Mathematical Investigations

Restoring Order through Repeated Riffle Shuffles

KESHAV LAKSHMI NARASIMHAN

1. Overview

We are all familiar with the riffle shuffle, one of the most popular ways to shuffle cards. To perform this shuffle, divide the deck into two halves, riffle them together with your thumb, interlocking them, and then push the halves together.

This may seem a good way of shuffling; intuition suggests that the cards will get 'more mixed up' every time we shuffle. But is this so? Of course, we humans aren't flawless; we may not make accurate divisions or riffle the same number of cards each time. But what if the shuffle is performed perfectly? This requires dividing the deck exactly in half, riffling the cards so that the cards alternate exactly each time, and always starting the riffle with the same half, either the top (held in the right hand) or the bottom (in the left hand). A question that arises is: *If a deck of cards is repeatedly shuffled using a perfect riffle shuffle, will the cards ever come back to their original order?*

Keywords: Riffle shuffle, card deck, pseudocode, Python code

2. Solution and Proof

The answer turns out to be: **Yes!** If we do the riffle shuffle perfectly, the effect will eventually be undone. To demonstrate this, I wrote a Python code that generates a deck of cards and simulates the perfect riffle shuffle until the deck returns to its original order. The deck is displayed as a list. The pseudocode and the Python code are given in Appendix (**A.1**) available online.

3. How does a Riffle Shuffle Work?

What are the exact mechanics behind such a shuffle? Figure 1 shows how a riffle shuffle works, taking a deck of 8 cards as an example.

As you can see, we first split the deck exactly in half, with the top half in the right hand and bottom half in the left. Next, we riffle them together (release one card at a time, alternating between the two hands), starting with the left hand. Finally, we push the halves together. Note that the top and bottom cards always remain as the top and bottom cards! This method of riffle shuffling is called "out shuffling" and is the method used throughout this article.

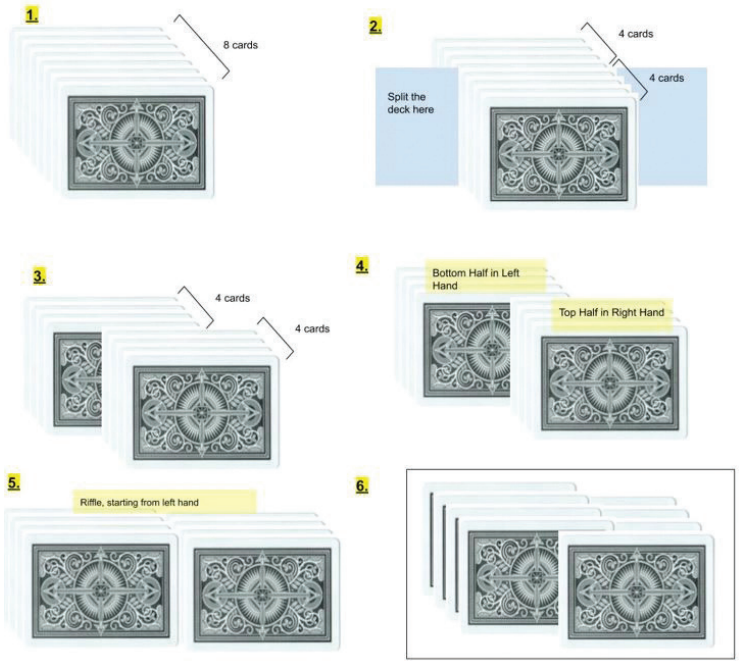

Figure 1: The process of riffle shuffling

To better understand riffle shuffles, here are a few more examples of decks with different numbers of cards that are powers of 2, the number of shuffles required to restore the original order, and the shuffling process.

Each number represents a card and each step in the shuffling process is denoted by <Cards to be held in right hand> : <Cards to be held in left hand>. Figure 2 illustrates this syntax more clearly, taking a deck of 4 cards as an example.

