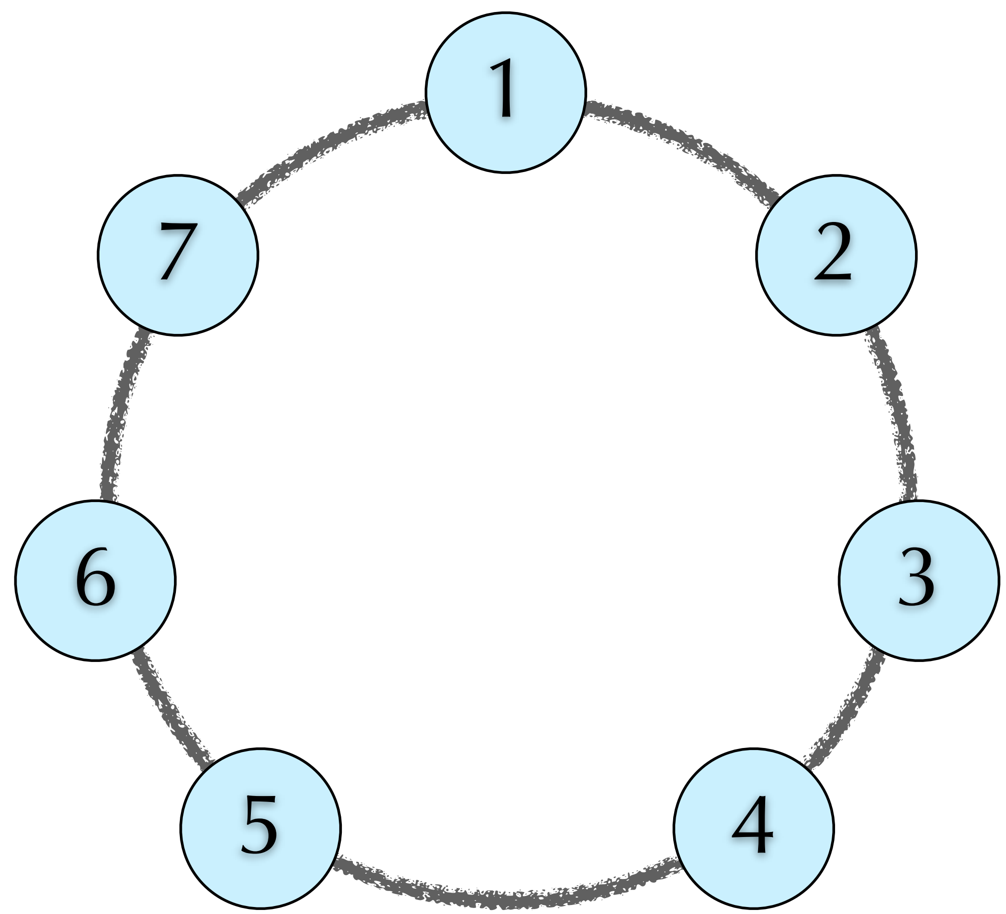
Joseph Problem

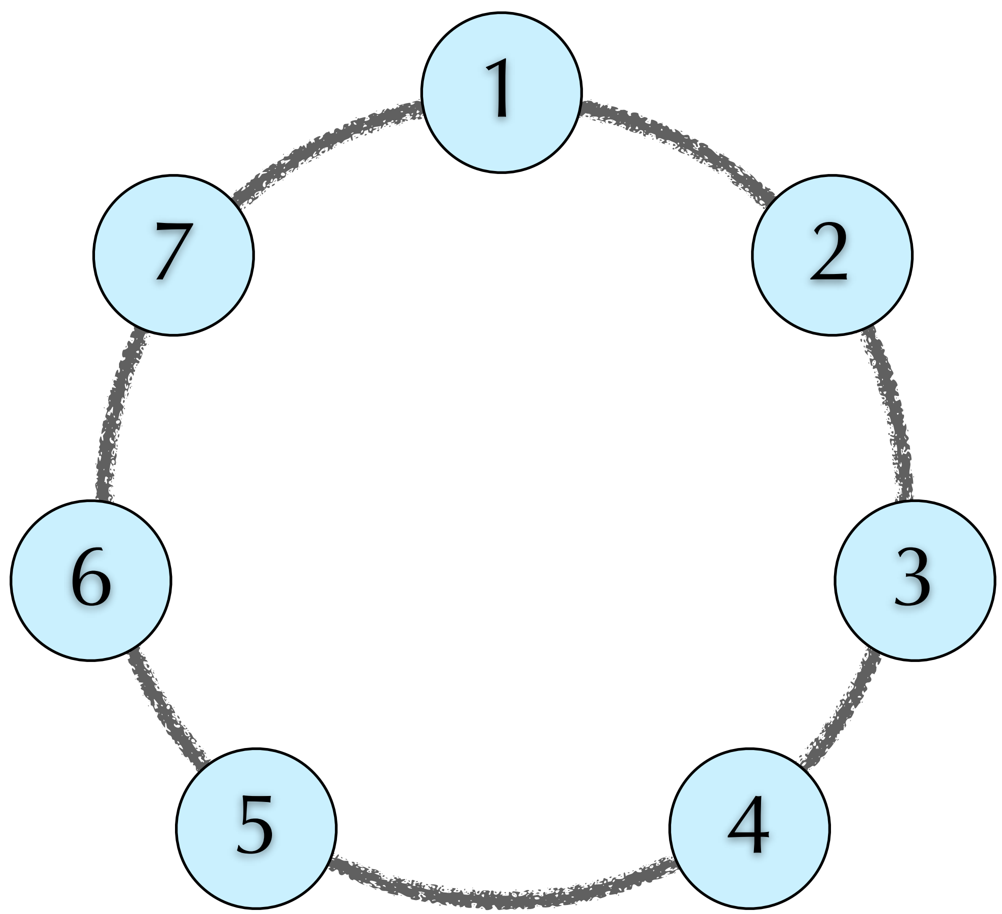
Joseph Problem

- n people stand in a circle.
- They are numbered 1, 2, ..., n in clockwise order.



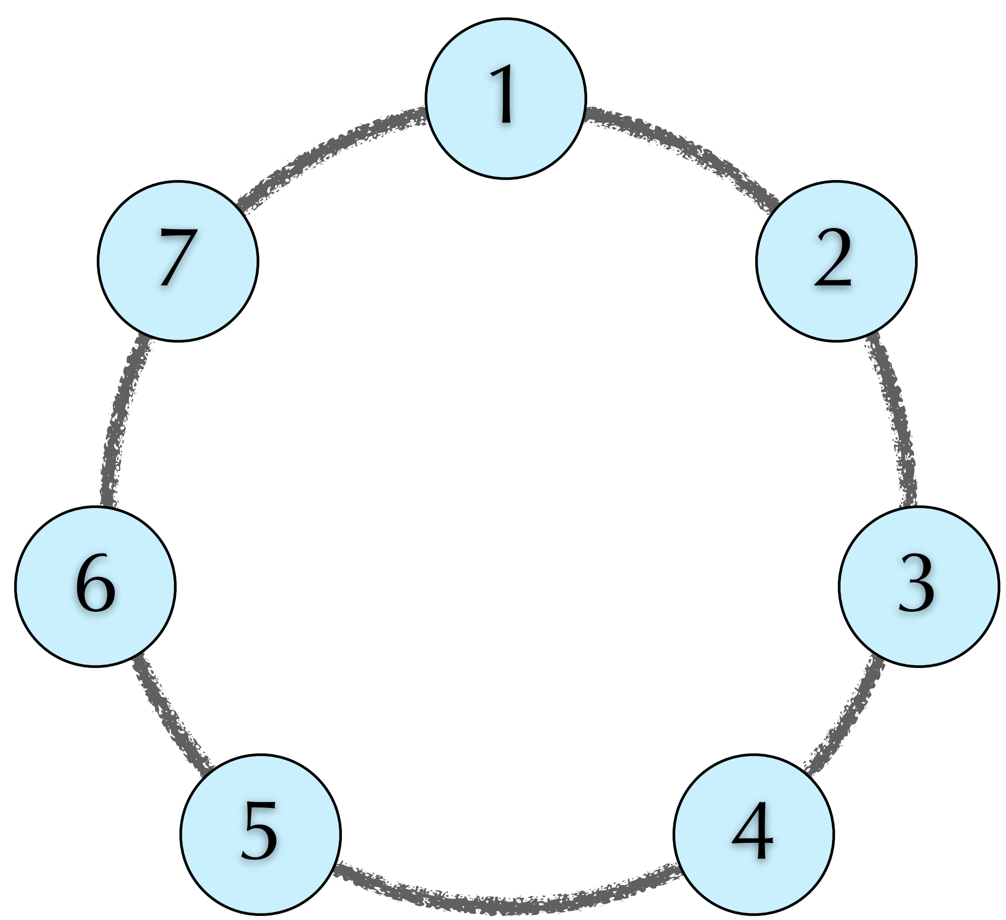
Joseph Problem

- In each iteration, counting the people in clockwise order. Kill the k-th persion.



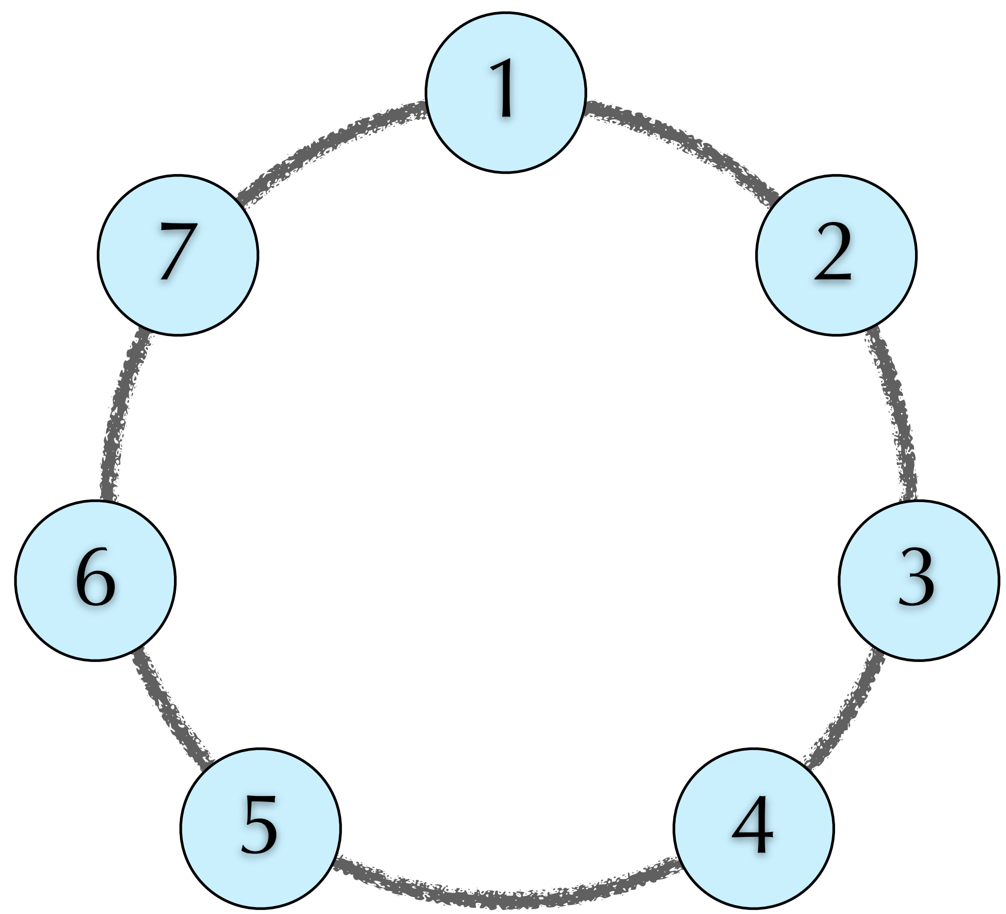
Joseph Problem

- The last person will survive.
- Who will survive?



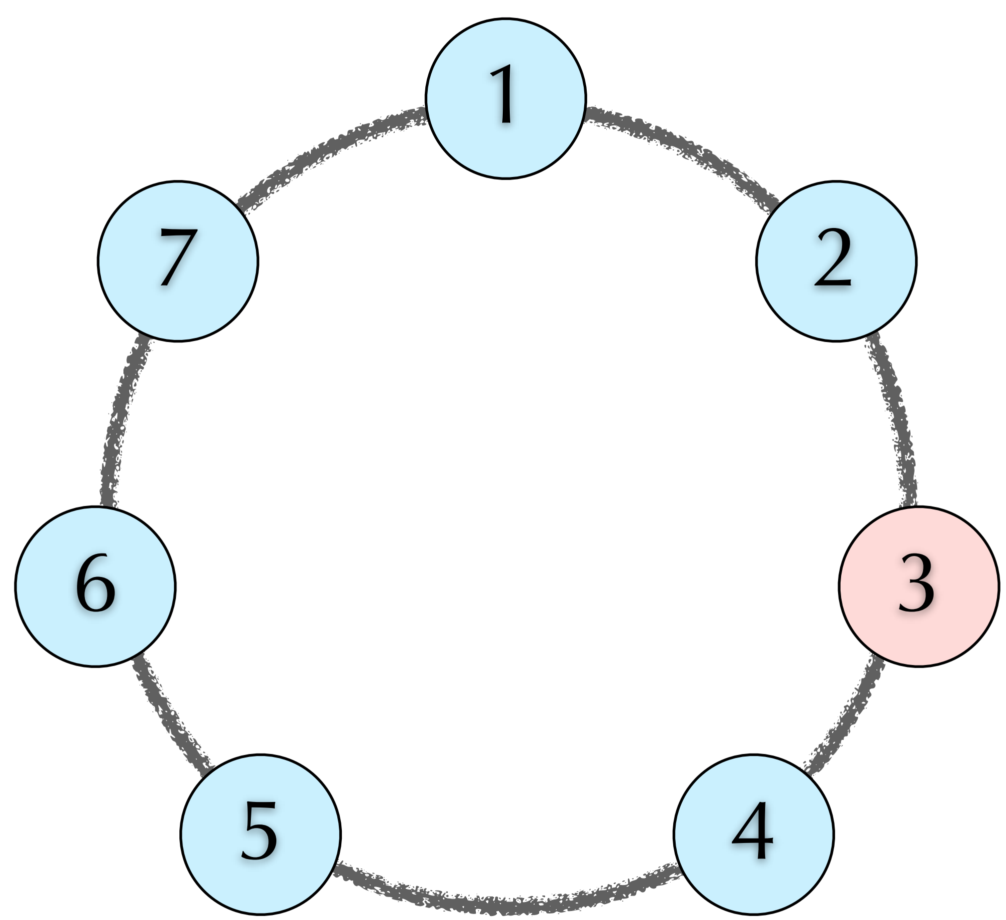
n=7, k=3

- 1st iteration: 1, 2, 3.
- 3 must die.



