**Day-7**

**Problem1**- [**215. Kth Largest Element in an Array**](https://leetcode.com/problems/kth-largest-element-in-an-array/)

Solution-

class Solution {

public:

    int findKthLargest(vector<int>& nums, int k) {

        priority\_queue<int>pq;

        for(auto i:nums)

        {

            pq.push(i);

        }

        int x;

        while(!pq.empty()&& k--)

        {

            x=pq.top();

           pq.pop();

        }

        return x;

    }

};

**Problem2**- [**264. Ugly Number II**](https://leetcode.com/problems/ugly-number-ii/)

Solution-

class Solution {

public:

    int nthUglyNumber(int n) {

        priority\_queue<long long,vector<long long>,greater<long long>>pq;

        pq.push(1);

        long long cnt=0;

        unordered\_map<long long,int>m;

        long long x;

        while(cnt<n)

        {

            x=pq.top();

            pq.pop();

            if(m[x]==0)

            {

                pq.push(x\*2);

                pq.push(x\*3);

                pq.push(x\*5);

                cnt++;

                m[x]=1;

            }

        }

        return x;

    }

};

**Problem 3**- [**1642. Furthest Building You Can Reach**](https://leetcode.com/problems/furthest-building-you-can-reach/)

class Solution {

public:

    int furthestBuilding(vector<int>& h, int bricks, int ladders) {

        int n=h.size();

        priority\_queue<int,vector<int>,greater<int>>pq;

        for(int i=1;i<n;i++)

        {

            int diff=h[i]-h[i-1];

            if(diff>0)

            {

                if(pq.size()==ladders)

                {

                    pq.push(diff);

                    int x=pq.top();

                    pq.pop();

                    if(bricks>=x)

                    {

                        bricks-=x;

                    }

                    else

                    return i-1;

                }

                else

               pq.push(diff);

            }

        }

        return n-1;

    }

};

**Problem 4**- [**234. Palindrome Linked List**](https://leetcode.com/problems/palindrome-linked-list/)

class Solution {

public:

bool isPalindrome(ListNode\* head) {

vector<int>v;

while(head!=NULL)

{

v.push\_back(head->val);

head=head->next;

}

int n=v.size();

cout<<n;

int flag=0;

for(int i=0;i<n/2;i++)

{

if(v[i]!=v[n-i-1])

return false;

}

return true;

}

};

**Problem 5**- [**70. Climbing Stairs**](https://leetcode.com/problems/climbing-stairs/)

class Solution {

public:

int solve(int n, vector<int>&dp)

{

if(n<=2)

return n;

if(dp[n]!=-1)return dp[n];

return dp[n]= solve(n-1,dp)+solve(n-2,dp);

}

int climbStairs(int n) {

// return climbStairs(n-1)+climbStairs(n-2);

vector<int>dp(n+1,-1);

return solve(n,dp);

}

};