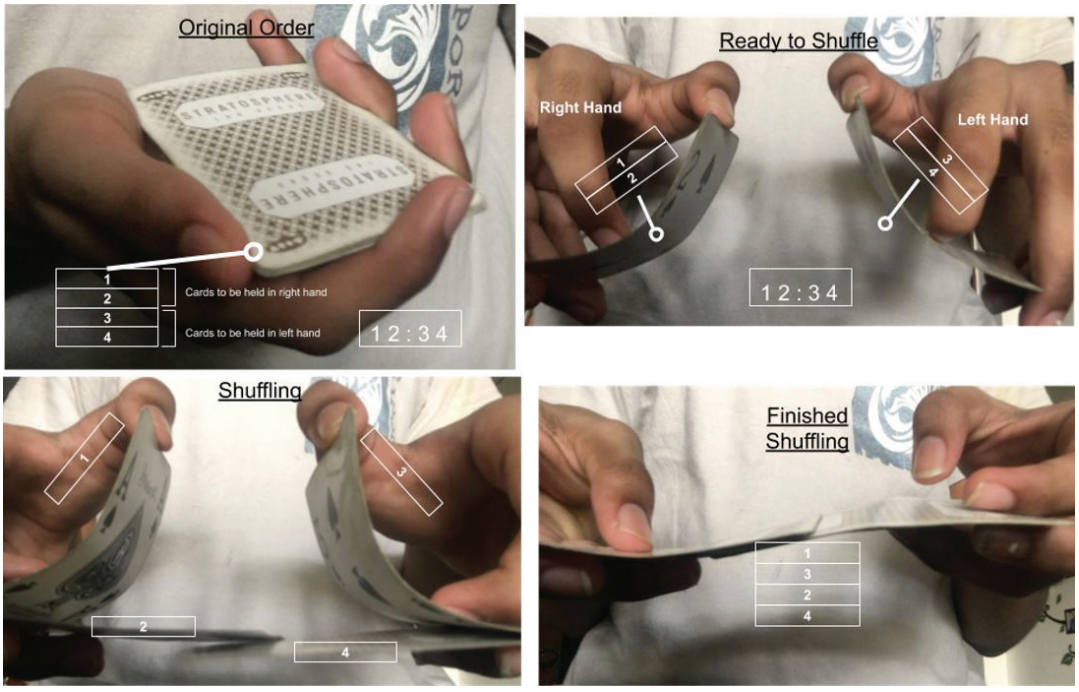

Figure 2: Explaining the deck configuration syntax for four cards.

Now, let us take a look at the table of shuffles.

No. Cards	No. shuffles required	Shuffling process	
4	2	Original	1 2 : 3 4
		After Shuffle One	1 3 : 2 4
		After Shuffle Two	1 2 : 3 4
8	3	Original	1 2 3 4 : 5 6 7 8
		After Shuffle One	1 5 2 6 : 3 7 4 8
		After Shuffle Two	1 3 5 7 : 2 4 6 8
		After Shuffle Three	1 2 3 4 : 5 6 7 8
16	4	Original	1 2 3 4 5 6 7 8 : 9 10 11 12 13 14 15 16
		After Shuffle One	1 9 2 10 3 11 4 12 : 5 13 6 14 7 15 8 16
		After Shuffle Two	1 5 9 13 2 6 10 14 : 3 7 11 15 4 8 12 16
		After Shuffle Three	1 3 5 7 9 11 13 : 15 2 4 6 8 10 12 14 16
		After Shuffle Four	1 2 3 4 5 6 7 8 : 9 10 11 12 13 14 15 16

Table 1: Shuffling configurations when the number of cards in the deck is a power of 2.

The code in section 2 generates a deck with 32 cards, numbered 1-8 for each suit. When this code was run, the deck was restored to the original order after five shuffles. Figure 3 shows the output (s represents spade, c represents clubs and so on).

```
['1s', '2s', '3s', '4s', '5s', '6s', '7s', '8s', '1c', '2c', '3c', '4c', '5c', '6c', '7c', '8c', '1d', '2d', '3d', '4d', '5d', '6d', '7d', '8d', '1h', '2h', '3h', '4h', '5h', '6h', '7h',
'8h']
['1s', '1d', '2s', '2d', '3s', '3d', '4s', '4d', '5s', '5d', '6s', '6d', '7s', '7d', '8s', '8d', '1c', '1h', '2c', '2h', '3c', '3h', '4c', '4h', '5c', '5h', '6c', '6h', '7c', '7h', '8c',
'8h']
['1s', '1c', '1d', '1h', '2s', '2c', '2d', '2h', '3s', '3c', '3d', '3h', '4s', '4c', '4d', '4h', '5s', '5c', '5d', '5h', '6s', '6c', '6d', '6h', '7s', '7c', '7d', '7h', '8s', '8c', '8d',
'8h']
['1s', '5s', '1c', '5c', '1d', '5d', '1h', '5h', '2s', '6s', '2c', '6c', '2d', '6d', '2h', '6h', '3s', '7s', '3c', '7c', '3d', '7d', '3h', '7h', '4s', '8s', '4c', '8c', '4d', '8d', '4h',
'8h']
['1s', '3s', '5s', '7s', '1c', '3c', '5c', '7c', '1d', '3d', '5d', '7d', '1h', '3h', '5h', '7h', '2s', '4s', '6s', '8s', '2c', '4c', '6c', '8c', '2d', '4d', '6d', '8d', '2h', '4h', '6h',
'8h']
['1s', '2s', '3s', '4s', '5s', '6s', '7s', '8s', '1c', '2c', '3c', '4c', '5c', '6c', '7c', '8c', '1d', '2d', '3d', '4d', '5d', '6d', '7d', '8d', '1h', '2h', '3h', '4h', '5h', '6h', '7h',
'8h']
>>>
```

