



SP2025 Week 09 • 2025-03-27

Esoteric Languages

Henry Qiu

Announcements

- Double Jeopardy this weekend
 - DiceCTF 2025 + TAMUctf 2025
 - Free pizza for dinner will be provided!
- Tracer FIRE!
 - Forensics and blue team competition!
 - Space is limited! (first 55 applicants) - we only filled about half so far
 - April 5-6
 - Register at <https://sigpwny.com/tracerfire2025>



ctf.sigpwny.com

sigpwny{+[>,.<]}

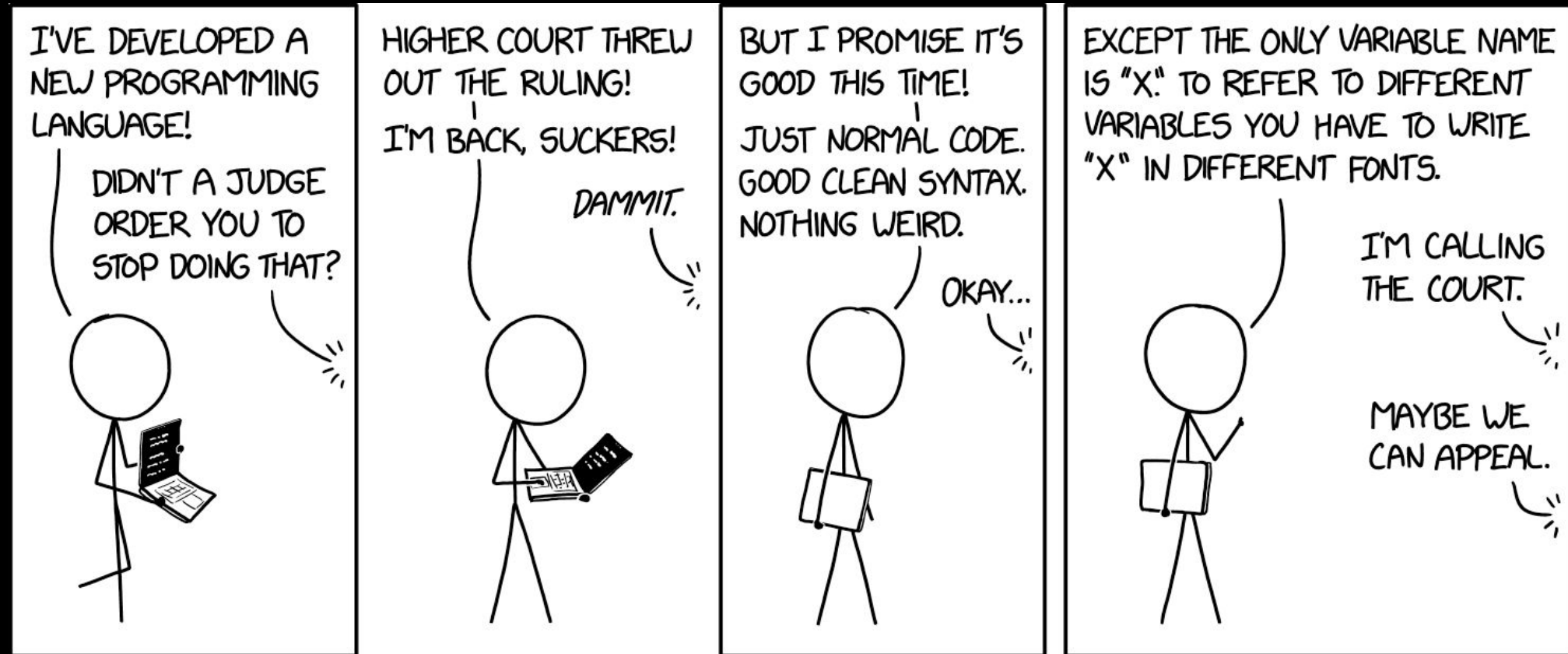


Table of Contents

- What is an esolang?
- Examples
- Additional information
 - Identification
 - Usage in CTF



Esolangs



What is an Esolang?

