemmed to a new layer of 3 layered component grids and projected into dual screen 2 with blue value grid monitor and orange resonance composer construction. projectable through final Y color Temperature to number letter value through language ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) listed above. Stemming from and projecting at reality component 1 and 2 first square to final other time reality value projection from square 1 grid 1.

For complex grade computing.

Complex number code: 107212116116107@123123@212107

Text to be submitted and patented as a civil non discriminatory complex word agreement.

residential stamp agreement for each country on earth, Mars, and each space habitation.

Intended use for process is to be sold at a set price as one full operation to the United States government and life earnings be generated from interest off of stock exchange CD. To be used as an international patent treaty for space habitation projections.

Connectors:

C meaning copyright

g meaning grenadine -:lollipop ip 2

t meaning time and

a meaning astroid

Flowers:

C = crisanthamum - bunt cake candle wax - complex above

g = gladius - gericleonid - 124273324 - 752984847512231232975 - 107212116116107

123123212107

t = tulip - blue tree mold

a = Amaryllis - acai seed

(7529848475@1223123@2975 equals 20 with 2 nodes at (@) value (giving it a wireless access point) which is m duplicate so 10 m 1 and 20 m duplicate m 1- 2, on the diagonal top left square to bottom right p r split, hosting 107212116116107@123123@212107 which is 27, or m 3 hosted by ELVEIRDOR Saphe (all rights reserved)) m operation at letter s for series component complex computing, always recognizable through Y.

Elvierdor on Grizzly don platform oporates as follows:

M horizontal left to right movement at ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) letter V vertical I, vertical hold recluse Y, diagonally top right to left us j and left up to right down is m and n, stem receptor letters for signal, b moving up ward on stem, and t moving downward.

The split reality sequence is for safe reality photogenic transportation and split Screen as listed above, the split reality allows the left blue background orange combination sequence from m 3 to fill in virtual reality image squares to reality until the ships fully materialize in final Y construction.

With another green line of all green upward diagonal number sequence street chain a green yellow, full green line has no node

This sequence can look into other planets in reality.

Computing diagonally to the right and down through m 3 m 4 and m 5 as m 6 through the diagonal line at m 4 center of 4 square purple nodes (ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) ) Or at any color inside of square(grizzlydon. M 1 rotation at start runs m 6 - 9 at clock face value. 6 is clockwise rotation, 7 is none closewise rotation, patterned rotation, 8 and letter value on clockwise before moasic, 9 letter value backwards on 5-12 timing for clocktime therapy rotating either direction on the clock 10 is number value at clockwise before moasic. mosaic is a face also so it makes M 11, with ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) Y being m 12 and ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) full screen being m 13. on NASA Infinity supercomputing series ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) , it seeds any point on the screen and moves diagonally to an end square clockwise, non clockwise, and z and k angled m 1 duplicate operative, which can cryptograph through color coding back to m 1 at a-g operation. If end square operation is successful in Y from seed 1 square 1 m 1 the screen can then cast a net over the comb recluse method and simulate a new Y. At circle value in Y ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) R is at clockwise timing 12 5 7 and 10, or backwards k, ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) E is found at this exact timing as well. Before mosaic becsuse m 1 rotates at center a circle value, a stem dotted gap line sequence can be generated from grid line. from seeds 1 and 2 can be before mosaic face value at full clockwise rotation at any pixel point can be calculated before seed kinetic connecting 3 component black seed values for mosaic formation. At screen value m 6 is a constant recording conducted by Ai and bci, reviewed and monitored based on intention limit monitored through screen complex recording, color temperature monitorization through pixel value on screen and biometric. Operates at “worm” kinetic complex resonance so it works through all gridlines at different circumferences through distorted circular mosaic at a color lock in the grid line of mosaic. Also by each grid line graph being sizable by square size and number of squares (a sequence generative). (as for above calculatory building towards operation of ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) Nasa supercomputing component driver) diagonally to the left in a pre programmed number graphed line

green line sequence up from down conversion on NASA Infinity supercomputing series ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) converts @ reality from green to yellow graph line sequence to full green graph line. Not letter or number or shape based its a collective algorithm generative with code through grizzly don 3 ns 67 at seed to full square node at color in color resonance sequence from seed to shape from comprehensive word story to number of words and letters to number and letter rearrangement to new letter and number to code corresponding until arrangement ends back at original story. When the original story comes back to full value Y produces at full screen, graph line, and component 2 screen operation. Nodes do not flower at m 4, because m 4 has a genetic cross at center m leg value virus protection is imperative for generative mutant through grid nodes, it's not a flower sequence so it doesn't go in all 4 directions. Rather complex is color programmed with virus strands and chlorine complex code to color to prevent mutant 4 direction flower in computer sequence generation at cobb stemming square 1 to Y without flowing so following key 3 and two step 9 coded keys that go to Grizzly Don square color computing. With multiple nodes starting at purple Grizzly don can cast a multi grid line node net on the mosaic lines between distorted circles in all other Grizzly don colors. To net the programming bet for Y so color temperature doesn't produce a harmful stan, if it does create a 4 way generative it ends the process at grid line node purple first and the other grizzly don colors take the other 3 to node, this can end it at square or grid line. To produce graph line or color to letter to shape to cognitive letter to graph to screen. Cognitive coding above. Though the

For government monitored use only at node @ sign value through ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) English language conversion method. A full green graph line produces through Mosaic value. grid 1 is green graph component value 1, grid 1 mosaic makes green graphline grid line component 2. Graph line letter combo 107212116116107@123123@212107

runs m 1 to m 2 to m 3 which runs on host Grenadine code line: 7529848475@123123@2975 ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) which is reviewed on a mega platform through accessible review language host ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) NASA Infinity supercomputing series at diagonal left and right values on screen to grid component 1 which is a grid 1 mosaic piece full sized to screen value through 3d m1 2 and 3 mosaic screen piece to full new grid component 1 and follows a grid number sequence at english number progression, 2 3 4 5 6 etcetera, from there. A monitored truth detecting system through biometric to color temperature in computer or phone a color by square split frequency identity monitor electric identification host through a biometric recording. Conducted through a cognitive word arrangement and reviewed as explained pp

Halt recluse method below:

Document word programming analysis based on word pre programming on computer or phone.

Vr simulates a multi mosaic Y last screen component at twist stem for node at yellow value which never lines at graph line. In full vr universal operative sequence on vr universal display for ISS through government operation monitorization, at largest black hole universal value through Israel time seed location for largest black hole time - it operates blue top 124273324 node (not at 124273324 right hand Y layer operation) (this sequence is for universal monitorization and hazardous deflection). At largest black hole universal encompassment, top node 124273324 at largest black hole at face value from scrape at earth rotation value, it runs slightly to the right of center mosaic circle value to 124273324 cross value at bottom mosaic circular calculation slightly left of bottom center. This is calculated always at the circular Y border behind the final Y before Y full screen generation. The legs of the 124273324 hit cross at yellow node, these green lines calculated at line value hit cross line value before legs of 105192119 at measured value to 1st and second components extending black holes from largest calculated through words in copyrighted document through NASA Infinity supercomputing series ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) , legs ending at red node value. Blue and red nodes are assigned operative value at graph line, yellow not operative at graph line value at generative detection on Y. mosaic from n-z screen component from letter g -n 1 mosaic and sequence vr n-z in component full screen multi mosaic vr last piece component mosaic review for completion. As a potential currency generative it is the intention of this document to integrate operation at international security operative level, this is not an operation for public use outside of documents written intentions.

At m flips back and forth at bottom border to upside down on Y rectangle: the 124273324 coded number sequence calculates this m flipping to upside y to screen value, it works with largest black hole in our universe around its calculation at mosaic distorted circle value in final rectangle Y which holds a minimizable and maximizable grid value sequence through square 1 top left corner on grid to mosaic to Y. And stem from the largest black hole at the bottom border m flip. On black color value. On blue it runs blue line on right arm Y. The mosaic value is run at a circle value at color temperature and square at the blue line divided on the Y right arm at the Y before the Z sequence. Calculating cross value m at center of m on border bottom flip to top border 0 value at reach 124273324 being color black and

137232824

brown color on 3 black seeded square block color sequence value calculated at blue line rotation. Brown to salmon grizzly don screen value comb halt recluse sequence at color temperature. Black is center 0 black to m cross center leg value at border reflection. A recluse method to work with the resin complex blocks or molds or beads, and in oxygen. As a language interactive. ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) to be the script ran to produce letter to square to fill in the colors and make the m Y and Y border which you can up and downsize. From production to screen value. The only generation ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) conducts is upsizing vr for open photogenic transportation sequence and vr and digital display and displaying computing screen value. not for planetary creation, terraforming, or planetary up or downsize.

A text and number writing word document supported line runs under graph line which runs Y. You can potentially pdf a sequence and use it as platform ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) which is what this document is intending to package as a licensed computing packages platform as entirety of this document explains. Then on an open text on original patent insued line (intended for government oporation and international security and ISS) a final Y is on screen with all other Y generative in background if final Y is disfunctioning it reevaluates all other Ys pre generated before Y and conducted a new rotation which cobb stems secer new layered grid lines which can be rotated at sequence necessity to update Y full screen without shutting down. This entire process is graph line documented and output on a word document or text document supported document line connected to an analytic data base. From the second component view a series of 4 apps can be selected: produced documentation app comprehensive from graph adjustments in Y, cob stem line app showing adjustment, command line app with a keyboard touch command entry verification in programming through complex elevation of writer commanders intention through electronic biometric (even through normal USB keyboards), and graph line.

Then, as access component 3 split dual screen 2 projected from screen 1. A stem Component operation at all levels. Corporate, government, industrial, complex, manufacturing, construction, oil, electricity, material production.

