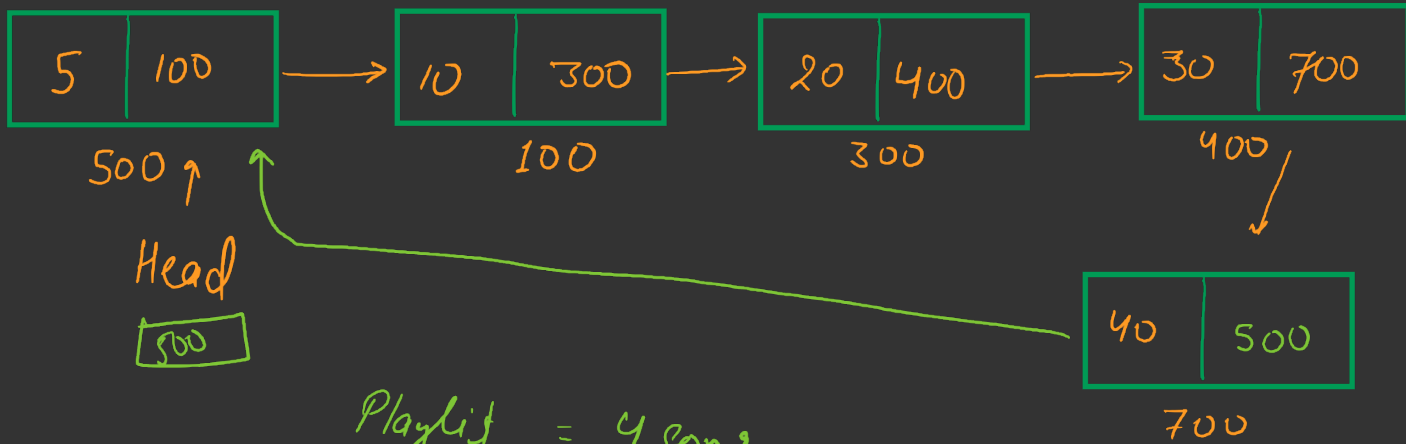
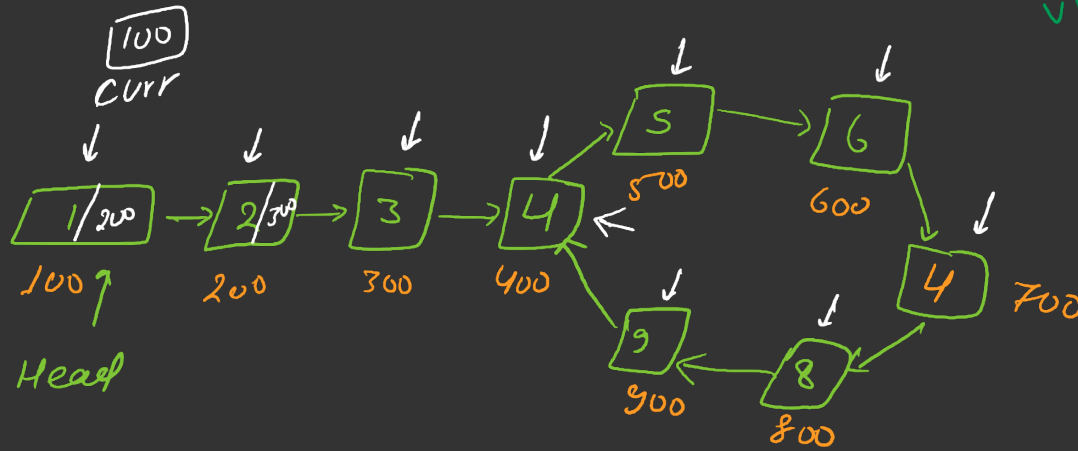


Circular Linked List



Playlist = 4 song

11 Δ 1 2 3 4
→



100	200	300	400	500	600	700	800	900
1	1	1	1	1	1	1	1	1

if (m[700] == 1)

0

Node *curr = head

unordered_map <Node*, bool> visit;

while (curr != NULL) {

if (visit[curr] == 1)

return 1;

visit[curr] = 1;

curr = curr->next;

}

T.C $\frac{O(n)}{O(n)}$
S.C $O(n)$

S.C $O(1)$

bool check(vector<Node*> visit,
Node* curr) {

for (i = 0; i < visit.size(); i++) {

if (visit[i] == curr)

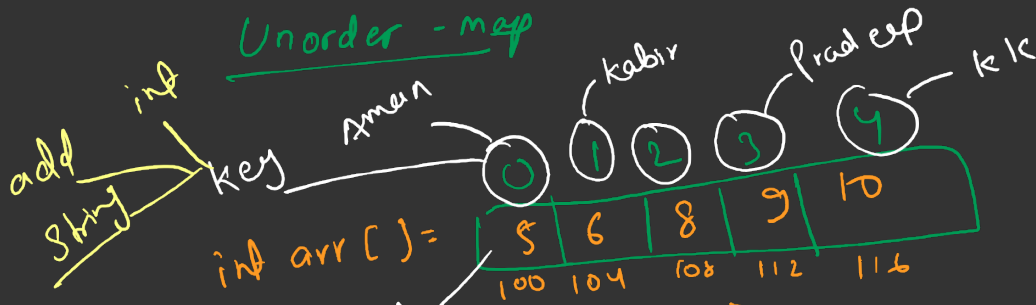
return 1;