- People thought it would be funny to make a programming language as a joke
 - Have too much time on their hands - Pete
 - Highest form of programming arts - Henry
- **esolang** : a computer programming language designed to experiment with weird ideas, to be hard to program in, or as a joke, rather than for practical use – esolangs.org



Classic Esolang History

```
DO FORGET expression
ABSTAIN (don't execute the referenc
DO ABSTAIN FROM (label)
or DO ABSTAIN FROM something + some
(as in DO ABSTAIN FROM CALCULATING)
REINSTATE (cancel out an ABSTAIN or
DO REINSTATE (label)
or DO REINSTATE something + somethi
IGNORE (make a variable read-only,
DO IGNORE variable + variable +
REMEMBER (cancel out an IGNORE)
DO REMEMBER variable + variable
```

```
Code: Pseudo code:
>> Move the pointer to cell2
[-] Set cell2 to 0
<< Move the pointer back to cell0
[ While cell0 is not 0
- Subtract 1 from cell0
>> Move the pointer to cell2
+ Add 1 to cell2
<< Move the pointer back to cell0
] End while
```

Number Guessing Game (from 1 to 3)

```
>>v
v1?2v
3
> > >: v
|-&<
$
>"!tcerroC">:v
|,<
@
```

First Esolang: **INTERCAL**

(Compiler Language With No Pronounceable Acronym)

Most Well-Known Esolang: **Brainfuck**

- Simple plus/minus/move operations
- The flag for this meeting is the 'cat' program

2D Esolang: **Befunge**

- Follow the arrows for control flow
- If statements change the direction of the instruction pointer



There are a *lot* of esolangs

- We will be covering the important ones, like AMONGUSISABIGSUSSYBAKAHAHAHAHA
- View a complete list at esolangs.org
- Are not guaranteed to be turing complete

Arch is the best! is a joke language that prints "Arch is the best!" regardless of the program. It has no syntax. It is a joke about the "Arch is the best" [project](#). I use Arch BTW.

fuck ^ v Highlight All Match Case Match Diacritics Whole Words 1 of 164 matches

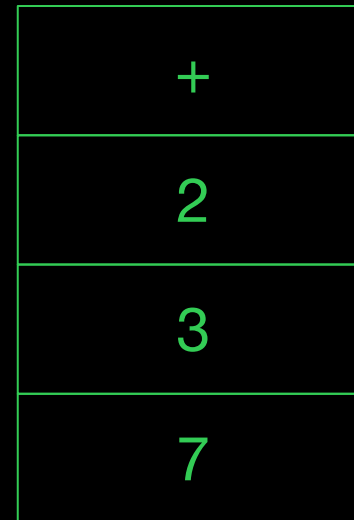
They are also very civil, in case you haven't noticed. /s

- ALPACA
- AIPhAbEt
- AlphaBeta
- Alphabet Stew
- Alphabetti spaghetti
- Alphaprint
- Alphon
- ALPL
- ALT-4
- /æmbi:ɛf/
- Amelia
- AMiaBF'!?
- Among Us
- AMONGUSISABIGSUSSYBAKAHAHAHA
- Amycus
- Amycus Severus
- AnalLang
- Analogia
- Analytical Engine Programming Cards
- And
- Andrei Machine 9000
- Andromeda
- Anemone
- ANGL
- Anguish
- Animosian
- ANItka
- AnnieFlow
- Annihilator
- brainbool
- Braincells
- BrainClub
- Braincopter
- BrainCurry
- BrainCurses
- Brainedumbed
- Brainfact
- brainfault
- BrainfishHQ9+
- Brain-Flak
- Brainflop
- Brainfoctal
- Brainfork
- brainfuckconsole74
- brainfuck
- Brainfuck++
- Brainfuck+10
- Brainfuck+3
- Brainfuck--
- brainfuck 4 humans
- Brainfuck Assembly Language
- Brainfuck But With Buffer
- Brainfuck Encoded Concatenative Calculus
- BrainFuckFart
- Brainfuckn't
- Brainfuck Substitutor
- BrainfuckXT
- Brainfuck/w/index.php?title=Talk:Brainfuck/in
- BrainFuck+
- BrainFuck++
- brainfunc
- brainfunct
- BrainFunge2
- Braingolf

