Linked Lists

Ulises Tirado Zatarain [1]
(ulises.tirado@cimat.mx)

[2]Algorists Group

April, 2016

Outline

A little problem: Josephus challenge

Statement

In the Romano-Jewish conict of 67 A. D., the Romans took the town Jotapata which Josephus was commanding. He and his companions escaped and were trapped in a cave. Fearing capture they decided to kill themselves. Josephus and a friend did not agree with that proposal but were afraid to be open in their opposition. So they suggested that they should arrange them in a circle and that counting around the circle in the same sense all the time, every third man should be killed until there was only one survivor who would kill himself. By choosing the position 31 and 16 in the circle, Josephus and his companion saved their lives.

A little problem: Josephus challenge

Input

The first line contains T, this represents the number of test cases. Each of following T lines contains two integers n and m, where n is the number of people in the cave and m is number of people that the group need to count in the circle to kill one.

Output

For each test case you need print single line with two integers indicating positions (in ascending order) in which Josephus and his friend need to set to survive.

Example

Sample input	Sample output
1	2 8
9 5	

Explanation

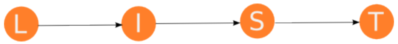
In this sample, the people will die in following order 5 1 7 4 3 6 9 2 8, then if Josephus and his friend be set at positions 2 and 8 they will survive.

Big question!

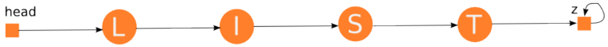
How can we solve it?
Some ideas...?

Definitions

- A **Linked List** is a recursive and linear data structure that can change its size dynamically. Also, is very flexible to insert, delete and reorder elements.

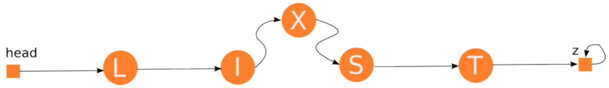


- So, each element needs to save a link to next element in the list. Then, to know when reach the end of the list we need a marker with its next element pointing itself.
- It's a nice practice having a marker at the begin of the list too.

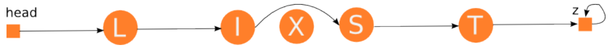


Definitions

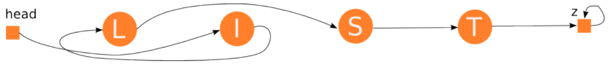
- Insertion of an element in the list.



- Deleting an element from the list.



- Reordering elements.



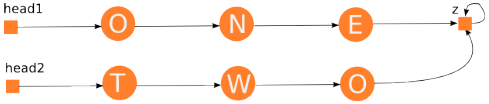
How to represent lists?

- The most simple way to represent a linked list is with an struct like following:

```
struct node <typename Type = int> begin
    Type data;
    struct node *next;
end
struct node *head, *z;
head = new node;
z = new node;
head->next = z;
z->next = z;
```

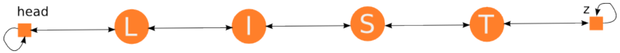
Shared memory and variants

- We can use only one z node for all list in our programs.



- Another posibility is having a bidirectional list.

```
**struct** node <typename Type = int> **begin**
    Type data;
    **struct** node *prev,*next;
**end**
```



How to add new key at the end of the list?

- To add a key, we can perform following algorithm:

```
function add-end-list(struct node *head , Type key) begin
    struct node $*t \leftarrow$ head;
    while $t->next \neq z$ do
        $t \leftarrow t->next$;
    end
    struct node $*q \leftarrow$ new node;
    $q->data \leftarrow key$;
    $t->next \leftarrow q$;
    $q->next \leftarrow z$;
end
```

How to remove an element of the list?

- To add a key, we can perform following algorithm:

```
function remove-from-list(struct node *head , Type key) begin
    struct node $*t \leftarrow$ head;
    while $t->next->data \neq key$ do
        $t \leftarrow t->next$;
    end
    struct node $*q \leftarrow t->next$;
    $t->next \leftarrow q->next$;
    delete q;
end
```

More usual operations

- Adding new key after/before some key.
- Adding/Removing key in specific position.
- Search an element.
- Removing first/last element.
- Adding at the head.
- ...

Josephus: Big question again!

How can we solve the initial problem with this?
Some ideas...?

Josephus: The idea

- What about if pointer next of last element to first one?
- Circular list.
- You only need to simulate.
- Another idea?

Josephus: The idea

- What about if pointer next of last element to first one?
- Circular list.
- You only need to simulate.
- Another idea?

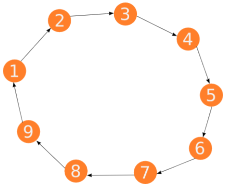
Josephus: The idea

- What about if pointer next of last element to first one?
- Circular list.
- You only need to simulate.
- Another idea?

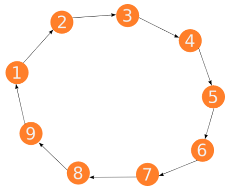
Josephus: The idea

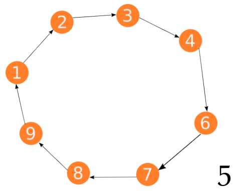
- What about if pointer next of last element to first one?
- Circular list.
- You only need to simulate.
- Another idea?

