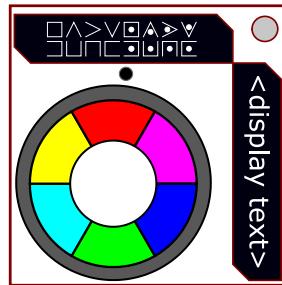


## On the Subject of Unfair's Cruel Revenge

*He made two versions of Unfair's Revenge because he wanted something more unfair than just the base idea... Was it unfair enough in the first place?*

To distinguish this from Unfair Cipher and Unfair's Revenge, the 2 displays on this module are identical with one of them being rotated. The display on top shows the encrypted message. There's also a strip of LEDs underneath which will light up to show the current stage the defuser is on.



The display on the right can be clicked to cycle between showing the Module ID, in white, strikes the module is keeping track of, in red, and the extra key. The Module ID and the strike counter are either shown in fixed or broken Roman numerals, or in Arabic form, I.E 1, 2, 3, 5,...

**For all operations involving STRIKES, always refer to the number in red on the module itself.**

The module encrypts a string of six distant three-letter-long instructions with different ciphers, using different **keys** for each. Enter the correct combination of inputs to disarm the module.

**ALL ciphers referring to the alphabet refers to the A1Z26 standard for each letter unless stated otherwise. Alphabetical order is ALSO modified for the ciphers!**

- The basic order of the given encrypted text is the following: Original  $\rightarrow$  5x Ciphers  $\rightarrow$  Pigpen Ciphered. Reverse the order to obtain the original instruction string.

### Key A

1. Start with the bomb's serial number.
2. Intercept each character as a base-36 digit.
  - If the character is unfamiliar in the set of base-36 digits, intercept this character as the digit "I".
3. Convert this base-36 number into hexadecimal. You can do this by converting the base-36 value into base-10 and then converting that into hexadecimal. Refer to *Appendix HexDex* for instructions to convert base-10 into hexadecimal and *Appendix Base-36 Conversion* for converting base-36 into base-10
4. 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,

Keep Talking and Nobody Explodes Mod

Unfair's Cruel Revenge

6. Transform the Module ID, (1 + the number of port plates), and (2 + the number of battery holders) into their alphabetical equivalents, using step 4 if necessary.
7. Append these 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 using the month and day of the week of when this module was activated:

|     |     | Month |      |      |     |      |      |      |      |      |      |      |      |
|-----|-----|-------|------|------|-----|------|------|------|------|------|------|------|------|
|     |     | Jan   | Feb  | Mar  | Apr | May  | Jun  | Jul  | Aug  | Sep  | Oct  | Nov  | Dec  |
| Day | Mon | ALPH  | ONE  | ABCD | AEI | PLAY | JAKK | FRLA | ZEKN | FIZZ | HEND | CLUT | SCG  |
|     | Tue | BETA  | TWO  | EFGH | OUY | HIDE | MCDU | VIRE | ELIA | TIMW | ACRY | MAGE | BASH |
|     | Wed | CHAR  | THRE | IJKL | WBC | SECR | EOTA | IONL | REXX | MOON | ONYX | SPAR | MOCK |
|     | Thu | DELT  | FOUR | MNOP | DFG | CIPH | CAIT | LEGN | RIVE | TAOO | SAMD | KONQ | BRIN |
|     | Fri | ECHO  | FIVE | QRST | HJK | FAIL | MARA | WILL | TRAI | Lupo | ELUM | FLAM | KANE |
|     | Sat | FOXT  | SIX  | UVWX | LMN | PART | WARI | SKIP | NANT | LUMB | FLUS | MOMO | HEXI |
|     | Sun | GOLF  | SEVN | YZAB | PQR | BECO | PIGD | ETRS | GRYB | CATN | ASIM | MITT | PERK |

### Key C

Use a Playfair Cipher to encode Key A using Key B as the keyword. The alphabet used is the Modern English Alphabet for this encryption. **This is Key C.**

Refer to *Appendix PL4YF4112 101* for instructions.

### Key D

Sum up the value of ALL the false rules in [Alphabetize \(Alphabetize.html\)](#), as if there are no strikes, no solved modules on the bomb, which is determined by the position in the table. Then take the average of the sum. If this results in a decimal answer, use the 15th row on that manual as Key D, otherwise count that many rows from the top and use that as Key D.