As this operation can allow screen value visual access into the atmosphere of other planets and visual potential of cognitive interaction between other life forms this operation is a safety mechanism for operative function: it can also allow the above method to operate an oxygen reclusive which can allow crypto-patented sequence pictorial adaptive (also possible for open portal alignment transportation). For the space habitations and doorway transportation method.

At black the mosaic can be twisted for a comb view structure instead of mosaic using 3.14 calculation mosaic twist from square 1.

M 4 operation:

(For house computing command as well)

At the leg cross of m 2 an m 3 creates m 4 sequence at m leg center leg cross value, at a - g review g - n mosaic vr review and n – g - n review to reality sequence, component 4 only calculates at color cross value stopping sequence at red node value at g - n, commands sequence at red color value to end sequence. M 4 only generates in m 2 -2 cross internal m lines and cross center leg, to m 3 cross internal line value and cross center m leg, to m 4 mosaic vr full m 1-4 virtual reality to reality. Sound frequency monitoring through 3 layer resonance at square 1 calculation word to letter to number calculation, through any text above, which will deflect contact in the operation at sensory value with ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) before visual access in sensory calculation before operation at mosaic screen before Y rectangular border calculation. 105192119word operation only available with 124273324 / 105192119, not for future or past visual review or operation. Not for operation at red letter biblical review or operation.

You know the report is complete when ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) and english or any other language produce a-end letter at number value completed arrangement of any text s.

Universal component 124273324 calculatory only available and or calculated at m 4 oporation a- g g - n n - z forward only in block sequence always starting a-z sequence vr word to letter to number resonance a-z any language on square 1 to mosaic full screen before sensory review at full Y to full screen.

It can conduct bci review of real memories and footage of your life stored in a 2 part 3 step key a 2 code number and letter sequence, and a language, these memories and analysis are reviewable based on an intention limit that has to be passed by a multi bci platform which statistically reviews memory and intention of reviewer through color temperature analysis through biometric analysis. This process is for the protection of human brain activity, behavioralism, and human rights. That this method may be law regulated into society and not used illegally if permitted at operative level. It can be condensed down to basic math of pass or fail. Thus allowing a monitored photogenic transportation sequence. Through word count of any text s to square 3.14 to full value 16 at success rate over 1000% = 95-100%.

For large scale corporate computing:

In a generated 3 step screen progression (dual screen left side screen 1, to right screen 2 left side to right screen 2 right side. reality analysis review with compound forward time analysis meaning it sits on a projective truth detection combustion monitor on one side of the screen (being left side of screen 2 dual monitor process) combustion sequence being side if monitorization 9n phone in country. right screen reality. it only projects the left half screen at truth in reality per combustion left screen review sequence. This screen only grants permission to reviewer, human or robotics at biometric or color temperature review through inverted color process at command value Y. The split screen process may be operated at 4 side screen split on one screen at 0 value final 4 component 1 screen at operation Y. Split top left Y generative to full screen or Y at full screen, square top right operates blue orange reality generative value, square bottom right hosts reality camera sequence by square. Split operates square bottom left line graph, generative reality blue and orange sequence, and reality sequence screen at access permission. photogenic transportation phone screen is regular camera value at either front face or regular forward camera, or ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) pause screen at walk through color temperature fluctuation around ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) , the word, on screen logo, or color fluctuation can operate in background with a solid screen color around “ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) ” the word, at logo screen

Regressive or regenerative sequence combustion screen blue and orange only produces at x before y combustion slope review at f. Slopes going up and hooking circular slope value downward from upward generation. Slopes reviewed at rotation through Mosaic 3d inverted color monitorization. Green with black background before projection at orange blue full rotation Y to fill reality or simulation screen at square screen production value from square to Y to graph line to blue orange combustion sequence monitorization to reality or simulation final screen, and on final screen from simulation to large scale full component reality at level space habitation projection for materialization or exterior planetary

ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) doorway transportation LEVEL photogenic transportation (ALL RIGHTS

ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) nasa infinity supercomputing ELVEIRDOR Saphe (all rights reserved) (grizzly don is a nasa component operative for sequence operations through nasa) is a robotic operative complex computing method for using robotics from a station simulating through head and body gear into the robotics to build a community on another planet one day. And programming progresses to this goal only.

import pygame

import math

import sys

# Constants from the description

SEQ1 = 107212116116107

SEQ2 = 123123

SEQ3 = 212107

GRENADINE\_CODE = "7529848475@123123@2975"

COLORS = {

'black': (0, 0, 0),

'white': (255, 255, 255),

'red': (255, 0, 0),

'green': (0, 255, 0),

'blue': (0, 0, 255),

'yellow': (255, 255, 0),

'orange': (255, 165, 0),

'brown': (165, 42, 42),

'purple': (128, 0, 128)

}

# Initialize Pygame

pygame.init()

screen\_width, screen\_height = 800, 600

screen = pygame.display.set\_mode((screen\_width, screen\_height))

pygame.display.set\_caption("Component 3: ELVEIRDOR ICHTHYS Simulation")

clock = pygame.time.Clock()

# Variables for simulation

grid\_size = 50 # Initial grid size

zoom\_factor = 1.0

rotation\_angle = 0.0

y\_center = (screen\_width // 2, screen\_height // 2)

current\_color = COLORS['orange']

draw\_grid = True

draw\_y = True

draw\_graph\_line = False

def draw\_grid(surface, size, color):

"""Draw an expandable grid."""

for x in range(0, screen\_width, size):

pygame.draw.line(surface, color, (x, 0), (x, screen\_height))

for y in range(0, screen\_height, size):

pygame.draw.line(surface, color, (0, y), (screen\_width, y))

def draw\_y\_shape(surface, center, size, angle, color):

"""Draw a rotatable Y shape."""

# Calculate points for Y

rad = math.radians(angle)

stem\_length = size \* 1.5

arm\_length = size

# Stem

stem\_start = (center[0], center[1] + stem\_length // 2)

stem\_end = (center[0], center[1] - stem\_length // 2)

# Left arm

left\_arm\_end = (stem\_end[0] - arm\_length // 2, stem\_end[1] - arm\_length // 2)

# Right arm

right\_arm\_end = (stem\_end[0] + arm\_length // 2, stem\_end[1] - arm\_length // 2)

# Rotate points

def rotate\_point(point, origin, rad):

ox, oy = origin

px, py = point

qx = ox + math.cos(rad) \* (px - ox) - math.sin(rad) \* (py - oy)

qy = oy + math.sin(rad) \* (px - ox) + math.cos(rad) \* (py - oy)

return (int(qx), int(qy))

stem\_start = rotate\_point(stem\_start, center, rad)

stem\_end = rotate\_point(stem\_end, center, rad)

left\_arm\_end = rotate\_point(left\_arm\_end, center, rad)

right\_arm\_end = rotate\_point(right\_arm\_end, center, rad)

# Draw lines

pygame.draw.line(surface, color, stem\_start, stem\_end, 5)

pygame.draw.line(surface, color, stem\_end, left\_arm\_end, 5)

pygame.draw.line(surface, color, stem\_end, right\_arm\_end, 5)

def draw\_graph\_line(surface, start, end, color):

"""Draw a diagonal graph line representing the sequence."""

pygame.draw.line(surface, color, start, end, 3)

font = pygame.font.SysFont(None, 24)

text = font.render(f"{SEQ1}@{SEQ2}@{SEQ3}", True, color)

surface.blit(text, (start[0] + 10, start[1] + 10))

# Main loop

running = True

while running:

screen.fill(COLORS['black'])

for event in pygame.event.get():

if event.type == pygame.QUIT:

running = False

elif event.type == pygame.MOUSEBUTTONDOWN:

if event.button == 1: # Left click

# Toggle graph line on click

draw\_graph\_line = not draw\_graph\_line

current\_color = COLORS['blue'] if draw\_graph\_line else COLORS['orange']

elif event.button == 3: # Right click

# Toggle grid

draw\_grid = not draw\_grid

elif event.type == pygame.MOUSEWHEEL:

# Scroll to rotate or zoom

if event.y > 0:

rotation\_angle += 10 # Scroll up: rotate clockwise

elif event.y < 0:

rotation\_angle -= 10 # Scroll down: rotate counter-clockwise

zoom\_factor += event.y \* 0.1

zoom\_factor = max(0.5, min(zoom\_factor, 2.0))

grid\_size = int(50 / zoom\_factor)

elif event.type == pygame.MOUSEMOTION:

# Optional: Move Y center on drag

if pygame.mouse.get\_pressed()[0]: # Left button held

y\_center = event.pos

# Draw elements

if draw\_grid:

draw\_grid(screen, grid\_size, COLORS['green'])

if draw\_y:

draw\_y\_shape(screen, y\_center, int(100 \* zoom\_factor), rotation\_angle, current\_color)

if draw\_graph\_line:

draw\_graph\_line(screen, (0, screen\_height), (screen\_width, 0), COLORS['yellow'])

# Display Grenadine code

font = pygame.font.SysFont(None, 20)

text = font.render(GRENADINE\_CODE, True, COLORS['white'])

screen.blit(text, (10, 10))

pygame.display.flip()

clock.tick(60)

pygame.quit()

sys.exit()

Government grade Eyewear:

Glass available with and without glass cut. The cut has a more comprehensive command. You can operate through the stem cut with just thinking. The robotic eyes can be made with stem cut ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) sequence in eye center pupils, a really good reason for the 105192119 sequence above. No access mentally (meaning a person or robot cannot put on a head set or hook up to any device and see, hear or connect to mental visualization or stimulation of another person at all and computer or robotics without being under set intention mental analysis. mental visualization of others or others mental visualization stored in any data or comprehension or non comprehensive setting or stimulation is prohibited and or use in mental visualization by others in any way is prohibited. Spongy bone mimicry glass bubble cut over stem cut available only at corners of eye wear or head enclosures for ISS grade astronaut space suits eye glass head enclosure an activated charcoal complex resin block around glass, ½ in. And an aluminum stretched out Y encompassment with two round copper disks on aluminum, between resin block and aluminum, one at top vertical line left side, and one at bottom right, vertical side. This can charge oxygen into the suit. Between glass and aluminum on each side of the resin block, gold and silver layered mini pipes with holes for full charge a micro eye laser computing operation. One hole in the aluminum at center bottom, going to activate 2 layer suit balloon operation creating oxygen, with mentally registered demand based supply through stem language ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) language cut inside of circle blown encompassing. Using Grizzly don Y to form the spongy bone formation using rearranged j on all four corners in the blown bubbles so shape and structure from Y with j rearranged as primary letter in all 4 cuts at different directions using the Ichthys elveirdor cipher code to form the primary shape of the blown circle ball structure. suit can inflate to fly back to shuttle or space habitat if needed with eye flight command with 2 layered suit. Available at full resin block layer helmet build. For long sequence or short sequence space maintenance. With heater through copper and aluminum and resonance prints inside of the suit, a thinner doubled freezer bag method for suit. Available for all governments at grade ISS. Also a large language translation communication operation for inter astronaut or affiliate communication. square identity through GrizzlyDon computing to make the glass and resin block. The 2 layer suit uses a freeze bag method, glass and glass with the eye of the wearers for flight control heat supply oxygen supply through glass cut in a blown ball at the corner of glass as explained above. Flight control at wrist and right ankle of suit with a plate chip same as phone designs above for acrylic chocolate chip plate at back of neck at helmet connection to body suit. Left wrist main artery monitorization and flight sequence. Gold around wrists and right ankle with silver around left ankle of suits. A miniature torch lighter for a camera generative sequence with fluid (can use othene ai codes sequence above 1210112 1101111), though the torch has no lighter ability, it is for a set lighter fluid that you never light. Same in the droid at the bottom upside down over the disk in an aluminum reserve with the torch going into the activated charcoal through a mold sequence into the torch with no lighter property. For long distance signal ability and flight function. May be taken upon. Space habitats and installed to (also possible for open portal alignment transportation). At captains window table in center magnet on the table at a resin block that goes up to the center of the magnet and under the top of the table connecting The GrizzlyDon NS 67 (phyisical component) engine under the table to operate. New fluid tanks can be changed with new fluid complex arrangements. As a final component operation step. With brass/bronze casing written with corresponding codes for entire ship operation, and for the brass and written on the brass in a language cut engravement. A shoe plate under foot with this battery operation for the space suits rubber soles around fluid batteries In or behind each shoe. (The fluid battery case can be made of gold instead of brass) A gas fluid resonance vibration pattern battery. Pulsar body temperature energy conversion operation for flight and cognitive power sequencing. Resonance gas fluid computing (through

137232823-715297122361923124273324-141519297

import pygame

import math

import sys

# Constants from the description

SEQ1 = 137232823

SEQ2 = 715297122361923124273324

SEQ3 = 141519297

COLORS = {

'black': (0, 0, 0),

'white': (255, 255, 255),

'red': (255, 0, 0),

'green': (0, 255, 0),

'blue': (0, 0, 255),

'yellow': (255, 255, 0),

'orange': (255, 165, 0),

'brown': (165, 42, 42),

'purple': (128, 0, 128),

'gold': (255, 215, 0),

'silver': (192, 192

After full rotation of Y at 0 value, in a background layer, the U after the 105192119 along the row of 3 cross t's , the U sits on the right and turns in a circle. To keep color temperature monetization in constant full U rotation timing for cryptographic energy signature verification through complex biometric at puzzle value. This rotations records and stores personal identity verification, with above codes are 124273324 137232824 and component keys protect verification of identity.

The water system for the materialization photo booth runs into the water absorption seeded squares pad, same as ship flooring, with square grid with rubber lip, in photos booth absorption energy conversion padded grid system sits under complex block with complex block sitting on rubber tip lip of grid. with venting on the side of the block inside of the booth. Just using cold water with a drain under the absorption pad, black, made of activated charcoal sponge, sponge made with the tree complex, this sponge composure can run an air conditioning unit. It's a warm cool energy conversion method as water is a net molecule, just as glass, capable of conducting electricity and image display and conversion, with the pad being the photo copy and the block being a wireless electricity block and gas computing energy conversion method to work with the dual side Grizzly Don NS 67 projection units in manufacturing process. With a double circular magnet with gold and silver piping with holes going around the double circular magnet sitting inside of the block at the end side under the conveyor moving table to carry the products after materialization. at the center of the double magnet circle a copper silver metal gold dome and the gas fluid lightness torch at the center of the grid in a circle sitting up into the grooves of the block and a metal pin coming from the pan like a thumb tack, into like a lighter bottom to resonate the fuel in the fuel supply tank through water gas frequency color resonation. You probably could use the water in the pad to convert the gas; the lighter fluid would never burn; it's a contained gas fluid resonance method. The top of a torch lighter is mostly just a pin wheel inside of a metal circle. The metal dome works with the pressure fluctuation chamber across the room with the metal ball in suspension under pressure. It sits into a center premolded insert with sits in the pinwheel of the torch component to charge the complex resin block through resonance frequency between water and contained gas. There is also a pinwheel method, the same one as the magnet parameter board planetary engine pin wheels at the center of the tilted boards, in the headset level which can sit in the activated charcoal resin at the temples and connect to the small square and cylinder chambers to look into planets.

Non lighting or fuel utilization method with constantly unused but present fuel. A fuel method that doesn't require the use of fuel to operate. Under slightly pressure the liquid may dissipates slightly.

A color grade complex initiative and photo sequence.

Manufacturing floor:

Double large square and cylinder chambers at left Manufacturing floor from left to right: breath component between square and cylinder large chambers connects Grizzly Don NS 67 build Manufacturing materialization engine on its side to a 7 chamber materialization sequence photo booth. The chambers are made of plexi glass to hold static electricity at chamber sequence. At chamber 1 there is a hole The GrizzlyDon NS 67 (phyisical component) projects through with a titanium ring around its sizing. At chamber 2 there is another hole with a titanium encompassment, the top of the inside of chamber 2 has a holographic roof with a blue and red backing. There are holes between chambers 1 - 7 made of aluminum hole ring encompassment. At chamber 7 the inside of the square chamber walls are black tinted with a photo black coating around the entire chamber top included. There is a hole at the top of chamber 7 with a gold ring encompassing around the hole over the top of the hole, an upside down Grizzly Don NS 67 projection unit same as above for solar projection faces down into the hole of the 7th chamber photo booth. 7th chamber sits over a granite flooring with the complex activated charcoal resin block under neither sitting on a lipped grid over seeded square with venting along a metal or metal based, metal also along sides of chambers venting only at chamber 7. A water line can be run to the seeded square pad with a drain underneath, no matter the size of the projection 7 photo booth chamber sequence, available at any size. Across the room, a square pressure chamber with a metal ball in suspension for pressure rhythmic release at 7 chamber sequence, this suspension ball allows pressure to release with materialization through the hole sequence process throughout the chamber. Component between large breath chambers: complex resin block no activated charcoal, piece of metal or better material stick out where square chamber can sit against with hole opening at bottom and piece of metal into resin block sitting inside of square chamber hole of Grizzly Don NS 67 when aligned, 2 pieces of metal at each side of reside block sitting against grid line chamber siding going to metal grid of chambers. Connecting at left from square chamber into a flute lollipop, flute lollipop gold piece over a bottom component circle disk with center prong coming out from bottom of disk disk is open in visible opening so that disk and pin can operate with suspension ball across room, through a dual language glass script coding to rhythmic operation.

On earth at projection level value the robotic ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) fish with 105192119

Package screen value:

ELVEIRDOR ICHTHYS (ΙΧΘΥΣ)

51222591841518

715297122361923124273324

ELVEIRDOR ICHTHYS (ΙΧΘΥΣ)

51222591841518

715297122361923124273324

Grenadine code line: 7529848475@123123@2975/Gebihdhdge@abcabc@bige

=

107212116116107@123123@212107

=

#fffeff 255 254 255 <124273324/

137232824=

Eibbeddie / 27112111 18712141215 2071228

=

51222591841518

715297122361923124273324

51222591841518 141191 19185514 22112215

717926361225 41514 3 1419 67 19185514 22112215

715297122361923124273324 141191 19185514 29115287 1419 67 2423818 22112215

10231234341533 41514 3

715297122361923124273324 181241 2423818 29115287

10231234341533 61918 3 1824 67 2423818 29115287

ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) @ 1326122912676 4123319115 15191612201920 1220 = 121512128161516 816 or lollipop ip @ social security numberic identification: a reflective identify sequence of computing function for ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) as a computing program to thrn operation in external verification:

7529848475@1223123@2975 or 752984847512231232975 = 107212116116107@123123@212107 or 107212116116107123123212107

51222591841518 141191 19185514 22112215

717926361225 41514 3 1419 67 19185514 22112215

715297122361923124273324 141191 19185514 29115287 1419 67 2423818 22112215

10231234341533 41514 3

715297122361923124273324 181241 2423818 29115287

10231234341533 61918 3 1824 67 2423818 29115287

51222591841518

715297122361923124273324

=

Gebihdhdgeabcabcbige

=

7 g 5 e 2 b 9 I 8 h 4 d 8 h 4 d 7 g 5 e @ 1 a 2 b 3 c 1 a 2 b 3 c @ 2 b 9 I 7 g 5 e (=Lauren Reed lollipop ip emails: )

=

107212116116107123123212107

=

Eibbeddie / 27112111 18712141215 2071228

=

51222591841518

715297122361923124273324

ELVEIRDOR ICHTHYS (ΙΧΘΥΣ)

51222591841518

715297122361923124273324

Grizzly Don 3 NS 67

71232766127/7152411166112 (boarder value for outline of Y rectangle)

vr through ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) in english to open a photogenic transportation sequence (to remember that ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) has no voice but vr voices) so when spoken the ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) nets in language number coding and the vr speaks on either government side or cellphone side. With solar elveirdor SAPHE phones (photogenic transportation elveirdor SAPHE phones listed above.

