Decoding The Thank You Card

James Brunyate

Homies

jamespbrunyate@gmail.com

February 6, 2018

Abstract

1 Introduction

After receiving terrible gifts from the Karneys for Christmas 2017, us squibs decided to send the Karneys a thank you card explaining how grateful we were for the gifts. We were, of course, all lying. The thank you card we sent was in the form of a cryptogram with two or three letters given to start off with.

In late January, after some time had passed, Kaija received an email from Kevin Karney which basically was an encrypted message back to us in the form of Python code... Annoyingly, Kevin knows that only I am capable of discerning a lot of the Python that he was written and so the task falls to me to decode it. It was become an occasional hobby for when I don't feel like doing anything else. I decided to document this in a proper article form on Monday 5th February 2018.

Kevin's code uses a multitude of layers to encrypt any message he decides to send. His code is not case-sensitive, does not contain punctuation and will not function properly if the character 'z' is used. Hence his original message does not contain the character 'z' or any punctuation.

2 Code operations

1. Remove the following letters from the alphabet: B, R, U, N, Y, A, T, E and Z. Form a list of letters in the order "B, R, U, N, Y, A, T, E" and then add on all the remaining letters of the alphabet to obtain "BRUNYATECDFGHIJKLMOPQSVWX". This list of letters shall be called the "matrix".

2. Assign integers from 0 to 24 respectively to each letter in the matrix, convert every letter in the original message to upper-case and remove any spaces in the original message. If there is an odd number of letters in the original message, add on an 'X' to the end of the message.

3. Count the number of letters now in the message. Divide this number by two. This will be our range.

4. For each number 'n' from 0 to 24, find 2*n. Numbering each letter in the message 0, 1, 2, 3, etc. Find the letter corresponding to 2n in the message. Using the numbers corresponding to the matrix, take the letter we just obtained and find it's corresponding number from that series of numbers. These numbers will be denoted by a_1, a_2, a_3, etc.

5. Repeat the procedure for 2n+1 instead of 2n, denoting the final outcome values as b_1, b_2, b_3, etc.

6. Let a_x be the remainder after dividing a_n by 5. This will produce values of a_{x1}, a_{x2}, a_{x3}, etc. Let b_x be the remainder after dividing b_n by 5. This will produce values of b_{x1}, b_{x2}, b_{x3}, etc. Any value subscripted with an x will hence be between 0 and 4 since we are dividing by 5 and the only possible remainders lie between 0 and 4 inclusive.

7. Let a_y be the quotient after dividing b_n by 5. This will produce values of a_{y1}, a_{y2}, a_{y3}, etc. Let b_n be the quotient after dividing b_n by 5. This will produce values of b_{y1}, b_{y2}, b_{y3}, etc.

8. Looking at all the values subscripted with the same number part, if $a_{xn} = b_{xn}$ then $c_{xn}, c_{yn}, d_{xn}, d_{yn} = a_{xn}, b_{yn}, a_{xn}, a_{yn}$. If this is not satisfied but $a_{yn} = b_{yn}$ then $c_{xn}, c_{yn}, d_{xn}, d_{yn} = b_{xn}, a_{yn}, a_{xn}, a_{yn}$. If neither of the previous two conditions are satisfied then $c_{xn}, c_{yn}, d_{xn}, d_{yn} = a_{xn}, b_{yn}, b_{xn}, a_{yn}$.

9. Using your values for c_{xn}, c_{yn}, d_{xn} and d_{yn}, create a pair of letters such that the first letter is the letter in the matrix corresponding to $c_x + 5c_y$ and that the second letter is the letter in the matrix corresponding to $d_x + 5d_y$, adding a space after each pair of letters generated in this process and sticking the next generated pair onto the end. The encrypted code has now been formed.

3 Methods and Results

If we look at the conditional statements described in step 8, we can make estimates as to what the positions of the numbers were in the original message. Unfortunately, there are many a number of possibilities and hence another method was derived which I'll illustrate later. The first pair of letters in Kevin's output was "RG".

Case 1

$R = c_x + 5c_y$ and $G = d_x + 5d_y$
$R = a_x + 5b_y$ and $G = a_x + 5a_y$
Corresponding matrix numbers:
$1 = a_x + 5b_y$ and $11 = a_x + 5a_y$
Eliminating a_x: $11 - 5a_y = 1 - 5b_y$
Hence: $2 = a_y - b_y$
The condition $a_x = b_x$ implies that
$a = ymod(5)$ and $b = ymod(5)$
In the example of RG:
$a - b = 10$
Therefore the original pair ab could be any of FB, KA, QF, or YK.

Case 2

$R = c_x + 5c_y$ and $G = d_x + 5d_y$
$R = b_x + 5a_y$ and $G = a_x + 5a_y$
Corresponding matrix numbers:
$1 = b_x + 5a_y$ and $11 = a_x + 5a_y$
Also $a = 5a_y + a_x$ and $b = 5b_y + b_x$
$1 - b_x = 11 - a_x$
$10 = a_x - b_x$
Since we are dividing by 5,
$a_x \neq b_x$ and $a_y = b_y$ imply
$b \neq a + 5n$ where n is an integer ≥ 0
This means that the arbitrary scenario
$\beta = a_x - b_x$ must provide
$\beta \neq 5n$ where n is an integer ≥ 0
In the case of RG, $a_x - b_x = 10$
which is outside of our range.
Therefore case 2 is invalid here.

Case 3

$R = c_x + 5c_y$ and $G = d_x + 5d_y$
$R = a_x + 5b_y$ and $G = b_x + 5a_y$
$a_x \neq b_x$ and $a_y \neq b_y$
Corresponding matrix numbers:
$1 = a_x + 5b_y$ and $11 = b_x + 5a_y$
If $a_y = 0, a_x \neq 0$. If $b_y = 0, b_x \neq 0$.
In this case, there are too many variables.
For RG, $a - b \neq 10 \neq a_x - b_x \neq a_y - b_y$
So ab can be any pair such that
$a - b \neq a_x - b_x \neq a_y - b_y$

On the 6th February 2018, I looked at the code from more of a programmer's perspective and ran all possible pairs of letters, such as $A, B, C, ..., Y, Z, AA, AB, AC, ..., YX, YY$. The results to this yielded the solution to the code and seemed to match both ways (though not all possibilities were tested). Eg. LX codes to SP and vice versa.

The solution is: "Greetings all you Brunyates. Thank you for your clever message and for the presents. Happy new year to you all from Kevin and Elixabeth."

4 References

The code described in this document was sent to me, James Brunyate, by Kevin Karney and is likely created by him. The code can be found here: http://bit.ly/2E7Pq2Q