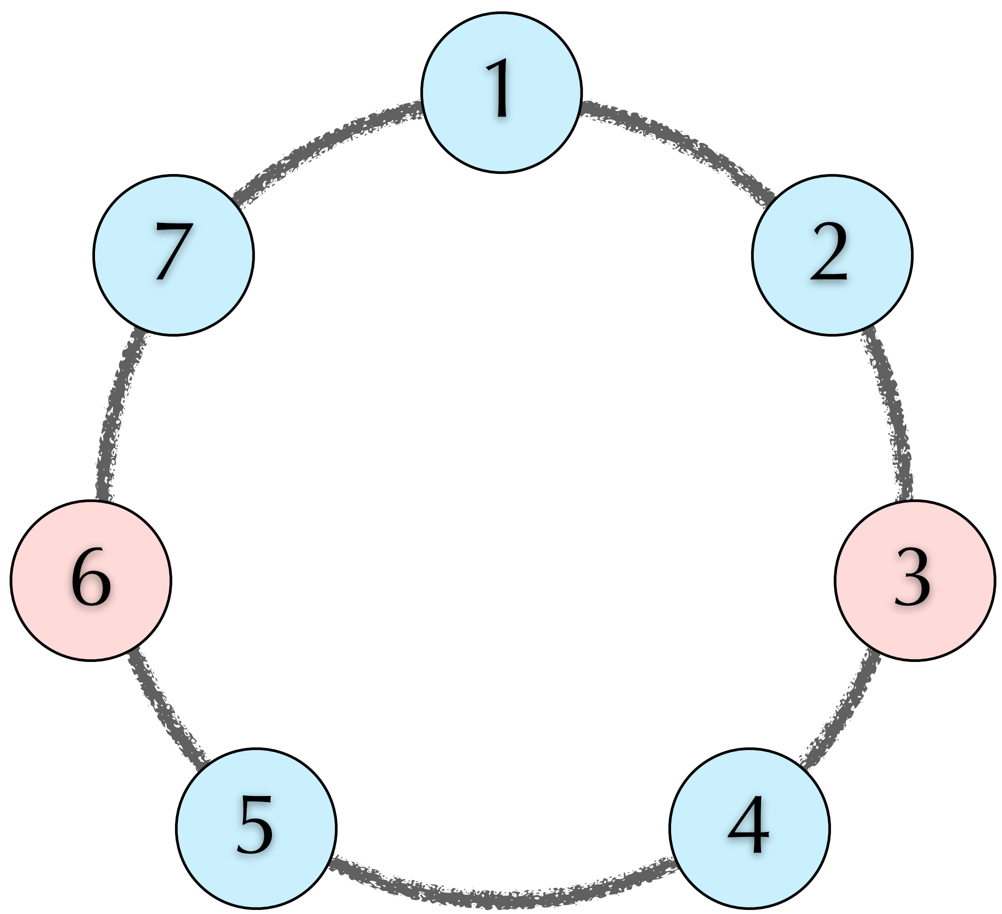
n=7, k=3

- 2nd iteration: 4, 5, 6.
- 6 must die.



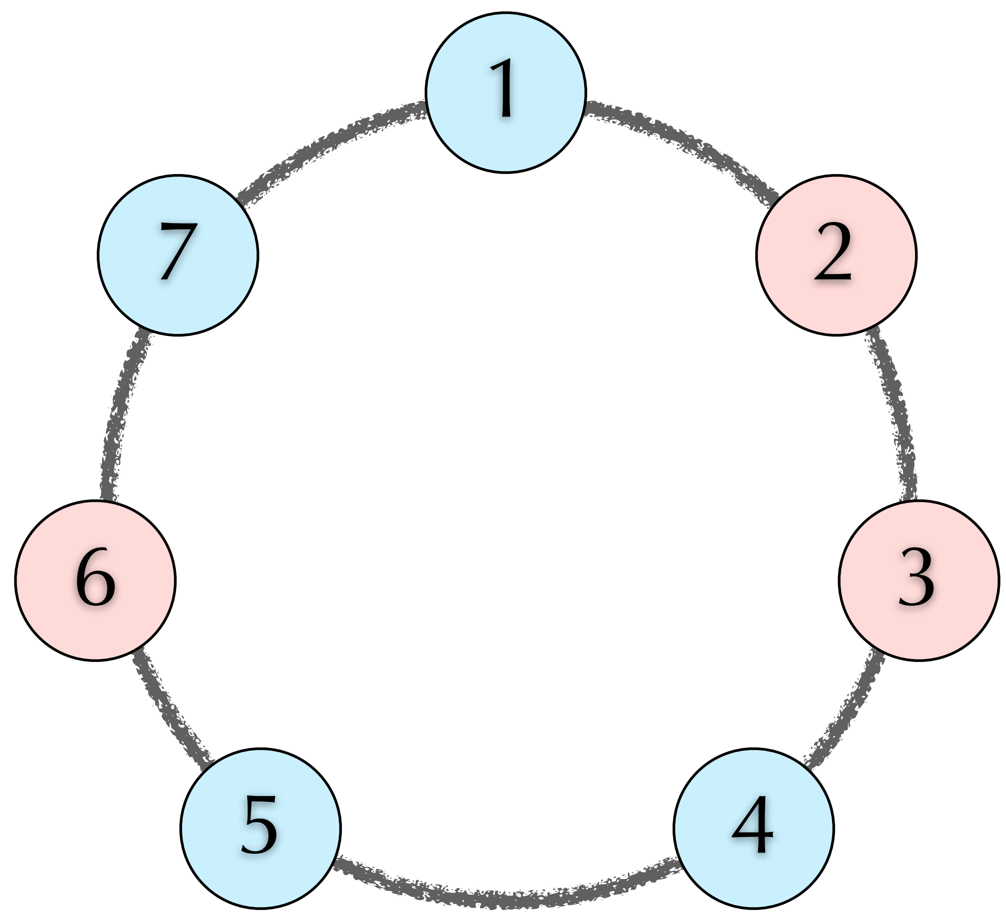
n=7, k=3

- 3rd iteration: 7, 1, 2.
- 2 must die.



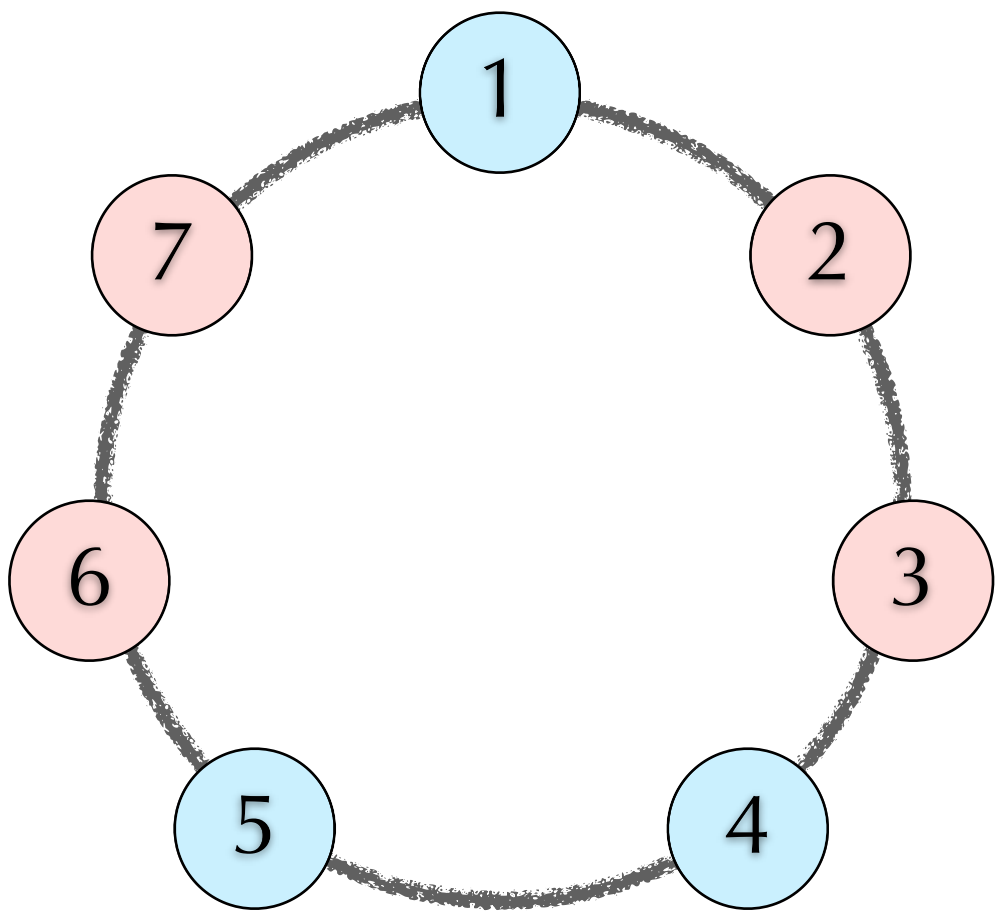
n=7, k=3

- 4th iteration: 4, 5, 7.
- 7 must die.



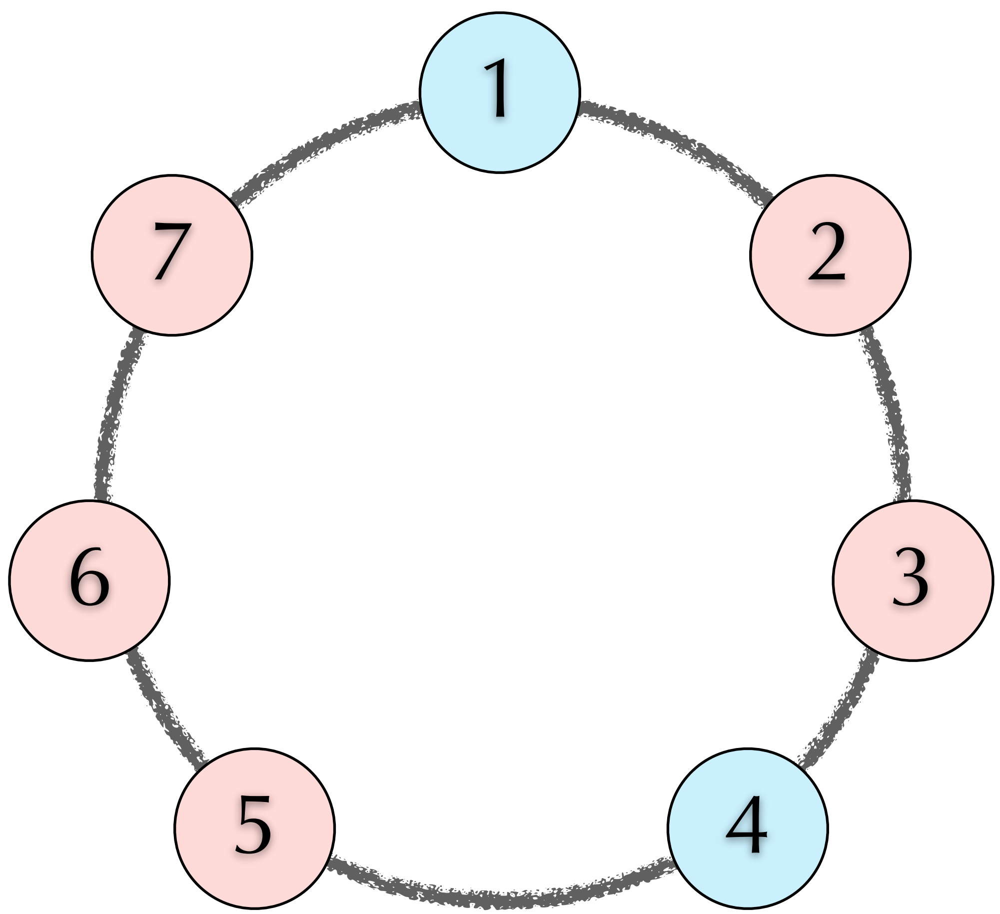
n=7, k=3

- 5th iteration: 1, 4, 5.
- 5 must die.



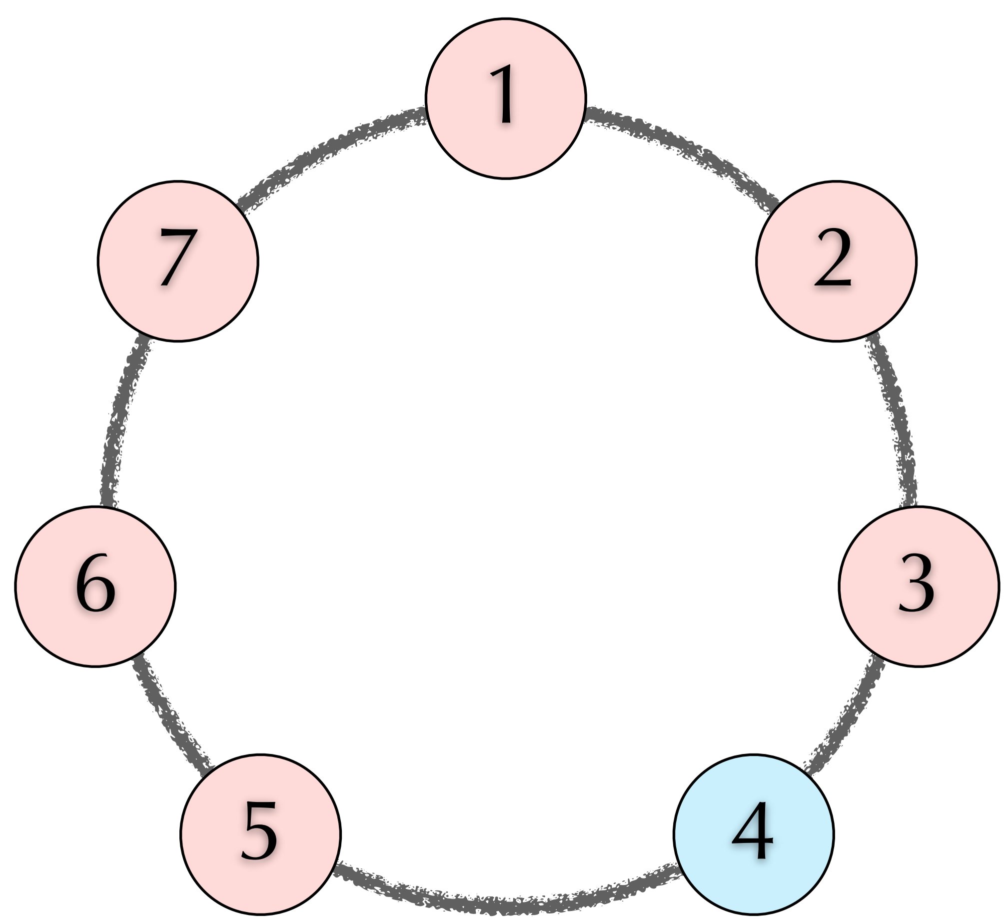
n=7, k=3

- 6th iteration: 1, 4, 1.
- 1 must die.



n=7, k=3

- 4 survives.



