Understanding The Josephus Problem

Introduction:

Imagine you are a part of a ring of people, and, starting with the person in position 1, each person must kill the person to their left, continuing this ring of death until only one person remains. You have an opportunity to choose your position in this ring, and you know exactly how many people are a part of this ring. However, you have neither the time nor the mental capability to run a simulation in your head of how the situation will turn out. What position would you need to choose in order to survive this? This is the question posed by the Josephus Problem. Given a number of participants and a starting position, how do you survive? This IA will attempt to give a method or equation in order to figure out the surviving position in the situation posed. This solution will be found by testing the problem up to a certain amount, say 36, in order to establish that a pattern exists, and then using that pattern in order to establish an equation to solve for the surviving position.

I chose this topic because I came across it while researching possible ideas, and the idea intrigued me, mostly due to its historical relevance. I really enjoy learning about interesting historical happenings, so when I came across this problem, it sparked in my head that this was what I wanted to research. In the year 67, the Romans sieged the Jewish city of Yodfat, and after over a month, finally won the battle. Right before this, however, a group of 41 hid in a cave inside the walls in order to figure out what to do. They decided that the best plan would be to kill each other, instead of everyone committing suicide, as suicide was a sin. They decided that circling up and having people kill a certain number of people would be best. One man, however, advocated for surrender instead, Yosef Ben-Matityahu. This man would be one of two survivors of this, as he convinced the other that their deaths were unnecessary. This man would be taken as a slave to Vespasian, as well as given the name of Flavius Josephus, for which this mathematical problem is named.

Credit cannot really be given to Josephus for creating this problem, however. I can't possibly imagine that 39 of the 41 people were willing to die. They weren't keeping a massive secret, they weren't hiding the Ark of the Covenant, they were just defending their honor. If everyone who would judge you is dead, what's the point in committing suicide? If, say, there were 42 people, then number 21 would have survived, and probably preferred capture to killing themselves. Maybe today it would be called the Octavius Problem, or the Julius Problem, if only someone else survived the Siege, but since Josephus was the survivor, we choose to honor him in his avoidance of an early death. Anyway it turned out, we would probably have an equivalent of the problem today, whether it be from history or from some ambitious or inquisitive computer programmer.

Taking Data and Learning From It:

While the Josephus Problem is a simple one to set up in concept, being a ring of numbers and then just following a set pattern, in reality it isn't as simple. It's repetitive, and higher numbers can take a lot of time to work through to their conclusion. It is for this reason that it is much easier to run the problem at lower numbers in order to establish the pattern, and then testing with larger numbers, such as the 41 from the origin of the problem.

The way that the Josephus Problem is set up, it is conducive to, instead of working out something beforehand, run through the problem with varying amounts of people in order to find a pattern. In essence, numbers are being plugged into the equation, without the equation actually being known, in order to reverse engineer how this equation actually plays out.