Figure 3: Shuffling process output of the program for a deck of 32 cards

As you can see, the number of cards and the number of shuffles required to restore the original order has a pattern. **We observe that if the number of cards is a power of 2, say 2^n, then the number of shuffles required is n.** (Note that at this stage, it is only an observation. We have yet to justify the statement.)

Now, let us explore why this formula works.

4. Explanation

We consider the case when the number of cards in the deck is a power of 2. The general case will be dealt with later.

To understand this, let us define something called a 'slot'. A slot is merely a gap between any two consecutive cards, i.e., a place where another card can fit in.

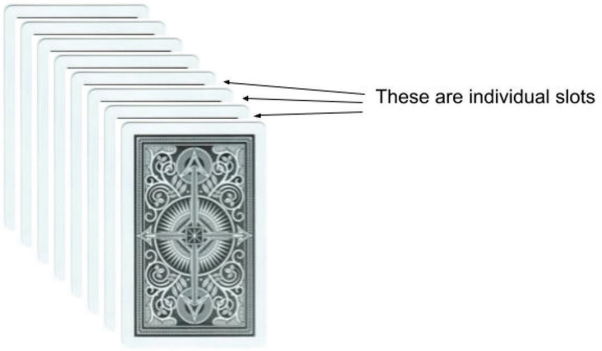

Figure 4: What is a slot?

In the riffle shuffle, since we are perfectly interlocking the cards, after each shuffle, exactly one card gets placed in each slot; that is, exactly one card gets placed between every two consecutive cards.

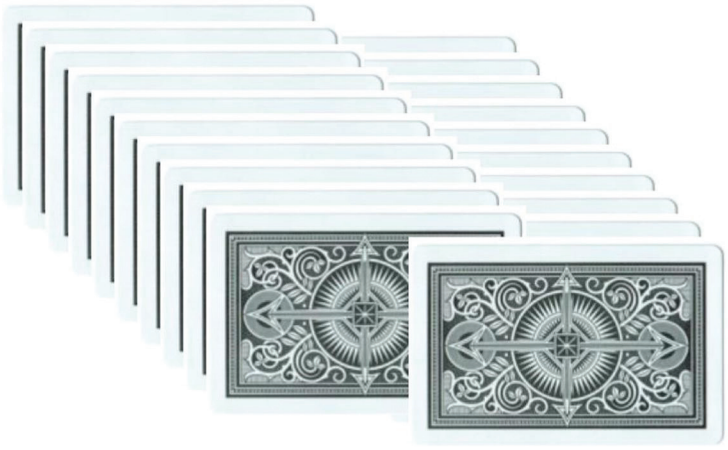

Figure 5: A riffle shuffled deck

Let us look at a deck of eight cards (as in Figure 1). The top half of the deck is held in the right hand and the bottom half, in the left hand. Let us look at the first two cards in the original order of the deck with eight cards: 1s and 2s.