How to solve?

- Straightforward: Simulate the process
- for i = 1 to n–1 do

```
    find the victim k
    kill k
next i
print the survivor
```

How to solve?

- Different implementation, different efficiency.
- Data structure
  - Array: find in $O(1)$, kill in $O(n)$
  - Linked list: find in $O(k)$, kill in $O(1)$
  - Fenwick tree + binary search: find in $O(\log^2 n)$, kill in $O(\log n)$
  - Augmented binary search tree: find in $O(\log n)$, kill in $O(\log n)$

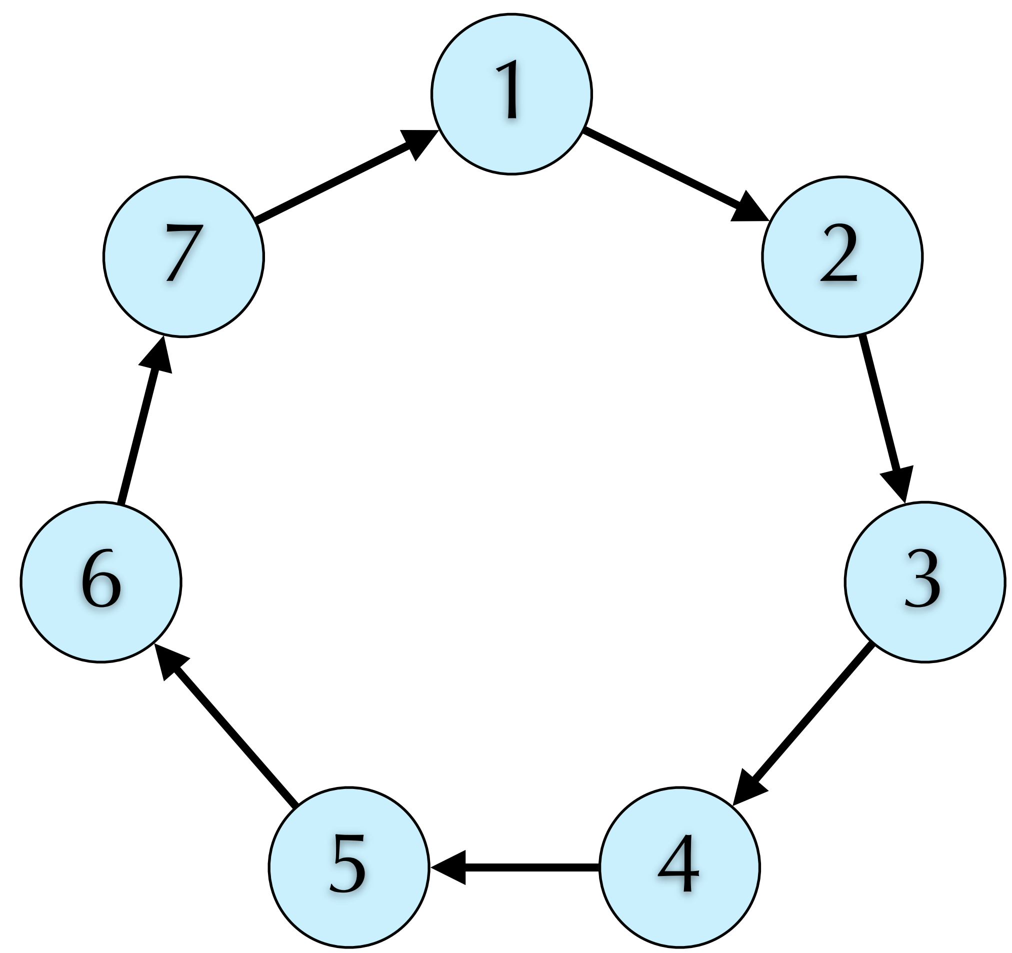
Array

- Use a single array to store the survivors.
- Find: O(1)
- Kill: O(n)
- Total running time: O(n^2)

1	2	3	4	5	6	7
1	2	4	5	6	7	
1	2	4	5	7		
1	4	5	7			
1	4	5				
1	4					
4						

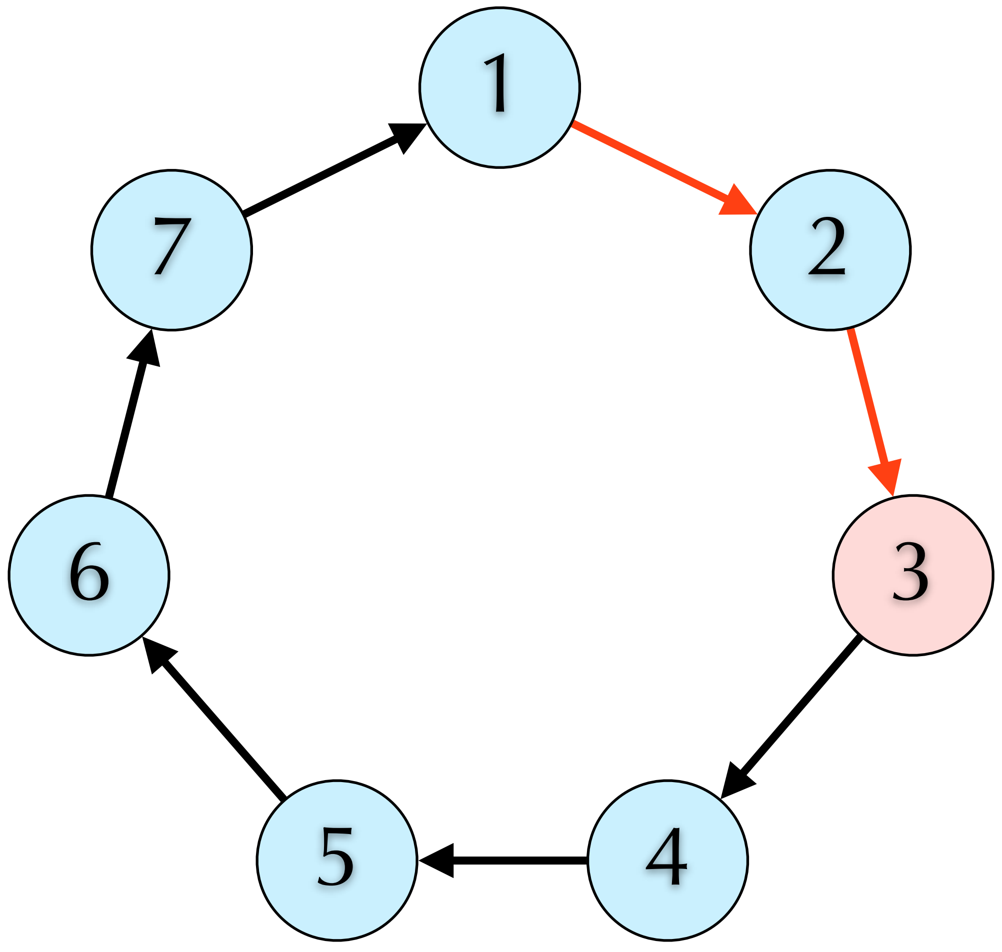
Linked list

- Find: O(k)
- Kill: O(1)
- Total running time: O(kn)



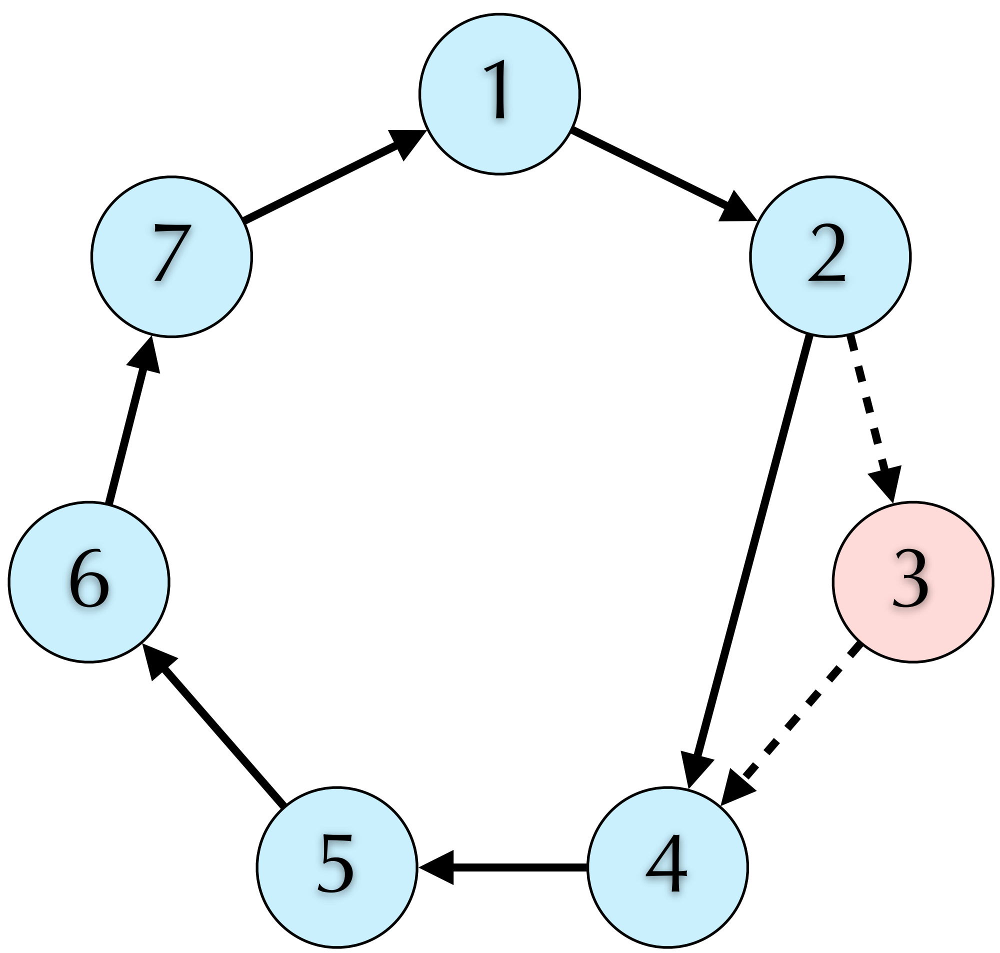
Linked list

- Find: O(k)
- Kill: O(1)
- Total running time: O(kn)



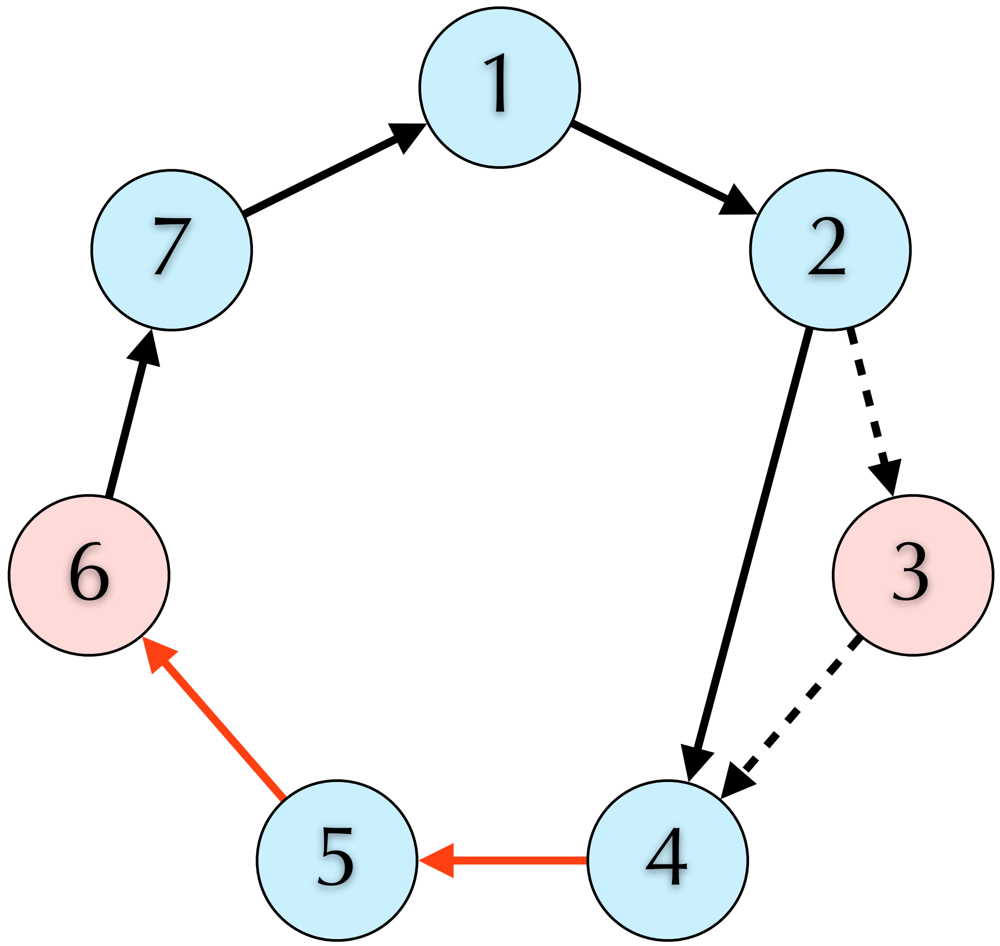
Linked list

- Find: O(k)
- Kill: O(1)
- Total running time: O(kn)