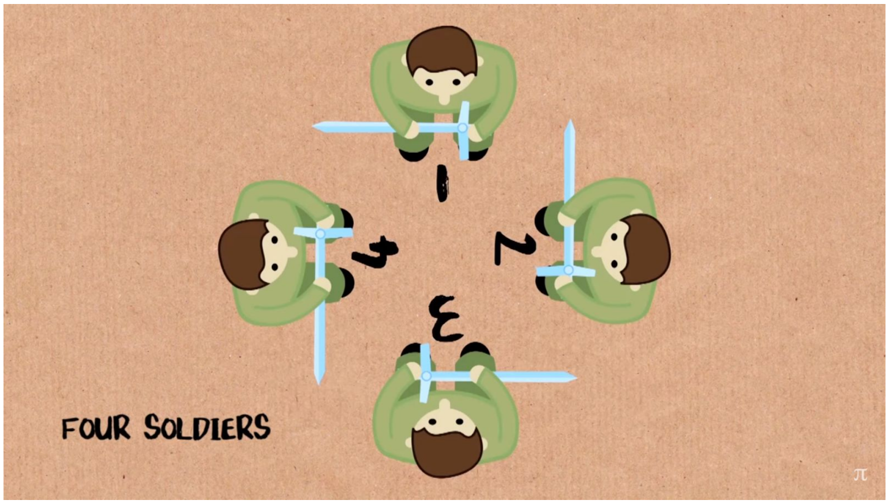


Fig.1 Example of the Josephus Problem with 4 soldiers

For example, 4 is a relatively simple one to do. In order, 1 removes 2, 3 removes 4, then 1 removes 3. So, with the input of 4, being the total number of soldiers, or just people, position 1 will survive. For 5, the method would go 1 removes 2, 3 removes 4, 5 removes 1, 3 removes 5. With an input of 5, the output must be 3. With an input of 6, 1 removes 2, 3 removes 4, 5 removes 6, 1 removes 3, 5 removes 1. This leaves 5 as the survivor.

There are a few important things to note from these tests of the problem. Firstly, there is never an even numbered survivor. This is due to the fact that the starting person will always be number 1, an odd number, and since any number after an odd number is even, all evens are removed in the first round. Secondly, it seems like the number of the survivor moves up by 2 when increasing the number of people.

In order to fully flesh out a pattern, iterations from 1 person to 16 people will be looked at.

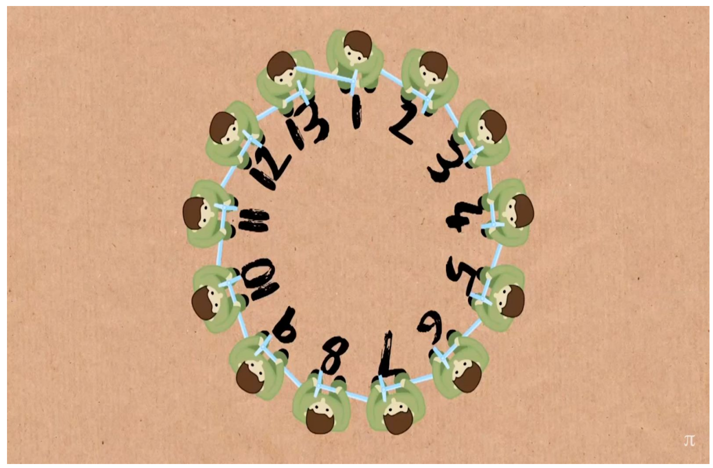
Number of People in Circle (n)	Surviving Position f(n)	Number of People in Circle (n)	Surviving Position f(n)
1	1	9	3
2	1	10	5
3	3	11	7
4	1	12	9
5	3	13	11
6	5	14	13
7	7	15	15
8	1	16	1

Table 1: Results from n=1 to n=16

A few important things can be gleaned by looking at more data. First, when the number of people equals 1,2,4,8, or 16, the person in position 1 survives. In other words, if the base number of people is a power of 2, then the resulting survivor is 1, or the person who begins the process. This will be important in describing an equation. Secondly, the pattern of skipping all even numbers and moving on to the next odd number holds true in this selection, continuing until the base number of people is a power of two, which in essence resets the survivor.

Constructing the Equation:

So with these patterns in mind, an equation can be built that should be able to predict the number of the survivor. But first, notation should be explained. f(n) will represent the number of the survivor. x cannot really be used in the equation as one would generally use it, because n will be split into two different terms. n will equal ($2^x + p$). The term 2^x will represent the largest power of 2 that can be subtracted from n. p represents the rest of what was once n. So, n= ($2^x +$ p), but where does this go? In order to break this down further, an odd example must be made, and by odd, it is meant that the number of people in the circle is not a power of 2.



So how is n represented here? The highest power of two in 13 is 8 so the 2^x term equals 8, while p equals 5. After 5 people have been eliminated (2,4,6,8, and 10), notice

Fig. 2 Jospehus Problem of 13

how many people are left in the circle.

