DSA SERIES

- Learn Coding

Topic to be Covered today

Queue Problems

LETS START TODAY'S LECTURE

Dota2 Senate

```
class Solution {
public:
    string predictPartyVictory(string senate) {

        int n = senate.length();
        queue<int> radiant, dire;

        for(int i =0;i<n;i++){
            if(senate[i]=='R'){
                radiant.push(i);
            } else{
                dire.push(i);
            }
        }

        while(!radiant.empty() && !dire.empty()){
            int r_idx = radiant.front();
            radiant.pop();
```

```
int d_idx = dire.front();
            dire.pop();

            if(r_idx < d_idx){
                radiant.push(r_idx+n);
            }else{
                dire.push(d_idx+n);
            }
        }

        return radiant.empty() ?
"Dire":"Radiant";

    }
};
```

Find the Winner of the Circular Game (1823)

```
class Solution {
public:
    int findTheWinner(int n, int k) {
       queue<int> q;

        for(int i =1;i<=n;i++){
            q.push(i);
        }

        int i =0;

        while(q.size()>1){
          for(int i =1;i<k;i++){
            q.push(q.front());
            q.pop();
          }
          q.pop();
        }

        return q.front();
    }
};
```

Reveal Cards In Increasing Order

```
class Solution {
public:
    vector<int> deckRevealedIncreasing(vector<int>& deck) {
       int n = deck.size();

       queue<int> que;
       vector<int> result(n);

       for(int i =0;i<n;i++){
        que.push(i);
       }

       sort(begin(deck),end(deck));

       for(int i =0;i<n;i++){
        int idx = que.front();
        que.pop();
        result[idx]= deck[i];
```

```
  if(!que.empty()){
            que.push(que.front());
            que.pop();
         }
        }
        return result;
    }
};
```

Learn coding

THANK YOU