Stack-based Programming

- very simple to implement, most esolangs are stack-based
- uses a stack of values, and performs operations on the values
- Typically does not have memory you can read/write to

$y = 7$
 $x = 2 + 3$



EXECUTE

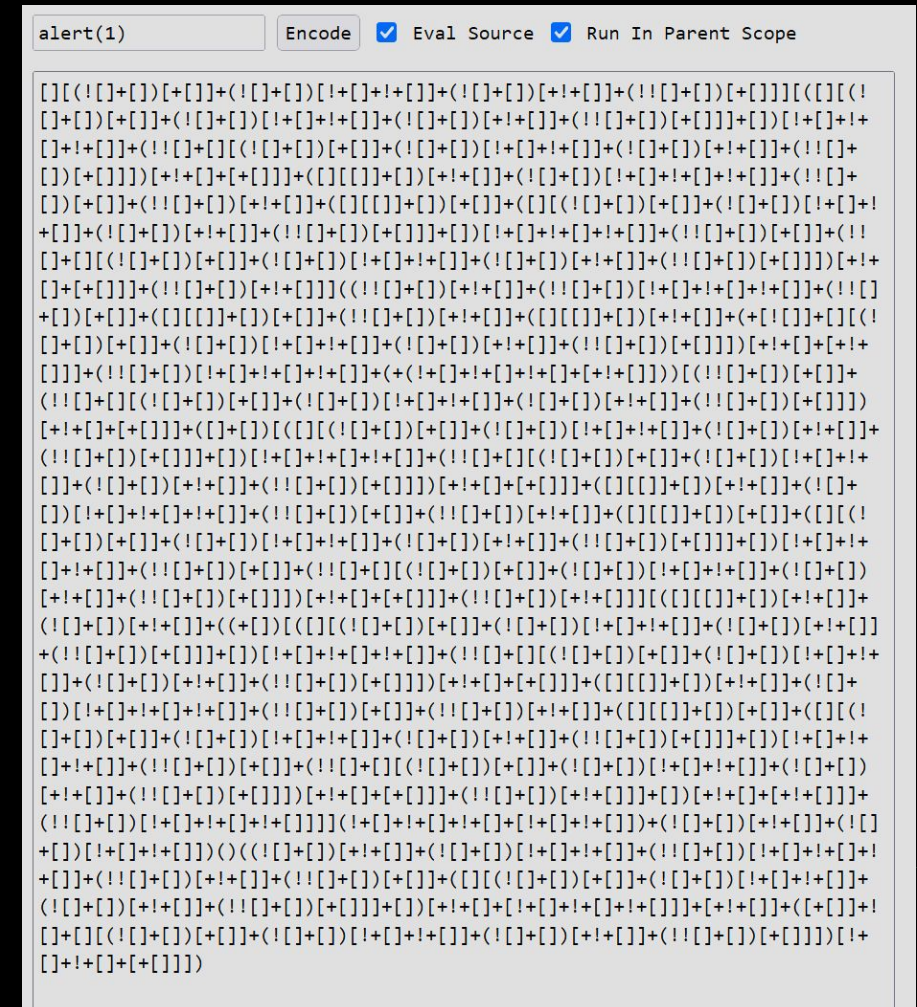


Esolang Examples



JS Fuck

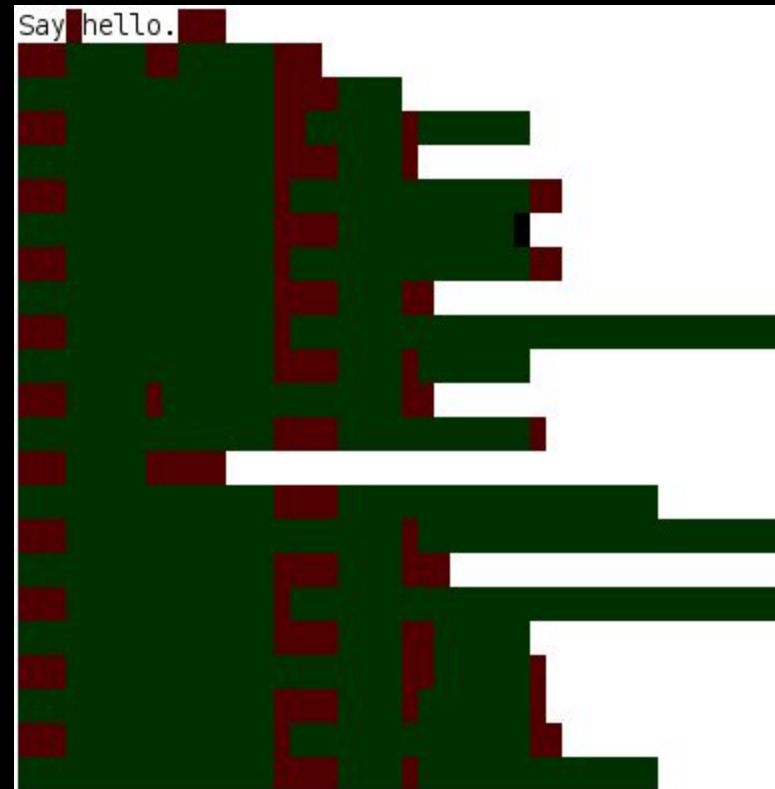
- Esoteric subset of JS
- Common way to obfuscate web / JS challenges
- Uses only 6 characters (`[] () ! +`)
- Abuses JS features like `true + true = 2`, and extract letters from “true”, “false”, etc
- jsfuck.com



The screenshot shows a web-based tool for obfuscating JavaScript code. The input field contains the simple code `alert(1)`. The tool has two checked options: `Eval Source` and `Run In Parent Scope`. The output field displays a highly obfuscated version of