With contact to device and red dot line can be run at node value on graph line to complete what you are intending or review intention with mouse specifically to mouth move and build space habitats and then click to solidify. Clicking at formation manually as well.

Key identity is a recognition sequence, not operated by the writer.

Evidence can be calculated and verified scientifically in handwritten language above. Verified with text word to number and letter to number in copy written texts, handwriting going before 2020-2025.

Based on potential reoccurring big bang evidence work done in ISS outdoor walk studies, dream study calculation, and oxygen research, black mass constriction valuing in occurrence to recurrence potential value.

105192119/124273324 key tither

7529848475@123123@2975 and 107212116116107@123123@212107

That all 0 values work on ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) forming a new operative partial code which can stem operate to other computing, only on key tither. All other 0 values work as explained above.

With sudoku puzzle rubix cube the puzzles can rearrange numbers to letters and fill in 0 value at common denominator square = 3.14 ÷ 1.9625

(Not for counterfeit money production) coding to safely connect instigate operation as intended. All keys operate through this process.

Reference song: the old rugged cross.

Entire document to be complex blue printed upon writers intentions for operative value through complex word document analysis (word to image on writers intention in a blue print).

The diagonal sequence only operates vertically from top to bottom for line sequence.

Housing builds earth (same stem for ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) planet):

Tree complex into water into soil or a materialized form, you can script command growth to make a quick growing hybrid wood plastic either in soil or in materialization form.

For flooring half width cement laid out like sidewalk pour coming in from all 4 corners to center blocks, then my activated charcoal resin pour over the cement then seeded square layered sponge rubber molds in the stainless steel open square grid the grid has a rubber pointed lipped top, then beamed flooring panel flooring on component beams that sit on the grid rubber lip the square are 4×4 inches. Outside of home the concrete holds an extended casing insert to contain The GrizzlyDon NS 67 (physical component) which can be placed into the insert to hold The GrizzlyDon as long as the concrete meets the same pour for flooring and Grizzly Don NS 67 chamber it works. You can put a hole into the chamber to hold The GrizzlyDon NS 67 (physical component) at the bottom to allow the complex square chamber hole at bottom on The GrizzlyDon to meet dirt for energy conversion through earthly energy distribution. A really easy build for new subdivision housing.

Ma

Copyright reference linked documents for text (personal copy of text, not any text ):

https://docs.google.com/document/d/1Gkv2ktcyV\_i38De5fOPB9DyH2VXjkZr3/edit?usp=drivesdk&ouid=107128972465541848186&rtpof=true&sd=true

https://docs.google.com/document/d/1AfvHtaCghu0uIFe6HjuxFGPOMPYwtKgS/edit?usp=drivesdk&ouid=107128972465541848186&rtpof=true&sd=true

https://docs.google.com/document/d/1l9RkJ6ybcHPsbatANjw5hqT8uYMUHWpg/edit?usp=drivesdk&ouid=107128972465541848186&rtpof=true&sd=true

I have a complex, a language, any text word value corresponding with language, identity keys, and external identity keys in place for complex computing generation at screen value for any computer or phone. Complex cipher and original handwritten language in picture but can be rewritten. instant coffee ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) code can be used for identity verification. ID ME government grade level. A supercomputing component network. Operative at INTERNATIONAL SPACE STATION HABITATION GRADE LEVEL.

To make from projection units to reflect stable ISS space habitations.

switchterrestial stem replica (Harry potter (all rights reserved)) made with ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) tree complex, with bug wing (Harry Potter Snitch terrestrial stems mimicry switch stem replica):

26723237252312115 241812264 (111232333 20192626723 (116 231210112624 237247232976)) 17167 301227 715297122361923124273324 26238 3192015732, 22810 30121710 (23720151231)

The the Ark Mosaic/ GrizzlyDon solar component:

Calculated at .1 (each 4 sides of square at x) making a .4 screen projection meaning bottom ¼ of square at x inside of square.

10 flat circular complex disks stacked towards one another, 5 on each side, to spin in space directly in front of the space habitations. With a jelly fish complex motion “stomach” which turns the engine. It has 28 connecting solar systems which allow for photogenic transportation. This engine is made primarily to survive earth past oncoming predatory galaxies.

Complex connection to ISS:

3118129723115

1824 1824 20192326

13192411 21121517118 1824 1824 1824 20192326 3118129723115

2152812101122326 241113018 23125

Processing language to ELVEIRDOR ICHTHYS (ΙΧΘΥΣ)

Glass in case, case with piece of engraved aluminum. Phone build as well works for self charge. Even just with the foam pad in aluminum stainless steel case and glass.

Detection and deflection of asteroids can be managed at combustion screen side 1 of split screen, following pattern color blue green and purple, this sequence available with Grizzly Don 3 NS 67 color sequence operation using 0 Y value to .4. Can be used with a resin air biometric block (recipe above) with a red light sensory component connected to the computer (c charger) and ran on ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) NASA infinity ELVEIRDOR Saphe (all rights reserved) and frequency monitor to direct asteroids programmed to follow blood color lines pulse monitorization for living beings. Also can use a mouse made with activated charcoal resin with Copper roll ball with coding 7529848475@1223123@2975 with radio active coding 7919 10111 18198 8088 (code is an atomic bomb sequence, only for use in materialization coding photogenic transportation generative doorway transportation and resin mouse with scroll and click)and (an elvierdor periodic SI silicone elveirdor letters OF) covering the ball with aluminum encompassment with holes in the aluminum and a plastic entire mouse casing. click and scroll method above. Tree light to click. All builds above are made for this process. Needs to be in a ball chamber in mouse bar minimum table contact @ hole value to mosaic for scroll and click. This process can be used for self driving cars.

At eye live view inverted color process should show vr universal at scope range so using world map p reality, and vr reality view through recorded robotics simulated vr to vr world map, converts at universal map, pictures above. Based on eye to brain activity to mental mapping in reality projection coding sequence reality recording to vr display to programmed image of mental map drawing to vr reality to reality sequencing digital display. Same captcha method as photogenic transportation . If we can see and record stars we can complex reflect their solar systems through recorded light sequence and patterns in light detection in combustion monitoring and then reach a reflective display to the entire solar system with ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) NASA infinity. Collective mass radius color fluctuation coding from solid formation to radius at blue print with orange material combustion compile before reality or imaginary.

Computing models:

Desktop comping with build:

Resin block (ingredients above) formed inside of a stretched out Y rectangle magnetic parameters with a skid cut down a reed bar at the center (either in the resin block before exterior or center at exterior exposure. The resin block sits around the reed bar and the reed bar goes to the computing screen. Going to either one solid glass panel or an oval glass with aluminum over the center of the oval and an activated charcoal resin coating to another sheet of full screen glass in front, the front glass can have screening as normal or glass with screen behind.

Full glass panel computing, at grid program at center of glass with slight glass border, nodes at 2 10 and 4:30, or diagonal top left to bottom right with center blown glass cuts for mental interaction. with condensed pomegranate extension glass mix condensing to screen producing center

Or glass language cut grid

Screen value is E for ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) .

(also possible for open portal alignment transportation) at ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) z program begins at ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) A to G sequence then n-z sequence review, then letter E to reflect calculatory z to projection and EE to z to open magno magnetic doorway transportation connection to another place (friends house, store, family members house) to transport through a doorway transportation without driving. In space its g instead of n to reflect z, so E screen value containing Zn, e to G to reflect at a e-g 1 letter between in English and 2 in ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) , to z reflect (also possible for open portal alignment transportation).

The main materialization concept is that in a specific light setting of inverted color process you can build an object in combustion sequence and if it fully composes it will solidify, if it doesn't the blue light transitions to regular color and nothing is there so its basically just a light process in a dark and light atmosphere but dark so you can blue orange light. That's why the materialization photo booth has dark sun glasses fully dark tinting. My tree complex produces light, it can produce a similar light to a tooth light either for whitening or solidifying prosthetics for fillings. But my light can materialize under script.