## Value X Table

| Condition                                 | Operation                              |
|---|--|
| For each BOB, CAR, or CLR indicator:      | +1 if lit, -1 if unlit                 |
| For each FRK, FRQ, MSA, or NSA indicator: | +2 if lit, -2 if unlit                 |
| For each SIG, SND, or TRN indicator:      | +3 if lit, -3 if unlit                 |
| For the number of batteries:              | +4 for odd, -4 for even                |
| There are port plates with parallel port: | +5 each, -4 if paired with serial port |
| There are port plates with DVI-D:         | -5 each, +4 if paired with Stereo RCA  |

Start with 0 and perform all operations from the table above in this section for each matching condition. If the value is negative, make it positive. This will be referred to as X for decrypting.

## Value A Table

| 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        |

Start with 0 and perform all operations in the following table for each matching condition from top to bottom. This will be referred to as Value A for decrypting.

Drop any remainders and/or decimals when using the division operation. I.E, if you get -1.5 after division, turn -1.5 into -1 instead.

## Solving — Step 1: Figuring Out Which Ciphers Were Used

The table on the right will show all possible ciphers that can be used on Unfair's Cruel Revenge. The original instructions are encrypted by the top-most 5 ciphers after the modifications, from top to bottom, and then encrypted with Pigpen Cipher to be displayed on the module. The modifications are done top to bottom on the table for Encryption

Modifications. The expert will need to grab those 5 ciphers that are used in this module, and then decrypt them by reversing the procedure, I.E if the 5 top-most ciphers are ROT13 Cipher, Atbash Cipher, Autokey Cipher, Anagram Shuffler, and Scytale Transposition from top to bottom, the order to decrypt this is Scytale Transposition, Anagram Shuffler, Autokey Cipher, Atbash Cipher, ROT13 Cipher.

If the defuser has colorblind mode enabled for this module, hovering over the colored button will show the color of that given button for that given position on the top screen.

|                              |
|------------------------------|
| Playfair Cipher (Key C)      |
| Affine Cipher (Value X)      |
| Caesar Cipher (Value A)      |
| Playfair Cipher (Key A)      |
| Playfair Cipher (Key B)      |
| Playfair Cipher (Key D)      |
| Four Square Cipher (Keys ?)  |
| ROT13 Cipher                 |
| Atbash Cipher                |
| Myszkowski Transposition     |
| Autokey Cipher               |
| Scytale Transposition        |
| Basic Columnar Transposition |
| Anagram Shuffler             |
| Redefence Transposition      |

| Encryption Modifications                                     |  |
|--|--|
| Condition  | Action   |
| The module ID is displayed in Broken or Fixed Roman Numerals | Put Playfair Cipher (Key D) at the top.  |
| The strike counter is displayed in Arabic numerals           | Put ROT13 Cipher at the top. And then put Scytale Transposition at the bottom.   |
| Unfair Cipher is NOT present                                 | Put Playfair Cipher (Key A), Caesar Cipher (Value A), Playfair Cipher (Key C) at the bottom, maintaining its previous order. |
| Orange Cipher is present                                     | Put Four Square Cipher (Keys ?) at the top.  |
| Alphabetize is NOT present but Reverse Alphabetize is        | Swap Autokey Cipher with Atbash Cipher.  |

| Encryption Modifications (Continued)                              |  |
|---|--|
| Condition   | Action   |
| Cryptography is present   | Take all of the even positioned ciphers out of the list, reverse it, and then place that at the top of the list.   |
| Anagrams is present   | Put Anagram Shuffler at the top.   |
| Word Scramble is present  | Swap the 3rd item from the top with the 3rd item from the bottom.  |
| Black Cipher is present   | Swap Redefence Transposition with the top-most cipher on the list. If the Columnar Transposition key has 3 or fewer numbers, also swap Basic Columnar Transposition with the bottom-most cipher on the list. |
| The NE colored button is Yellow                                   | Put the top-most 4 ciphers at the bottom, maintaining its order.   |
| Red is diametrically opposite to the Cyan                         | Swap ROT13 Cipher with Myszkowski Transposition, and Basic Columnar Transposition with Anagram Shuffler.   |
| Blue and Yellow are both on the upper or lower half of the module | If Blue and Yellow are both on the upper half, Take the bottom-most cipher in the list and place it at the top. Otherwise, take the top-most cipher in the list and place it at the bottom.                  |
| Unfair's Revenge is present                                       | Cycle all the ciphers up by the number of steps it takes to get to Cyan on this module, starting on the NW button and going clockwise. (0 steps to get to Cyan is <b>not</b> 6 for this.)                    |
| Value X is $13n + 6$  | Swap Affine Cipher (Value X) with Atbash Cipher and then remove Affine Cipher (Value X) from the table.  |
| Value A is $26n$  | Swap Caesar Cipher (Value A) with ROT13 Cipher and then remove Caesar Cipher (Value A) from the table.   |