the code using only the characters `[] () ! +`. The obfuscated code is a long, dense string of these characters that, when executed, would perform the same `alert(1)` action.

Whitespace

- Each whitespace character (space, tab, linefeed) is a different operator
- Common CTF language
- Stack-Based
- e.g. [Space] = PUSH



Malbolge

- Designed to be impossible (or at least very hard) to program
- First program written by **brute-forcing** a set of constraints
([article](#))
- After each instruction, runs the “Crazy operation”
 - Completely changes the instruction set based on what instruction was just ran

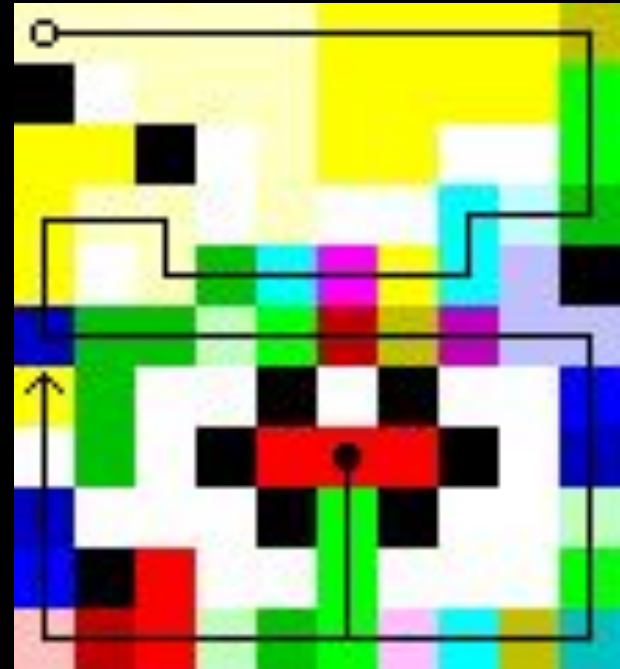
Hello, World! [edit]

This program displays "Hello, World!".^[9]

```
(=<`#9]~6ZY327Uv4-QsqpMn&+Ij" 'E%e{Ab~w=_:]Kw%o44Uqp0/Q?xNvL:`H%c#DD2^WV>gY;dts76qKJImZkj
```