There are 8, a power of 2. Based on the pattern established by Table 1, it is known that any circle with a population equal to a power of 2 will end on the person who began it. Once 10 has been removed, and the circle now consists of 8, 11 becomes the starter of the circle, and once done to completion, 11 is the survivor if n=13.

How does 11 relate to 5, however? If other circles are looked at, say, 10 and 11, a new pattern can be established to determine how the survivor (f(n)) is connected to p. 10 can be broken down into 8+2, so two people must be removed (2 and 4), meaning 5 is the survivor. 11

is broken into 8+3, so three must be removed (2,4, and 6) meaning 7 survives. In all three of these cases, $f(n)$ is equal to $2p+1$.

Finally, there must a slight addendum to the equation. Not any number can be used for the equation. Since the variable n, the number of people in the circle, must be a possible number of people, the set of numbers that n can be chosen from must be restricted. The number cannot be negative, and cannot be a decimal or fraction. Therefore, n must be a subset of all natural counting numbers, anywhere from 1 to infinity, so $n \in \mathbb{N}$. With this information, a final equation can finally be constructed.

$f(n)=2p+1$ where $p=n-2^x$ and $2^x>p$ and $n \in \mathbb{N}$

The actual function portion of this gives the survivor's number, while the other three parts set the conditions for this to actually work correctly. If $2^x<p$, then the equation spits out a survivor number higher than the total amount of people within the circle, which obviously does not make sense. This includes the possibility of $n=0$. If n were to equal 0, p would equal -1, which makes even less sense, which is why n must be restricted to the natural counting numbers.

Final Check:

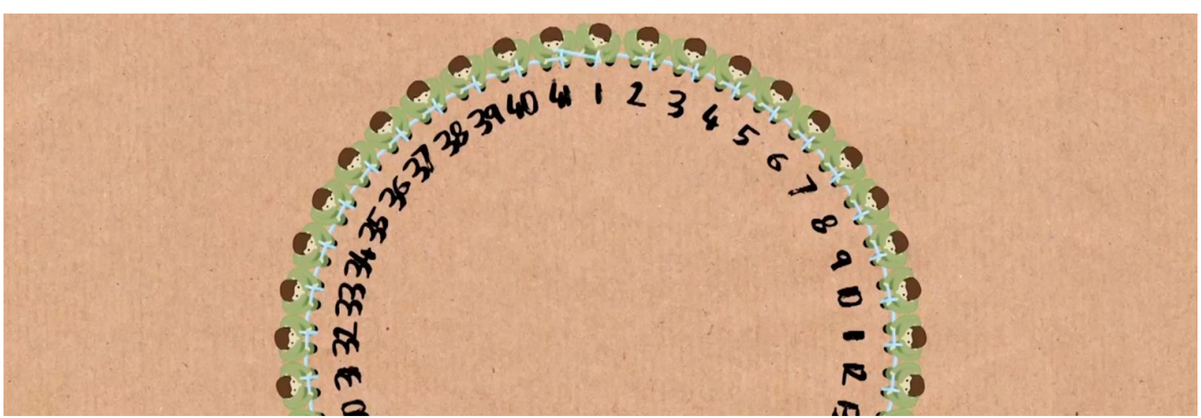


Fig. 3 The Accurate-To-Life Example of the Josephus problem

Going back to the origin of the Josephus problem, there were 41 men who were within the circle, and Josephus needed to know what position he needed to be in. n=41, and 41 can be broken into 32 and 9. Plugging that 9 into the function from the last section gives us 19 as the position of the survivor. If the arduous process of drawing out the circle with 41 people is done, the surviving position is indeed the 19th position in the Circle.

