LENGTH OF LOOP IN A LINKED LIST

✡ In this problem, we are given a LinkedList and our job is to return the length of the loop that occurs in the linkedlist (0 if there is no loop).

Bruteforce solution is to obviously use a map datastructure where Node is a key and its index is value. At each iteration we check if node was already there in the map or not. If it was, we just subtract its index from current as that is our length. Time Complexity is O(N) and Space Complexity is O(N)

Optimal Solution is using Tortoise & Hare algorithm. Once we detect the loop using the algorithm, we just make one of slow or fast pointer do one more pass to calculate the length.

```
int detectLoop(Node * head) {
  Node * s = head;
  Node * f = head;
  while(f != nullptr && f->next != nullptr){
        s = s->next;
```

```
        f = f -> next -> next;
        if (s == f) {
            int c = 1;
            f = f -> next;
            while (s1 = f) {
                f = f -> next;
                c++;
            }
            return c;
        }
    }
    return 0;
}
```