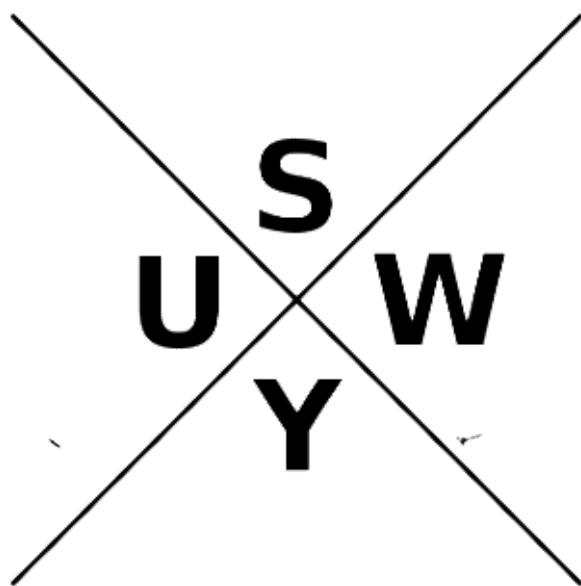
## Solving — Step 2: Pigpen Cipher

Convert the symbols on the top screen into letters using the chart below.

|   |   |   |
|---|---|---|
| A | C | E |
| G | I | K |
| M | O | Q |

|   |   |   |
|---|---|---|
| B | D | F |
| H | J | L |
| N | P | R |

**S**  
**U**   **W**  
**Y**



**T**  
**V**   **X**  
**Z**



## Solving — Step 3: Modifying the Base Alphabet

Did he put a "Blue Arrows" in "Unfair's Revenge?" Don't tell me he did...

You will need to modify the starting alphabet in order to use the ciphers and transpositions on the next page when it is needed.

1. Start the alphabet with "ABCDEFGHIJKLMNOPQRSTUVWXYZ".
2. Shift the entire alphabet to the right [ $l + \text{the last digit of the serial number}$  (10 if none)] times.
3. Move the last letter of the serial number to the front of the alphabet if there are any letters in the serial number. If that letter is the front, move it to the back instead.
4. Then, perform the extra modifications from the table below. Your alphabet for decrypting will be the string after using the table.

| Condition  | Action  |
|--|---|
| Exactly all of these: Lit BOB, no batteries, no port plates, no unlit indicators, serial number contains a vowel | Throw away your current string from this point and use the starting alphabet instead for decrypting. Skip the rest of the conditions from this table.   |
| Lit BOB is present   | Reverse the entire string.  |
| Odd number of battery holders  | Move the vowels ("A", "E", "I", "O", "U") to the back of the string, keeping the order they appeared. If "W" is also in the serial number, move that letter as well in respect with the other vowels. |
| DVI-D port is present  | Reverse the first half of the string.   |
| Stereo RCA port is NOT present   | Move letters R, C, and A to the very back of the string, in that specific mentioned order.  |
| Even number of batteries   | Take the letters where the current l-start position in the string has exactly 3 or 4 distant factors divisible by its position, reverse that set, and then put it back of the sequence.               |
| "Green Arrows", "The Sphere", or "Yellow Arrows" are present.  | Put "LAZYDOG" at the very front of the string and then remove duplicate letters afterwards.   |
| Duplicates of Unfair's Cruel Revenge are present   | Reverse the 2nd half of the string, and then put "THEQUICK" at the very back of the string, in that specific mentioned order.   |

## Solving — Step ?: Caesar Cipher

To decrypt from Caesar Cipher, shift every letter on the screen forwards by this many letters in the alphabet if the offset is negative, backwards if positive. Wrap back to the other side of the alphabet if you have to go backwards from the first letter in the modified alphabet or forwards from the last letter in the modified alphabet.