Further Evidence and Explanations:

In order to prove further that the equation given works, it is necessary to take the equation to a different base of numbers, specifically base 2. It's necessary to understand binary, or base 2, somewhat well in order to understand the following explanation. Base refers to the amount of digits in a counting system, so the standard counting system for most of the world is base 10, using the arabic numerals 1-9. Once the number needs to increment past 9, the place to the left of it increments by one and the original place returns to 0. This should be understood well. In binary, the exact same thing happens, but only the digits 0 and 1 are used. So counting to 5, instead of going 1, 2, 3, 4, 5, would go 1, 10, 11, 100, 101. In essence, each place is equal to the number of digits to a certain power. The ones place is any base to the 0th power, the second place is to the 1st power, third place is to the 2nd, etc. So any base 10 number can be converted into binary. For example, 41 is converted to 101001.

Now how does this relate to the Josephus Problem? Think back to the equation created.

$$\mathbf{f(n)=2p+1 \text{ where } p=n-2^x \text{ and } 2^x>p \text{ and } n\in\mathbb{N}}$$

What is the first step to solving the equation? Subtract 2^x from the number of people. For any number in binary, this is equivalent to removing the furthest left 1. 41 (101001) minus 32 (100000) is equal to 9 (1001). The next step after removing the highest power of 2 is to multiply the number by 2. In binary, this is equivalent to moving every number to the left by 1, as that is how . 9 (1001) multiplied by 2 is equal to 18 (10010). The final step is to add 1, which is the exact same in binary. 18 (10010) plus 1 (1) is equal to 19 (10011). In essence, the way that the equation works is the furthest left 1 is moved to the far right of the number. 101001 (41) becomes 10011 (19) through the function created to solve the Josephus problem. This will work no matter what number you use, because the basic principles used to create the function don't change. Any number that is a power of 2, such as 1000000 (64) is reduced to just 1, which is exactly what is wanted.

Now how exactly is it that no matter what power of 2 is used, it will always equal 1? Consider how the problem works out with 2, 4, and 8 people. With 2, 1 will kill 2, so 1 is victorious. With 4, 2 and 4 are killed in the first round, leaving only 2 people, and as previously stated, with 2 people, 1 will always win. For 8, positions 2, 4, 6, and 8 are killed, leaving 4, which, as previously shown, leaves 1 as the winner. The powers of 2 always leave 1 as the winner because they all cut the amount of people in half each iteration, which always bring the people to 1. Now, how can non-powers of 2 be explained? It's simple. At some point in going around the circle, there will be a number of people alive equal to a power of 2. For 67 people, once three people die, there are 64 people left. From that point, the problem can be worked out as if those three people were never there and position 7 (the next position after three deaths) can be relabeled as position 1 in a circle of 64, and it's easily known how that turns out.

Real World Application:

The Josephus Problem has its basis in history, and while the story likely got mixed up somewhere along the line, the basis of it is that it was essentially a more lethal "counting-out game", which is something that will end up choosing a single person through some process, whether that be from a rhyme or a complicated math function. These counting-out games have their place in society, with many games, such as Eeny meeny miny moe or Tinker, Tailor, taking place at local elementary schools, but the Josephus problem in particular has applications in computer science. While computers cannot rhyme, at least not yet, they can compute massive amounts of data, say if the problem were scaled up to 3000 people and instead of removing every other person, 6 people are skipped, meaning 1 removes 8, 9 removes 17, and so on. The iterations are, quite literally, infinite. As an example for a practical application of the Josephus Problem, a random number generator can be built from the Problem. By having a set amount of people, the number of people skipped can be varied, causing a "random" number to be generated. While other random number generators may be more efficient, or perhaps more random, this is still a practical application.

Works Cited:

Josephus problem. (2017, December 13). Retrieved February 05, 2018, from https://en.wikipedia.org/wiki/Josephus_problem

Counting-out game. (2018, January 23). Retrieved February 05, 2018, from https://en.wikipedia.org/wiki/Counting-out_game

(2016, October 28). Retrieved February 05, 2018, from https://www.youtube.com/watch?v=uCsD3ZGzMgE