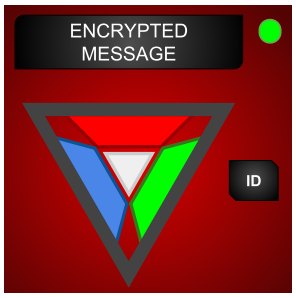
On the Subject of Unfair Cipher



It doesn't play that fair anymore.

This module has two displays. The display on top shows the encrypted message.

The display on the right can be clicked to toggle between showing the Module ID, in white, or strikes the module is keeping track of, in red. Both of these are shown in Roman numerals.

The module encrypts a string of ten three-letter-long instructions with two **Playfair Ciphers**, using different **keys** for each. Enter the correct combination of inputs to disarm the module.

- The order of encryptions is the following: **Original -> Key A Encrypted -> Key C Encrypted -> Caesar Ciphered.** Reverse the order to obtain the **original** instruction string.

Key A

1. Start with the bomb's serial number.
2. Transform each letter into its numerical equivalent (A=1, B=2, etc.).
    - Make a single string of digits.
    - Ignore the first character if its numerical equivalent is 20 or above.
3. Remove the last digit if either the 4th or the 5th characters of the serial are **vowels.**
    - You should only do this once, even in the case both characters are vowels.
4. Convert this number into **hexadecimal.** Refer to *Appendix D3K2H3X* for instructions.
5. Now read the string of hexadecimal digits as a string of decimal digits and letters. Going from left to right, for every digit:
    - If the digit is followed by another digit and they form a number in the range 10-26, convert the pair into its alphabetical equivalent.
    - Otherwise, convert the single digit into its alphabetical equivalent, or skip it if it is a zero.
6. Transform the Module ID, the number of port plates and the number of battery holders into their alphabetical equivalents.
7. Append these three characters at the end of the result of the previous conversion.
8. **This is Key A.**

Key B

Obtain Key B from the following table:

		Month											
		Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
Day	Mon	ABDA	FEV	DBHC	BLD	DBIE	AFEF	AFCG	CQH	DEAI	FEAA	EFAB	DECC
	Tue	ABDB	FEW	DBHD	BLE	DBIF	AFEG	AFCH	CQI	DEAA	FEAB	EFAC	DECD
	Wed	ABDC	FEX	DBHE	BLF	DBIG	AFEH	AFCI	CQA	DEAB	FEAC	EFAD	DECE
	Thu	ABDD	FEY	DBHF	BLG	DBIH	AFEI	AFCA	CQB	DEAC	FEAD	EFAE	DECF
	Fri	ABDE	FEZ	DBHG	BLH	DBII	AFEA	AFCB	CQC	DEAD	FEAE	EFAF	DED
	Sat	ABDF	FEBG	DBHH	BLI	DBIA	AFEB	AFCC	CQD	DEAE	FEAF	EFB	DEDA
	Sun	ABDG	FEBH	DBHI	BLA	DBIB	AFEC	AFCD	CQE	DEAF	FET	EFBA	DEDB

Key C

Use a Playfair Cipher to encode Key A using Key B as the keyword. **This is Key C.**

Refer to *Appendix PL4YF411Z 101* for instructions.

Solving — Step 1: Caesar Cipher

Calculate an offset used for the Caesar Cipher. Start with 0 and perform all operations in the following table for each matching condition.

Condition	Operation
For every **port type**	− 2
For every **port plate**	+ 1
For every **consonant** in the serial number	+ 1
For every **vowel** in the serial number	− 2
For every **lit** indicator	+ 2
For every **unlit** indicator	− 2
For every **battery**	− 1
No batteries	+ 10
No ports	× 2
31 or more modules	÷ 2

In the division operation, drop any remainders.

To decipher the message, shift every letter on the screen forwards by this many letters if the offset is negative, backwards if positive. Wrap back to the other side of the alphabet if you have to go backwards from **A** or forwards from **Z**.

Solving — Step 2: Playfair Ciphers

Use a Playfair cipher with Key C as the keyword to decrypt the string you just deciphered.

Repeat this process once more, but with Key A as the keyword.

You now have the original message.

Solving — Step 3: Executing the Instructions

The message consists of 10 instructions. Execute the instructions left to right.

Tap the small screen on the right to toggle between showing Module ID or strikes. Strikes are shown in red. Both of these are in Roman numerals.

Instructions:

'%' refers to the modulo (remainder) operation.
Inner Center refers to the white button in the middle.
Outer Center refers to the grey triangular frame around the colored buttons.
Refer to *Appendix PR1M3* for a list of prime numbers.

- **PCR:** Press the Red button.
- **PCG:** Press the Green button
- **PCB:** Press the Blue button
- **SUB:** Press **Outer Center** when the seconds digits on the timer match.
- **MIT:** Press **Inner Center** when the seconds on the timer match $(m + c) \% 60$, with m being the Module ID, and c being the number a colored (R, G, B) button has been pressed since the last strike on this module.
  - "MIT" will accept inputs up to 2 seconds early or late.
- **PRN:** Press **Inner Center** if Module ID % 20 is a prime number; otherwise, press Outer Center.
- **CHK:** Press **Outer Center** if Module ID % 20 is a prime number; otherwise, press Inner Center.
- **BOB:** Press **Inner Center.**
  - If there is a lit BOB as the only indicator on the bomb and two batteries in total, this **instantly solves the module!**
- **REP** or **EAT:** Repeat the last input, or press **Inner Center** if this is the first instruction.
- **STR** or **IKE:** Starting from Red (0) at the top, count as many colored buttons clockwise as there are strikes and press the resulting button.

Appendix — D3K2H3X

Follow these steps to convert an integer to hexadecimal:

1. Divide the number by 16. Obtain the remainder and quotient.
2. Convert the remainder into a hexadecimal digit (see table).
3. Repeat these steps with the quotient as the new number. Keep repeating until the quotient is zero.
4. Reverse the order of the hexadecimal digits obtained.
5. Remove any leading zeroes.

DEC	HEX
0 - 9	0 - 9
10	A
11	B
12	C
13	D
14	E
15	F
16	10
26	1A
...	...

Appendix — PR1M3

- Prime numbers (to 20): 2, 3, 5, 7, 11, 13, 17, 19

Appendix — PL4YF4112 101

- Create a 5×5 matrix of letters. Start with your **keyword** and fill the rest with the unused letters of the alphabet. Each letter must occur only **once** in the matrix, so only add the first occurrence. 'J' and 'I' are interchangeable.
- In the following text, use the instructions marked (d) when decrypting and those marked (e) when encrypting.
- Split the **message** into character pairs. If you cannot form a pair, add an **'X'**. If the characters are the **same**, transform the second character into an **'X'**. For each pair:
  - If the letters appear on the same row of your matrix, replace them with the letters to their immediate left (d)/right (e) respectively, wrapping around to the other side of the row if necessary.
  - If the letters are on the same column of your matrix, replace them with the letters immediately above (d)/below (e), wrapping to the other side if necessary.
  - If the letters are on different rows and columns, replace each of them with the letter on the same row but in the column of the other letter in the original pair.
- Drop any final X's that don't make sense and locate any I's that should be J's.