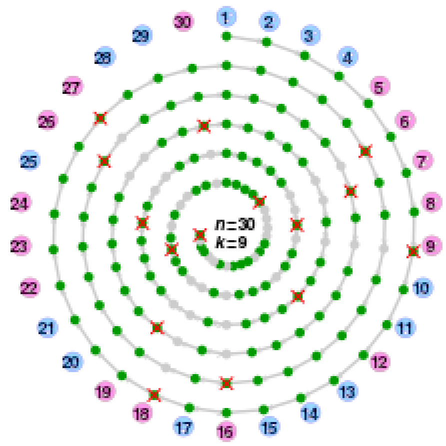
Josephus Problem - The Last Man Standing



Abstract:

Josephus's Problem is a logical puzzle that relates to a counting-out game. This puzzle is used to pick out a person from a given group of people using an algorithm to eliminate people. It includes a circle elimination algorithm where n people are placed in the circle, and counting starts from a specific point. It is a classical example illustrating concepts like Modular Arithmetic, Recursion, and Linked Lists. A Circular Linked List can be created to represent the positions. Modular Arithmetic is used to determine the position of the person to be eliminated next, and Recursion is used to traverse through the whole circle until only one person remains. This algorithm can have many other variations, which will be covered in this report later.

Historical Context and Real-World Applications:

The Josephus problem is named after the Jewish Historian Flavius Josephus, who documented the related event during the Jewish War. In his text, he had written that 40 soldiers and himself were trapped in a cave by Roman Forces. The whole group decided to commit suicide rather than surrendering to the Romans. They all formed a circle, and every second person in a particular counting method killed the person on their left till only one person who would surrender to the Romans remained. At last, Josephus survived, and he surrendered to the Romans.

Though it was a life/death situation for the soldiers, this algorithm was studied extensively by computer scientists and mathematicians for more variations and different scenarios in the real world.

For Example - It can be used in

1. Resource Allocation in Systems: Resources can be distributed in a circular fashion by making a counting method to allocate the resources among the users.
2. Task Scheduling: It can be used for recurring tasks or processes.
3. Queue Management: Handling requests in a service center

Special Facts for the case when k = 2

1. When $n=2^a$ →
   Fact: The winning position W(n) is always 1, meaning the person who starts the killing.
   Explanation: When k=2 and n is even, every even position is eliminated until (n-1)th person kills nth person, and it will be the first person's chance to kill again i.e. the person who started the killing. As n is a power of 2 in this case, every round has an even number of people, so everytime the chance for starting the killing comes to the first person and the process continues until only one person remains. Hence, the pattern is such that the person who starts the killing always survives.

2. When $n=2^a+l$ →
   Fact: The winning position W(n) is 2*l+1.
   Explanation: Using the above fact, when 2^a people are left in the circle, then the person who starts the killing survives. So, in this case, after l persons are killed, there are 2^a people left in the circle, so the person whose chance will be to kill after killing l people will survive in the end. And the position at which it lands after killing l people will be 2*l+1.

3. Fact: If n is expressed in binary, then W(n) is obtained by a one-bit left shift of the binary representation of n.
   Explanation: $n = (b_m,b_{m-1}...b_1,b_0)_2 = b_m2^m + b_{m-1}2^{m-1} +\ldots+b_12 + b_0$ where $b_i \in 0,1$ & $b_m= 1$,
   Hence, when n is expressed as 2^a+l then, a = m and $l = 0 + b_{m-1}2^{m-1} +\ldots+ b_12 + b_0$;
   Therefore, $l = (0,b_{m-1}...b_1,b_0)_2$
   Hence, $2*l = (b_{m-1}...b_1,b_0,0)_2$ & $2*l+1 = (b_{m-1}...b_1,b_0,1)_2$ which is the same as $(b_{m-1}...b_1,b_0,b_m)_2$ as $b_m = 1$. Therefore, the binary representation of the survivor position is just one-bit left shift of the binary representation of n.

Now, we will explore how different elimination rules (such as varying n & k) and counting patterns affect the final survivor position using concepts of discrete mathematics.

Mathematical Analysis and Algorithmic Solutions:

To solve Josephus Problem we can do it by two ways:

1. Modular Arithmetic and Recursion
2. Graph Theory

Let us first solve it using Modular Arithmetic And Recursion:

Let $J(n,k)$ be the position of the survivor when starting with n-people and elimination every k-th person

Recursion

Base Case - If $n=1$ that is $J(1,k) = 1$ survivor at position 1

Recursive Step - If $n>1$, then apply modular arithmetic to handle the circular arrangement

Modular Arithmetic Formula:

$J(n,k) = (J(n-1,k)+k-1) \bmod (n+1)$ if $n>1$

Pseudocode

```
josephus(n,k)
if(n==1):
   return 1
else:
   return ((josephus(n-1,k)+k-1) mod n+1
```

Time Complexity:

- The time complexity for the above code is only due to the recursive calls
- The recurrence relation in else statement requires n calls;
- Therefore time complexity will O(n)

Space Complexity:

- It will be determined by the depth of the recursive calls
- Worst case will be n calls
- Therefore space complexity will be O(n)

Proof:

For (josephus(n-1,k)+k-1) mod n+1 gives the survivor position

Base Case:

For $n = 1$ the survivor position is 1, which is consistent with the formula.

Inductive Step:

Assume that the formula holds for $n = m$, where $m > 1$.

Now, consider $n = m + 1$. We need to show that the formula holds for this case as well.

We know that the survivor position for $n = m + 1$ is determined by excluding one person and finding the survivor position for $n = m$.