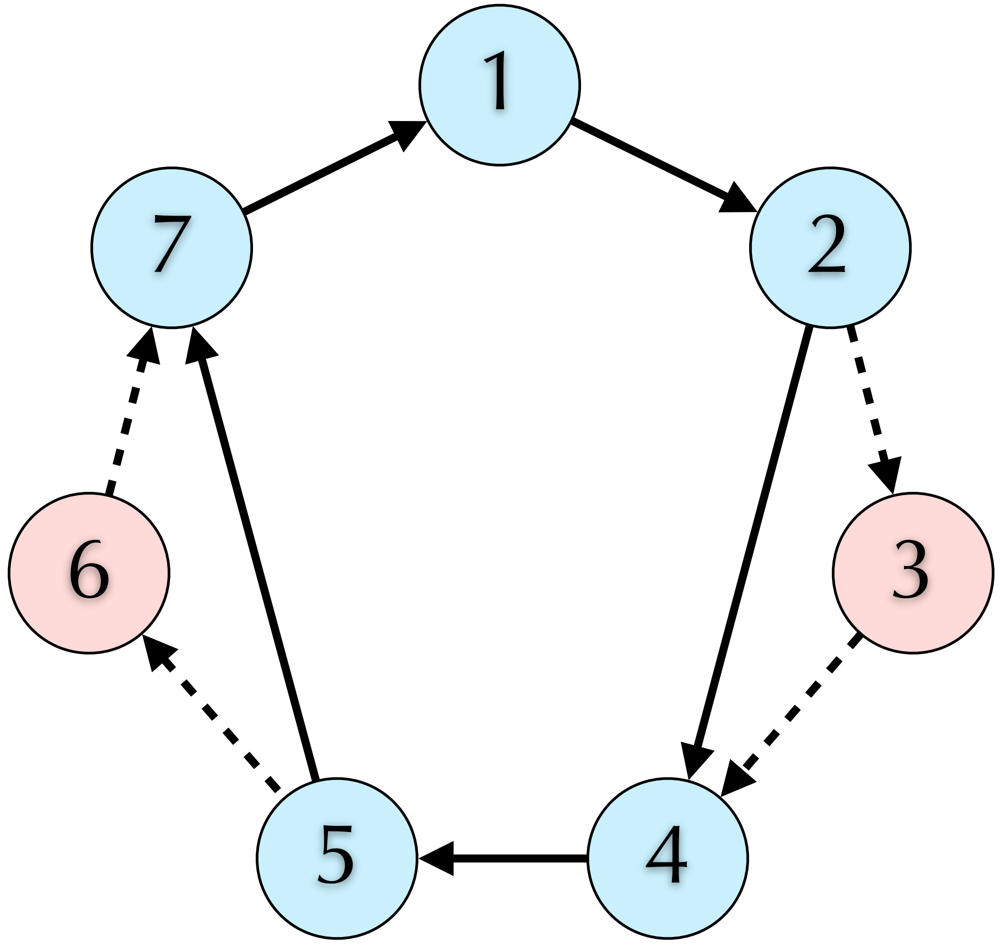
Linked list

- Find: O(k)
- Kill: O(1)
- Total running time: O(kn)



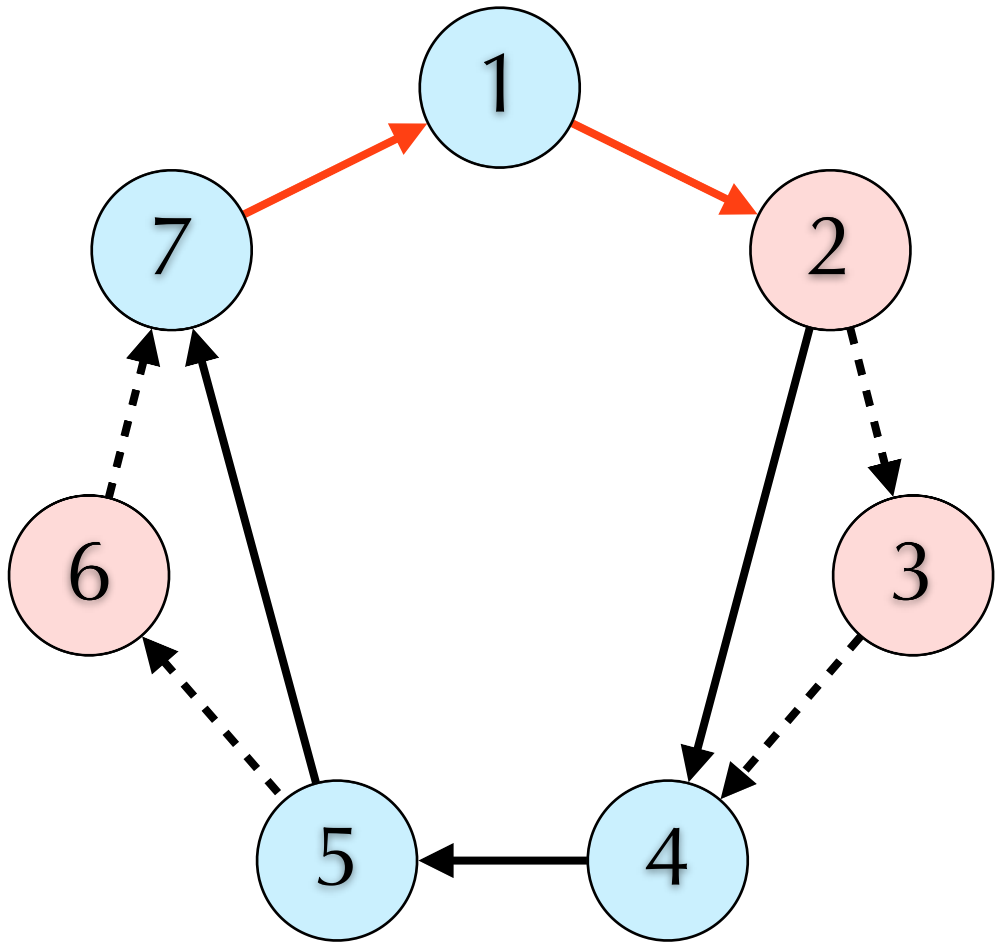
Linked list

- Find: O(k)
- Kill: O(1)
- Total running time: O(kn)



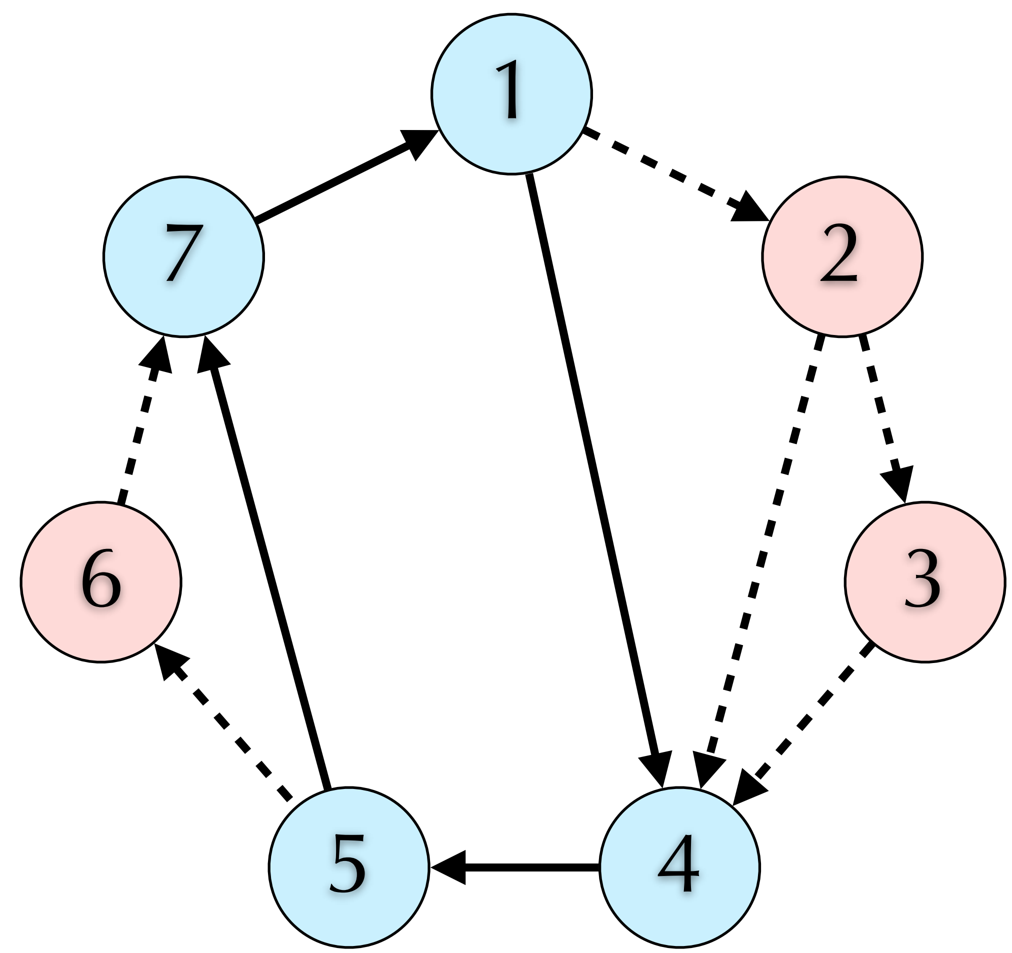
Linked list

- Find: O(k)
- Kill: O(1)
- Total running time: O(kn)



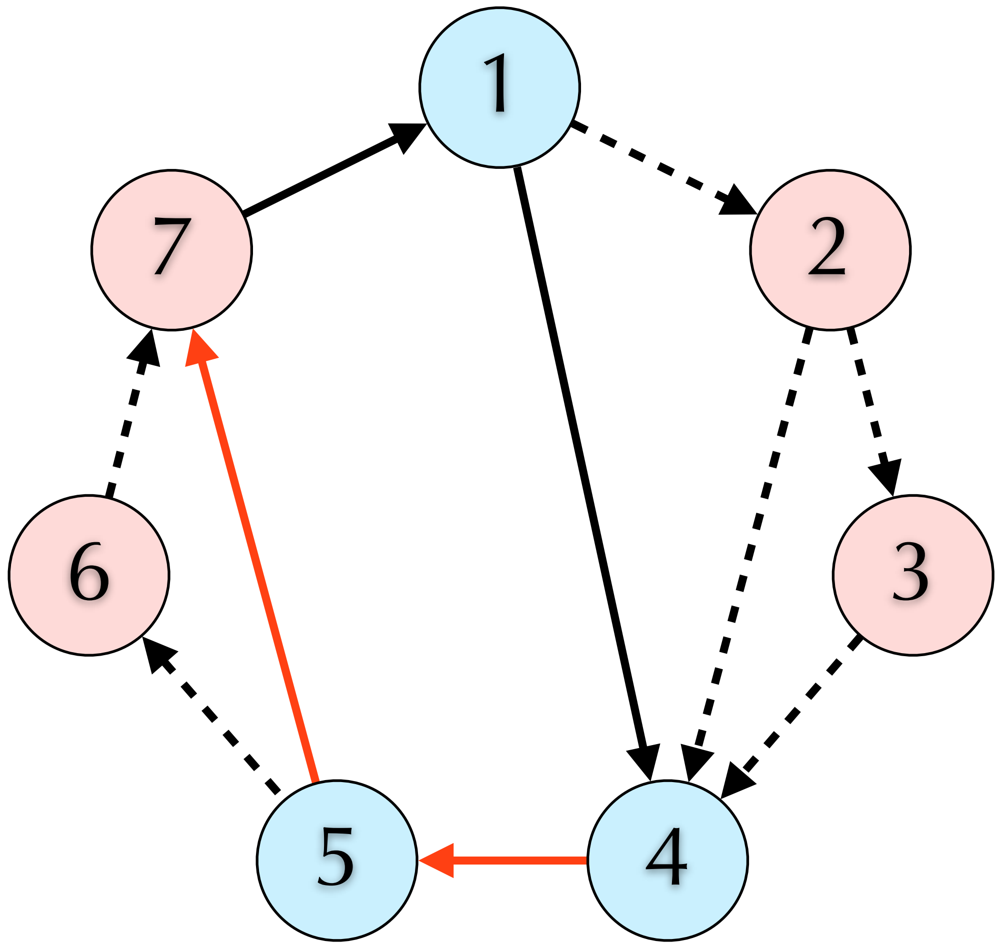
Linked list

- Find: O(k)
- Kill: O(1)
- Total running time: O(kn)



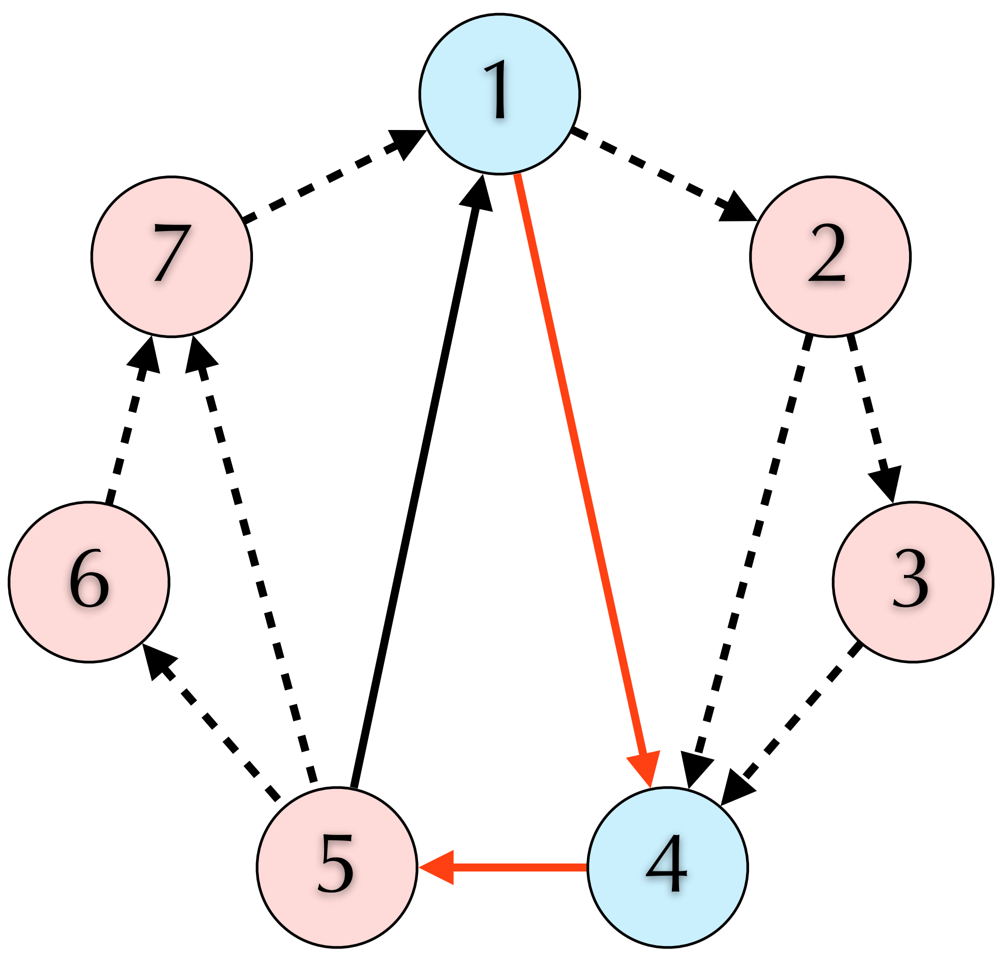
Linked list

- Find: O(k)
- Kill: O(1)
- Total running time: O(kn)



