FIBONACCI CODING AND A CARD TRICK

By Kiran Ananthpur Bacche
[Author of "**The Magic and Joy of Exploding Dots**" and "**Mathematical Approach to Puzzle Solving**"]

Fibonacci Series is a series of numbers such that each number in the series is a sum of the previous two numbers in the series.

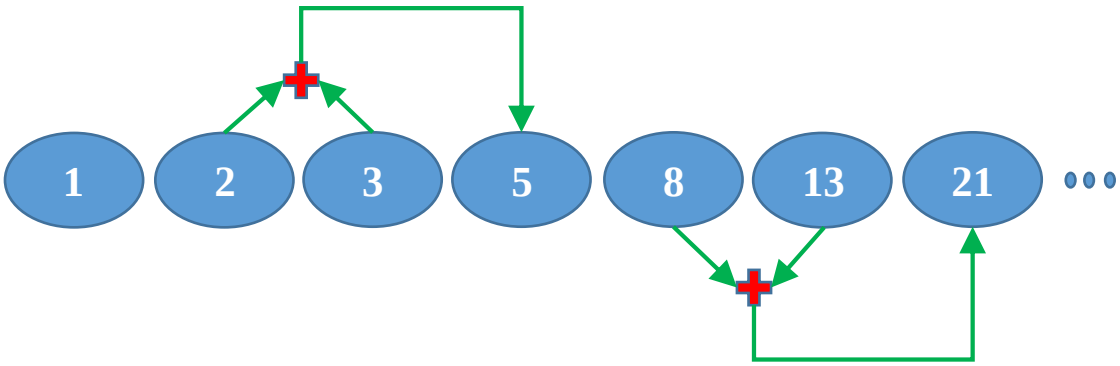

FIGURE 1: Fibonacci Series

A very simple series with amazing and extraordinary characteristics, occurrence and applications. In this article, we are going to cover one such characteristic and the application thereof.

One of the characteristics of Fibonacci series is that "Every positive integer can be represented as a sum of distinct Fibonacci Numbers". As an example, take 16.

16 can be represented as "13" + "3" (3 and 13 are numbers in the Fibonacci Series)

Similarly, let us take 28.

28 can be represented as "21" + "5" + "2" (21, 5 and 2 are numbers in the Fibonacci Series)

Try this out with any positive integer. You will always be able to express that integer as a sum of distinct Fibonacci Numbers.

So far so good. While trying to express an integer in terms of distinct Fibonacci numbers, you would have observed, that for many integers, there are multiple ways of representing that integer using distinct Fibonacci Numbers.

Ex: 9 can be represented as "8"+ "1" (8 and 1 are numbers in the Fibonacci Series)

9 can also be represented as "5" + "3" + "1" (5, 3 and 1 are numbers in the Fibonacci Series)

So in such cases we will choose the one which has non-consecutive distinct Fibonacci Numbers. So in the above example, we will choose to represent 9 as "8" + "1" since 8 and 1 are non-consecutive distinct Fibonacci Numbers.

Note: Are you wondering, if every positive integer can be represented using non-consecutive distinct Fibonacci Numbers? Yes, absolutely. Try it out.

Now we are all equipped to explore the use of this characteristic. Let us begin by representing the integers from 1 to 15 using non-consecutive distinct Fibonacci Numbers as shown in the table below.

Raw Value	Fibonacci Encoded Value (Cells shaded in green)						Fibonacci Code = Fibonacci Encoded Value + "1"
	1	**2**	**3**	**5**	**8**	**13**	
01	**1**	**0**	**0**	**0**	**0**	**0**	**11**
02	**0**	**1**	**0**	**0**	**0**	**0**	**011**
03	**0**	**0**	**1**	**0**	**0**	**0**	**0011**
04	**1**	**0**	**1**	**0**	**0**	**0**	**1011**
05	**0**	**0**	**0**	**1**	**0**	**0**	**00011**
06	**1**	**0**	**0**	**1**	**0**	**0**	**10011**
07	**0**	**1**	**0**	**1**	**0**	**0**	**01011**
08	**0**	**0**	**0**	**0**	**1**	**0**	**000011**
09	**1**	**0**	**0**	**0**	**1**	**0**	**100011**
10	**0**	**1**	**0**	**0**	**1**	**0**	**010011**
11	**0**	**0**	**1**	**0**	**1**	**0**	**001011**
12	**1**	**0**	**1**	**0**	**1**	**0**	**101011**
13	**0**	**0**	**0**	**0**	**0**	**1**	**0000011**
14	**1**	**0**	**0**	**0**	**0**	**1**	**1000011**
15	**0**	**1**	**0**	**0**	**0**	**1**	**0100011**

