Question :

There is a group of students playing in the playground and joins each other's hands. Find if they forms a circle. If yes, print 1 else 0

Constraints:

1 <= N <= 1000

Input Description:

Number of Students - N, Number of connections - M
N student’s names
M students connections each in new line

Output Description:

Print 1, if they forms a circle

Solution:

```
class Node:
   def __init__(self, data):
       self.data = data
       self.next = None

class CircularLinkedList:
   def __init__(self):
       self.head = None
   def push(self, data):
       ptr1 = Node(data)
       temp = self.head
       ptr1.next = self.head
       if self.head is not None:
           while (temp.next != self.head):
               temp = temp.next
           temp.next = ptr1
       else:
           ptr1.next = ptr1  # For the first node
       self.head = ptr1

   def printList(self):
       temp = self.head
       if self.head is not None:
           while (True):
               print("%d" % (temp.data))
               temp = temp.next
               if (temp == self.head):
                   break
```

```
    def splitList(self, head1, head2):
        slow_ptr = self.head
        fast_ptr = self.head
        if self.head is None:
            return
        while (fast_ptr.next != self.head and
               fast_ptr.next.next != self.head):
            fast_ptr = fast_ptr.next.next
            slow_ptr = slow_ptr.next
        if fast_ptr.next.next == self.head:
            fast_ptr = fast_ptr.next
        head1.head = self.head
        if self.head.next != self.head:
            head2.head = slow_ptr.next
        fast_ptr.next = slow_ptr.next
        slow_ptr.next = self.head

head = CircularLinkedList()
head1 = CircularLinkedList()
head2 = CircularLinkedList()

head.push(12)
head.push(56)
head.push(2)
head.push(11)

print("Original Circular Linked List")
head.printList()

# Split the list
head.splitList(head1, head2)

print("\nFirst Circular Linked List")
head1.printList()

print("\nSecond Circular Linked List")
head2.printList()
```

Test Cases:

Test Case 1:
Input:
6 6
2 5 9 1 4 7
2 5
5 9
9 1
1 4
4 7
7 2
Output:
1

Test Case 2:
Input:
4 3
2 5 3 6
2 5
5 3
3 6
6 2
Output:
1

Test Case 3:
Input:
7 6
A B C G H R E
A B
B C
C G
G H
H R
R E
Output:
0

Test Case 4:
Input:
3 3
2 9 7
2 9
9 7
7 2
Output:
1

Test Case 5:
Input:
5 5
2 5 9 1 4
2 5
5 9
9 1
1 4
4 2
Output:
1