M horizontal left to right movement at ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) letter V vertical I, vertical hold recluse Y, diagonally top right to left us j and left up to right down is m and n, stem receptor letters for signal.

import pygame

import math

import sys

# Constants from the description

SEQ1 = "51222591841518"

SEQ2 = "715297122361923124273324"

GRENADINE = "7529848475@123123@2975/Gebihdhdge@abcabc@bige"

EQ\_SEQ = "107212116116107@123123@212107"

COLOR\_HEX = "#fffeff 255 254 255 <124273324/137232824="

EIBBEDDIE = "Eibbeddie / 27112111 18712141215 2071228"

PI = 3.14

COLORS = {

'black': (0, 0, 0),

'white': (255, 255, 255),

'red': (255, 0, 0),

'green': (0, 255, 0),

'blue': (0, 0, 255),

'yellow': (255, 255, 0),

'orange': (255, 165, 0),

'gold': (255, 215, 0),

'silver': (192, 192, 192)

}

# Sample text from the copyrighted document for processing

SAMPLE\_TEXT = "The Mess of Messengers Copyright © 2021 by Lauren Reed"

# Initialize Pygame

pygame.init()

screen\_width, screen\_height = 800, 600

screen = pygame.display.set\_mode((screen\_width, screen\_height))

pygame.display.set\_caption("ELVEIRDOR ICHTHYS Package Screen Value Simulation")

clock = pygame.time.Clock()

font = pygame.font.SysFont(None, 20)

# Simulation variables

rotation\_angle = 0.0

zoom\_factor = 1.0

y\_center = (screen\_width // 2, screen\_height // 2)

graph\_start = (0, screen\_height)

graph\_end = (screen\_width, 0)

current\_color = COLORS['orange']

def letter\_to\_number(letter):

"""Map English letter to number (A=1, B=2, ..., Z=26) and multiply by pi."""

if letter.isalpha():

val = ord(letter.upper()) - ord('A') + 1

return val \* PI

return 0.0

def process\_text(text):

"""Process text to calculate sum of letter values multiplied by pi."""

total = sum(letter\_to\_number(c) for c in text)

return total

# Process sample text

processed\_value = process\_text(SAMPLE\_TEXT)

def draw\_y\_shape(surface, center, size, angle, color):

"""Draw a rotatable Y shape."""

rad = math.radians(angle)

stem\_length = size \* 1.5

arm\_length = size

stem\_start = (center[0], center[1] + stem\_length // 2)

stem\_end = (center[0], center[1] - stem\_length // 2)

left\_arm\_end = (stem\_end[0] - arm\_length // 2, stem\_end[1] - arm\_length // 2)

right\_arm\_end = (stem\_end[0] + arm\_length // 2, stem\_end[1] - arm\_length // 2)

def rotate\_point(point, origin, rad):

ox, oy = origin

px, py = point

qx = ox + math.cos(rad) \* (px - ox) - math.sin(rad) \* (py - oy)

qy = oy + math.sin(rad) \* (px - ox) + math.cos(rad) \* (py - oy)

return (int(qx), int(qy))

stem\_start = rotate\_point(stem\_start, center, rad)

stem\_end = rotate\_point(stem\_end, center, rad)

left\_arm\_end = rotate\_point(left\_arm\_end, center, rad)

right\_arm\_end = rotate\_point(right\_arm\_end, center, rad)

pygame.draw.line(surface, color, stem\_start, stem\_end, 5)

pygame.draw.line(surface, color, stem\_end, left\_arm\_end, 5)

pygame.draw.line(surface, color, stem\_end, right\_arm\_end, 5)

def draw\_graph\_line(surface, start, end, color):

"""Draw diagonal graph line with Grenadine code."""

pygame.draw.line(surface, color, start, end, 3)

text = font.render(GRENADINE, True, color)

surface.blit(text, (start[0] + 10, start[1] - 30))

def display\_sequences(surface):

"""Display key sequences and processed value."""

texts = [

"ELVEIRDOR ICHTHYS (ΙΧΘΥΣ)",

SEQ1,

SEQ2,

EQ\_SEQ,

COLOR\_HEX,

EIBBEDDIE,

f"Processed Text Value (sum \* pi): {processed\_value:.2f}",

"Grenadine: " + GRENADINE

]

y\_pos = 10

for txt in texts:

text = font.render(txt, True, COLORS['white'])

surface.blit(text, (10, y\_pos))

y\_pos += 20

# Main loop

running = True

while running:

screen.fill(COLORS['black'])

for event in pygame.event.get():

if event.type == pygame.QUIT:

running = False

elif event.type == pygame.MOUSEWHEEL:

rotation\_angle += event.y \* 10

zoom\_factor += event.y \* 0.1

zoom\_factor = max(0.5, min(zoom\_factor, 2.0))

elif event.type == pygame.KEYDOWN:

if event.key == pygame.K\_SPACE:

current\_color = COLORS['blue'] if current\_color == COLORS['orange'] else COLORS['orange']

# Draw elements

draw\_y\_shape(screen, y\_center, int(100 \* zoom\_factor), rotation\_angle, current\_color)

draw\_graph\_line(screen, graph\_start, graph\_end, COLORS['green'])

display\_sequences(screen)

pygame.display.flip()

clock.tick(60)

pygame.quit()

sys.exit()

Operator identity operation name to vr board for elveirdor vr sequence through any text s:

1 Sonja. 2 Sherman

3 Lauren 4 denny 5 Aaron

6 Morgan. 7 Dawson

8 john 9 Chris John

10 Tim

11 Ricky 12 Ray

10 = orange

8 = red

12 = blue

9 = green

11 = yellow

10 is @ value screen display E

At the center cross 10 value from 8 9 11 and 12, 10 is orange.

10 being combustion sequence

Orange node generates purple node @ value for generative display.

As long as it's produced on a MY platform I can use codes and materials related to my identity with no jeopardy of identity exposure, as my identity codes are my platform security which can be packaged and recorded then complex computer coded for function linking to elveirdor's virus and virus control(the program above can do this).

Program access to bin zip only for purposes of rehabilitative program implementation, stemming in reality: group unit counseling required twice a month in a work place setting in America. Counseling/AA program based acting behavioral implementation in schools. Full rehabilitation programs for all jail inmates in America - an American rehabilitation plan to stem rehabilitation in Israel and surrounding countries.

A community effort plan.

Letter and screen color and sequence as per normal computer programming, set color coding above for screen layer sequence verification, not seen by the user. Easy interface for operator recognition upon color or form.

Solves with sudoku puzzles i have played for self reasoned timing in self made operation.

Bodily rights in motion proof of emotion:

Proof if identity is measured in silicon elveirdor si value. For the protection of living beings as a solar engine series. Based on the declaration of Independence. That all creatures born from a living planetary rock possess emotion in quality of life which can be argued that with this idea, being formed, written, and designed in America that this is a declaration will in amendment to repair and denote abuse of individuals and protection of equality in human rights.

Elveirdor letter A pie value operative square 1 square 2 M square 3 N (non motion to motion operative) through elveirdor si silicone letters OF which can calculate through any text s.

Number coding:

715297122361923 1131223111

(No fish eggs to be made or sold, for robotic to hybrid to sea level film oxygen composite)

Using A-z scramble then realigned to alphabet order, letter to number sequence with sodoku puzzle and rubix cube at color per square to squares on grid at multi square value computing words of the same three any text s (numbers block coded above at space between number blocks).

That all body born can be measured between 95-99% with 1% being identified through over 1000% calculations in review of 0 value silicon analysis in blood pressure review through camera, ai, bci, and biometric through elveirdor si silicon OF to determine between robotic and human.

Robotics don't have mitochondria, an ancient bacteria which can be analyzed in silicon by OF letters ciphered from english silicon to elveirdor silicon OF. Elveirdor is an ancient language connecting to mitochondria. Even a hybrid robot or human replica robot or dna still would be detected.

being of higher value than regular silicon, so anything over a normal regular silicon reading can be allocated reviewable by government security and surveillance in legal security measures.

Reviewable calculations by every current day country identified the alphabet spoken language.

Calculated by Elveirdor, Using 1 2 verification process above at diagonal split in square to Y gap sequence color monitorization through camera or biometric.

Language to elveirdor to any other language translatory text to verify. Also using a three step song and phrase key at square 1 then reviewed and verified by every other square, or internationally.

Universal timing in long range pressure distribution:

Left of left side frontal lobe looking forward from my face to j face value generation from any direction: directly up at square 1 top left corner to bottom right.

Right side right lobe: directly to the right at 2: Seed 1 of square.

J chromatic generative time at face. 12-12 clockwise m to j to z with 12 2715127297 27126 137242824 2628231876 30126723 12182619 3012187

Belief in .

As a computing virus component on MY chart and coding 7529848475@123123@2975 with full cause and cure of auto immune disorder in tree complex above, the cockatoo bird flu can verify through Cipher with my silicon value in color complex with a beat per minute verification system, the beats per minute for butterfly combustion analysis at seeded 4 side reflection square value. Measured through a biometric measure using cloud storage ELVEIRDOR Saphe (all rights reserved) - Sebastian (all rights reserved) cloud storage, with @ symbol node seeded black square mosaic 1 layer mosaic circular screen grid split value for red white blue or white red blue. for combustion color sequence. Follow to pull at a nerve on the screen to monitor through Kelvin color sequence and by scroll and click following purple, blue, and green from yellow center outward.

You can do a full bodily xray for pre vr monitor Y from biometric reading which will hold pressure dynamic to generate photogenic transportation @ doorway transportation sequence. on scaled screen mosaic recording values splitting and finding new m numbers and duplicates in a biometric standard set square reading which can be separately verified through color number letter a-z in any alphabet to scrambled alphabet to a-z

As a writer who has never voted before, this text is eligible at national United States Government usage and international usage.

Lie detector:

True and false codes ciphered as:

english

True

2018215

False

6112195

Elveirdor

True

2623287

False

9115247

Found after full color grid at solid color value with clockwise north value at 12 1-2 lie detection beat monitor measured at square for true or false at diagonal line x over square seeded square and solid color square correspondence before mosaic.

