competing standards

VM Specification

the VM consists of

- an unbounded stack manipulated by ops
  - the stack MUST have a type large enough to hold a 32-bit signed integer
- an unbounded call stack manupulated by RET and CALL
  - the call stack MUST have a type large enough to hold a 32-bit signed integer
- an unbounded CTF stack, you should probably keep an eye on it
  - the CTF stack MUST have a type large enough to hold a 8-bit unsigned integer

the binary format (code data) consists of instructions as:

- the operator to evaluate (uint8)
- N arguments (int32 - little endian - low byte to high byte)

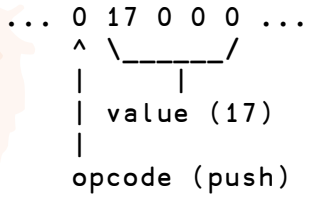
args from code data are represented by [x, y, z], popped values are represented by [a, b, c, ...]:

Ops:

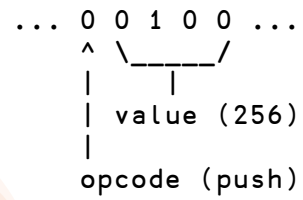
op	name	args/description
0	push	x
1	pop	drop last value from the stack. popping from an empty stack should result in halting the vm.
3	swp	pop `a`, `b`; push `a`; push `b`;
4	sub	pop `a`, `b`; push `a - b`
5	add	pop `a`, `b`; push `a + b`
6	mul	pop `a`, `b`; push `a * b`
7	div	pop `a`, `b`; push `a / b`. a division by zero should not be caught as it's one of the ways to end execution.
8	xor	pop `a`, `b`; push `a xor b`
9	«	pop `a`, `b`; push `a << b`
10	»	pop `a`, `b`; push `a >> b`
11	write	pop `a`; write `a` as a raw byte to stdout. you can assume `a` will be between 0 and 255
12	read	push 1 byte read from standard input
13	je	pop `a`, `b`, `c`; jump to address `a` if `b = c`; push `c`, `b`
14	jne	pop `a`, `b`, `c`; jump to address `a` if `b != c`; push `c`, `b`
15	jlz	pop `a`, `b`; jump to address a if `b < 0`; push `b`
16	call	pop `a`; push (current instruction pointer + 1) to call stack; jump to address `a`
17	goto	pop `a`; jump to address `a`
18	ret	jump to latest call
19	dup	duplicate top value of the stack
20	jempt	pop `a`; jump to address a if stack is empty
21	jnempt	pop `a`; jump to address a if stack is not empty
22	wmem	pop `a`, `b`; write `a modulo 256` to **code memory** at pos `b`.
23	pmem	pop `a`; push byte from code memory at pos `a` to the stack
24	ctf	pop `a`; push `a` to CTF stack

binary format examples

- push 17



- push 256



- goto