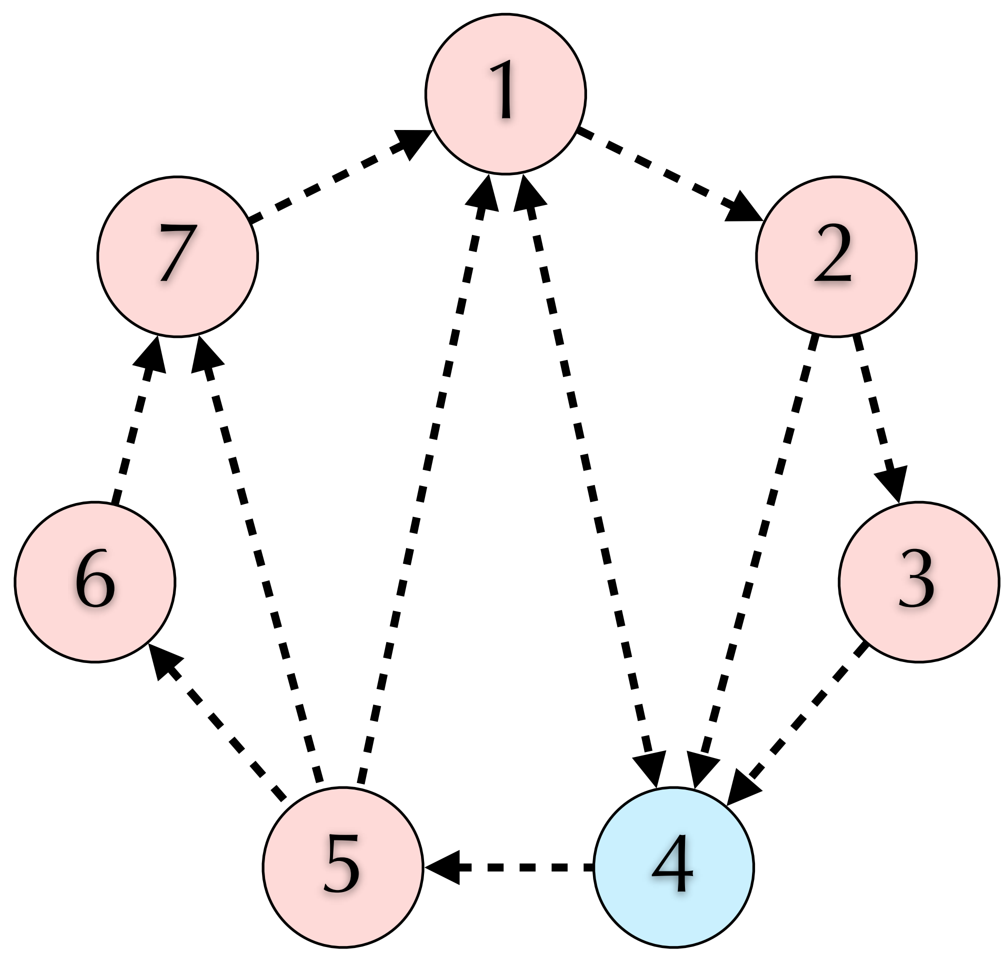
Linked list

- Find: O(k)
- Kill: O(1)
- Total running time: O(kn)



Linked list

- Find: O(k)
- Kill: O(1)
- Total running time: O(kn)



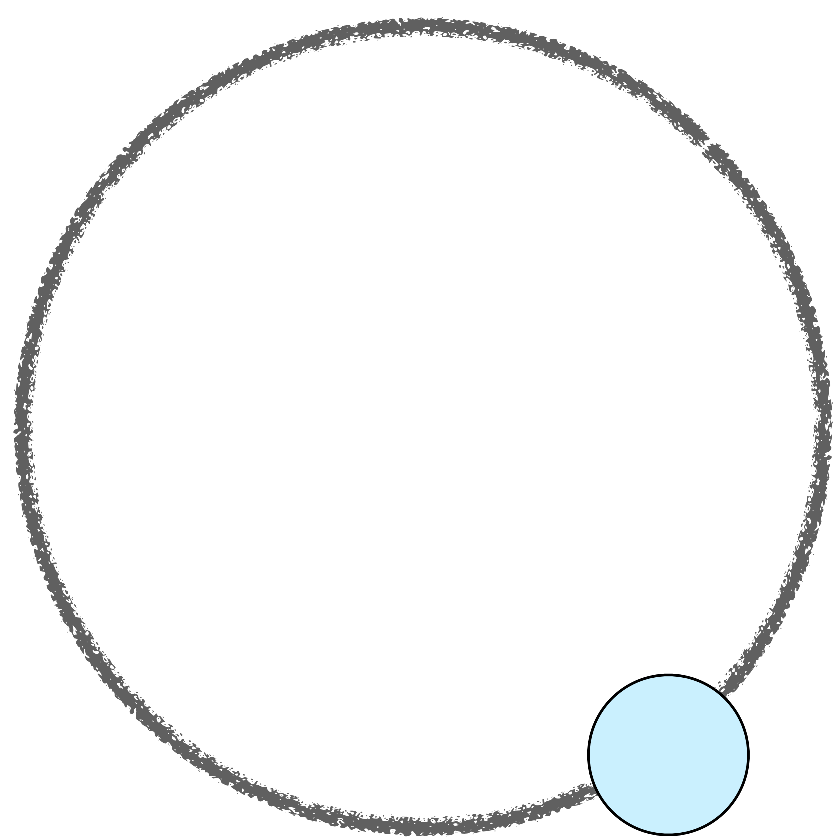
Other methods?

- Actually, Joseph problem can be solved in shorter time.
- It can be done in O(n) and ≤25 lines code.
- How?
  - Backward simulation

n=7, k=3

Backward simulation

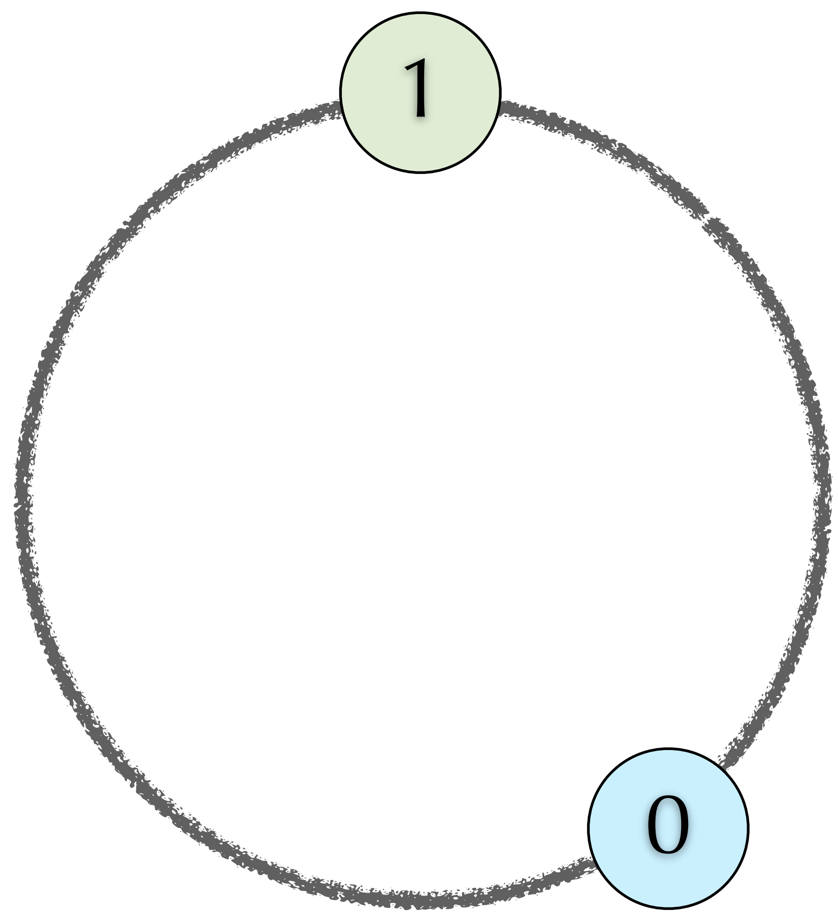
- At the end, there is only the survivor.



n=7, k=3

Backward simulation

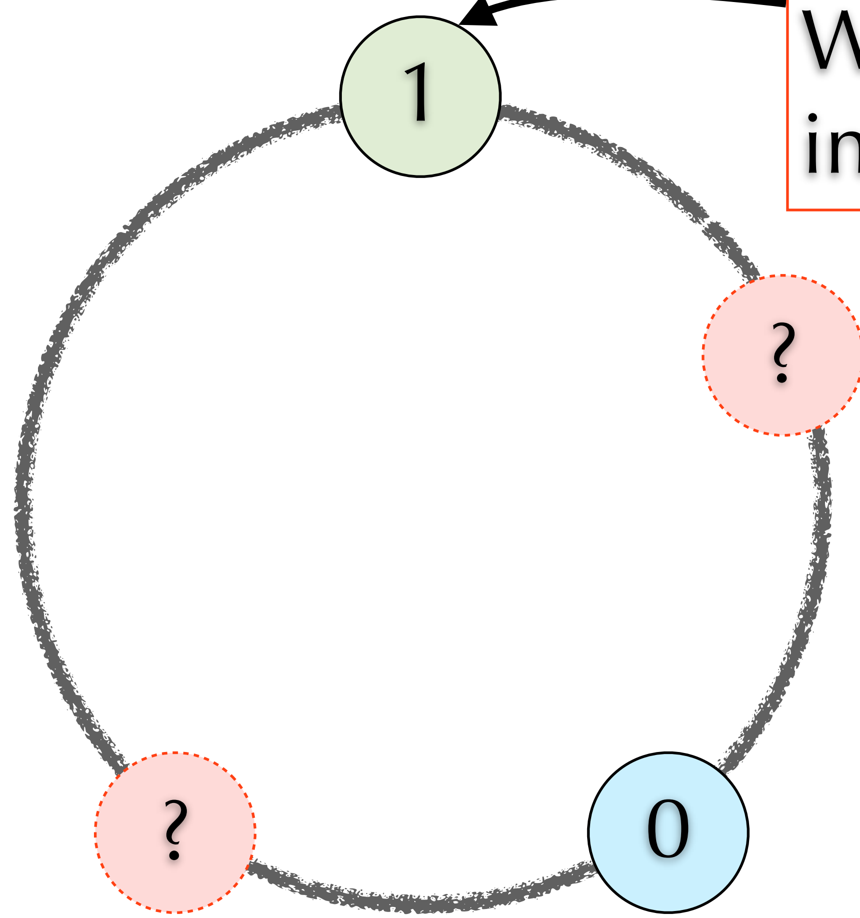
- 1 iteration before the end: there are 2 people.



n=7, k=3

Backward simulation

- 2 iterations before the end: there are 3 people. Where is the person killed in this iteration?