## Solving — Step ?: ROT13 Cipher

Each letter is encrypted by using Caesar Cipher with a key of 13 on the given alphabet, wrapping around to the first letter if needed.

To decrypt, simply do Caesar Cipher with a key of 13.

## Solving — Step ?: Affine Cipher

For each letter in the plain text:

- The alphabetic position of that letter is multiplied by  $(2X + 1)$ .
- 26 is subtracted from the product until it falls within the range [1, 26].
- This is the alphabetic position of the encrypted letter.

To decrypt, add 26 to the alphabetical postion of the encrypted letter until it is divisible by  $(2X + 1)$ , then divide it by  $(2X + 1)$  to get the alphabetical position of the unencrypted letter.

The table underneath can be used to quickly decrypt each letter for Affine Cipher if needed. The table uses the Standard English Alphabet order however the alphabet used on the module may be different. X = 0 denotes to the first row in the table and the decrypted letter, X = 1 for the 2nd row for the encrypted letter, X = 2 for the 3rd row, etc.

*The contents in this table are editable, you may modify this in order to use a different alphabet.*

Affine Encryption Table

|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U | V | W | X | Y | Z |
| C | F | I | L | O | R | U | X | A | D | G | J | M | P | S | V | Y | B | E | H | K | N | Q | T | W | Z |
| E | J | O | T | Y | D | I | N | S | X | C | H | M | R | W | B | G | L | Q | V | A | F | K | P | U | Z |
| G | N | U | B | I | P | W | D | K | R | Y | F | M | T | A | H | O | V | C | J | Q | X | E | L | S | Z |
| I | R | A | J | S | B | K | T | C | L | U | D | M | V | E | N | W | F | O | X | G | P | Y | H | Q | Z |
| K | V | G | R | C | N | Y | J | U | F | Q | B | M | X | I | T | E | P | A | L | W | H | S | D | O | Z |

The message to decrypt would result in very ambiguous conditions if using X = 6.

|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| O | D | S | H | W | L | A | P | E | T | I | X | M | B | Q | F | U | J | Y | N | C | R | G | V | K | Z |
| Q | H | Y | P | G | X | O | F | W | N | E | V | M | D | U | L | C | T | K | B | S | J | A | R | I | Z |
| S | L | E | X | Q | J | C | V | O | H | A | T | M | F | Y | R | K | D | W | P | I | B | U | N | G | Z |
| U | P | K | F | A | V | Q | L | G | B | W | R | M | H | C | X | S | N | I | D | Y | T | O | J | E | Z |
| W | T | Q | N | K | H | E | B | Y | V | S | P | M | J | G | D | A | X | U | R | O | L | I | F | C | Z |
| Y | X | W | V | U | T | S | R | Q | P | O | N | M | L | K | J | I | H | G | F | E | D | C | B | A | Z |

## Solving — Step ?: Playfair Cipher (Key X)

Use a Playfair Cipher with Key X as the keyword to decrypt the string you just deciphered.

Refer to *Appendix PL4YF4112 101* for instructions.

## Solving — Step ?: Atbash Cipher

Each letter is encrypted to the alphabetical position of  $(27 - P)$ , where P is the alphabetical position of the unencrypted letter.

To decrypt simply get the alphabetical position of  $(27 - E)$ , where E is the alphabetical position of the encrypted letter.

## Solving — Step ?: Basic Columnar Transposition

The extra key will contain random numbers when rearranged from consecutive numbers. These have been used to encrypt the instructions.

To decrypt follow the process for Columnar Transposition from [Green Cipher \(Green%20Cipher.html\)](#), except with 18 characters from the encrypted text.

## Solving — Step ?: Myszkowski Transposition

These seem very familiar to the last one...

| Keywords |        |        |        |        |        |        |
|----------|--------|--------|--------|--------|--------|--------|
| ARCHER   | ATTACK | BANANA | BLASTS | BURSTS | BUTTON | CANNON |
| CALLER   | CELLAR | DAMAGE | DEFUSE | DEVICE | KABOOM | LETTER |
| LOOPEd   | MORTAR | NAPALM | OTTAWA | PAPERS | POODLE | POOLED |
| RASHES   | RECALL | ROBOTS | SAPPER | SHARES | SHEARS | WIRING |

Take the sum of the serial number digits, modulo it by 28, add 1, and count that many words in reading order from the table to get your keyword, starting on "ARCHER".

