Intersection in C++

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

// Node class definition
class Node {
public:
    int data;
    Node* next;

    Node(int d) {
        data = d;
        next = nullptr;
    }
};

// Intersection2LL class definition
class Intersection2LL {
public:
    Node* head1;
    Node* head2;

    int getCount(Node* node) {
        Node* current = node;
        int count = 0;

        while (current != nullptr) {
            count++;
            current = current->next;
        }
        return count;
    }

    int getNode() {
        int c1 = getCount(head1);
        int c2 = getCount(head2);
        int d;
        if (c1 > c2) {
            d = c1 - c2;
            return getIntesectionNode(d, head1, head2);
        } else {
            d = c2 - c1;
            return getIntesectionNode(d, head2, head1);
        }
    }

    int getIntesectionNode(int d, Node* node1, Node*
node2) {
        Node* current1 = node1;
        Node* current2 = node2;

        for (int i = 0; i < d; i++) {
            if (current1 == nullptr) {
                return -1;
            }
            current1 = current1->next;
        }

        while (current1 != nullptr && current2 !=
nullptr) {
            if (current1->data == current2->data) {
                return current1->data;
```

Final Linked Lists

List 1	List 2
3 → 6 → 9 → 15 → 30	10 → 15 → 30

- Intersection starts at **node 15** (shared memory).

🔄 Dry Run of getNode()

1. Count Nodes

Operation	Result
Count of List 1	5
Count of List 2	3
d = c1 - c2	2

2. Advance Longer List by d = 2 Nodes

After Skipping in List 1	Current Node 1	Current Node 2
Skip 1st → 3	6	
Skip 2nd → 6	9	

Now:

- current1 = 9
- current2 = 10

🔁 Start Comparing Nodes

Step	current1->data	current2->data	Same Node Address?	Action
1	9	10	✗	Move both forward
2	15	15	✓✓✓	**Return 15**

✅ Output

The node of intersection is 15

📜 Summary Table

Phase	Details
Total Nodes in List1	5
Total Nodes in List2	3
Difference d	2
First match by addr	Node with data 15
Final Answer	15

```
            }
            current1 = current1->next;
            current2 = current2->next;
        }

        return -1;
    }
};

int main() {
    // Creating an instance of Intersection2LL
    Intersection2LL list;

    // Creating first linked list
    list.head1 = new Node(3);
    list.head1->next = new Node(6);
    list.head1->next->next = new Node(9);
    list.head1->next->next->next = new Node(15);
    list.head1->next->next->next->next = new
Node(30);

    // Creating second linked list
    list.head2 = new Node(10);
    list.head2->next = new Node(15);
    list.head2->next->next = new Node(30);

    // Finding the intersection node
    cout << "The node of intersection is " <<
list.getNode() << endl;

    // Clean up memory
    delete list.head1->next->next->next->next;
    delete list.head1->next->next->next;
    delete list.head1->next->next;
    delete list.head1->next;
    delete list.head2->next->next;
    delete list.head2->next;
    delete list.head2;

    return 0;
}
```

```
The node of intersection is 15
```