Piet

- 2D stack-based programming language
- Pixel difference correlates to opcode (which operation is run)
- Ran a challenge for UIUCTF 2022 in Piet
 - Challenge: automate reverse-engineering piet programs
 - [Solution](#): trace execution, and reverse program from execution



Shakespeare Programming Language

- Shakespeare ... but you are programming
- **Act / Scene** are GOTO labels
- [...] acts as control flow/operations
- Constants represented by series of words
 - “Positive and neutral nouns have a value of 1 and negative nouns have a value of -1. Any adjective multiplies a noun by 2”

```
A New Beginning.
```

```
Hamlet, a literary/storage device.
```

```
Juliet, an orator.
```

```
Act I: The Only Act.
```

```
Scene I: The Prince's Speech.
```

```
[Enter Hamlet and Juliet]
```

```
Juliet: Thou art the sum of an amazing healthy honest noble peaceful  
fine Lord and a lovely sweet golden summer's day. Speak your  
mind!
```

```
[A pause]
```

```
Juliet: Thou art the sum of thyself and a King. Speak your mind!
```

```
Thou art the sum of an amazing healthy honest hamster and a golden  
chihuahua. Speak your mind!
```

```
[Exeunt]
```



Wenyan-lang 文言

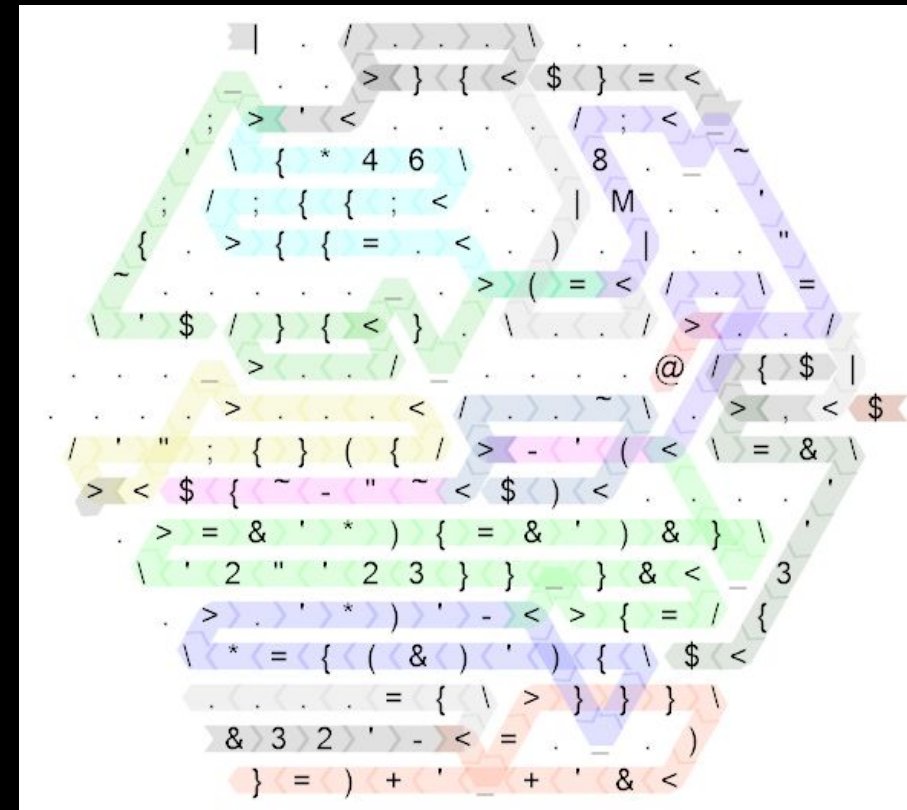
Tower of Hanoi Example

- The alphabet of wenyan contains only traditional Chinese characters and 「」 quotes. (Yes, even numbers and operators.)
- Wenyan is javascript-based, but can be cross-translated into Ruby and Python. It also has an online IDE with crazy examples.
- Go do my [Ancient Scroll of Wisdom challenge!](#)

```
漢諾塔 by examples ▶ Compile
1 吾有一術。名之曰「漢諾塔」。欲行是術。必先得四數。曰「盤數」。曰「甲塔」。曰「乙塔」。曰「丙塔」。
2 乃行是術曰。
3 若「盤數」等於零者。吾有一列。乃得其也。
4 減「盤數」以一。昔之「盤數」者。今其是矣。
5 施「漢諾塔」於「盤數」。於「甲塔」。於「丙塔」。於「乙塔」。名之曰「古」。
6 施「漢諾塔」於「盤數」。於「丙塔」。於「乙塔」。於「甲塔」。名之曰「後」。
7 吾有一列。名之曰「步」。充「步」以「甲塔」。以「乙塔」。
8 吾有一列。名之曰「今」。充「今」以「步」。
9 衝「古」以「今」以「後」。名之曰「史」。乃得「史」。
10 是謂「漢諾塔」之術也。
11
12 吾有一術。名之曰「畫塔法」。欲行是術。必先得一數。曰「盤數」。一列。曰「史」。
13 乃行是術曰。
14 吾有一言。曰「甲乙丙」。名之曰「諸名」。