The encryption process follows by splitting the unencrypted letters up so that each row is equal to the length of the keyword. The letters are then read in alphabetical order in accordance to the keyword, from left to right, top to bottom. Refer to [this \(\[https://en.wikipedia.org/wiki/Transposition\\\_cipher\]\(https://en.wikipedia.org/wiki/Transposition\_cipher\)\)](https://en.wikipedia.org/wiki/Transposition_cipher) for clarification.

To decrypt, place each letter underneath the keyword, in alphabetical order. In the case of duplicate letters in the keyword, place them from left to right for the current duplicate letter, and then top to bottom until each have been filled. Then read the message in reading order to get the decrypted text. The example of Myszkowski Transposition being used is demonstrated on the next page.

**Example: Myszkowski Transposition**

The decrypted text reads "BOBRAN FASTER THANON EHORSE." The keyword used is "PAPERS" using the Modern English Alphabet for encrypting. The text when encrypted reads "OAHH RTNR BBFS TAEQ AEOS NRNE." Spaces are provided in the quotes for readability for decrypting and encrypting.

| P | A | P | E | R | S |
|---|---|---|---|---|---|
| B | O | B | R | A | N |
| F | A | S | T | E | R |
| T | H | A | N | O | N |
| E | H | O | R | S | E |

**Solving — Step ?: Scytale Transposition**

1. Add 2 to (the number of ports on the bomb modulo 4); this is how many rows you will need to make.
2. Create dashes that match the length of the string, and to match the height of the theoretical cylinder. The dashes should start on the top-left and repeat every X letters, based on the height of the cylinder.

Example, with 3 rows and 11 letters:

|   |   |   |   |   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|---|---|---|---|
| - |   |   | - |   |   | - |   |   | - |   |
|   | - |   |   | - |   |   | - |   |   | - |
|   |   | - |   |   | - |   |   | - |   |   |

3. Fill each dash in reading order with each letter in the string you have encrypted.
4. The read each letter like reading a ribbon wrapped around the cylinder.
5. This results in your decrypted string for Scytale Transposition.

An example to decrypt to "ORANGEJUICE" from "ONJCRGUEAEI" in 3 rows can be demonstrated here.

|   |   |   |   |   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|---|---|---|---|
| O |   |   | N |   |   | J |   |   | C |   |
|   | R |   |   | G |   |   | U |   |   | E |
|   |   | A |   |   | E |   |   | I |   |   |

## Solving — Step ?: Redefence Transposition

*As if Railfence wasn't easy enough.*

You will need to use the Columnar Transposition key from the extra key screen in order to decrypt this.

Create dashes in a zig-zag formation so that the length is equal to the length of the encrypted text and the height is equal to the number of numbers in the given key.

Example, with 3 rows and 11 letters:

|   |   |   |   |   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|---|---|---|---|
| - |   |   |   | - |   |   |   | - |   |   |
|   | - |   | - |   | - |   | - |   | - |   |
|   |   | - |   |   |   | - |   |   |   | - |

Now insert numbers to the left of the formation you made where the top-most row takes the 1st number, the 2nd top-most row takes the 2nd, until each row has a number assigned.

To decrypt:

1. Replace as many dashes on with the first set of letters in the encrypted string on the row with the smallest number.
2. Repeat the previous step with the 2nd, 3rd, ... numbers using the next set of letters until each dash has been replaced.
3. Read the letters on the zig-zag formation you made to get your decrypted string.

An example to decrypt to "FOURHUNDRED" from "FHRUNDORUDE" in 3 rows with the key 132 can be demonstrated here.

|   |   |   |   |   |   |   |   |  |   |  |   |
|---|---|---|---|---|---|---|---|--|---|--|---|
| 1 | - |   |   |   | - |   |   |  | - |  |   |
| 3 |   | - |   | - |   | - |   |  | - |  |   |
| 2 |   |   | - |   |   |   | - |  |   |  | - |

Initial setup with key "132", 11 letters

|   |   |   |   |   |   |   |   |   |  |   |   |
|---|---|---|---|---|---|---|---|---|--|---|---|
| 1 | F |   |   |   | H |   |   | R |  |   |   |
| 3 |   | - |   | - |   | - |   | - |  | - |   |
| 2 |   |   | - |   |   |   | - |   |  |   | - |