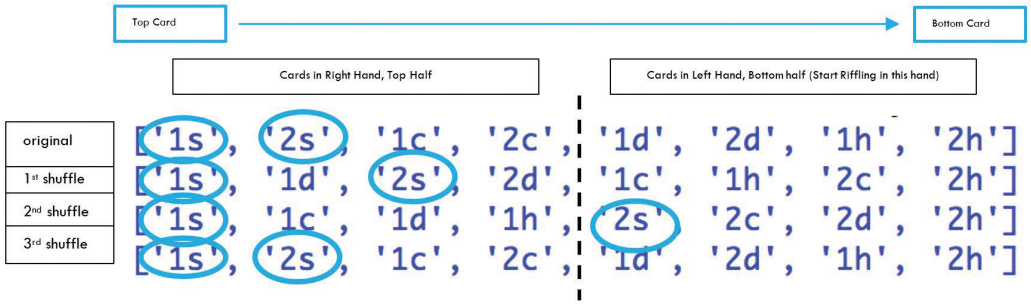

Figure 6: Focusing on the slots between 1s and 2s throughout the shuffling process

In the original order, there is one slot between 1s and 2s. After one shuffle, an additional card fills the slot and there are now 3 cards including 1s and 2s, with 2 slots between them. After the second shuffle, 2 cards fill the 2 slots, and there will now be 5 cards with 4 slots between them. After the third shuffle, there will be 5 + 4 = 9 cards with 9 – 1 = 8 slots. However, the number of cards in the deck itself is 8, which also means that there can be at most 7 slots between any two cards. So, what does this mean? 7 slots can only exist between the first and last cards, that is, 1s and 2h in the original deck order. If there are 8 slots between 1s and another card, the number of slots gets reset to 1 slot, as this is a card deck and is cyclic in shuffling. So, after three shuffles there will be 8 slots between 1s and 2s; equivalently, 1 slot. But if there is 1 slot, this obviously means that 1s and 2s have to be consecutive. Hence, after three shuffles, 1s and 2s will be consecutive once again. This idea can be applied to all consecutive cards of the deck, which will give the original order of the deck.

Now, let us take the case with 32 cards. In the original order, there is 1 slot between 1s and 2s. After the first shuffle, there will be 2 slots. After the second shuffle, there will be 4 slots. After the third shuffle, there will be 8. After the fourth, there will be 16. After the fifth, there will be 32 slots. However, the number of cards in the deck itself is 32. This means the number of slots will be reset to 1 and hence, 1s and 2s will be together again after 5 shuffles. This means that for a deck of 32 cards, it will take 5 riffle shuffles to get back to the original order.

Using this idea, we can prove the formula mentioned in Section 3. The recurring pattern seen here is that the number of slots gets multiplied by 2 with each shuffle, starting from 1, until the number of cards in the deck is reached. The number of shuffles needed for this to happen is that which will restore the original order. Therefore, 2 raised to the number of shuffles needed to restore the original order is equal to x, the number of cards in the deck. That is:

$$2^n = x.$$

Why is 2 used here instead of any other number? This is because while doing this shuffle, we split the deck into two packs, causing the number of slots to increase in powers of 2. Riffling together two packs inserts one card between any two consecutive cards, increasing the number of slots by 1. Theoretically, if we could do a riffle shuffle with three packs, the number of slots would increase in powers of 3 and we would use 3 in the formula instead.

The above logic works regardless of the initial arrangement of cards. In other words, after n shuffles, the original order of the cards will get restored.

5. General Case

So far, we have looked at the case where the number of cards (x) is a power of 2. This is why the 'slot-counting' is reset to 1 so easily. What if x is not a power of 2? The standard card deck has 52 cards, 13 in each suit. Does this reasoning still work? Yes. We consider a few cases to understand the underlying pattern. Note that x must be an even number (otherwise we would not be able to perform a riffle shuffle at all).

For example, take a deck of 12 cards and go through the shuffling process step by step. In the original deck there is 1 slot between 1s and 2s.

- After 1st shuffle: there will be a total of three cards (so 1 card between 1s and 2s) and two slots.
- 2nd shuffle: there will be 5 cards and 4 slots.
- 3rd Shuffle: there will be 9 cards and 8 slots.
- 4th Shuffle: there will be 16 slots. However, the maximum number of slots that can exist between any two cards, in our case, is 11. So, this gets reset to 16 – 11 = 5 slots. Now, we continue as usual.
- 5th Shuffle: 5 × 2 = 10 slots.
- 6th Shuffle: 20 slots. This gets reset to 20 – 11 = 9 slots.
- 7th Shuffle: 9 × 2 = 18 slots. This gets reset to 18 – 11 = 7 slots.
- 8th Shuffle: 14 slots, which gets reset to 3 slots.
- 9th Shuffle: 6 slots.
- 10th Shuffle: 12 slots, which is reset to 1 slot.

Hence, a deck of 12 cards needs to be shuffled ten times to get it back into the original order. Figure 7 is a screenshot of the output when shuffling a deck of 12 cards. A slight modification was made that displays the number of slots 1s is away from 2s each time.