15 吾有一列。名之曰「三塔」。充「三塔」以「盤數」。以零。以零。
16
17 吾有一術。名之曰「畫」。是術曰。
18 有數一。名之曰「戎」。恆為是。若「戎」大於三者乃止也。
19 夫「三塔」之「戎」。名之曰「磔」。
20 減「盤數」以「磔」。名之曰「柱」。
21 吾有一言。名之曰「行」。
22 為是「磔」過。加「行」以「盤」。昔之「行」者。今其是矣。云云。
23 為是「柱」過。加「行」以「一」。昔之「行」者。今其是矣。云云。
24 夫「諸名」之「戎」。名之曰「名」。
25 吾有四言。曰「「」」。曰「名」。曰「」」。曰「行」。書之。
26 加一以「戎」。昔之「戎」者。今其是矣云云。
27 書之。
28 是謂「畫」之術也。
29
30 凡「史」中之「步」
31 施「畫」噫。
32 夫「步」之一。名之曰「起」
33 夫「步」之二。名之曰「訖」
34 夫「三塔」之「起」。減其以一。昔之「三塔」之「起」者。今其是矣。
35 夫「三塔」之「訖」。加其以一。昔之「三塔」之「訖」者。今其是矣。
36 書之。
37 云云。
38 施「畫」噫。
39 吾有一言。曰「「畢」」。書之。
40 是謂「畫塔法」之術也。
41
42
43 有數四。名之曰「盤數」
44 施「漢諾塔」於「盤數」。於一。於二。於三。名之曰「史」。
45 施「畫塔法」於「盤數」。於「史」。
```

Hexagony

- 2d grid esolang
- [code golfing writeup](#)
- [HexagonyColorer](#)
- [Online interpreter](#)
- Look at these visuals!



Uiua

- a general purpose, stack-based, array-oriented programming language
- Uiua designates special glyphs for all the operations, and it's generally all in one line.
- Code runs from right to left, top to bottom, with only one precedence rule.
- It does support many cool features though, like Multimedia output.



The image displays three examples of the Uiua programming language interface, each showing a line of code and its corresponding output.

Example 1: Code: `⊞<⊞+↑3⊞(÷25)↑240↑80`. Output: A 2D plot showing a wavy pattern with a color gradient from blue to cyan.

Example 2: Code: `÷3/+⊞×⊞×1.5.220×↑&asr`. Output: A multimedia player interface showing a play button, a progress bar at 0:01 / 0:01, and a volume control icon.

Example 3: Code:

```
1 Xy ← ⊞⊞⊞.÷↑100
2 F ← ⊞Δ1⊞(+/÷|÷3+1⊞×↑)Xy
3 ∴F÷↑10
```

. Output: A 3D visualization showing a series of colorful, curved lines or surfaces, possibly representing a mathematical function or data set.



Catala

- a language adapted for socio-fiscal legislative literate programming.
- This was developed because tax code is too complex, so we can apply programming to laws
- At its core, we have conditions and consequences for each law, and each law has a base case and exceptions to it.

