DSA_College\circular_linked_list.cpp

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

class Node
{
    public :
    int data ;
    Node* next ;
    Node(int data)
    {
        this->data = data ;
        this->next = NULL ;
    }
    ~Node()
    {
        int value = this->data ;
        while(this->next != NULL)
        {
            delete next ;
            next = NULL ;
        }
        cout<<"memory is free for node with data= "<<value<<endl;
    }
};

void insertNode(Node* &tail, int d, int element)
{
    if(tail == NULL) // linked list is empty
    {
        Node* temp = new Node(d) ;
        tail = temp ;
        temp->next = temp ; // single node wale ko bhi circular bna diya
    }
    else{ // assume that element is present in the list
        Node* curr = tail ;
        while(curr->data != element)
        {
            curr= curr->next ;
        }
        Node* temp = new Node(d) ;
        temp->next = curr->next ;
        curr->next = temp ;
    }
}

void print(Node* &tail)
{
    Node* temp = tail ;
    do
    {
        cout<<temp->data<<" ";
```

```
52          temp= temp->next ;
53      } while (temp != tail);
54      cout<<endl ;
55  }
56
57  void deleteNode(Node* &tail, int value)
58  {
59      if(tail == NULL)
60      {
61          cout<<"list is empty"<<endl;
62          return ;
63      }
64      else{
65          Node* prev= tail ;
66          Node* curr= prev->next ;
67          while(curr->data != value)
68          {
69              prev= curr ;
70              curr= curr->next ;
71          }
72          prev->next = curr->next ;
73          if(curr == prev)
74          {
75              tail = NULL ;
76          }
77          else if(tail == curr) // for two or more than two nodes case
78         {
79          tail = prev ;
80         }
81         curr->next = NULL ;
82         delete curr ;
83      }
84  }
85
86  int main()
87  {
88      Node* tail = NULL ;
89
90       // insert in empty list
91       insertNode(tail,5,3) ; // 5
92       print(tail) ;
93
94       insertNode(tail,3,5) ; // 5 3
95       print(tail) ;
96
97       insertNode(tail,5,3) ; // 5 3 5
98       print(tail) ;
99
100      deleteNode(tail,5) ;
101      print(tail) ;
102
103      return 0;
104  }
```