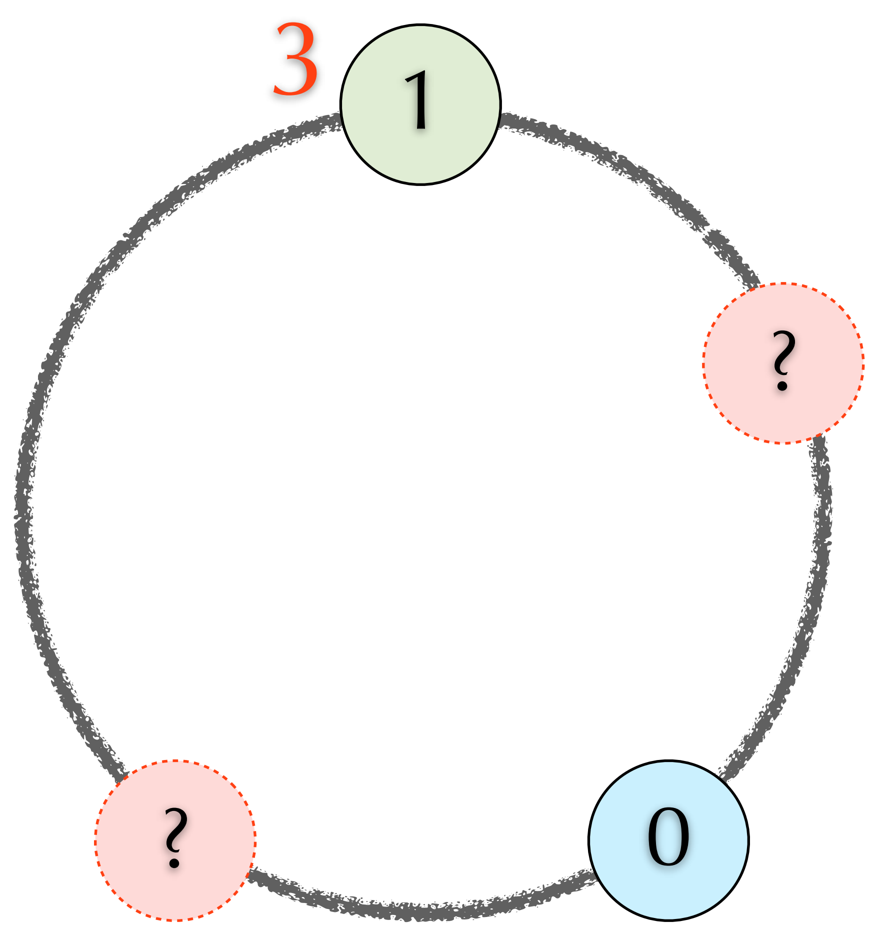


Why is this person killed in the last iteration?

n=7, k=3

Backward simulation

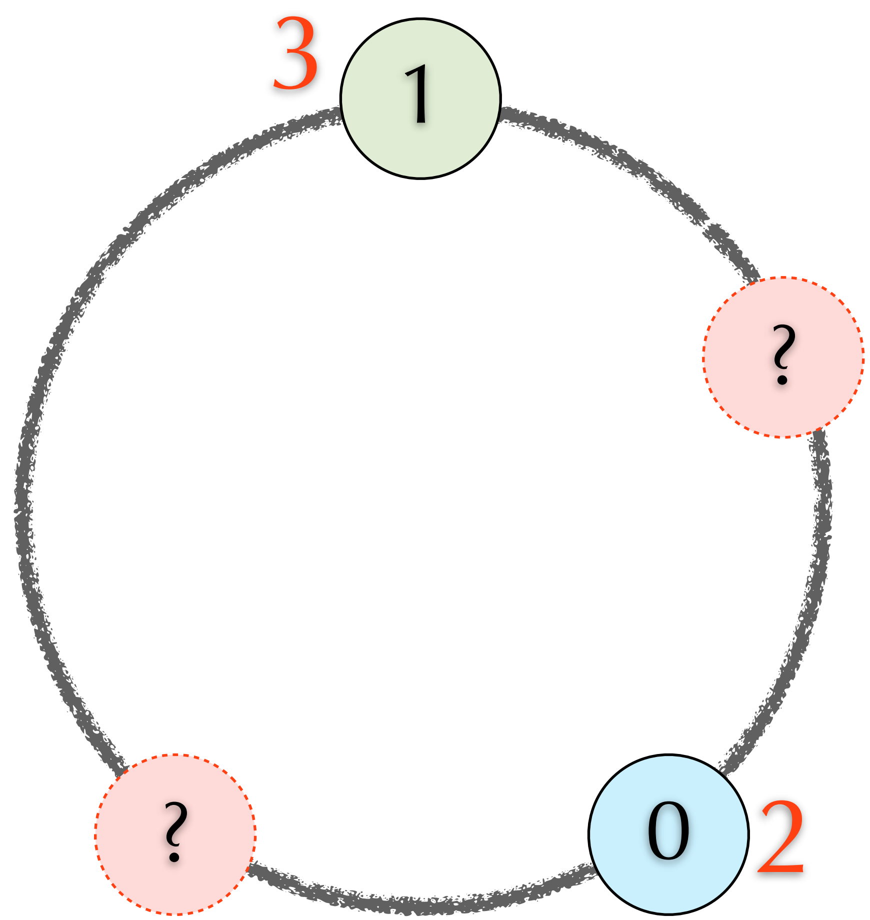
- 2 iterations before the end: there are 3 people. Where is the person killed in this iteration?



n=7, k=3

Backward simulation

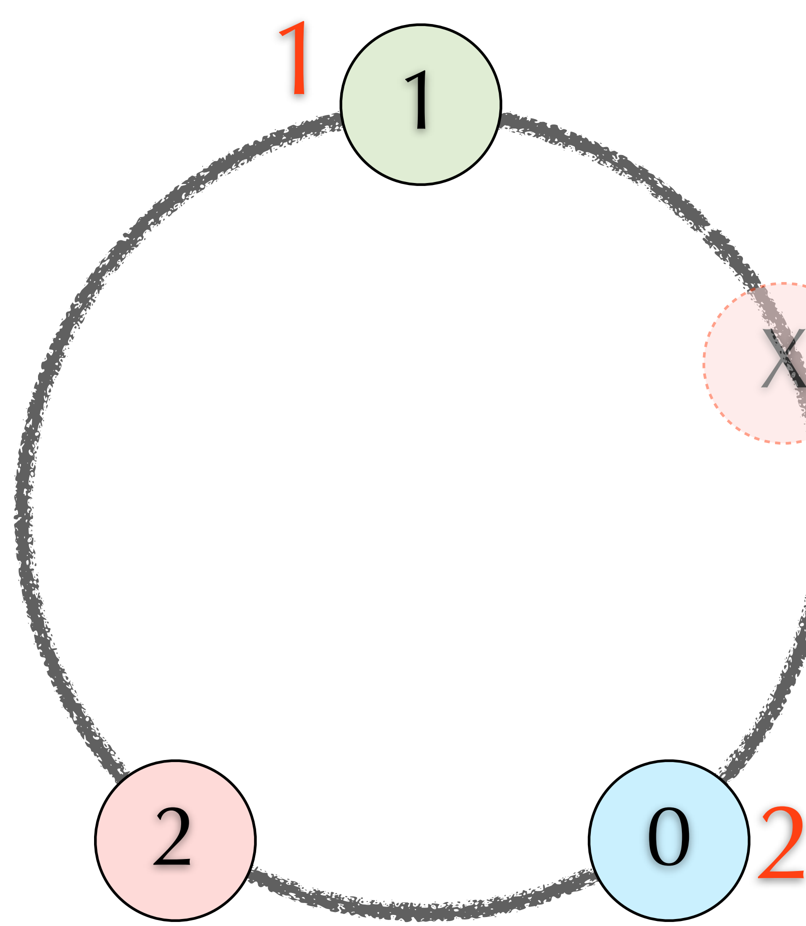
- 2 iterations before the end: there are 3 people. Where is the person killed in this iteration?



n=7, k=3

Backward simulation

- 2 iterations before the end: there are 3 people. Where is the person killed in this iteration?

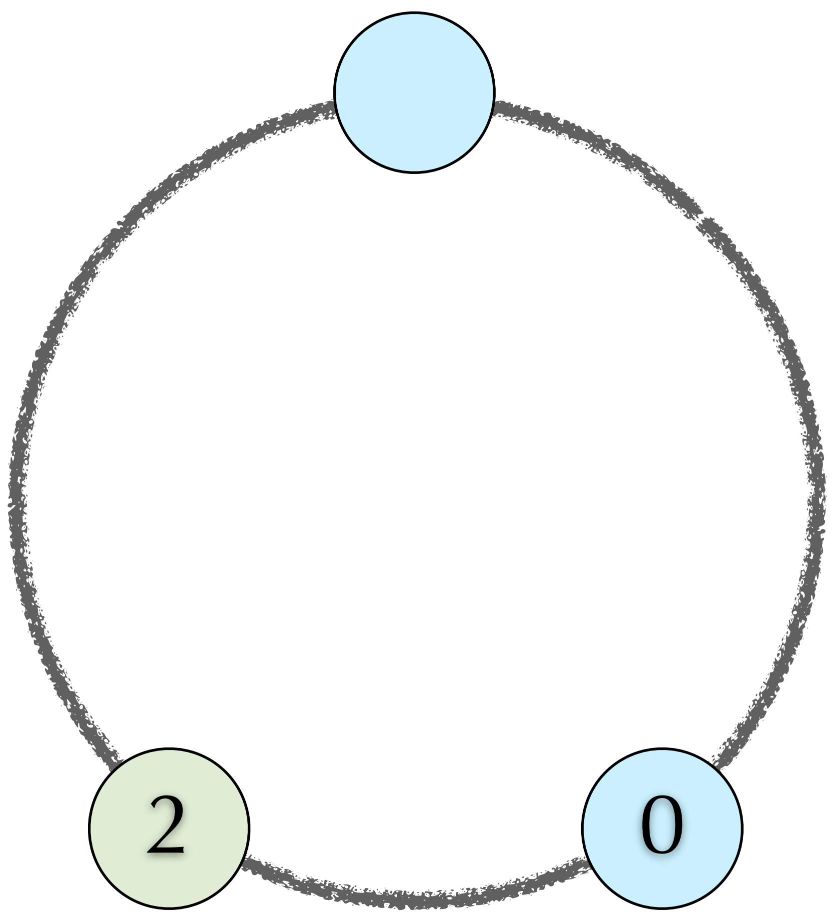


$1+3 \equiv 2 \pmod{2}$

n=7, k=3

Backward simulation

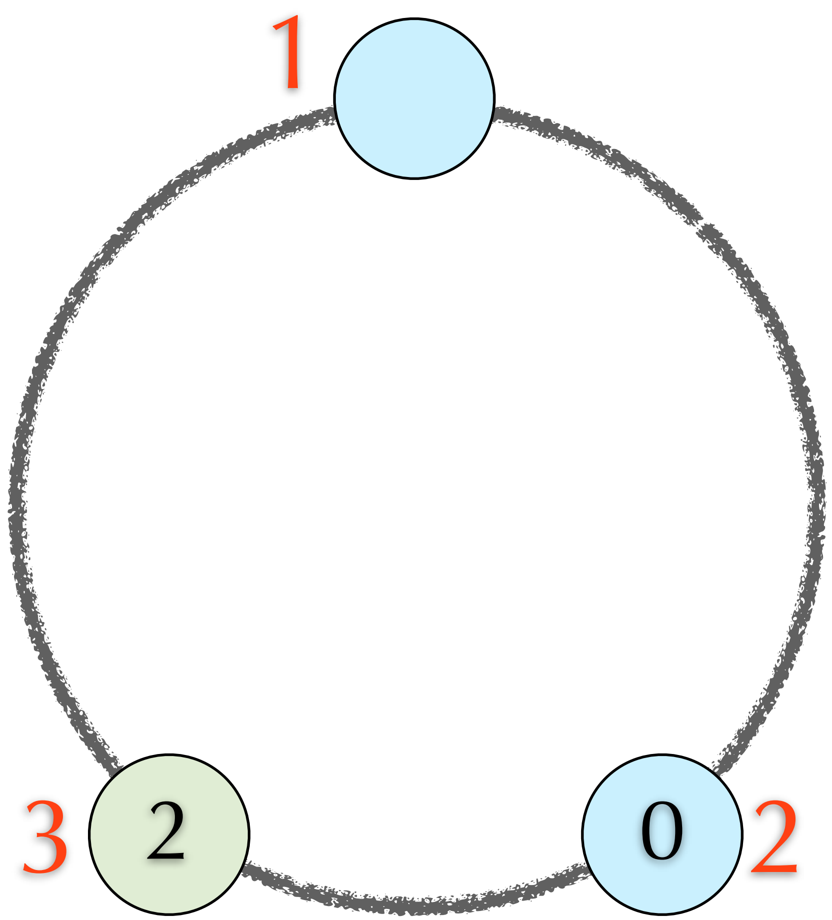
- 3 iterations before the end: there are 4 people. Where is the person killed in this iteration?



n=7, k=3

Backward simulation

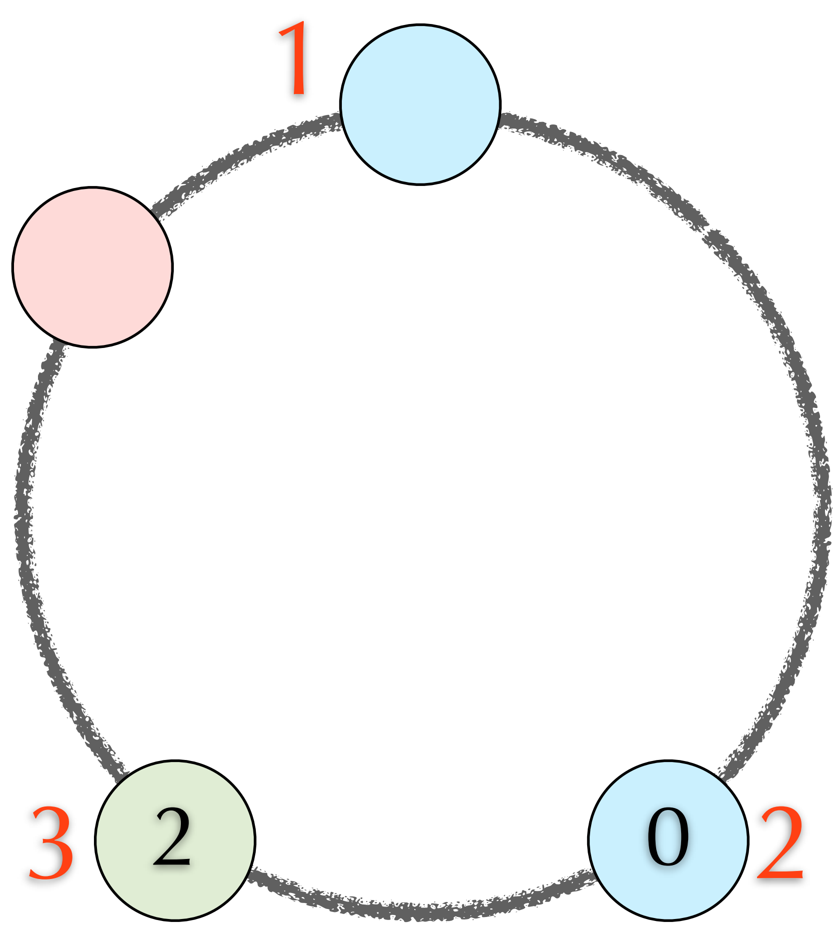
- 3 iterations before the end: there are 4 people. Where is the person killed in this iteration?



n=7, k=3

Backward simulation

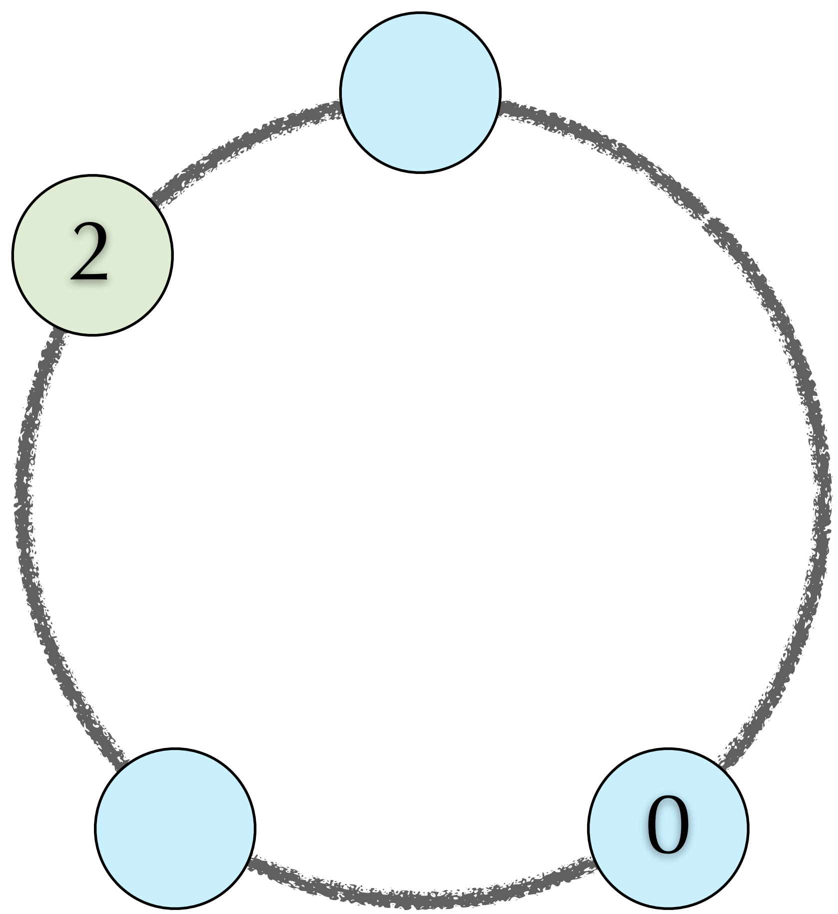
- 3 iterations before the end: there are 4 people. Where is the person killed in this iteration?



