

## On the Subject of Variety

*Life is like a box of chocolates. You never know what you're gonna get.*

Begin with a list of the items on the module, sorted into the order shown on the right.

Observe the number displayed at the top of the module.

Decoding this number involves an iterative process. In each iteration, a number  $n$  is relevant. Perform these calculations:

- Take the number modulo  $n$  to obtain a value.
- Divide the number by  $n$  (rounding down) to obtain the number for the next iteration.

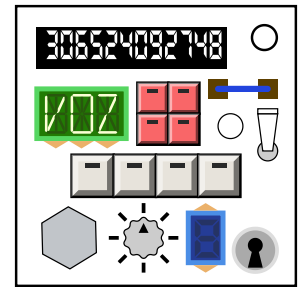
First, apply the iteration with  $n$  equal to the number of items in your list. The value obtained identifies which item to interact with, numbered from 0.

Next, apply the iteration in a way described by the below section for the relevant item. In general,  $n$  is the number of states the item can be in and the value identifies the state the item must be set to.

Afterwards, remove the item from your list. Continue this process, each time with the reduced list of remaining items.

Variety will only issue a strike if an item is interacted with while an item earlier in this process is in an invalid state. Incorrectly cut wires will only cause one strike. All other types of items must be corrected to avoid further strikes.

In the special case where the last item is put into the correct state, but an incorrectly cut wire causes a strike, an item may need to have its state changed and changed back to disarm the module.



- Vertical slider
- Yellow keypad
- 1x3 white keypad (tall)
- Yellow wire
- Blue keypad
- Braille display
- 1x4 white keypad (tall)
- Red switch
- Knob
- 3x4 maze (tall)
- 4x4 maze
- Blue wire
- Horizontal slider
- Red keypad
- 3x3 maze
- Red wire
- 4x1 white keypad (wide)
- Blue button
- Yellow button
- Key-in-lock
- Digit display
- White button
- 4x3 maze (wide)
- Red button
- Blue switch
- White wire
- 3x1 white keypad (wide)
- 2x2 white keypad
- Letter display
- Yellow switch
- White switch
- Black wire
- LED

## Knobs

$n$  is the number of tickmarks. The states are numbered from 0 going clockwise. To find out which tickmark is state 0, start with the up-facing tickmark and move clockwise a number of tickmarks equal to the numeric value of the first character in the serial number (letters are A=1 to Z=26).

## White keypads

$n$  is the factorial of the number of keys. Obtain the order in which to press the keys by applying the iterative process to the obtained value. In each iteration, the value of  $n$  for the nested process is the number of remaining keys, which are numbered from 0 in reading order.

## Colored keypads

$n$  is equal to  $b$  choose  $k$ , where  $b$  is the number of buttons and  $k$  is obtained from the table based on the keys' color.

Color	$k$
Red	2
Yellow	3
Blue	4

You will need to press  $k$  of the buttons. Write down all ways of choosing  $k$  out of  $b$  buttons (with the buttons in reading order) and sort them as if they were binary numbers (1 = press, 0 = no press). The value identifies the combination to press (numbered from 0).

" $b$  choose  $k$ " =  $b! / (k! \times (b - k)!)$ .  
 $!$  is the factorial function.

## Sliders

$n$  is the number of tickmarks. The states are numbered from 0, starting on the left for horizontal sliders and the top for vertical ones.

## LEDs

An LED starts out flashing between two colors. Look up that combination of colors in the following table to obtain a new list of colors.  $n$  is the size of that list. The value identifies a color from that list, numbered from 0.

Tap the LED once to switch to input mode. The LED will cycle all five possible colors. Tap the LED again when the correct color is shown to submit it.

R	Y	B	W	
K B Y R W	K Y W B R	K Y R W	B R W K	K
	Y W K B R	W B	B K Y	R
		R Y K B	Y W B K	Y
			R W	B

B = blue, K = black, R = red, W = white, Y = yellow

## Digit displays

$n$  is the number of prior items in the process excluding wires. The value identifies a prior item, numbered from 0 in the order of the process.