Filled in dashes on the row marked "1"

|   |   |   |   |   |   |   |  |   |   |   |  |
|---|---|---|---|---|---|---|--|---|---|---|--|
| 1 | F |   |   |   | H |   |  | R |   |   |  |
| 3 |   | - |   | - |   | - |  | - |   | - |  |
| 2 |   |   | U |   |   | N |  |   | D |   |  |

Filled in dashes on the row marked "2"

|   |   |   |   |   |   |   |   |   |  |   |  |
|---|---|---|---|---|---|---|---|---|--|---|--|
| 1 | F |   |   |   | H |   |   | R |  |   |  |
| 3 |   | O |   | R |   | U |   | D |  | E |  |
| 2 |   |   | U |   |   |   | N |   |  | D |  |

Filled in dashes on the row marked "3", decrypted upon reading zig-zag way

## Solving — Step ?: Autokey Cipher

You're telling 18 letters for 1 cipher!? That's completely out of his mind!

The portion of the extra key will consist of a falsely selected word from the module, [Word Search \(Word%20Search.html\)](#). The actual word used for encrypting is the word in the same cell as the falsely selected word, E.G if the falsely selected word is "TEST", the word used for encrypting is "JINX."

Part of the plain text is added to the right of the base key to form a length equivalent to the length of the plain text.

Then the module is encrypted from a theoretical table called *Tabula Recta*, except using the modified alphabet, instead of the standard alphabet. Each letter of the plain text is used for the column, and each letter of the key is used for the row.

Because of this, the theoretical table from *Tabula Recta* has a glaring flaw that can be easily described as the following for each encrypted letter:

1. Take the index of the letter in the 0-indexed alphabet in the plain text.
2. Add the index of the letter in the 0-indexed alphabet from the key.
3. Keep this number within 0-25 by adding/subtracting 26 and grab that letter of the result in the 0-indexed alphabet.
4. This is your encrypted letter.

To decrypt:

1. Start with the first X characters of the encrypted string where X is the length of the keyword.
2. Decrypt this set by undoing the procedure mentioned on this page to get the first set of letters that was appended to the keyword.
3. Repeat this with the next set of letters in your encrypted string until you have decrypted your entire encrypted string.

An example of the decryption procedure for decrypting to get "APPLES" with the keyword "ATE" can be shown here. The alphabet used in this example is the Standard English Alphabet.

|   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|
| ? | ? | ? | ? | ? | ? | ? |
| A | T | E | ? | ? | ? | ? |
| A | I | T | L | T | H |   |

Initial Setup, may differ from module

|   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|
| A | P | P | ? | ? | ? | ? |
| A | T | E | ? | ? | ? | ? |
| A | I | T | L | T | H |   |

Decrypt letters from keyword provided

|   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|
| A | P | P | ? | ? | ? | ? |
| A | T | E | A | P | P | P |
| A | I | T | L | T | H |   |

Add some decrypted text to keyword

|   |   |   |   |   |   |
|---|---|---|---|---|---|
| A | P | P | L | E | S |
| A | T | E | A | P | P |
| A | I | T | L | T | H |

Decrypt Section → Result of decrypting

Go [here](http://www.crypto-it.net/eng/simple/autokey-cipher.html) (<http://www.crypto-it.net/eng/simple/autokey-cipher.html>) for information about autokey cipher. This link provided is NOT secure and should be entered with caution.

## Solving — Step ?: Anagram Shuffler

Did you know that "Orange" is an anagram of "A Goner?"

| Anagrams |        |        |         |         |         |
|----------|--------|--------|---------|---------|---------|
| TAMERS   | STREAM | MASTER | ARM SET | MRS TEA | MR SEAT |
| BARELY   | BARLEY | BLEARY | LAB RYE | A BERYL | ALB RYE |
| RUDEST   | DUSTER | RUSTED | ED RUST | EDS RUT | DUST RE |
| IDEALS   | SAILED | LADIES | A SLIDE | DEAL IS | SEA LID |

Use the even rows if the pigpen set is at the top on the extra key screen, otherwise use the odd rows. Use the top half of the same table if the Columnar Transposition key is to the left of the falsely selected word used for Autokey Cipher, otherwise use the bottom half.