Josephus: The idea $n = 9$, $m = 5$

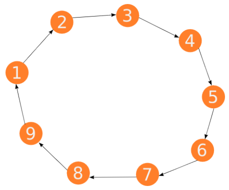


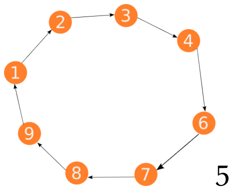
Josephus: The idea $n = 9$, $m = 5$

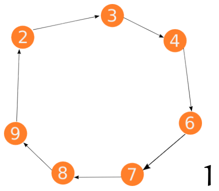




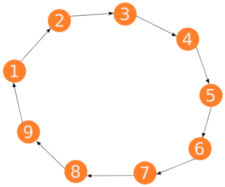
Josephus: The idea $n = 9$, $m = 5$

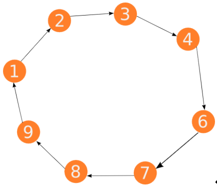




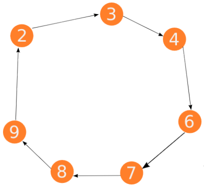


Josephus: The idea $n = 9$, $m = 5$

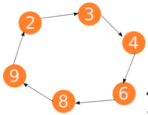




5

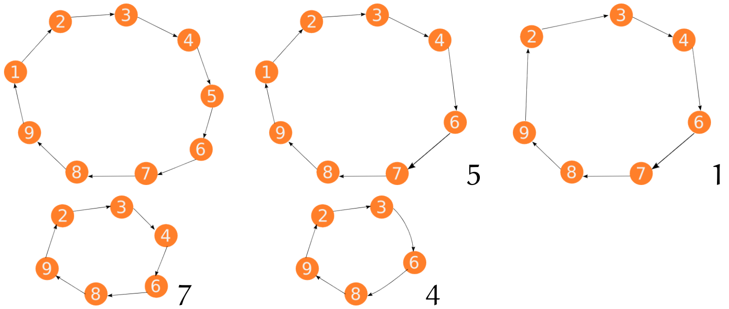


1

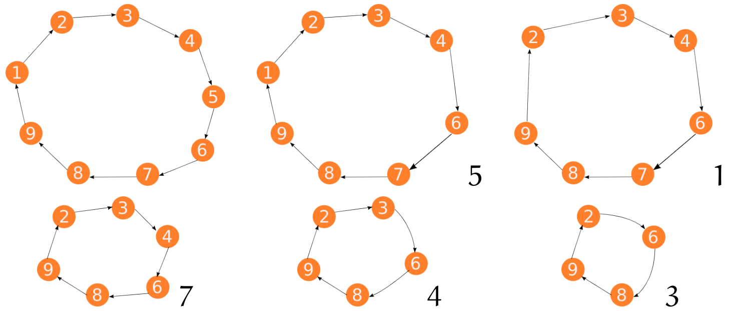


7

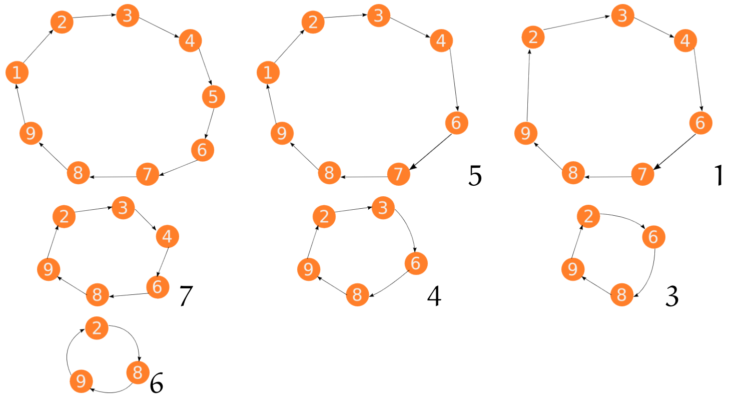
Josephus: The idea $n = 9$, $m = 5$



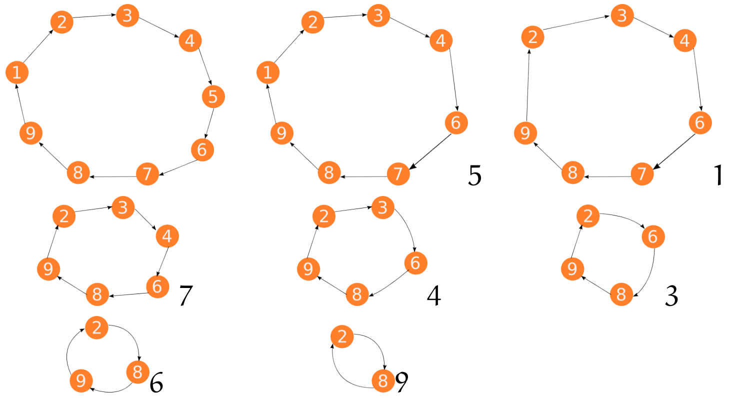
Josephus: The idea $n = 9$, $m = 5$



Josephus: The idea $n = 9$, $m = 5$



Josephus: The idea $n = 9$, $m = 5$



More aplications & variants

- Base of other data structures
  - Stacks
  - Queues
  - Deques
  - Graphs
  - Polygons
- Sparse vectors & sparse matrix

References I

- Codeforces
- TopCoder
- Wikipedia
- Quora
- HackerRank