Set the display to the digit it showed in blue when that item was interacted with. Entered digits are shown in yellow.

## Wires

$n$  is 2. If the condition in the following table is false, 1 means “cut” and 0 means “don’t cut”; if it is true, they are reversed.

Wire color	Condition
Black	more battery holders than port plates
Blue	more letters than digits in the serial number
Red	more parallel+serial ports than any other ports
Yellow	more lit than unlit indicators
White	more D batteries than AA batteries

## Buttons

$n$  is  $k + p$ , where  $k$  is obtained from the table based on the button’s color, while  $p$  is the number of corners of the button’s shape.

Color	$k$
Red	2
Yellow	1
Blue	0
White	3

If the obtained value is  $< k$ , hold the button across that many timer ticks. Otherwise, subtract  $k - 2$  from the value and tap the button that many times.

Then wait for two timer ticks for the input to be recognized.

A “timer tick” is a change in the seconds value of the bomb’s countdown timer.

## Letter Displays


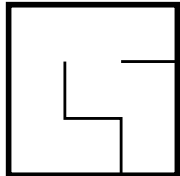
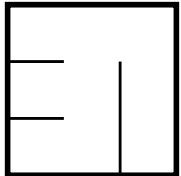
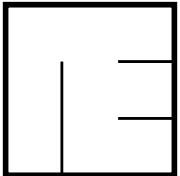

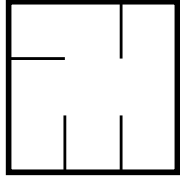
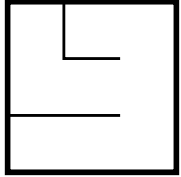
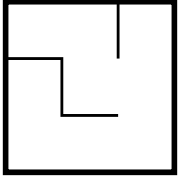

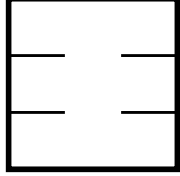
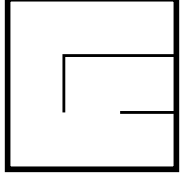
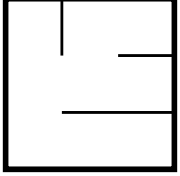
$n$  is the number of words from the below word list that can be formed using the provided options. The words are numbered from 0 in alphabetical order.

ACE BIT DOG FRQ IND LUA NUT QUA SEE TIP  
 ACT BIZ DOT FRY INK LUG OIL QUE SET TOE  
 AID BOB DRY FUN IRK MAD OPT QUO SHE TOO  
 AIM BOW DUE FUR JAM MAN OUR RAG SHY TOP  
 AIR BOY DUG GET JAR MAP OUT RAM SIC TOY  
 ALE BUT DUO GIG JAW MAT OWE RAT SIG TRN  
 ALL BUY DYE GIN JOB MAX OWL RAW SIN TRY  
 AND BYE EAR GUM JOY MAY PAD RED SIR TUB  
 ANT CAN EAT GUT KID MIC PAN RGB SIT VAT  
 APT CAP FAN GUY KIN MID PAR RIB SIX VET  
 ARM CAR FAQ HAM KIT MIX PAT RID SLY WAR  
 ART CAT FAR HAT LAD MOB PAY RIG SND WAX  
 AWE COP FAT HAY LAP MOD PEG RIM SUE WAY  
 AYE COT FAX HEN LAW MUD PEN ROB SUM WEE  
 BAD COW FED HER LAY MUG PER ROD SUN WET  
 BAG CUE FEE HEY LEG MUM PET ROT TAG WHY  
 BAR CUP FEN HIM LET NET PIE ROW TAP WIG  
 BAT CUT FEW HIP LID NEW PIG RUB TAX WIN  
 BAY DAD FIN HIT LIE NIL PIN RUG TEA WIT  
 BED DAM FIT HOP LIP NLL PIT RUM TEE WIZ  
 BEE DAY FIX HOT LIT NOD POP RUN TEN WRY  
 BEG DIE FLY HOW LOG NOR POT SAD TGB YEN  
 BET DIG FOG HUT LOO NOT POW SAW THY YET  
 BID DIM FOR ILK LOT NOW PUB SAY TIE ZAG  
 BIG DIP FRK ILL LOW NUN PUT SEA TIN ZIG


