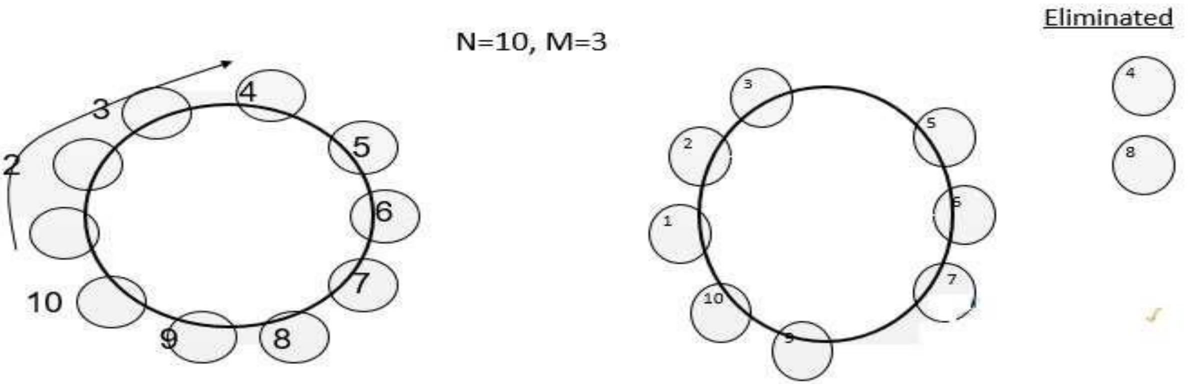
DSA Assignment 02

Name:	Syed Muhammad Raza Ali
Enrolment:	02-134231-028
Course:	DSA
Faculty:	Miss Lubna Siddiqui

Scenario **CLO-4, PLO-4, C6**

N people, numbered 1 to N are sitting in a circle. Starting at person 1, a hot potato is passed. After M passes the person holding the hot potato is eliminated, the circle closes ranks and the game continues with the person who was sitting after the eliminated person picking up the hot potato. The last remaining person wins. Thus if *M=0* and *N=5*, players are eliminated in order and player 5 wins. If *M=1 and N=5* the order of elimination is 2,4,1,5



Design two different algorithms with different approaches to solve the above problem.

Create a code in C++ to implement both algorithms by applying appropriate data structure for general values of M and N.

Test your program with three different values of *M* and *N*.

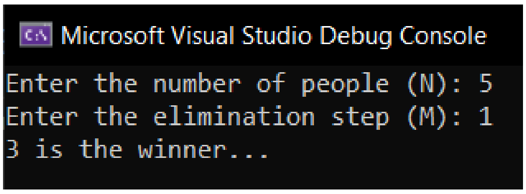
First Approach: Circular Linked List

Code:

```
#include <iostream>
using namespace std;

struct player {
    int data;
    player* next;
};

int eliminate(int N, int M) {
    if (N <= 0 || M < 0) return -1;

    player* head = new player;
    head->data = 1;
    head->next = nullptr;

    player* current = head;

    for (int i = 2; i <= N; i++) {
        player* newplayer = new player;
        newplayer->data = i;
        newplayer->next = nullptr;
        current->next = newplayer;
        current = newplayer;
    }

    current->next = head;

    player* prev = current;

    while (current->next != current) {
        for (int i = 0; i < M; i++) {
            prev = current;
            current = current->next;
        }

        prev->next = current->next;
        player* nextplayer = current->next;
        delete current;
        current = nextplayer;
    }

    int winner = current->data;

    delete current;

    return winner + 1;
}

int main() {
    int N, M;

    cout << "Enter the number of people (N): ";
    cin >> N;
    cout << "Enter the elimination step (M): ";
    cin >> M;
    int winner = eliminate(N, M);

    cout <<winner << " is the winner..." <<  endl;


    return 0;
      }
```

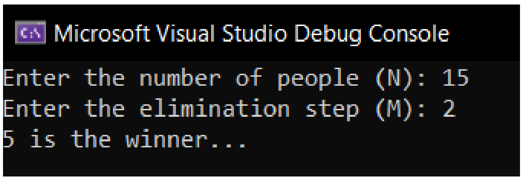
Output:

When M = 1 and N = 5



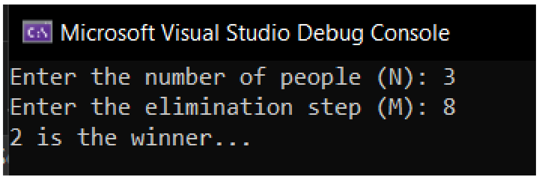
```
Enter the number of people (N): 5
Enter the elimination step (M): 1
3 is the winner...
```

When M = 3 and N = 15



```
Enter the number of people (N): 15
Enter the elimination step (M): 2
5 is the winner...
```

When M = 0 and N = 3



```
Enter the number of people (N): 3
Enter the elimination step (M): 8
2 is the winner...
```

Second Approach: Recursion

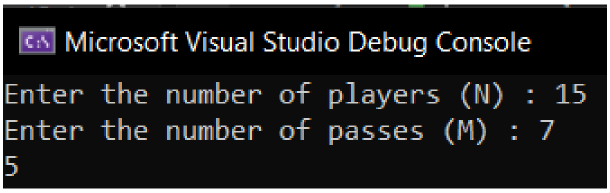
The Josephus Problem can also be solved by making a recursive function

Code:

```
#include <iostream>
using namespace std;

int eliminate(int N, int M) {
    if (N == 1) {
        return N;
    }
    else {
        int winner = eliminate(N-1, M);
        return (winner +M-1) % N + 1;
    }
}

int main() {

      int N = 5;
      int M = 1;
      cout << eliminate(N, M);

      return 0;
}
```

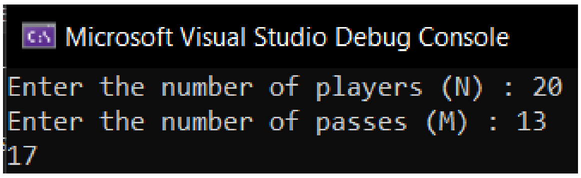
Output:

When N = 15 and M = 7



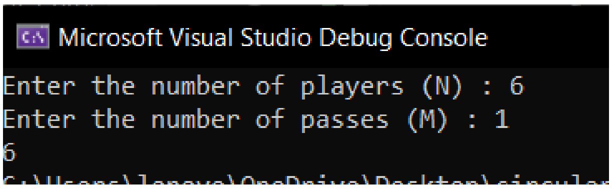
```
Enter the number of players (N) : 15
Enter the number of passes (M) : 7
5
```

When N = 20 and M = 13



```
Enter the number of players (N) : 20
Enter the number of passes (M) : 13
17
```

When N = 6 and M = 1



```
Enter the number of players (N) : 6
Enter the number of passes (M) : 1
6
C:\Users\lenovo\OneDrive\Desktop\circular
```