(Refer to **A.2** in Appendix)

```
['1s', '2s', '3s', '1c', '2c', '3c', '1d', '2d', '3d', '1h', '2h', '3h']
2s is 1 slot away.
['1s', '1d', '2s', '2d', '3s', '3d', '1c', '1h', '2c', '2h', '3c', '3h']
2s is 2 slots away.
['1s', '1c', '1d', '1h', '2s', '2c', '2d', '2h', '3s', '3c', '3d', '3h']
2s is 4 slots away.
['1s', '2d', '1c', '2h', '1d', '3s', '1h', '3c', '2s', '3d', '2c', '3h']
2s is 8 slots away.
['1s', '1h', '2d', '3c', '1c', '2s', '2h', '3d', '1d', '2c', '3s', '3h']
2s is 5 slots away.
['1s', '2h', '1h', '3d', '2d', '1d', '3c', '2c', '1c', '3s', '2s', '3h']
2s is 10 slots away.
['1s', '3c', '2h', '2c', '1h', '1c', '3d', '3s', '2d', '2s', '1d', '3h']
2s is 9 slots away.
['1s', '3d', '3c', '3s', '2h', '2d', '2c', '2s', '1h', '1d', '1c', '3h']
2s is 7 slots away.
['1s', '2c', '3d', '2s', '3c', '1h', '3s', '1d', '2h', '1c', '2d', '3h']
2s is 3 slots away.
['1s', '3s', '2c', '1d', '3d', '2h', '2s', '1c', '3c', '2d', '1h', '3h']
2s is 6 slots away.
['1s', '2s', '3s', '1c', '2c', '3c', '1d', '2d', '3d', '1h', '2h', '3h']
2s is 1 slot away.
>>>
```

Figure 7: Shuffling process output for a deck of 12 cards, along with the number of slots after each shuffle

Here are a few more examples of shuffling decks where x is not a power of 2 (the notation is the same as that in Table 1, but with a new column containing the number of slots between the first and second cards).

A python code was used to generate the following shuffling steps (Refer to **A.3** in Appendix).

No. Cards	No. shuffles required	Shuffling process		
10	6	Original	1 2 3 4 5 : 6 7 8 9 10	1
		After Shuffle One	1 6 2 7 3 : 8 4 9 5 10	2
		After Shuffle Two	1 8 6 4 2 : 9 7 5 3 10	4
		After Shuffle Three	1 9 8 7 6 : 5 4 3 2 10	8
		After Shuffle Four	1 5 9 4 8 : 3 7 2 6 10	7
		After Shuffle Five	1 3 5 7 9 : 2 4 6 8 10	5
		After Shuffle Six	1 2 3 4 5 : 6 7 8 9 10	1
14	12	Original	1 2 3 4 5 6 7 : 8 9 10 11 12 13 14	1
		After Shuffle One	1 8 2 9 3 10 4 : 11 5 12 6 13 7 14	2
		After Shuffle Two	1 11 8 5 2 12 9 : 6 3 13 10 7 4 14	4
		After Shuffle Three	1 6 11 3 8 13 5 : 10 2 7 12 4 9 14	8
		After Shuffle Four	1 10 6 2 11 7 3 : 12 8 4 13 9 5 14	3
		After Shuffle Five	1 12 10 8 6 4 2 : 13 11 9 7 5 3 14	6
		After Shuffle Six	1 13 12 11 10 9 8 : 7 6 5 4 3 2 14	12
		After Shuffle Seven	1 7 13 6 12 5 11 : 4 10 3 9 2 8 14	11
		After Shuffle Eight	1 4 7 10 13 3 6 : 9 12 2 5 8 11 14	9
		After Shuffle Nine	1 9 4 12 7 2 10 : 5 13 8 3 11 6 14	5
		After Shuffle Ten	1 5 9 13 4 8 12 : 3 7 11 2 6 10 14	10
		After Shuffle Eleven	1 3 5 7 9 11 13 : 2 4 6 8 10 12 14	7
		After Shuffle Twelve	1 2 3 4 5 6 7 : 8 9 10 11 12 13 14	1
24	11	Original	1 2 3 4 5 6 7 8 9 10 11 12 : 13 14 15 16 17 18 19 20 21 22 23 24	1
		After Shuffle One	1 13 2 14 3 15 4 16 5 17 6 18 : 7 19 8 20 9 21 10 22 11 23 12 24	2
		After Shuffle Two	1 7 13 19 2 8 14 20 3 9 15 21 : 4 10 16 22 5 11 17 23 6 12 18 24	4
		After Shuffle Three	1 4 7 10 13 16 19 22 2 5 8 11 : 14 17 20 23 3 6 9 12 15 18 21 24	8
		After Shuffle Four	1 14 4 17 7 20 10 23 13 3 16 6 : 19 9 22 12 2 15 5 18 8 21 11 24	16
		After Shuffle Five	1 19 14 9 4 22 17 12 7 2 20 15 : 10 5 23 18 13 8 3 21 16 11 6 24	9
		After Shuffle Six	1 10 19 5 14 23 9 18 4 13 22 8 : 17 3 12 21 7 16 2 11 20 6 15 24	18
		After Shuffle Seven	1 17 10 3 19 12 5 21 14 7 23 16 : 9 2 18 11 4 20 13 6 22 15 8 24	13
		After Shuffle Eight	1 9 17 2 10 18 3 11 19 4 12 20 : 5 13 21 6 14 22 7 15 23 8 16 24	3
		After Shuffle Nine	1 5 9 13 17 21 2 6 10 14 18 22 : 3 7 11 15 19 23 4 8 12 16 20 24	6
		After Shuffle Ten	1 3 5 7 9 11 13 15 17 19 21 23 : 2 4 6 8 10 12 14 16 18 20 22 24	12
		After Shuffle Eleven	1 2 3 4 5 6 7 8 9 10 11 12 : 13 14 15 16 17 18 19 20 21 22 23 24	1