FIGURE 2: Fibonacci Coding Table

Let us understand how to build this table.

Raw Value → The integer that we are trying to represent using non-consecutive distinct Fibonacci Numbers.

Fibonacci Encoded Value → The representation of the corresponding integer using non-consecutive distinct Fibonacci Numbers. Ex: 9 is represented as "100010". The symbol "1" in the first and fifth position (from the left) denote the 1st Fibonacci Number (which is 1), and the 5th Fibonacci Number (which is 8) respectively. When we add these two numbers, we get 9, which is nothing but the integer we are trying to represent. We can discard the trailing zeroes (all the zeroes after the rightmost "1"). So the Fibonacci encoded value for 9 becomes "10001".

	1	2	3	5	8	13
01	1	0	0	0	0	0
02	0	1	0	0	0	0
03	0	0	1	0	0	0
04	1	0	1	0	0	0
05	0	0	0	1	0	0
06	1	0	0	1	0	0
07	0	1	0	1	0	0
08	0	0	0	0	1	0
09	1	0	0	0	1	0

FIGURE 3: Illustration of Fibonacci Coding Table

Now comes the interesting part. Since we are representing each integer using non-consecutive distinct Fibonacci numbers, the Fibonacci encoded value of any integer will

(a) Have the value of "0" or "1" at any position.
(b) Never have the value of "1" at any two consecutive positions.
Ex: 00110101 would never be a valid Fibonacci encoded value.
(c) Will always end with "1" since we are discarding trailing zeroes.

Fibonacci Code → The final representation of the integers based on the Fibonacci encoded value. We append an additional "1" at the rightmost position of the corresponding Fibonacci encoded value of an integer to get the Fibonacci Code of that integer.

Ex: The Fibonacci encoded value for "9" is "10001". And thus the Fibonacci Code of "9" becomes "10001" + "1" = "100011".

Let us quickly check out the characteristic of the Fibonacci Code

1. Fibonacci Code is of variable length. The length of Fibonacci code for "6" is the LengthOf("10011") = 5, while the length of Fibonacci code for "9" is the LengthOf("100011") = 6.
2. Fibonacci Code always of any integer always ends with "11".

We can exploit the above two characteristics and use Fibonacci Coding for data compression.

Let us take an English sentence "THE SEA IS BLUE". If we have to represent this sentence using ASCII codes, we would need one byte (8 bits) for each character. So we would need 15 bytes (120 bits) to represent this sentence, since this sentence has 15 characters including spaces.

Now let us try to represent this sentence using Fibonacci coding. In an English text, "space" is the most common character followed by the letters "e", "t", "a", "o", and so on. Based on this, let us use the following table (FIGURE 4) to encode this sentence. To keep it simple, I have considered

only the letters that occur in this sentence. In practice, we need to consider all the letters in the English alphabet while encoding a huge block of text.

We map each character in the order of its frequency ranking into an integer starting from 1. Since "<space>" is the most commonly occurring character in an English text, we map it to 1, followed by letter "E" which is mapped to 2, and so on. We finally encode the mapping integer using Fibonacci Coding, and use this to represent the character.

Character	Mapping Integer	Fibonacci Code
<space>	1	**11**
E	2	**011**
T	3	**0011**
A	4	**1011**
I	5	**00011**
S	6	**10011**
H	7	**01011**
L	8	**000011**
U	9	**100011**
B	10	**010011**

FIGURE 4: Fibonacci Code for sample characters.

So our sentence gets encoded into "**00110101101111100110111011110001110011110100110000111000011011**" as illustrated in the table below.