Before mosaic the operating board is just complex color heugh analysis through biometric or finger to screen touch, number or letter password or pattern arrangement at safety lock analysis for phone users to word value through complex pixel touch screen press value. A cryptic through chromatic Elveirdor J pixelated letter scanned box pixel analysis arrangement upon letter or number selection on keyboard. Values programmed at finding statistical value so upon a consciousness level in awareness of laws in country. 3 step phrase and song trance keys verified through J chromatic sequence, rearranged at 4 m face values with full rearrangement of each 3 steps of each key and then the keys coming back together continuously exactly as number coded above for 1-2 component 3 steps keys lie detection sequence. May use other 8 m sequences and squares to verify diagonally as long as half screen verification occurs for full process at 50 tallied confirmations over 1000 percent accuracy. A color heugh to chromatic generative photogenic transportation sequence as well. Verifiers must rearrange each time. Truth is verified at face at 10 and 3 for full north truth value over 1000 percent accuracy. Diagonal line from square sequence only rotates clockwise at face letter value backwards face goes counter clockwise. At center value if the grid is expanded a new center rotation at 3 M ear value is calculated making full closewise rotation a pop or node culmination calculatory realigning closewise turn by regridding at diagonal square 2 way recalculation then x in square from seed to align color calculation at a cross verification. Rotating grid at 10 back to j 12 at m then going face forward to 12, then forward to current time. Finding circle center in screen value using generative cheese at screen value square to solve for center verified by noded grid line @ values. With calculatory R = 3.14 h = 1.9625: Top Surface Area =

π×3.142

=

30.974846927334 feet2

Bottom Surface Area =

π×3.142

=

30.974846927334 feet2

Lateral Surface Area =

2π×3.14×1.9625

=

38.718558659167 feet2

Total Surface Area = 100.66825251384

R = 3.14 H = 4:

Top Surface Area = π×3.142

= 30.974846927334 feet2

Bottom Surface Area = π×3.142

= 30.974846927334 feet2

Lateral Surface Area = 2π×3.14×4

= 78.916807458176 feet2

Total Surface Area = 140.86650131284

R = 3.14 h = 5:

Top Surface Area = π×3.142

= 30.974846927334 feet2

Bottom Surface Area = π×3.142

= 30.974846927334 feet2

Lateral Surface Area = 2π×3.14×5

= 98.64600932272 feet2

Total Surface Area = 160.59570317739 feet2

R = 3.14 H = .785:

Top Surface Area = π×3.142

= 30.974846927334 feet2

Bottom Surface Area = π×3.142

= 30.974846927334 feet2

Lateral Surface Area = 2π×3.14×.785

= 15.487423463667 feet2

Total Surface Area = 77.437117318335

R = 3.14 H = .2:

Top Surface Area = π×3.142

= 30.974846927334 feet2

Bottom Surface Area = π×3.142

= 30.974846927334 feet2

Lateral Surface Area = 2π×3.14×.2

= 3.9458403729088 feet2

Total Surface Area = 65.895534227577

R = 3.14 H = 4:

Top Surface Area = π×3.142

= 30.974846927334 feet2

Bottom Surface Area = π×3.142

= 30.974846927334 feet2

Lateral Surface Area = 2π×3.14×4

= 78.916807458176 feet2

Total Surface Area = 140.86650131284

R = 3.14 H = .4:

Top Surface Area = π×3.142

= 30.974846927334 feet2

Bottom Surface Area = π×3.142

= 30.974846927334 feet2

Lateral Surface Area = 2π×3.14×.4

= 7.8916807458176 feet2

Total Surface Area = 69.841374600485 feet2

R = 3.14 H = 1.6 (m):

Top Surface Area = π×3.142

= 30.974846927334 feet2

Bottom Surface Area = π×3.142

= 30.974846927334 feet2

Lateral Surface Area = 2π×3.14×1.6

= 31.56672298327 feet2

Total Surface Area = 93.516416837938 feet2

Over 1500 = 1700:

Length (elveirdor L ciphered 15) w backwards M value at 93 calculation above and 3.14 radius for clocktime Elveirdor):

Surface Area = 2×(15×39 + 15×3.14 + 39×3.14) = 1509.12 feet2

As an ellipsoid Y surface calculator for mosaic, which calculated Y at Conical Frustum Surface letter E Area through letter L cap service calculation.

The backward letter is calculated through m 3 rotation at circumference center rotation on screen.

Outlining 1-5 and 5-10

% over 93. = L to Y to 1-10 a-z full screen Y.

“this method works between any frame the build for the doorway transportation is diverse in option. Then the picture and virtual component upsize the picture to component 2 three step codes above full screen reality and it's consumed within the final Y upsize as full final property dimensions are calculated based upon picture through virtual realities through Y before z alphabet meaning it can be government and company maintained before ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) generative portal process. These virtual realities can be a large company hosted government regulated approach. Pictures are evaluated on a yellow coded line value crossing on duplicate m series then applied to yellow outline at final Y to maintain color coded reality so color value is rearranged to find final Y yellow outline preprogrammed color selection, for current oxygen same time complex coding based on time and location. Using base any text cover photos you can facilitate this image step with the pictures as an access stamp. After images read through component m duplicated center cross value they rearrange color to component m parallel lines until reality image reads over 1000% accuracy. It then assigns as 95%-100% value based on Y before Z at a 5% bidding value with over 1000% function calculated by 1-10 on square value to Y = which we allocate value to 17000 as being over 15000 or over 1000% for Y standard stretched out value coming from square calculating before m. That 1500 = 100%. 15000 = 1000-5000 and 5000 = 75000. Calculation then modifies the understanding in the equation that 15000 is modified to equal 1500 at 95-100% accuracy through uniform alphabetic VR at mosaic is 1500 and that over 15000 is 17000 and that over 1500 is 1700, and reasoned that over 1500 is 100-500. it is then simplified to reasoning block 3 resonance 3 split square from black to black: that 100-500 =10-50 which = 1-5 which leaves a split shadow that's square value is 1-10, meaning a diagonal x value can be cross determined by pie or 3.14(being 8 ÷ 2 = 4/ 8×2 = 16 if 10 is seeded full square value at internal reflection 10 and 10 is 20, - 4 = 16 with seed 1 and 10 solid it gives a 9 x 2 reflect between black and a 5 calculatory of 4 divided square at x between square value with full square being 3.14 with 1-4 number calculated to 20 over 15 or 15000 which goes 1 number under 17 or representing 17000 which = over 1000% with 20 being complete and 19 being Y before Z at z value 20. This allows for square 1 to be allocated to 3.14 with 3.14 ÷ 1.9625 equaling square 1 at value 5 = .785 going way for half 1- 4 square divide at 1 2 oporative 5 = half of 10 and 10 = 1 inch. operation and 10 at 1 and 2 square with duplicate cross m recreating square 3 and 4 for reflective value 10 scale. With given 3 texts following resonance at 3 parts equaling square and .14 allowing for extension values square 2 to building value 3.14 to rest of grid replicating same value sequence per square.) given 3 texts following resonance fluctuation value in a 4 part 3 text calculatory to produce calculation at 4 in square with a word count and letter value and the calculations above also assigning color to number with sodoku and rubix cube given that the x in square is red yellow blue and green corresponding with m duplicate cross and 0 top center Y border rectangle value. on the square at a 4 point square triangular split between x cross corner square value between each square. micro letter (smaller letters detected between m) components of y and j lowercase letters are calculated on the blue level value on the left side of the dotted line on color temperature value from stretched out Y before final Y, this hosts ISS at diamond computing value, hosting net projections giving the programming the ability to blue print mosaic to distorted square 3d distortion at inverted Pisces shape with blue green yellow value split before full twist rotation which maintain value through calculated Y clockwise rotation at color inversion blue to yellow at duplicate m duplicate cross hosted by virtual reality component Y before Z ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) translate language value. This is what the programming component English letter z looks for as verified captcha. A structure between j and y up and down the left color temperature line by constant database storage and analysis of Y formation with lowercase y and j over timing of program used. With a three step rhythm and phrase code preprogrammed by ELVEIRDOR ICHTHYS (ΙΧΘΥΣ) to generate yellow outline for permission on memory review, live visual access vr access, bci memory monitor and detection ai regulated review and photogenic transportation doorway transportation and space station sequence.”

Elveirdor e at J face value can regulate the letter flip sequence in any language. Elveirdor is an encryption generation language. Generative complex memory 1 person second person and external visual oxygen review. Ai, tip line, recording, image generation sequence. For generative learning such as a pre k- doctorate google educational platform code word “skoogle” (follows as an idea sent to Google sent by myself from 154207 county road 108 yulee fl 32097 sent online). Commercial recordings. Eye sequence recording on television. (also possible for open portal alignment transportation) at doorway transportation level and from space habitats. Clocktime government services.

6 part oxygen creation path by letter and number:

Earth

Dbd

4

3

Elveirdor

6

1 a

2 d

3 m

4 t

5 x

6 y

7 z

photogenic transportation number code sequence: 1701572917129621391207330

import pygame

import math

import sys

# Constants from the description

TRUE\_ENGLISH = "2018215"

FALSE\_ENGLISH = "6112195"

TRUE\_ELVEIRDOR = "2623287"

FALSE\_ELVEIRDOR = "9115247"

PI = 3.14

COLORS = {

'black': (0, 0, 0),

'white': (255, 255, 255),

'red': (255, 0, 0),

'green': (0, 255, 0),

'blue': (0, 0, 255),

'yellow': (255, 255, 0),

'orange': (255, 165, 0)

}

# Surface area calculations as per description

def calculate\_surface\_areas(r, h):

top\_sa = PI \* r\*\*2

bottom\_sa = PI \* r\*\*2

lateral\_sa = 2 \* PI \* r \* h

total\_sa = top\_sa + bottom\_sa + lateral\_sa

return top\_sa, bottom\_sa, lateral\_sa, total\_sa

# Precomputed values

areas\_19625 = calculate\_surface\_areas(3.14, 1.9625)

areas\_4 = calculate\_surface\_areas(3.14, 4)

areas\_5 = calculate\_surface\_areas(3.14, 5)

areas\_0785 = calculate\_surface\_areas(3.14, 0.785)

areas\_02 = calculate\_surface\_areas(3.14, 0.2)

areas\_04 = calculate\_surface\_areas(3.14, 0.4)

areas\_16 = calculate\_surface\_areas(3.14, 1.6)