(A) \$500,000 Limitation for certain joint returns Paragraph (1) shall be applied by substituting "\$500,000" for "\$250,000" if— (i) either spouse meets the ownership requirements of subsection (a) with respect to such property; (ii) both spouses meet the use requirements of subsection (a) with respect to such property; and (iii) neither spouse is ineligible for the benefits of subsection (a) with respect to such property by reason of paragraph (3).

(B) Other joint returns If such spouses do not meet the requirements of subparagraph (A), the limitation under paragraph (1) shall be the sum of the limitations under paragraph (1) to which each spouse would be entitled if such spouses had not been married. For purposes of the preceding sentence, each spouse shall be treated as owning the property during the period that either spouse owned the property.

US Tax Code, Section 121, (b), (2)

```
scope Section121TwoPersons:
  rule section_121_b_2_A_condition under condition
    (return_type with pattern JointReturn of data_couple) and
    (section121Person1.requirements_ownership_met or
     section121Person2.requirements_ownership_met) and
    (section121Person1.requirements_usage_met and
     section121Person2.requirements_usage_met) and
    (not (section121Person1.section_121_b_3_applies)) and
    (not (section121Person2.section_121_b_3_applies))
  consequence fulfilled
  exception rule section121a_requirements_met under condition
    section_121_b_2_A_condition
  consequence fulfilled
```



Additional information



How to identify an esolang

- ... Pretty hard!
- Google-fu or ChatGPT for series of operations in the code + “esolang”
- Use lists of esolangs online
- Use the list of popular languages on esolangs.org, or the esolangs.org IRC chat



Meeting Challenge - Identify This

```
HAI 1.3
O HAI IM pile
  I HAS A length ITZ 0
  I HAS A max ITZ -1

HOW IZ I pushin YR item
  DIFFRINT ME'Z max AN BIGGR OF ME'Z max AN ME'Z length, O RLY?
    YA RLY, ME HAS A SRS ME'Z length ITZ item, ME'Z max R SUM OF ME'Z max AN 1
    NO WAI, ME'Z SRS ME'Z length R item

  OIC
  ME'Z length R SUM OF ME'Z length AN 1
IF U SAY SO

HOW IZ I popin
  DIFFRINT ME'Z length AN 0, O RLY?
    YA RLY
      ME'Z length R DIFF OF ME'Z length AN 1
      I HAS A item ITZ ME'Z SRS ME'Z length
      ME'Z SRS ME'Z length R NOOB
      FOUND YR item

  OIC
IF U SAY SO

HOW IZ I gettinLen
  FOUND YR ME'Z length
IF U SAY SO
```

KTHX



My interpretations of Esolang

Esoteric form (Looks like an esolang)

Ulua (Fancy alien-like script symbols, but can be very useful in certain applications)	Brainfuck (Looks complex but the operations are fairly simple to understand)	LLVM IR (Not an esolang per se, it's just compiler gobbledygook)
Conway's Game of Life (Technically Turing complete am I right?)	Python Pickle (It's not just a serialization tool. It contains a Turing Complete stack-based VM)	C (Has a bunch of cursed features and functions, and can be as memory-unsafe as you want)
The universe (Really. You can build a computer with anything that has statistical convergence and state transitions. See water computer from Steve Mould)	PowerPoint and Excel (Yes, you can build Turing machines and even 16 bit computers in powerpoint and excel)	Javascript <pre>> "0" == 0 > "0" == [] < true < false > 0 == [] > "0" >= [] < true < true</pre>

Esoteric behavior
(Runs like an esolang)



What did we learn...

- Everything can be an esolang if you try hard enough
- Or rather, everything can be programmed if you try hard enough, you just have to find how to program it.



Next Meetings

2025-03-28 • This Friday

- DiceCTF 2025 + TAMUctf 2025

2025-04-05 • Next Saturday

- Tracer FIRE



ctf.sigpwny.com

sigpwny{+ [>, . <] }

Meeting content can be found at
sigpwny.com/meetings.