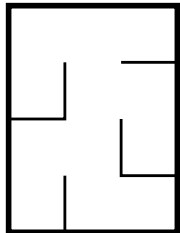
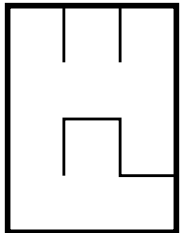
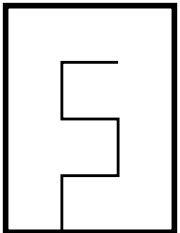

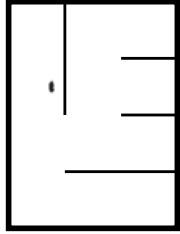
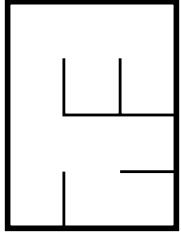
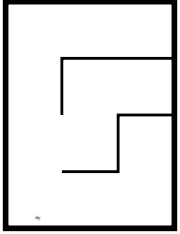

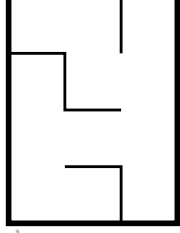
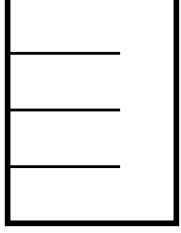
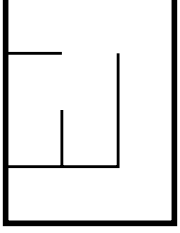
## Braille displays

$n$  is the number of distinct characters in the serial number. The value identifies which of those characters to input in [Braille](#), numbered from 0 based on their first occurrence. Digits are input as 1=A, ..., 9=I, 0=J, except shifted down one row.

**3×3 mazes**

	Red	Yellow	Blue
			
			
			

**3×4 mazes**

	Red	Yellow	Blue
			
			
			

**Mazes**

$n$  is the number of positions in the maze (width times height).

Use the arrow buttons to move the symbol to the goal position identified by the value. The positions are numbered in reading order from 0 in the top-left corner.


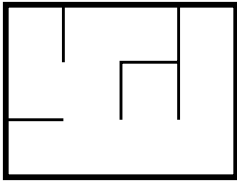
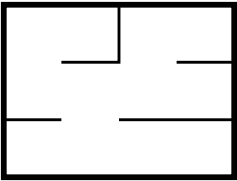
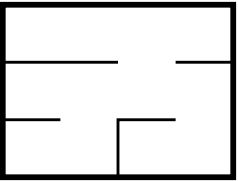

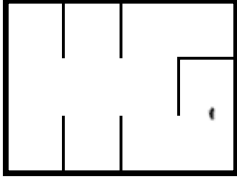
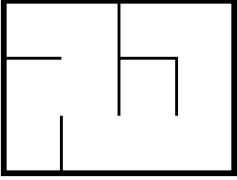
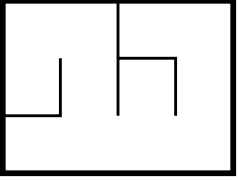

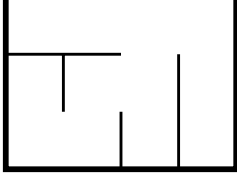
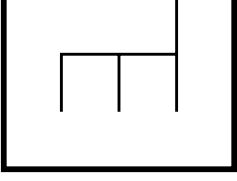
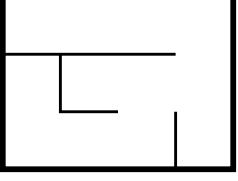
**Keys-in-lock**

$n$  is 10. Turn the key clockwise when the last seconds digit of the bomb's timer equals the value.

**Switches**

$n$  is the number of positions the switch can be toggled to (between 2 and 4). These positions are numbered from 0 starting from the full "up" position.

4×3 mazes

	Red	Yellow	Blue
			
			
			

4×4 mazes

	Red	Yellow	Blue