}

return 0;

}

Slow, fast



Rohit	Am	kk
100	200	250

250

value

cout << arr[2]

O(n)

m["Aman"]

3

Key

100

200

250

Rohit

Aman

→ kk

arr + 2

100 + 2 × 7

100

Rohit	Aman	kk
100	200	250

O(1)

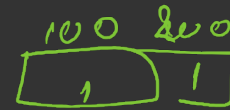
m["kk"] = 250

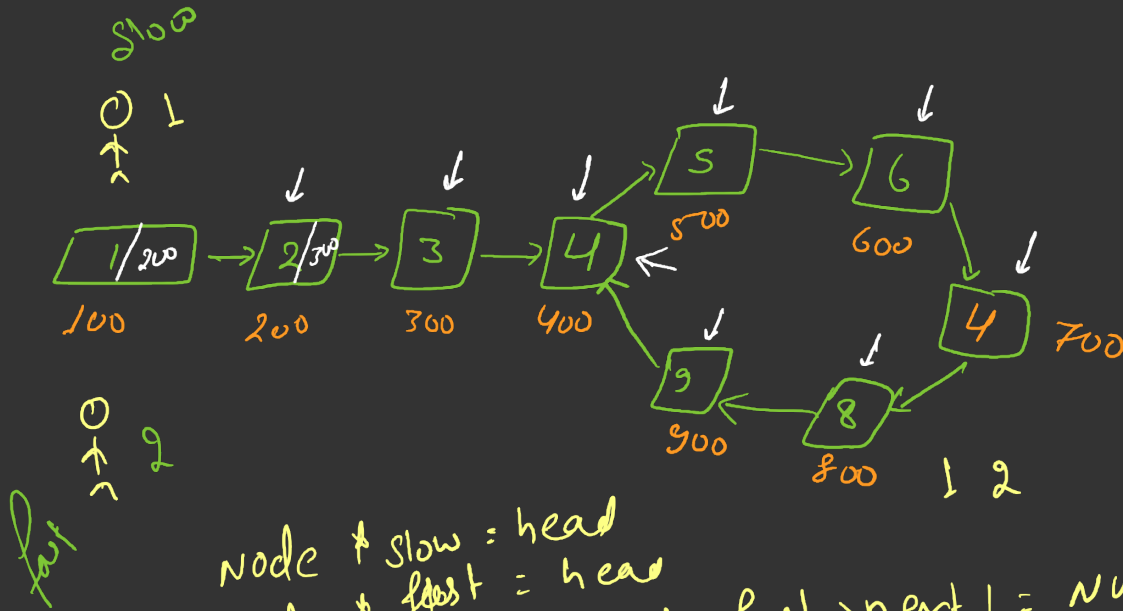
Unordered_map <key, value> map_name

unordered_map <Node*, bool> visit;

visit [100] = 1

visit [200] = 1



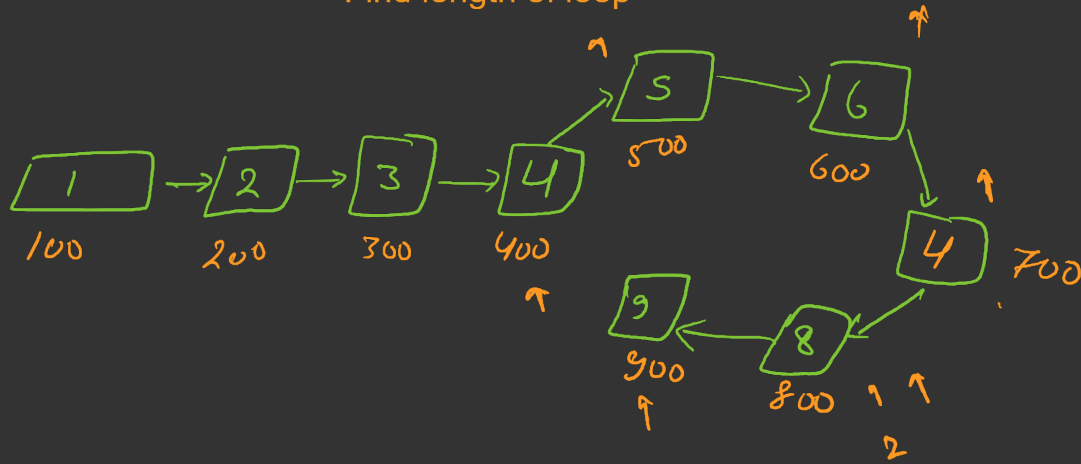


```

node * slow = head
node * fast = head
while (fast != NULL && fast->next != NULL) {
    slow = slow->next;
    fast = fast->next->next;
    if (slow == fast) {
        return 1;
    }
}
return 0;

```

Find length of loop



slow count = 123

```

node *slow = head
node *fast = head
→ while (fast != NULL && fast->next != NULL) {
    slow = slow->next;
    fast = fast->next->next;
    → if (slow == fast) {
        break;
    }
}

```

```

if (fast == NULL || fast->next == NULL)
    return 0;

```

```

int count = 1
slow = fast->next;
while (slow != fast)
{
    count++;
    slow = slow->next;
}
return count;

```