**RYBE++**

THE OFFICIAL REFERENCE

CONCEPT 1

CONCEPT 0 EXTENSION

MORE BASICS

Unicode:Characters

CONCEPT 2

CONCEPT 1 EXTENSION

VERSION

CONCEPT 3

CONCEPT 2 EXTENSION

MORE INFORMATION CREATION TECHNIQUES

UNICODE TECHNIQUE

=================

Elements

========

Unicode Character

Sequence

========

Hello world!:uncd

|| :: = :

CARDINE TECHNIQUE

=================

Elements

========

0, 1, 2, 3, 4, 5, 6, 7, 8, 9

Sequence

========

254145748544:crdn

|| :: = :

CONCEPT 4

CONCEPT 3 EXTENSION

EXTERNAL STREAM

In place of raw stream (not information):

=========================================

Computer:File \*cfle:/home/qeetell/xyz/source\*

Hypertext \*hypr:http://qeetell.vi/source\*

\*\* = \*

CONCEPT 5

INSTRUCTION SET REAPPLICABILITY

Pause and communicate

CONCEPT 6

CODE GROUPING

**####-####-####-####-####-####-####-####-####-###**

**####-####-####-####-####-####-####-####-####-###**

**####-####-####-####-####-####-####-####-####-###**

**####-####-####-####-####-####-####-####-####-###**

**####-####-####-####-####-####-####-####-####-###**

**####-####-####-####-####-####-####-####-####-###**

**####-####-####-####-####-####-####-####-####-###**

**####-####-####-####-####-####-####-####-####-###**

CONCEPT 5

INSTRUCTION-SET IMPORTATION

instruction w

$-[sequence]![-1/+1] || Seed set

==== sequence: 1st (A), 2nd (B), +Info:[-1/+1], (+Info:[-1/+1])

==== 1st seed: Seed set’s cardinality

instruction x

eftd| yeld [+Info, +InfoXY2]

instruction y

instruction z

eftd| yeld [+InfoXY3, +InfoXY4]

eftd| sspn

eftd| +Info: eais || Execute as an instruction set

eftd| +Info: eais [+Info, +InfoXY2]

eftd| nlin @Inst

@Inst-%-[-1/+1]

**Comment**

+Info: hrdn || Harden

+Info: sftn || Soften

**Unit 1: Thread**

Thread:: Standard and Creation;

---- [and Premier-standard]

Program: Standard and Fragment; One-time and Repeatable;

---- Local and Imported

---- [and Premier-standard]

eofp +Info (--seed--)

---- || Execute as a one-time imported fragment program

mrfp +Info ($Xyz)

---- || Make a repeatable imported fragment program

mtpc +Info (@Xyz)

---- || Make an information type repeatable imported fragment program collection

+Information

$Program

@InformationType

!LastInstruction

Unit 2: Thread Structure

^^^^ 1: RFPS ^^^^

^^^^ 2: RPFP ^^^^

N: @Hello

\*\*\*\* 2: RPFP \*\*\*\*

Xxxx

---- 2: RPFP ----

^^^^ 2: ITYP ^^^^

N: @Gello

\*\*\*\* 2: ITYP \*\*\*\*

^^^^ 3: RPFP ^^^^

T: T

N: $Hello

\*\*\*\* 3: RPFP \*\*\*\*

Xxxx

---- 3: RPFP ----

^^^^ 3: RPFP ^^^^

T: I

N: $Hello

\*\*\*\* 3: RPFP \*\*\*\*

Xxxx

---- 3: RPFP ----

---- 2: ITYP ----

---- 1: RFPS ----

^^^^ 1: LOSS ^^^^

^^^^ 2: LOSP ^^^^

N: @Main1XYZ

S: xxxx

E: xxxx

\*\*\*\* 2: LOSP \*\*\*\*

^^^^ 3: RPFP ^^^^

xxxx

xxxx

---- 3: RPFP ----

^^^^ 3: APRG ^^^^

1^^1

2^^2

2--2

xxxx

xxxx

2--2

2^^2

xxxx

xxxx

2--2

1--1

1==1

---- 3: APRG ----

---- 2: LOSP ----

I: xxxx

==== 2: LOSP ====

^^^^ 2: LOSP ^^^^

N: @Main2XYZ

\*\*\*\* 2: LOSP \*\*\*\*

xxxx

xxxx

xxxx

xxxx

---- 2: LOSP ----

---- 1: LOSS ----

**Unit 2: Premier Program**

Access to connected components

nlin !Inst ||Name last instruction

!Inst [1]

end!

**Unit 2: Instruction Type: Conditional, Fated, Selectional, and Repetitional**

Name: alphabet + alpha numeric and “ ’ ”; max 48 characters

**Unit 4: Repeatability: Repeatable Fragment Program and Information Type**

+Info: type

+@: wpot

$SayHello: exct (--seed--)

@InformationType:$CreateAnInformation1: exct ()

+Info:$ProcessInformation1: exct ()

+Info:$ProcessInformation2: exct

**Unit 5: Imported Program (One-time & Repeatable (Program & Type))**

**Unit 6: Step Packaging**

A^^A

T: This is an algorithm step to be carried out.

D: Description.

I: DrawInData1, DrawInData2

A\*\*A

instruction w

instruction x

B^^B

T: This is an algorithm step to be carried out.

D: Description.

I: DrawInData1, DrawInData2

B\*\*B

instruction w

instruction x

C^^C

T: This is an algorithm step to be carried out.

D: Description.

I: DrawInData1, DrawInData2

C\*\*C

instruction w

instruction x

instruction y

instruction z

C--C

PullOutData1, PullOutData2

C==C

instruction y

instruction z

B--B

PullOutData1, PullOutData2

B==B

instruction y

instruction z

B^^B

T: This is an algorithm step to be carried out.

D: Description.

I: DrawInData1, DrawInData2

B\*\*B

instruction w

instruction x

instruction y

instruction z

B--B

PullOutData1, PullOutData2

B==B

A--A

PullOutData1, PullOutData2

A==A

instruction x

instruction y

instruction z