Let S_m denote the survivor position for $n = m$. According to the inductive assumption, S_m = (josephus(m-1, k) + k - 1) % m + 1.

For $n = m + 1$, the survivor position will be different due to the circular nature of the problem after each elimination of the k_{th} person.

When we eliminate the k_{th} person in a circle of m + 1 people, the position changes to the survivor position in a circle of m people, which is S_m, plus k - 1 to account for the gap due to excluding one person.

So, the survivor position for n = m + 1 will be $(S_m + k - 1) \% (m + 1) + 1$.

Therefore, we've shown that if the formula holds for n = m , it also holds for n = m + 1.

By induction, since the formula holds for n = 1 and it holds for n = m + 1 whenever it holds for n = m ,it holds true for all positive integers n .

Now, Let us solve it using Graph Theory:

This approach will use a circular graph or circular linked list to get to the solution.
First, create a circular graph to represent all the people present. The last node should point to the first node to create a circular linked list.

```
Example 1–>2–>3–>4–>5–>6
          ^________________|
```

Start the from any point in the graph and iterate through the graph using k-elimination step

Pseudocode:

```
josephus(n,k)
Create a circular linked list
Initialize a variable current to head (1st position in the circle)
Length = length(Circular List)
while(current->next != current): //until there is only one person left in the circular linked list
    Increment current by k - 1 (to find the k-th person)
    Eliminate the person at index 'current' from the circular structure
    set 'current' to the next person (node) in the list
return the last position as the survivor
```

Time Complexity:

The program iterates through the whole circular list to find the current position which takes O(n) time
This happens for n persons
Therefore, Time Complexity is O(n*n)

Space Complexity:

The space complexity is determined by the size of the circular linked list.
Therefore, it is O(n)

Variations of Josephus Problem:

1. Josephus Problem with Revivals

   The Josephus Problem with revivals is an extension of the classic Josephus Problem where individuals are arranged in a circle, and every (k)-th person is eliminated and added to a queue, until only one person remains. In this variant, after every (m)-th person is eliminated, the first person in the queue is revived and reintroduced into the circle at its original position. The process continues cyclically until only one person remains in the circle, creating a dynamic where eliminated individuals may rejoin.

   Note: k>=m for this process to terminate.

   Algorithm:

   Represent the circular arrangement of people as a set C with elements 1, 2, ..., n, where n is the number of people in the circle.

   The algorithm will be the same as implementation of graphical (circular linked list) solution of normal Josephus problem, just maintain a queue of eliminated people and after m eliminations, dequeue the first person from the queue and add it to its original position in the circle before elimination.

2. Josephus Problem in Multi Dimensional

The Multidimensional Josephus Problem expands the classic Josephus dilemma into multiple dimensions, entailing eliminations across various axes or structures. Unlike the original linear or circular arrangement, this variant involves intricate multi-dimensional setups with distinct elimination rules for each dimension. Determining the surviving positions or entities within these complex, multi-axis configurations poses significant mathematical and computational complexities, demanding innovative strategies for solution.

Algorithm:

1. Define D as the number of dimensions or structures.
2. Each dimension or structure d (1<=d<=D) has its arrangement of entities, often forming circles, lines, or other configurations.
3. Introduce elimination rules specific to each dimension, denoted by k_d, indicating the elimination step within that dimension.
4. The elimination process involves iterating through entities across all dimensions, eliminating individuals at the specified intervals based on the rules of each dimension.
5. Determine the survivor(s) or remaining positions within this multi-dimensional setup after all eliminations.

Code for different algorithms of Josephus Problem and Interactive Implementations are present in the Github Repository provided below:

- https://github.com/SGx3377/DM-Project.git
- For the interactive implementation, we have used the 'Turtle' module of python, for which we have referenced GeeksforGeeks.

Conclusion

Our exploration into the Josephus Problem and its array of algorithmic variations has been an illuminating journey through the depths of mathematics and algorithmic methodologies. This ancient puzzle, centering around the survival of individuals arranged in a circular formation and facing sequential elimination, sparked our curiosity to delve into its multifaceted computational complexities. Our exploration didn't merely scratch the surface; instead, it delved into a spectrum of algorithmic approaches – from recursive models to iterative solutions – all aimed at deciphering the position of the final survivor within the circular structure.

Diving deeper into the problem's extensions and variations, we ventured into realms like the Josephus Problem with revivals, multidimensional expansions, circular permutations, and the integration of graph theory. These forays allowed us to dissect diverse computational strategies, unraveling the intricacies stemming from altered elimination rules, multi-dimensional constructs, and unique mathematical representations. Navigating through graph-based interpretations, permutation cycles, recurrence relations, and other facets of the problem, our exploration not only honed our problem-solving prowess but also underscored the interdisciplinary nature of mathematical challenges. This journey has affirmed that the Josephus Problem and its large number of variations serve as a fertile ground for exploration and innovation, embodying the essence of computational exploration across diverse domains.

References

1. Cse547 DISCRETE MATHEMATICS Lecture 2 slides by Professor Anita Wasilewska
   https://www3.cs.stonybrook.edu/~cse547/newLec2.pdf
2. "The Josephus Problem - Numberphile" youtube video featuring Daniel Erman from University of Wisconsin-Madison uploaded to channel "Numberphile"
   https://www.youtube.com/watch?v=uCsD3ZGzMgE
3. Recurrence Relations and Their Solutions (Josephus Problem) by P Sam Johnson (NITK)
   https://sam.nitk.ac.in/courses/MA201/Josephus_problem.pdf
4. The Josephus Problem lecture notes from University College of Dublin
   https://www.ucd.ie/mathstat/t4media/1.%20The%20Josephus%20problem.pdf