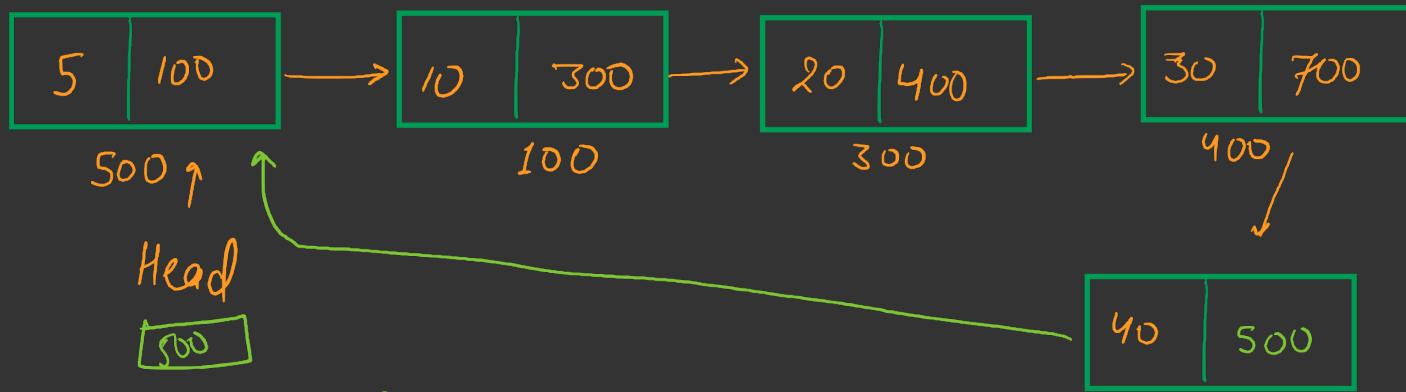
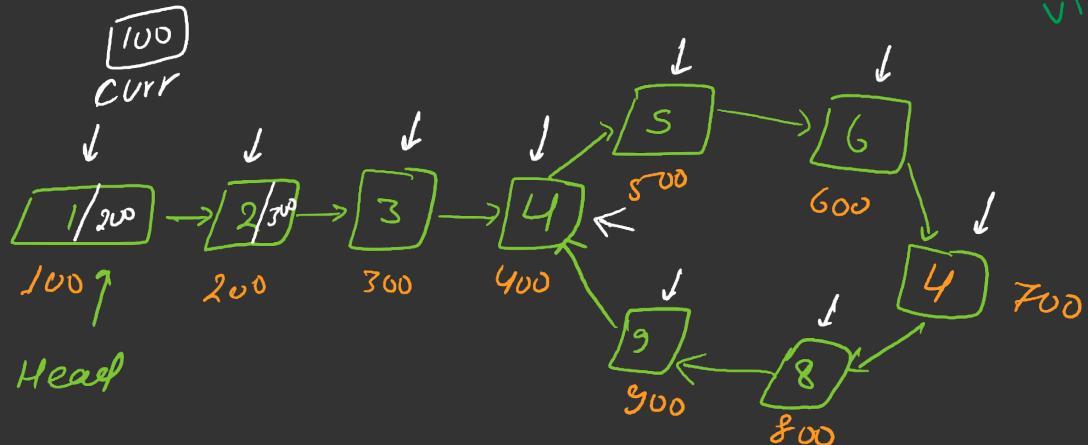


Circular Linked List



Playlist = 4 song
[11 △] 1 2 3 4



visit - $100 \cdot 200$
 $300 \cdot 400$



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;
}

```

$$\begin{array}{ll}
T.C & \underline{\mathcal{O}(n)} \\
S.C & \underline{\mathcal{O}(n)}
\end{array}$$

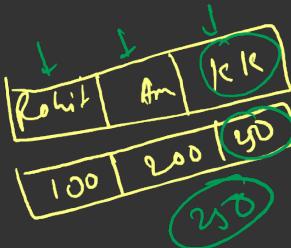
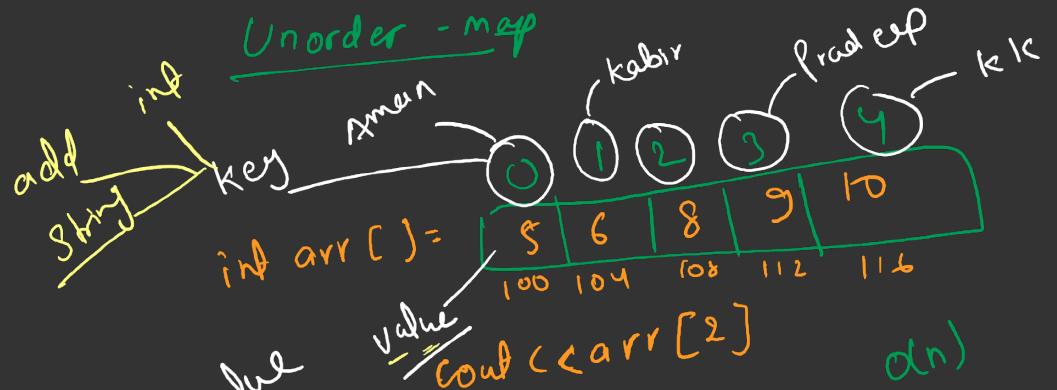
$$S \subset \underline{\mathcal{O}(1)}$$