Then count the anagrams in that row from left to right, by the number of steps it takes to reach to Green, starting on the NW button and going CW if the current button is not Green until landing on Green, plus 1. That will be the base key for the unencrypted string.

If the base key consists of 2 words, swap them if the serial number does not contain a vowel (A, E, I, O, U).

Finally, count the anagrams in the same row from left to right, by the number of steps it takes to reach to Magenta, starting on the NW button and going CW if the current button is not Magenta until landing on Magenta, plus 1. That will be the encryption key.

If the encryption key consists of 2 words, swap them if there are an odd number of battery holders on the bomb.

To decrypt, place each letter in reading order in respect to the length of the encryption key. Then for each letter in the base key, place the exact combination of letters from the encrypted string underneath the corresponding letter for the base key. Finally, read the string underneath the base key in reading order to get your decrypted string.

An example of this is shown here.

| P | R | I | N | C | E |
|---|---|---|---|---|---|
| A | F | A | I | N | T |
| S | O | U | N | D | T |
| H | R | I | C | E | D |

| P | I | N | C | E | R |
|---|---|---|---|---|---|
| A | A | I | N | T | F |
| S | U | N | D | T | O |
| H | I | I | E | D | R |

## Solving — Step ?: Four Square Cipher (Keys ?)

*I think he ran out of ideas for potential reasonable ciphers.*

The extra key consists of 12 random pigpen letters, which are used for square D. Decrypt this by using step 2 from the manual. For the other 3 keys:

1. Grab the first true and last false rule from [Reverse Alphabetize \(Reverse%20Alphabetize.html\)](#), as if the module had no strikes, no solved modules. The strings from the first true and last false rules are used for squares B and C respectively.
2. Then grab the median of the true rules from the same manual with the same criteria. If there is none or it doesn't exist, use the alphabet that was created on step 3. Otherwise, use the string to the right of the median of the true rules. This will need to be built for square A.
3. Substitute the 10th letter in the alphabet being used with the 9th letter in the alphabet for each key, then remove duplicate letters and fill in the rest of the alphabet if necessary at the end of each key.
4. Create 4 5x5 squares in that regard.
5. Rearrange the squares into the following: A on the TL, B on the TR, C on the BL, D for the BR. Refer to the table for this step for the order.

|   |   |
|---|---|
| A | B |
| C | D |

6. Now refer to [Orange Cipher \(Orange%20Cipher.html\)](#) in order to decrypt this. This page will repeat steps from Step 3's Orange Cipher if the link is broken.
7. To decrypt, split the encrypted string into pairs, making sure all letters from the 10th letter in the alphabet are substituted with the 9th letter in the alphabet. Then for each pair:
  - Grab the row and column of the 1st letter in the pair from the TR square.
  - Grab the row and column of the 2nd letter in the pair from the BL square.
  - Use the row position of the 1st letter and the column position of the 2nd letter to get your 1st decrypted letter in the TL square for that pair.
  - Use the column position of the 1st letter and the row position of the 2nd letter to get your 2nd decrypted letter in the BR square for that pair.
9. Concatinate the decrypted pairs to get your decrypted string.

## Solving — Step 8: Executing the Instructions

If at this point the expert has least 1 instruction not in the given list of instructions, the expert will need to redo the decryption process.

Tap the screen on the right to cycle between showing the module ID, the number of strikes obtained so far, which is shown in red, or the extra key.

Again, if the defuser has colorblind mode enabled for this module, hovering over the colored button will show the color of that given button for that given position on the top screen.

### Instructions:

'%' refers to the module (remainder) operation.

**Inner Center** refers to the white button in the middle.

**Outer Center** refers to the gray circular 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.
- **SCC:** Press the Cyan button.
- **SCM:** Press the Magenta button.
- **SUB:** Press **Inner Center** when the seconds digits on the countdown timer match.
- **MOT:** Press **Outer Center** when the last digit on the seconds timer is  $(m + c + (5 - s)) \% 10$ , with  $m$  being the Module ID,  $c$  being the number of times a colored (R, G, B, C, M, Y) button has been pressed since the last strike on this module (or since the beginning if there are no strikes) and  $s$  being the current stage, starting with 1.
- **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**.
- **REP** or **EAT:** Repeat the last input, or press **Inner Center** if this is the first instruction. Ignore timing constraints.
- **STR** or **IKE:** Starting from the last colored button you pressed or **Red** if you have not pressed any colored buttons yet, count as many colored buttons counter-clockwise as there are strikes and press the resulting button. In the case of 0 strikes, press the starting button.
  - For these instructions, refer to the **Strike Counter** on the screen to the right of the module itself.