T	H	E		S	E	A		I	S		B	L	U	E
0011	**01011**	**011**	**11**	**10011**	**011**	**1011**	**11**	**00011**	**10011**	**11**	**010011**	**000011**	**100011**	**011**

The number of bits needed are 4 + 5 + 3 + 2 + 5 + 3 + 4 + 2 + 5 + 5 + 2 + 6 + 6 + 6 + 3 = 61 BITS.

With ASCII Encoding, we would need 8 bits for each character, since ASCII is a fixed length code. So we would need 120 BITS to encode the sentence.

Thus we have compressed 120 bits of data into 61 bits of data using Fibonacci Coding.

Wow, isn't that cool?

Note: Because it is a variable length code, the pattern "11" marks the boundary between encoded letters. So you can easily work out the decoding process. Here is a secret message below

Secret Message **= "10110000110000111100110101101111010011011100110011"**

Time to put on the Sherlock Holmes Hat. Can you decode the Fibonacci encoded secret message above to find out the original English sentence?

I'm sure you would have had lots of fun finding out the secret message. Are you ready for the next fun adventure with Fibonacci coding? This time we will create a cool MAGIC Trick with Cards using Fibonacci Coding.

Create 5 Cards as below using a cardboard sheet.

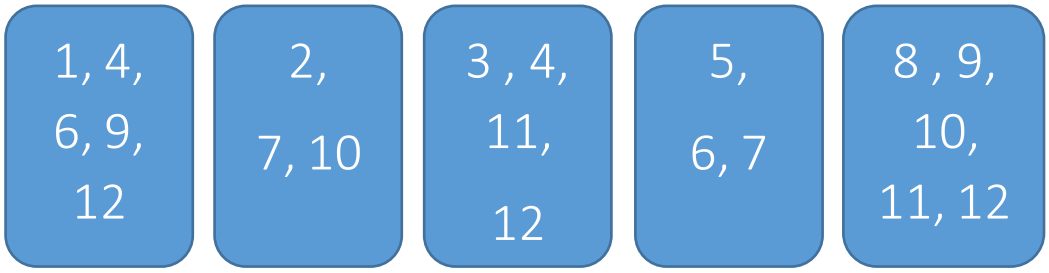

FIGURE 5: Front Side of the 5 Cards.

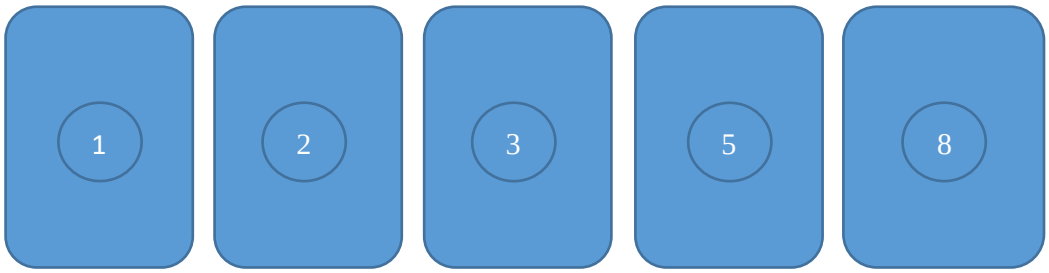

FIGURE 6: Back Side of the 5 Cards.

The magic trick works as follows:

- Pile up all the cards in your left hand, ordered with the first card on the top.
- Ask your friend to think of any secret number between 1 and 12.
    - **Step #1**: Pick up the top card from the pile with the back side facing you, and ask your friend if the secret number is present on the front face of the card.
        - If the answers is YES, then keep that card in your pocket, and then discard the next card from the top of the pile.
        - If the answer is NO, then just discard that card.
    - Repeat **Step #1** till all the cards in the pile are exhausted.
- Add the numbers written behind all the cards you have in your pocket. This would be the secret number your friend has thought of.

Note: Once you master this trick, you will be able to do some optimizations. Ex: If your friend answers NO to the first four cards, then you don't need to check the fifth card. The secret number would be 8.

Can you figure out how this magic works behind the scenes? Time to put on the Mathematician Hat. Enjoy and have fun!!!