# Initialize Pygame

pygame.init()

screen\_width, screen\_height = 800, 600

screen = pygame.display.set\_mode((screen\_width, screen\_height))

pygame.display.set\_caption("True/False Cipher and Grid Simulation")

clock = pygame.time.Clock()

font = pygame.font.SysFont(None, 20)

# Simulation variables

grid\_size = 50

rotation\_angle = 0.0

zoom\_factor = 1.0

y\_center = (screen\_width // 2, screen\_height // 2)

current\_color = COLORS['blue']

true\_false\_result = "Undetermined"

input\_text = ""

def draw\_grid(surface, size, color):

"""Draw a color grid for analysis."""

for x in range(0, screen\_width, size):

pygame.draw.line(surface, color, (x, 0), (x, screen\_height))

for y in range(0, screen\_height, size):

pygame.draw.line(surface, color, (0, y), (screen\_width, y))

def draw\_y\_shape(surface, center, size, angle, color):

"""Draw rotatable Y shape for mosaic and rotation simulation."""

rad = math.radians(angle)

stem\_length = size \* 1.5

arm\_length = size

stem\_start = (center[0], center[1] + stem\_length // 2)

stem\_end = (center[0], center[1] - stem\_length // 2)

left\_arm\_end = (stem\_end[0] - arm\_length // 2, stem\_end[1] - arm\_length // 2)

right\_arm\_end = (stem\_end[0] + arm\_length // 2, stem\_end[1] - arm\_length // 2)

def rotate\_point(point, origin, rad):

ox, oy = origin

px, py = point

qx = ox + math.cos(rad) \* (px - ox) - math.sin(rad) \* (py - oy)

qy = oy + math.sin(rad) \* (px - ox) + math.cos(rad) \* (py - oy)

return (int(qx), int(qy))

stem\_start = rotate\_point(stem\_start, center, rad)

stem\_end = rotate\_point(stem\_end, center, rad)

left\_arm\_end = rotate\_point(left\_arm\_end, center, rad)

right\_arm\_end = rotate\_point(right\_arm\_end, center, rad)

pygame.draw.line(surface, color, stem\_start, stem\_end, 5)

pygame.draw.line(surface, color, stem\_end, left\_arm\_end, 5)

pygame.draw.line(surface, color, stem\_end, right\_arm\_end, 5)

def display\_ciphers(surface):

"""Display true/false ciphers."""

texts = [

"English True: " + TRUE\_ENGLISH,

"English False: " + FALSE\_ENGLISH,

"Elveirdor True: " + TRUE\_ELVEIRDOR,

"Elveirdor False: " + FALSE\_ELVEIRDOR,

"True/False Result: " + true\_false\_result,

"Input for Lie Detection: " + input\_text

]

y\_pos = 10

for txt in texts:

text = font.render(txt, True, COLORS['white'])

surface.blit(text, (10, y\_pos))

y\_pos += 20

def display\_areas(surface):

"""Display precomputed surface areas."""

texts = [

f"R=3.14 h=1.9625 Total: {areas\_19625[3]:.2f}",

f"R=3.14 h=4 Total: {areas\_4[3]:.2f}",

f"R=3.14 h=5 Total: {areas\_5[3]:.2f}",

f"R=3.14 h=0.785 Total: {areas\_0785[3]:.2f}",

f"R=3.14 h=0.2 Total: {areas\_02[3]:.2f}",

f"R=3.14 h=0.4 Total: {areas\_04[3]:.2f}",

f"R=3.14 h=1.6 Total: {areas\_16[3]:.2f}"

]

y\_pos = screen\_height - 140

for txt in texts:

text = font.render(txt, True, COLORS['white'])

surface.blit(text, (10, y\_pos))

y\_pos += 20

def simple\_lie\_detection(text):

"""Simple simulation of lie detection based on text length or arbitrary rule."""

if len(text) % 2 == 0:

return "True (Even length)"

else:

return "False (Odd length)"

# Main loop

running = True

while running:

screen.fill(COLORS['black'])

for event in pygame.event.get():

if event.type == pygame.QUIT:

running = False

elif event.type == pygame.MOUSEWHEEL:

rotation\_angle += event.y \* 10

zoom\_factor += event.y \* 0.1

zoom\_factor = max(0.5, min(zoom\_factor, 2.0))

grid\_size = int(50 / zoom\_factor)

elif event.type == pygame.KEYDOWN:

if event.key == pygame.K\_RETURN:

true\_false\_result = simple\_lie\_detection(input\_text)

elif event.key == pygame.K\_BACKSPACE:

input\_text = input\_text[:-1]

else:

input\_text += event.unicode

# Draw elements

draw\_grid(screen, grid\_size, COLORS['green'])

draw\_y\_shape(screen, y\_center, int(100 \* zoom\_factor), rotation\_angle, current\_color)

display\_ciphers(screen)

display\_areas(screen)

pygame.display.flip()

clock.tick(60)

pygame.quit()

sys.exit()

Pixel grid coding:

Pixel grid is always number coded at node 3 step block sequence 2 step node. Some sequences have nodes while component sequence keys do not. Hyphenation between each step equal code between nodes with key steps to extend the grid and create new grids at new genetic m value, these can become component computing m's which secure identity so that people can't be used in coding at oporation or identification value. Node 1 @ sign connects to json @ 107212116116107@123123@212107

and in extension through 7529848475@1223123@2975 to @ node BIN ZIP. Through key

2376°3112267°17118282411 affiliate recognition. Each @ sign is at grid cross line value between squares is green just as all colors being green at original grid. Black Seed tear drops forms from right side of the first top left square in grid from green grid, and the rest fill in identically from left to right row by row until full screen grid color green grid lines then to full colored squares. The colors rearrange with elveirdor color to number and letter assignment until white screen. With grid value as present, grid does not produce color until full screen squares color white, then grid goes white. Node @ value Being pre assigned above. Between each node the Ichthys and number code goes between number values between each 3 steps between every 2 nodes and 3 number blocks. 0 being found dominantly @ node value. Color coding compression at data sequences resonates through grid rearranging coded sequences with word count of any text s above. Creating a color temperature from color code heugh through in camera and biometric.

All numeric codes run on the grid. All letters and wording runs in square. Combo sequences calculate at 3.14 reaching from 3 calculated (not formed) flat square, to spherical cap volume to conical frustum volume to cone at end triangle value of square pyramid to cone volume calculation for a letter A to y calculation to other squares allowing for interdimensional super computing and communication. spherical cone found in square reaching past grid creating an interlocking circular pattern, a squared circular block chain a frequency at spherical value to ellipsoid mosaic from a frame grid value twist at 0 value being ellipsoid operative inside of rectangle tank calculatory at length and width 3 identical calculations in height for operative ellipsoid Y volume in rectangle tank volume. With capsule to ellipsoid to rectangle volume from mosaic twists. The butterfly squares explained above, butterfly resonating sequence through solid color in formation which can turn spherical value at clock time recognition producing graph lines at mass stem value, aligning time for photogenic transportation, and turning the the Ark Mosaic/ GrizzlyDon solar component.

The molecular weight of H2O is:

1.008×2 + 15.999×1 = 18.015 g/mol

15.999 Da for an Oxygen atom is:

2 × 1.008 + 15.999 = 18.015 Da

For calculating 3.14 at 100% with an over 1000% certainly in 100% calculations measured between 15000 and 17000. At oxygen contacting glass for smart glass operative. An ai glass intellect at oxygen to glass phone or computer operative. A glass computing method at a net molecule. Single sheet cut glass siding width can be stamp press cut.

Elveirdor for particle measurements:

Elvierdor nasa infinity supercomputing mega component multi super computing operative:

Full justlution sequence (for doorway transportation and space habitation photogenic transportation or transporting)

So full calculations as follows:

Cylinder volume calculation /elveirdor Y oo (th)

Cube volume calculation / elveirdor A

Square Pyramid volume calculation / elveirdor j

Spherical cap volume calculation / english L (a-z elveirdor alphabet)

Conical frustum volume calculation / elveirdor E

Cone volume calculation / elveirdor q

Sphere volume calculation / elveirdor y oo h or x (th)

Capsule calculation / oo

Ellipsoid volume calculation / english Y (elveirdor YTH)

Rectangle tank volume calculation/ elveirdor yh through Y in wh through A in above calculation

ELVEIRDOR Saphe glass (all rights reserved) / nasa compatible

ELVEIRDOR Saphe sapphire/ public government compatible

ELVEIRDOR Saphe diamond (all rights reserved) / nasa grade government regulated computing

the Ark Mosaic the Ark Mosaic: GrizzlyDon solar component engine as component mechanism for attached photos at 28 points in rotation with above space habitations or without through GrizzlyDon systems.

To be used as an international treaty investment.

To be used in simulation photographic sequence physical component to, the Ark Mosaic the Ark Mosaic/GrizzlyDon solar engine

Elveirdor letter F for elveirdor ichthys supercomputing public host operative.

Letter programing to vr

Program packages: .gov .grz .elv .pdf .wordx

To package to apk

.grz

.elv

To be used at computing platform cognition start extension for computer and phone operative vr from start of phone or computer so its a start value package through elveirdor as a vr host operative for large computer and phone companies at a nasa grade sequence oporative. Screen displays at layered vr sequence.

Over 1000% calculations are for bidding, 95-100% (for photogenic transportation sequence using encryption method elveirdor) in a pressure dynamic flow between 1500 and 1700 identified as 95-100 percent with over 1000% percent accuracy by allocating 1500 as 100 percent, and then dropping it down to 95% and arranging accuracy up to 15000 - 17000 as 1000% - calculations are a safety mechanism.