$2+3 \equiv 2 \pmod 3$

Backward simulation

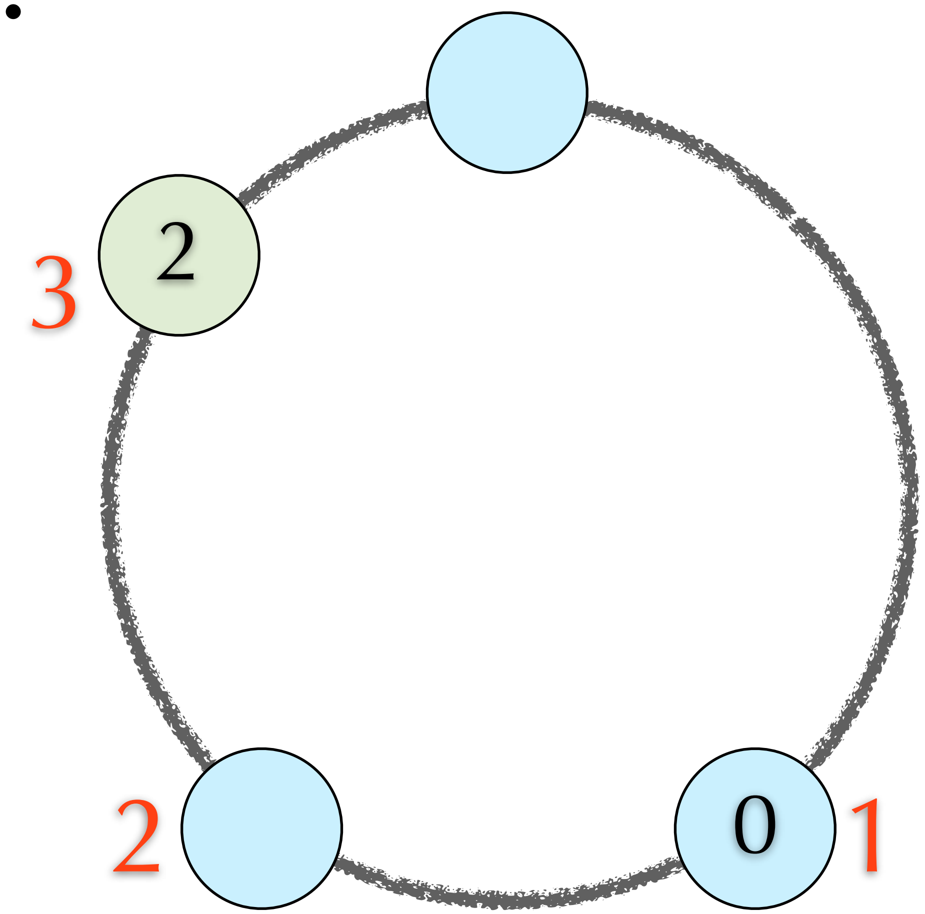
- 3 iterations before the end: there are 4 people. Where is the person killed in this iteration?



n=7, k=3

Backward simulation

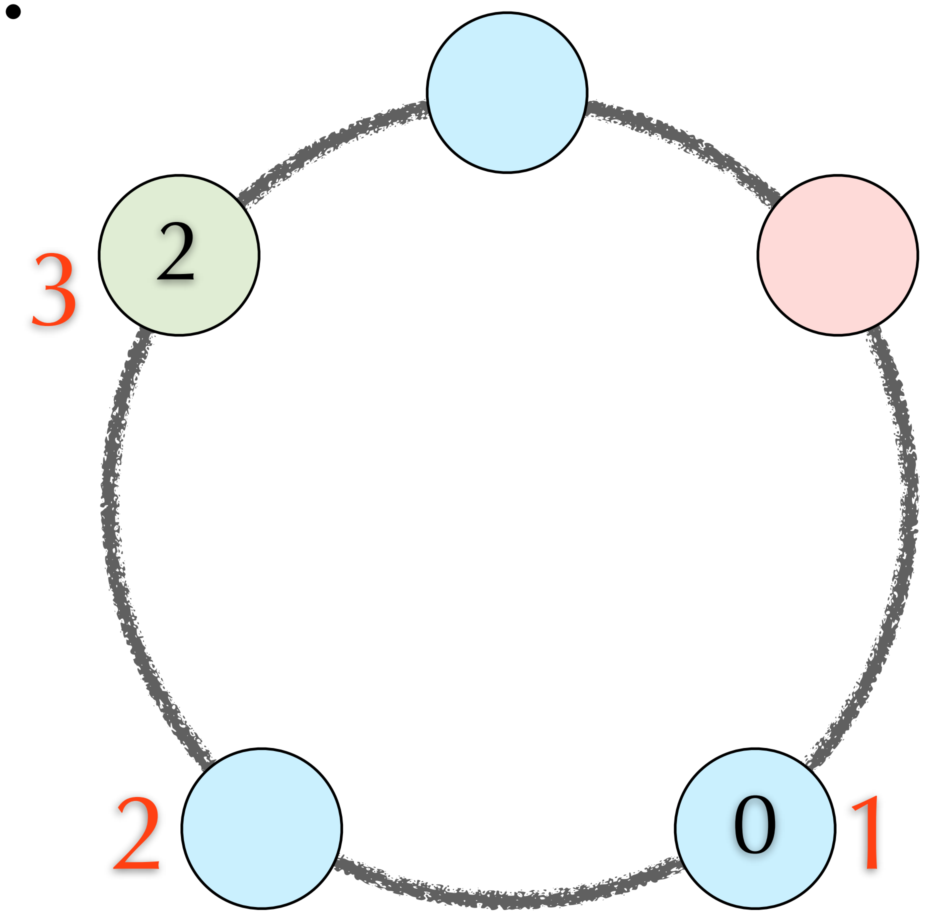
- 4 iterations before the end: there are 5 people. Where is the person killed in this iteration?



n=7, k=3

Backward simulation

- 4 iterations before the end: there are 5 people. Where is the person killed in this iteration?

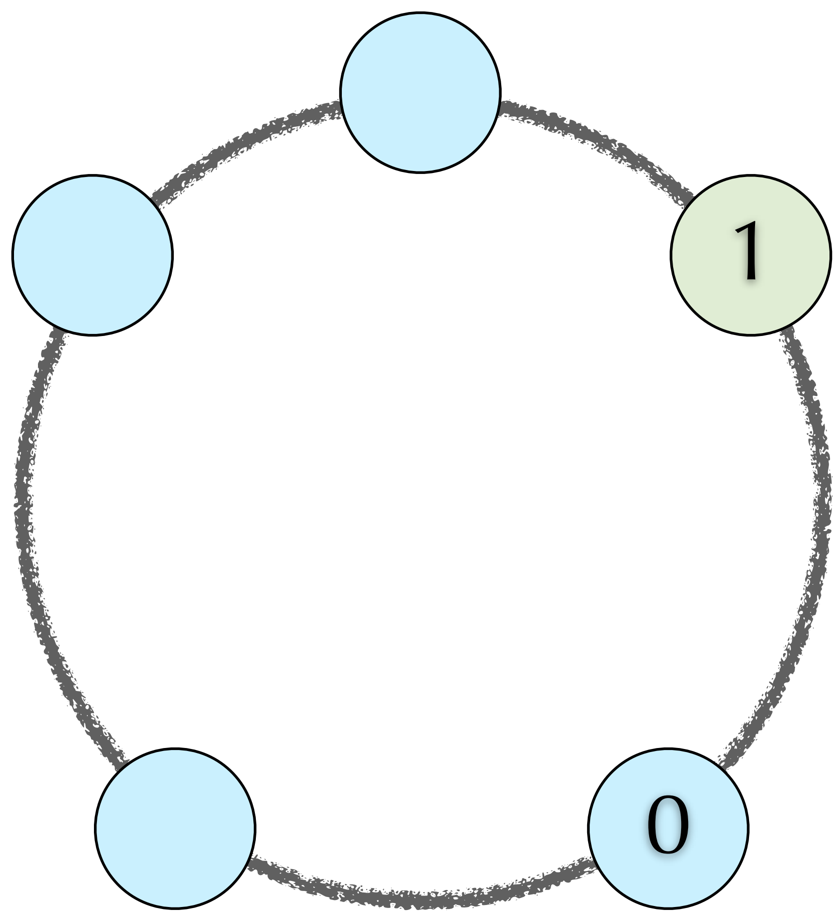


$2+3 \equiv 1 \pmod 4$

n=7, k=3

Backward simulation

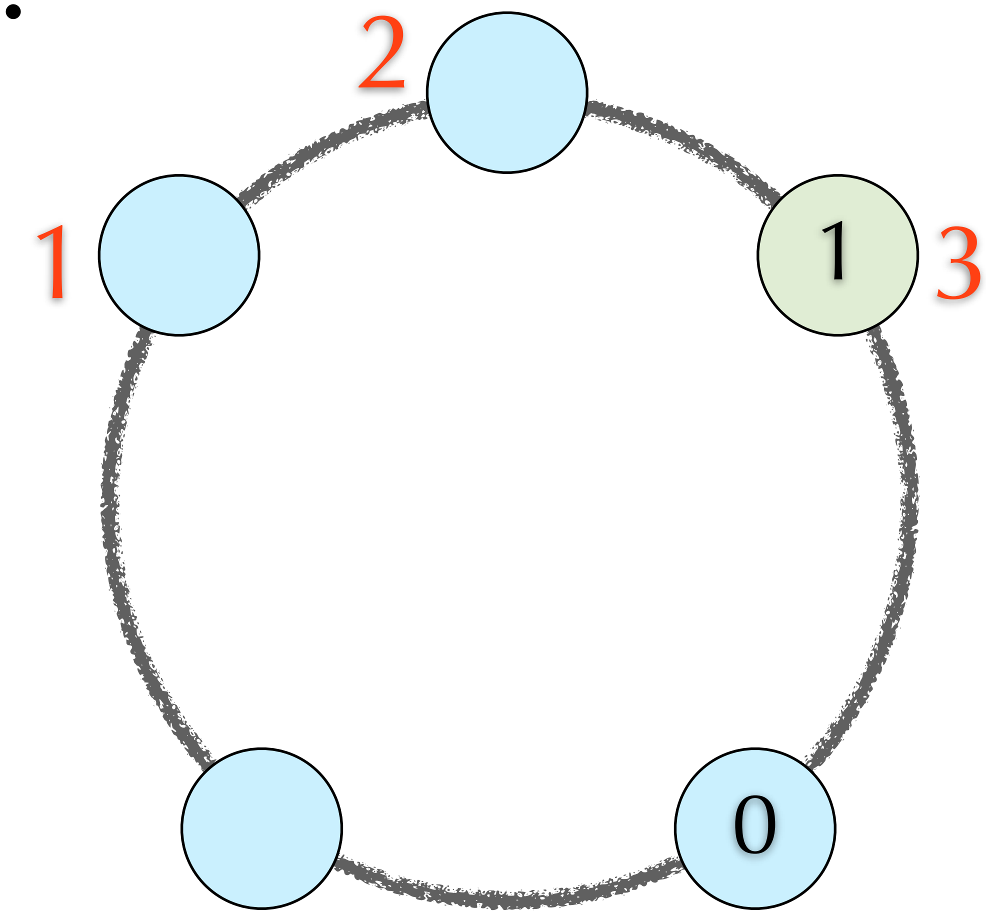
- 4 iterations before the end: there are 5 people. Where is the person killed in this iteration?



n=7, k=3

Backward simulation

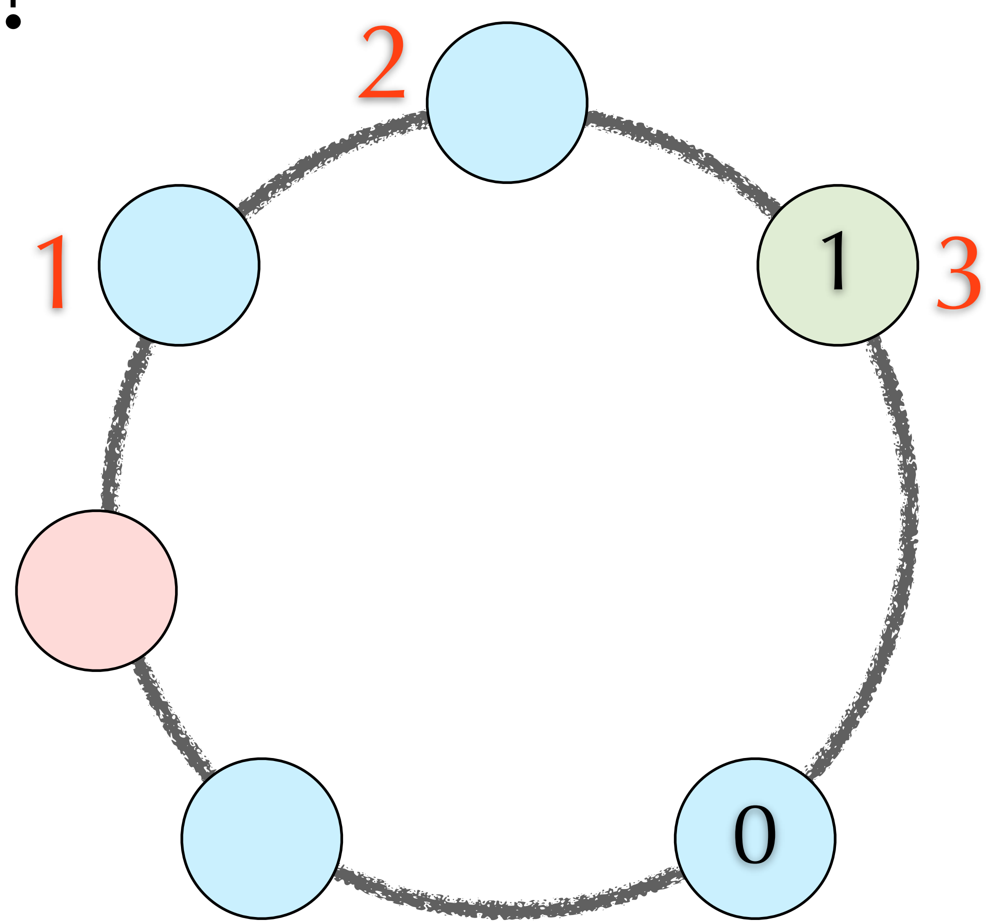
- 5 iterations before the end: there are 6 people. Where is the person killed in this iteration?



n=7, k=3

Backward simulation

- 5 iterations before the end: there are 6 people. Where is the person killed in this iteration?