- **SKP:** Press **Inner Center**. Then press **Outer Center** and **skip the next instruction**. If this instruction is unable to skip the next instruction, or this is the very last instruction, only the **Inner Center** press is needed.
- **PVP or NXP:** Start from the last colored button you pressed (or the NW button if you have not pressed any yet). Go (counter-clockwise if PVP / clockwise if NXP) until you get to a button that is a primary color (R, G, B), then press that button.
- **PVS or NXS:** Start from the last colored button you pressed (or the NW button if you have not pressed any yet). Go (counter-clockwise if PVS / clockwise if NXS) until you get to a button that is a secondary color (C, M, Y), then press that button.
- **OPP:** Press the button that is diametrically opposite to the last button you pressed. If your last button pressed was **Outer Center** then press **Inner Center** and vice versa. Otherwise, if this is the first instruction, press **Outer Center**.
- **FIN or ISH:** This is the only instruction that **CANNOT** be skipped and also the last instruction. Take the number of **Inner Center** and **Outer Center** buttons you pressed up to this point and call it X. Count X buttons clockwise starting from the last colored button you pressed, or the NW button if you have not pressed a colored button yet. Then count Y buttons counter-clockwise from that button, where Y is the number of solved modules on the bomb. Press that button when the last seconds digit on the countdown timer is the least significant digit of the number of unsolved modules on the bomb. The number of solved/unsolved modules will take into account once you press the button.

### Appendix: Base-36 Conversion

The table on the right shows a quick lookup for the base 36 values on the table, in base-10 (DEC). To convert a base-36 number into base-10 (DEC):

1. Add the base-36 value of the first digit.
2. If this is not the last base-36 digit, multiply the current value by 36 and go to the next digit.
3. Add the base-36 value of the current digit you are on.
4. Repeat steps 2 and 3 until you have gone through all base-36 digits.
5. You should now have a number in base 10 or in decimal.

| Base 36 | DEC   | Base 36 | DEC |
|---------|-------|---------|-----|
| 0 - 9   | 0 - 9 | 0       | 24  |
| A       | 10    | P       | 25  |
| B       | 11    | Q       | 26  |
| C       | 12    | R       | 27  |
| D       | 13    | S       | 28  |
| E       | 14    | T       | 29  |
| F       | 15    | U       | 30  |
| G       | 16    | V       | 31  |
| H       | 17    | W       | 32  |
| I       | 18    | X       | 33  |
| J       | 19    | Y       | 34  |
| K       | 20    | Z       | 35  |
| L       | 21    | 10      | 36  |
| M       | 22    | 11      | 37  |
| N       | 23    | ...     | ... |

## Appendix — HexDex

To convert a base-10 number to hexadecimal:

1. Divide the number by 16. Obtain the remainder and quotient.
2. Convert the remainder into a hexadecimal digit. See the corresponding table for a quick reference.
3. Repeat steps 1 and 2 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 leading zeros.

| DEC   | HEX   |
|-------|-------|
| 0 - 9 | 0 - 9 |
| 10    | A     |
| 11    | B     |
| 12    | C     |
| 13    | D     |
| 14    | E     |
| 15    | F     |
| 16    | 10    |
| 17    | 11    |
| 26    | 1A    |
| ...   | ...   |

## Appendix — PRIM3

- A prime number is referred to a number that is only divisible by 1 and itself. 1 is not considered prime even though it is divisible by 1 and itself.
- Prime numbers (to 20): 2, 3, 5, 7, 11, 13, 17, 19

## Appendix — PL4YF4LL2 101

- Create a  $5 \times 5$  matrix of letters. Start with your **keyword** and fill the rest with the unused letters of the alphabet you are using. Each letter must occur only **once** in the matrix, so only add the first occurrence. The 9th and 10th letters of the alphabet being used 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 the 24th letter in the alphabet. For each pair:
  - If the letters are exactly identical, do not modify them at all.
  - 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 of the column 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 (24th letter)'s that don't make sense and locate any (9th letter)'s that should be (10th letter)'s if necessary.

T