Algorithm Problem Solving
17ECSE309

JOSEPHUS PROBLEM

USN: 01FE15BCS172

NAME: SARJITH R M

Josephus Problem Definition

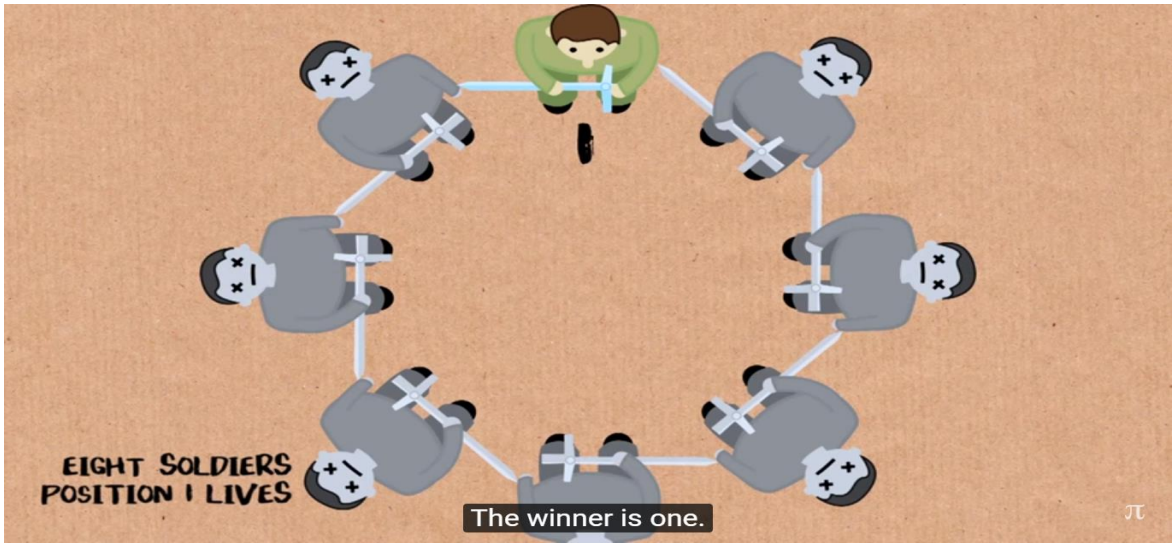
The **Josephus problem** (or **Josephus permutation**) is a theoretical problem related to a certain counting-out game.

People are standing in a circle waiting to be executed. Counting begins at a specified point in the circle and proceeds around the circle in a specified direction. After a specified number of people are skipped, the next person is executed. The procedure is repeated with the remaining people, starting with the next person, going in the same direction and skipping the same number of people, until only one person remains, and is freed.

The problem — given the number of people, starting point, direction— is to choose the position in the initial circle to avoid execution.

Josephus Problem Solution

- If n(number of people) is power of 2, then safest position is 1.
  - Ex: If n=8, safest position is 1



- If n(number of people) other than power of 2, then express the number in the form of

$$n = 2^a + L$$

Ex: If n=41, then

$$41 = 2^5 + 9$$

$$41 = 32 + 9$$

if L<2a, then W(Winning position)=2L+1.

For the above example the winning position is

$$W(n)=2(9)+1=19.$$

NOTE: Even applicable to power of 2.

Another approach for the above condition

- Find the binary notation of n(binary notation of 41 is 101001)
- Later left shift the number by one and add one to it.

(n<<1)+1 => 010010+000001 => 010011

Binary notation of 19 = 010011

Code

```
public int getSafePosition(int n) {
int valueOfL = n - Integer.highestOneBit(n);
int safePosition = 2 * valueOfL + 1;
 return safePosition;
}
```

References

[1] https://en.wikipedia.org/wiki/Josephus_problem.

[2] https://www.youtube.com/watch?v=uCsD3ZGzMgE