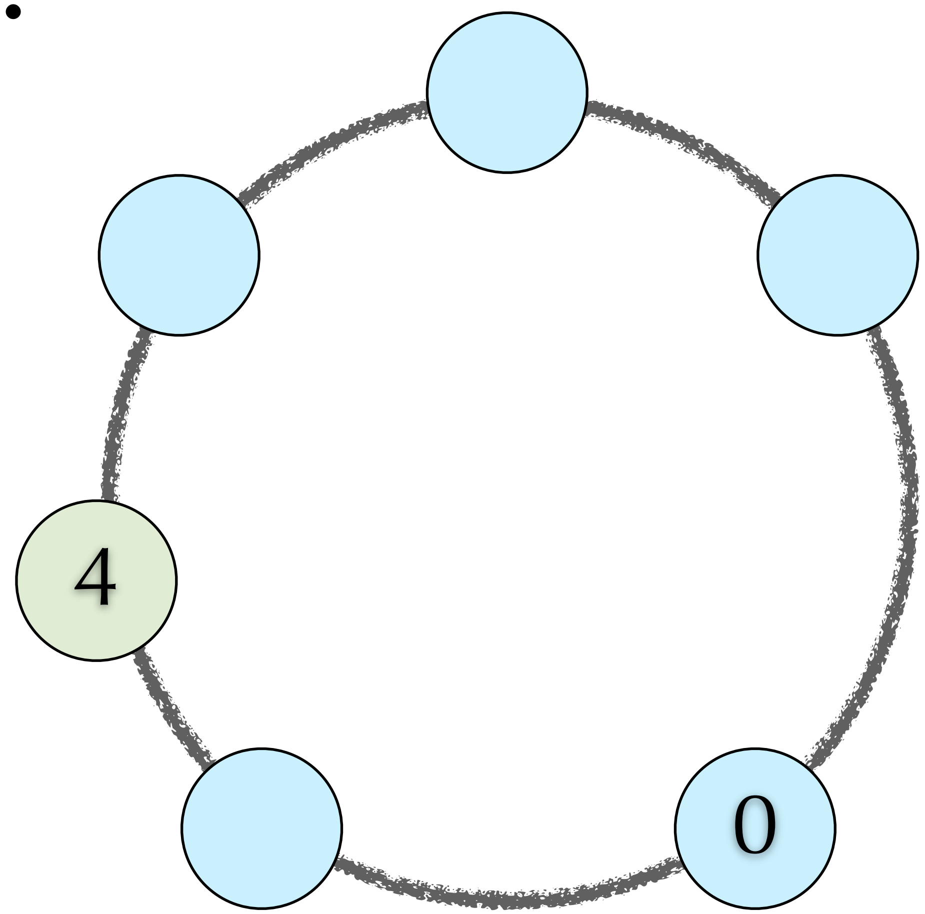


$1+3 \equiv 4 \pmod 5$

n=7, k=3

Backward simulation

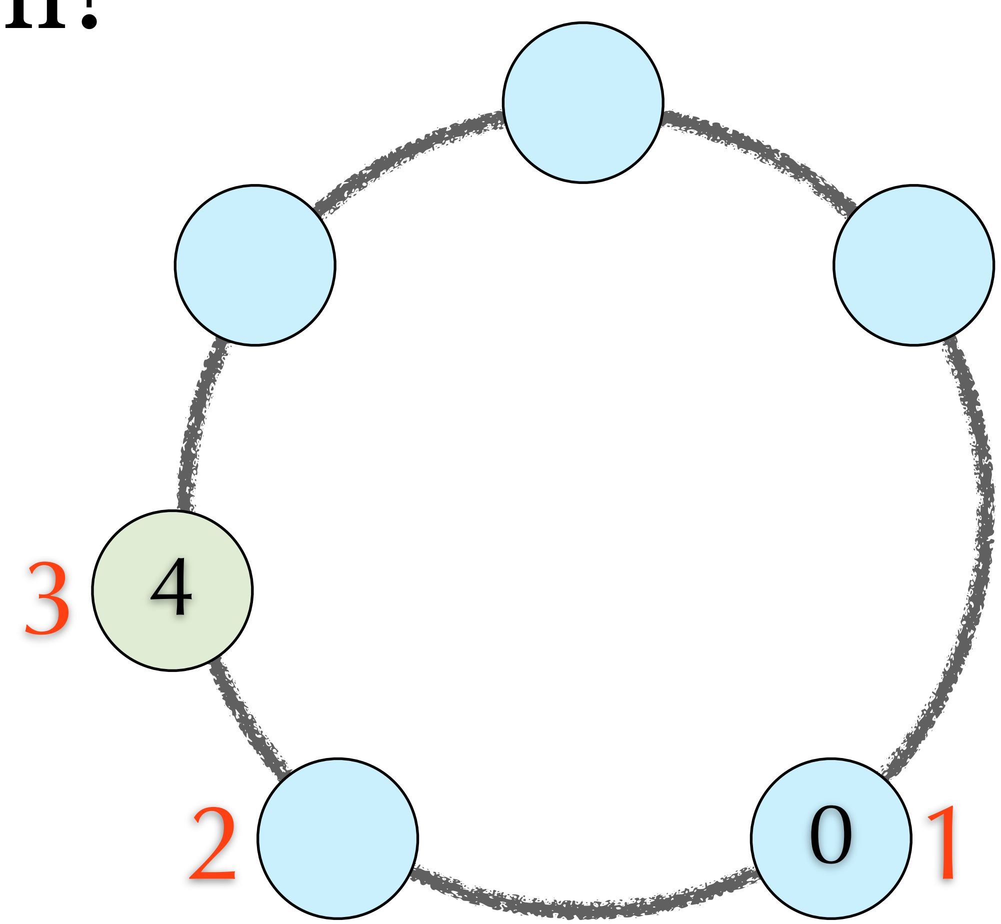
- 5 iterations before the end: there are 6 people. Where is the person killed in this iteration?



n=7, k=3

Backward simulation

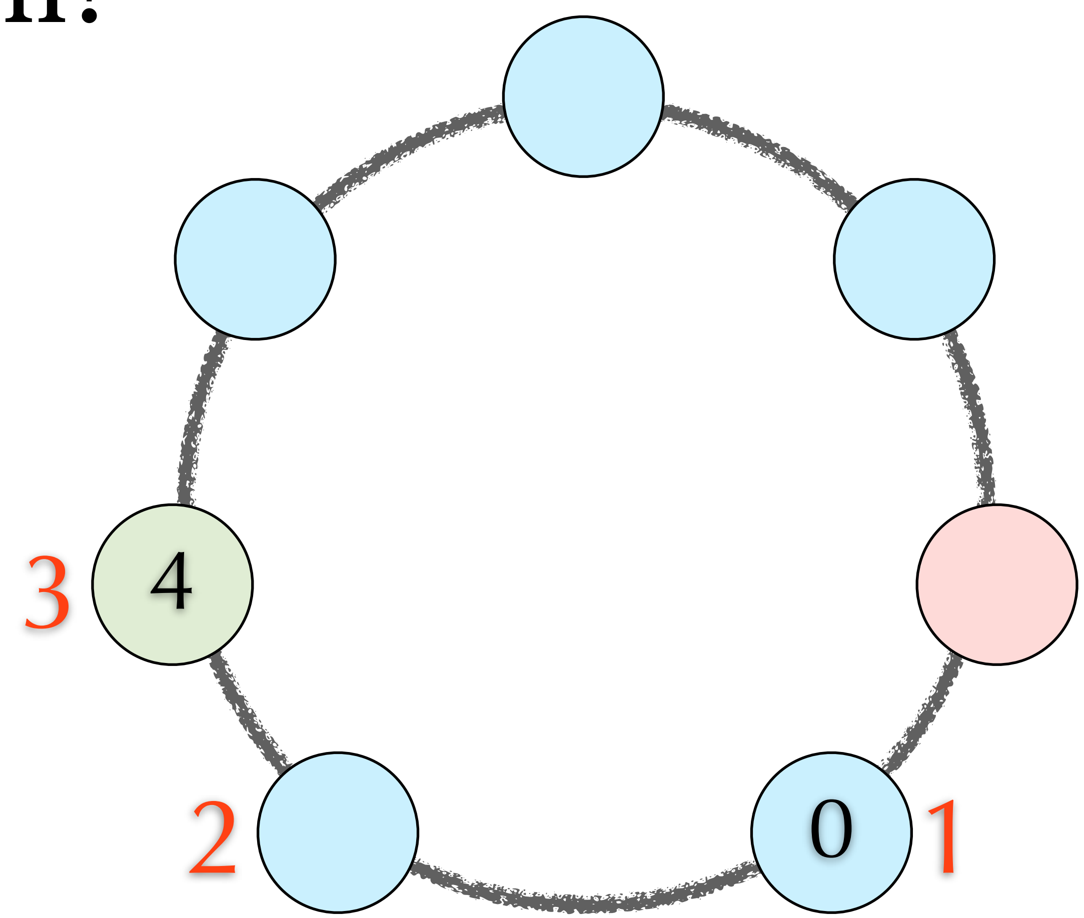
- 6 iterations before the end: there are 7 people. Where is the person killed in this iteration?



n=7, k=3

Backward simulation

- 6 iterations before the end: there are 7 people. Where is the person killed in this iteration?

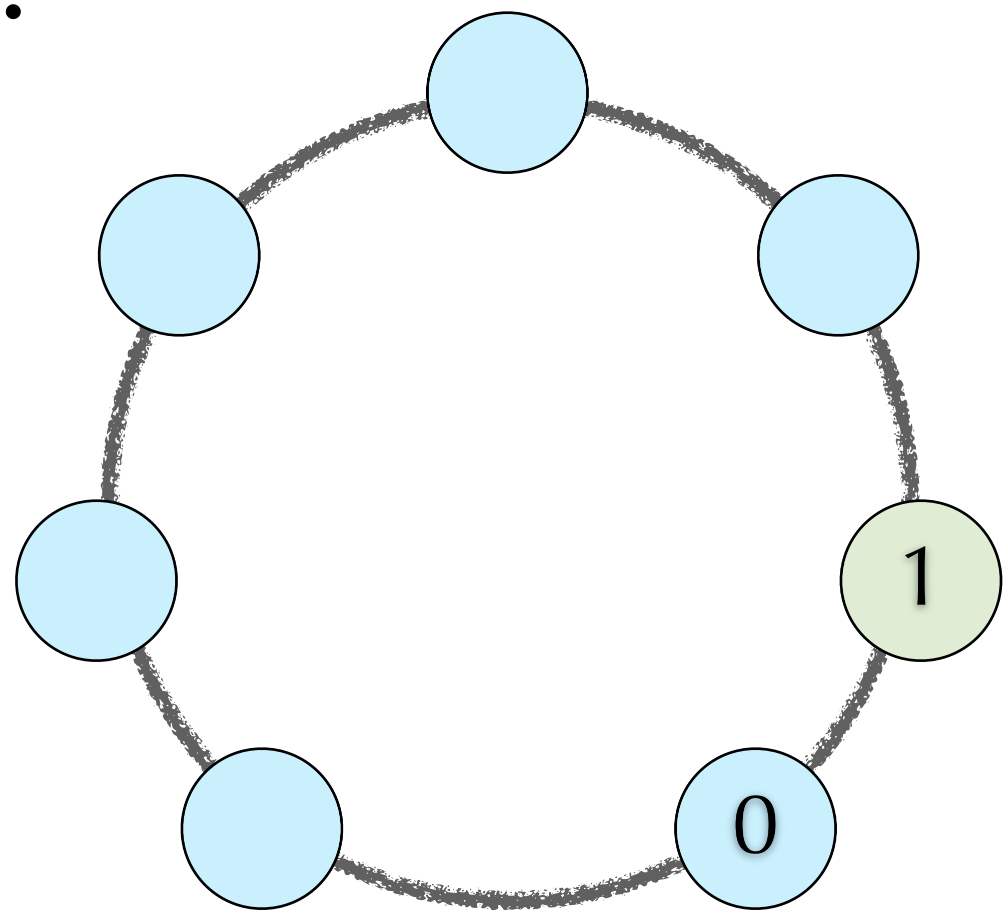


$4+3 \equiv 1 \pmod 6$

n=7, k=3

Backward simulation

- 6 iterations before the end: there are 7 people. Where is the person killed in this iteration?



Ans: 1+3=4