Project-1

(CSE-220-Data Structure and Algorithm)

Title: ***The Josephus Problem- Solving using a Circular Linked List***

Made By:
Ankit Vadehra(12BCE0282)
Poorva Arora(1BCE0277)

Introduction:

In computer science and mathematics, the **Josephus Problem** (or **Josephus permutation**) is a theoretical problem related to a certain counting-out game.

There are people standing in a circle waiting to be executed. The counting out begins at some point in the circle and proceeds around the circle in a fixed direction. In each step, a certain number of people are skipped and the next person is executed. The elimination proceeds around the circle (which is becoming smaller and smaller as the executed people are removed), until only the last person remains, who is given freedom.

The task is to choose the place in the initial circle so that you are the last one remaining and so survive.

Imagine a game where N people stand on a circle. Every third person is eliminated from the circle until only one person remains. For example, with 6 people in the circle (labeled 1 through 6), the order of elimination would be:

3, 6, 4, 2, 5

and player 1 would be the last person standing.

The best method to approach this problem is by using a Circular-Linked-List.

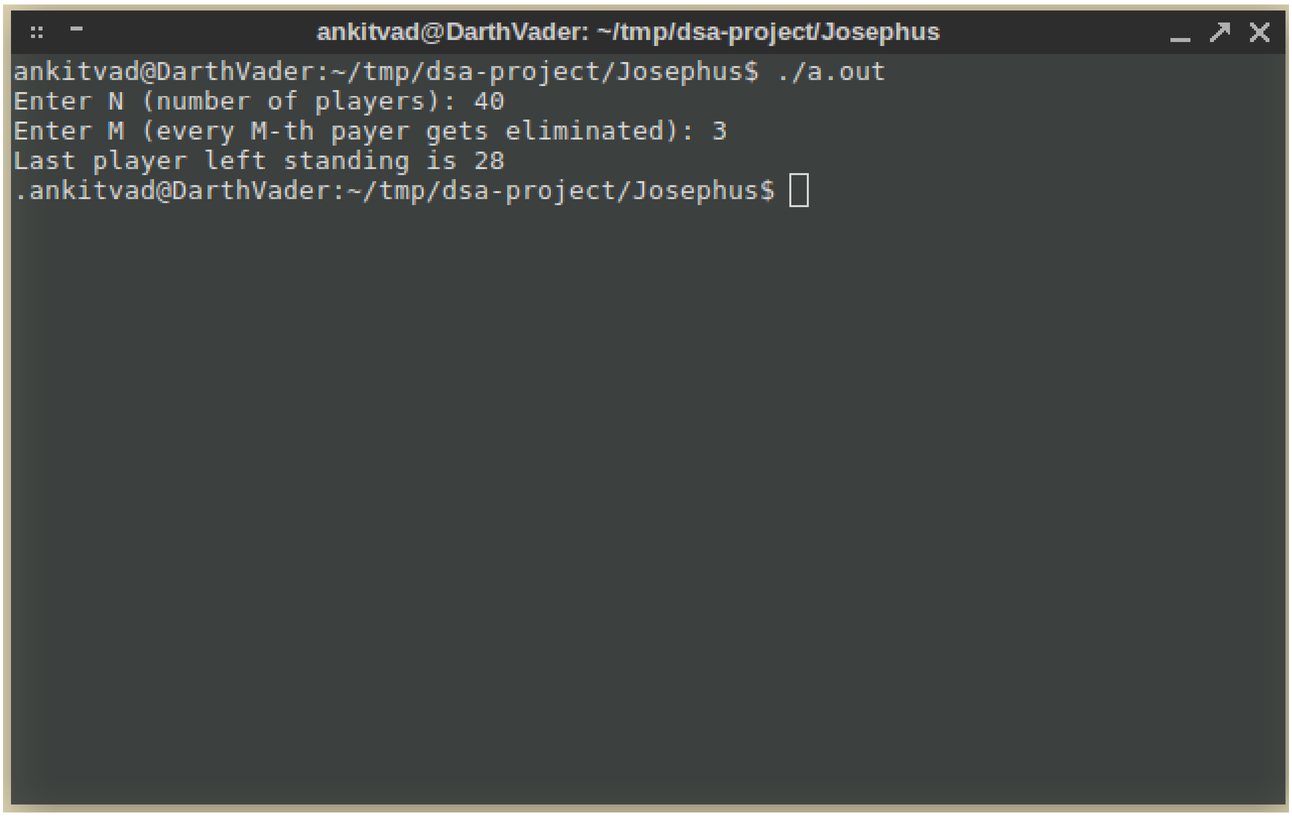
That way we can dynamically eliminate the alues and keep changing the list as we proceed.

Code:

```
#include <stdio.h>
#include <stdlib.h>



int main()
{
    int N, M;
    struct node { int player_id; struct node *next; };
    struct node *p, *q;
    int i, count;

    printf("Enter N (number of players): "); scanf("%d", &N);
    printf("Enter M (every M-th payer gets eliminated): "); scanf("%d", &M);

    // Create circular linked list containing all the players:
```

```
    p = q = malloc(sizeof(struct node));

    p->player_id = 1;

    for (i = 2; i <= N; ++i) {
        p->next = malloc(sizeof(struct node));
        p = p->next;
        p->player_id = i;
    }

    p->next = q;  // Close the circular linked list by having the last node point to the
                     first.

    // Eliminate every M-th player as long as more than one player remains:

    for (count = N; count > 1; --count) {
       for (i = 0; i < M - 1; ++i)
          p = p->next;

       p->next = p->next->next;  // Remove the eiminated player from the circular linked
                                    list.
    }

    printf("Last player left standing is %d\n.", p->player_id);

    return 0;
}
```

Output:



```
ankitvad@DarthVader:~/tmp/dsa-project/Josephus$ ./a.out
Enter N (number of players): 40
Enter M (every M-th payer gets eliminated): 3
Last player left standing is 28
.ankitvad@DarthVader:~/tmp/dsa-project/Josephus$ 
```