```

bool check(vector<Node*> visit,
           Node* curr) {
    for (i=0; i<visit.size(); i++) {
        if (visit[i] == curr)
            return 1;
    }
    return 0;
}

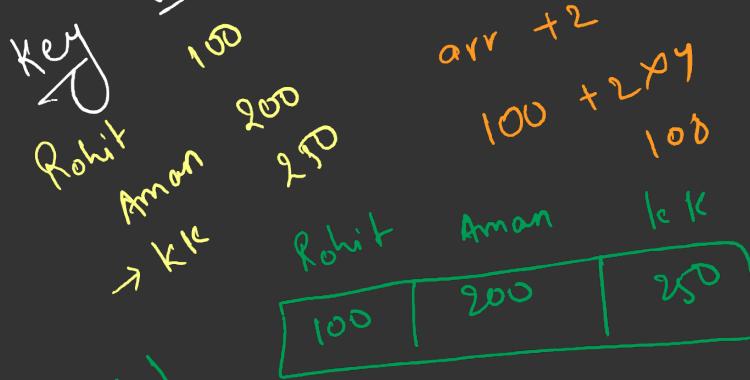
```

Slow, fast



m["Aman"]

(5)



$$m["kkk"] = 250$$

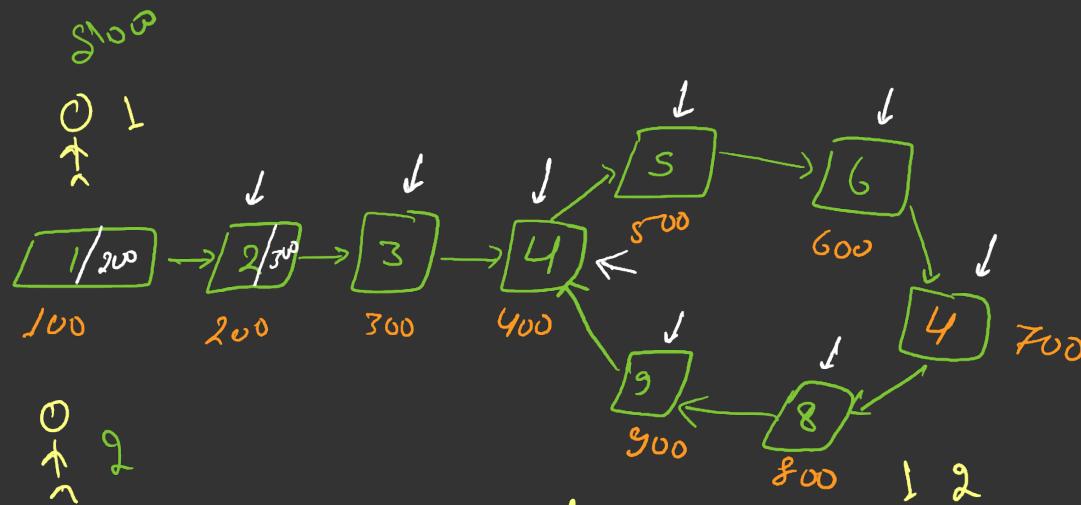
Unordered-map<key, value> map-name

Unordered-map<Node*, bool> visit;

visit[100] = 1

visit[200] = 1

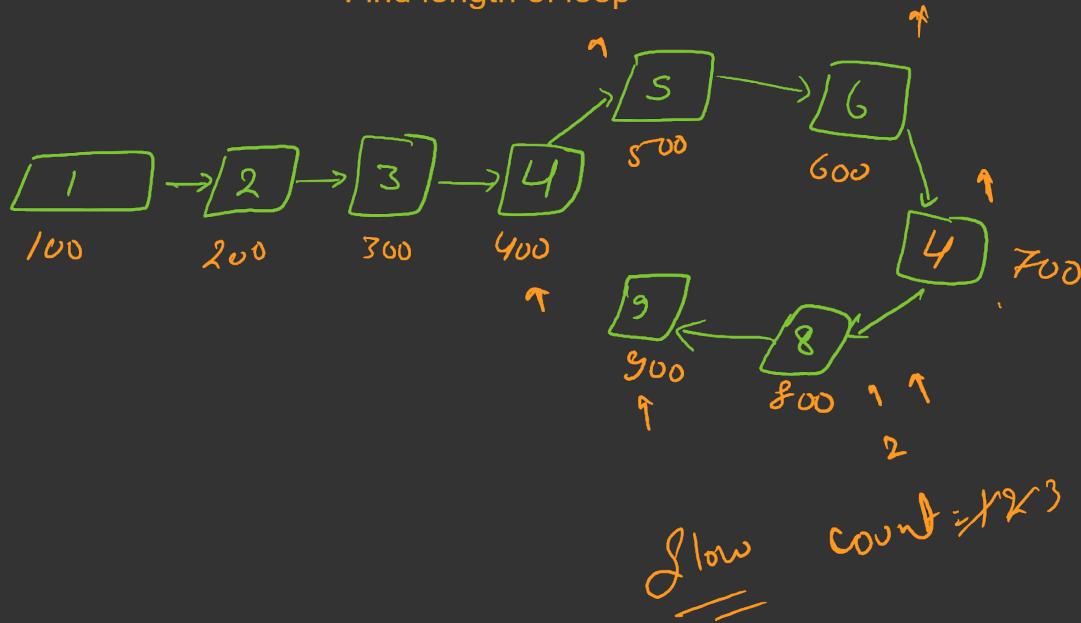




```

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



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;