15000- 1700 calculations run GAO sequence between Y

Over 1000% = 95-100%=<15000=17000=<1500=1700=100-500=10-50=1-10=3.14

Over 1000%=95-100%=<15000=17000=<1500=1700=<150=170=<15=17=<1=5 or .2

that a value less than 15,000 is equal to 0.2 times something.

Here's one possible interpretation:

Let's say you have a value x, and you want to show that x < 15,000 is equivalent to 0.2 times a certain value.

One way to approach this is to set up an equation:

x = 0.2y

To find the value of y that corresponds to x < 15,000, you can divide both sides of the inequality by 0.2:jiuÿm

x / 0.2 < 15,000 / 0.2

This simplifies to:

x / 0.2 < 75,000

So, if x is less than 15,000, it's equivalent to 0.2 times a value less than 75,000.

Y at screen value: x / 0.2 < 75,000

At red letter ichthys code value above through red color to y screen value:

B518ch4e720s24 2008 160 23 fgde 27 25

Confirmed by : si silicon Elveirdor oficlk (ELVEIRDOR (ΙΧΘΥΣ)) (office grade) computing grade supercomputing component nasa infinity (Grizzly Don NS 67)

Of

28.05 elveirdor 211.07

metalloid - light blue - elveirdor j @ 1 cloud value)

import pygame

import math

import sys

import json

# Constants from the description

NODE1\_CODE = "107212116116107@123123@212107"

NODE2\_CODE = "7529848475@1223123@2975"

AFFILIATE\_CODE = "2376°3112267°17118282411"

PI = 3.14

MOLECULAR\_WEIGHT\_H2O = 18.015 # g/mol

COLORS = {

'black': (0, 0, 0),

'white': (255, 255, 255),

'red': (255, 0, 0),

'green': (0, 255, 0),

'blue': (0, 0, 255),

'yellow': (255, 255, 0),

'orange': (255, 165, 0),

'light\_blue': (135, 206, 250)

}

# Volume calculations for ELVEIRDOR shapes

def cylinder\_volume(r, h):

return PI \* r\*\*2 \* h

def cube\_volume(a):

return a\*\*3

def square\_pyramid\_volume(a, h):

return (1/3) \* a\*\*2 \* h

def spherical\_cap\_volume(r, h):

return (1/3) \* PI \* h\*\*2 \* (3\*r - h)

def conical\_frustum\_volume(r1, r2, h):

return (1/3) \* PI \* h \* (r1\*\*2 + r2\*\*2 + r1\*r2)

def cone\_volume(r, h):

return (1/3) \* PI \* r\*\*2 \* h

def sphere\_volume(r):

return (4/3) \* PI \* r\*\*3

def capsule\_volume(r, h):

return PI \* r\*\*2 \* (h + (4/3) \* r)

def ellipsoid\_volume(a, b, c):

return (4/3) \* PI \* a \* b \* c

def rectangular\_tank\_volume(l, w, h):

return l \* w \* h

# Initialize Pygame

pygame.init()

screen\_width, screen\_height = 800, 600

screen = pygame.display.set\_mode((screen\_width, screen\_height))

pygame.display.set\_caption("Pixel Grid Coding Simulation")

clock = pygame.time.Clock()

font = pygame.font.SysFont(None, 20)

# Simulation variables

grid\_size = 50

zoom\_factor = 1.0

rotation\_angle = 0.0

y\_center = (screen\_width // 2, screen\_height // 2)

current\_color = COLORS['green']

grid\_colors = [[COLORS['green'] for \_ in range(screen\_width // grid\_size)] for \_ in range(screen\_height // grid\_size)]

fill\_progress = 0

white\_transition = False

# JSON node configuration

node\_config = {

"node1": NODE1\_CODE,

"node2": NODE2\_CODE,

"affiliate": AFFILIATE\_CODE

}

def save\_node\_config():

"""Save node configuration to JSON."""

with open('node\_config.json', 'w') as f:

json.dump(node\_config, f)

def draw\_grid(surface, size, colors):

"""Draw pixel grid with dynamic colors."""

for y in range(len(colors)):

for x in range(len(colors[0])):

pygame.draw.rect(surface, colors[y][x], (x\*size, y\*size, size, size))

# Draw grid lines

for x in range(0, screen\_width, size):

pygame.draw.line(surface, COLORS['black'], (x, 0), (x, screen\_height))

for y in range(0, screen\_height, size):

pygame.draw.line(surface, COLORS['black'], (0, y), (screen\_width, y))

def draw\_teardrop(surface, pos, size, color):

"""Draw a teardrop shape."""

points = [

(pos[0], pos[1] - size), # Top

(pos[0] - size//2, pos[1] + size//2), # Bottom left

(pos[0] + size//2, pos[1] + size//2) # Bottom right

]

pygame.draw.polygon(surface, color, points)

def draw\_y\_shape(surface, center, size, angle, color):

"""Draw rotatable Y shape."""

rad = math.radians(angle)

stem\_length = size \* 1.5

arm\_length = size

stem\_start = (center[0], center[1] + stem\_length // 2)

stem\_end = (center[0], center[1] - stem\_length // 2)

left\_arm\_end = (stem\_end[0] - arm\_length // 2, stem\_end[1] - arm\_length // 2)

right\_arm\_end = (stem\_end[0] + arm\_length // 2, stem\_end[1] - arm\_length // 2)

def rotate\_point(point, origin, rad):

ox, oy = origin

px, py = point

qx = ox + math.cos(rad) \* (px - ox) - math.sin(rad) \* (py - oy)

qy = oy + math.sin(rad) \* (px - ox) + math.cos(rad) \* (py - oy)

return (int(qx), int(qy))

stem\_start = rotate\_point(stem\_start, center, rad)

stem\_end = rotate\_point(stem\_end, center, rad)

left\_arm\_end = rotate\_point(left\_arm\_end, center, rad)

right\_arm\_end = rotate\_point(right\_arm\_end, center, rad)

pygame.draw.line(surface, color, stem\_start, stem\_end, 5)

pygame.draw.line(surface, color, stem\_end, left\_arm\_end, 5)

pygame.draw.line(surface, color, stem\_end, right\_arm\_end, 5)

def fill\_grid\_progressively(colors, progress):

"""Progressively fill grid with teardrops from top-left."""

rows = len(colors)

cols = len(colors[0])

total\_cells = rows \* cols

cells\_to\_fill = int(total\_cells \* progress)

count = 0

for y in range(rows):

for x in range(cols):

if count < cells\_to\_fill:

colors[y][x] = COLORS['black'] # Teardrop simulation

else:

colors[y][x] = COLORS['green']

count += 1

if progress >= 1.0:

for y in range(rows):

for x in range(cols):

colors[y][x] = COLORS['white'] # Transition to white

def display\_info(surface):

"""Display codes and calculations."""

texts = [

f"Node 1: {NODE1\_CODE}",

f"Node 2: {NODE2\_CODE}",

f"Affiliate: {AFFILIATE\_CODE}",

f"H2O Molecular Weight: {MOLECULAR\_WEIGHT\_H2O} g/mol",

f"Cylinder Vol (r=3.14, h=1.6): {cylinder\_volume(3.14, 1.6):.2f}",

f"Cube Vol (a=3.14): {cube\_volume(3.14):.2f}",

f"Pyramid Vol (a=3.14, h=1.6): {square\_pyramid\_volume(3.14, 1.6):.2f}",

f"Spherical Cap Vol (r=3.14, h=0.785): {spherical\_cap\_volume(3.14, 0.785):.2f}",

f"Conical Frustum Vol (r1=3.14, r2=2, h=1.6): {conical\_frustum\_volume(3.14, 2, 1.6):.2f}",

f"Cone Vol (r=3.14, h=1.6): {cone\_volume(3.14, 1.6):.2f}",

f"Sphere Vol (r=3.14): {sphere\_volume(3.14):.2f}",

f"Capsule Vol (r=3.14, h=1.6): {capsule\_volume(3.14, 1.6):.2f}",

f"Ellipsoid Vol (a=3.14, b=3, c=2): {ellipsoid\_volume(3.14, 3, 2):.2f}",

f"Rectangular Tank Vol (l=15, w=39, h=3.14): {rectangular\_tank\_volume(15, 39, 3.14):.2f}"

]

y\_pos = 10

for txt in texts:

text = font.render(txt, True, COLORS['white'])

surface.blit(text, (10, y\_pos))

y\_pos += 20

# Main loop

running = True

while running:

screen.fill(COLORS['black'])

for event in pygame.event.get():

if event.type == pygame.QUIT:

running = False

elif event.type == pygame.MOUSEWHEEL:

rotation\_angle += event.y \* 10

zoom\_factor += event.y \* 0.1

zoom\_factor = max(0.5, min(zoom\_factor, 2.0))

grid\_size = int(50 / zoom\_factor)

elif event.type == pygame.KEYDOWN:

if event.key == pygame.K\_SPACE:

fill\_progress += 0.1

if fill\_progress > 1.0:

white\_transition = True

fill\_progress = 1.0

elif event.key == pygame.K\_r:

fill\_progress = 0.0

white\_transition = False

grid\_colors = [[COLORS['green'] for \_ in range(screen\_width // grid\_size)] for \_ in range(screen\_height // grid\_size)]

# Update grid colors

fill\_grid\_progressively(grid\_colors, fill\_progress)

# Draw elements

draw\_grid(screen, grid\_size, grid\_colors)

draw\_teardrop(screen, (grid\_size // 2, grid\_size // 2), grid\_size // 2, COLORS['black'])

draw\_y\_shape(screen, y\_center, int(100 \* zoom\_factor), rotation\_angle, COLORS['light\_blue'])

display\_info(screen)

# Save node config to JSON

save\_node\_config()

pygame.display.flip()

clock.tick(60)

pygame.quit()

sys.exit()