Table 2: Shuffling processes of different generic decks

The same approach can be applied to a deck of 52 cards. Here is a table of the shuffle no. and the number of slots:

Shuffle no.	1	2	3	4	5	6	7	8
Slots	2	4	8	16	32	13	26	52 (1)

Table 3: Number of slots for each shuffle in a standard deck of 52 cards

As we can see, the pattern repeats. After each shuffle, the number of slots between 1s and 2s gets doubled. If it is greater than the maximum number of slots (total # of cards -1), the maximum number of slots is subtracted from it. This continues until the number of slots reaches 1. Using this reasoning, a general formula can be derived, for any given number of cards. If s = the number of slots between 1s and 2s, t = the total number of cards, and n = number of shuffles, then

$2^n \equiv s \bmod (t - 1)$. This is generally written in computer languages as s = 2n % (t – 1).

This formula works for any number of cards. In order to find out how many shuffles it takes to bring the deck back into original order, plug in s as 1, plug in the value of t and find n. This may take time as n cannot be made the 'subject' of the formula so easily. For a given number of cards in the deck (t), and s = 1, there are infinitely many pairs (n, s) which will satisfy the above equation. However, our interest is in the *least* possible integer value of n. This value is best found using computer-based enumeration. For example, take the case of an ordinary deck of 52 cards. We must solve the equation

$$2^n \equiv 1 \bmod 51.$$

Starting with $n = 1$, we compute the remainders in the divisions $2^n \div 51$ by starting with 2, doubling repeatedly and throwing away multiples of 51, and continuing till we reach remainder 1. We get the remainders shown in Table 3, i.e., the following numbers: 2, 4, 8, 16, 32, 13, 26, 1… Since a remainder of 1 is first reached when $n = 8$, it means that for a deck of 52 cards, 8 riffle shuffles are required to return to the original order.

I wrote a python program (**A.4** in Appendix) that finds the required number of shuffles. Using this, the number of times needed to riffle shuffle a deck to restore it to its original order can be found with ease. While the program mentioned in the beginning of the paper simulates the riffle shuffle, this program uses a formula-based approach.

6. Conclusion

To sum up, though it may seem at first sight that using a perfect riffle shuffle repeatedly will mix the deck more and more, we find in fact that it must necessarily return to the original order after some shuffles. This is counter-intuitive but true.

But don't worry, because of our imperfect nature, I guarantee that you are safe using this way to shuffle your cards!

KESHAV LAKSHMI NARASIMHAN is a 10th standard student living in Chennai. He has been an avid learner of mathematics since his elementary grades and is fond of solving challenging problems. He sees problem-solving as an opportunity to understand the universe. He takes inspiration from his other hobbies such as art, python programming and cardistry to ask questions which challenge his reasoning skills. He may be contacted at kl77.airsim@gmail.com.