EASY :

1. Number of 1 bits

```
class Solution {
  public:
    int setBits(int N)
    {
        int c=0;
        while(N)
        {
            c+=(N%2);
            N/=2;
        }
        return c;
    }
};
```

2. Non repeating elements

```
class Solution{
public:
    vector<int> singleNumber(vector<int> nums)
    {
        // Code here.
        int ans=0;
        int ans1=0;
        unordered_map<int,int>mp;
        for(int i=0;i<nums.size();i++)
        mp[nums[i]]++;

        for(auto it:mp)
        {
            if(ans==0 && it.second==1)
            ans=it.first;
            else if(it.second==1)
            ans1=it.first;
        }

        return {min(ans,ans1),max(ans,ans1)};

    }
};
```

3. Bit difference

```
class Solution{
    public:
    int countBitsFlip(int a, int b)
    {
        int c=0;
        while(a || b)
        {
            c+=abs(a%2 - b%2);
            a/=2;
            b/=2;
        }
        return c;
    }
};
```

4. Is num Power of 2

```
class Solution{
    public:
    bool isPowerofTwo(long long n)
    {
        int c=0;
        while(n)
        {
            c+=(n%2);
            n/=2;
        }
        return c==1;
    }
};
```

5. Power of four

```
class Solution {
public:
     bool solve(int n)
    {

        if(n==0 || n==1)
        return true;

        if(n%4!=0)
        return false;
        return solve(n/4);
    }

    bool isPowerOfFour(int n)
    {
        if(n==0)
        return false;

       return solve(n);
    }
};
```

MEDIUM :

1. Divide two num without *, / and %.

```
class Solution {
public:
    int divide(int dividend, int divisor)
    {
        bool sign=(divisor<0)^(dividend<0);

        long divi=abs(dividend);
        long div=abs(divisor);
        long quot=0;

        while(divi>=div)
        {
            long long temp=div;
            long long c=1;
            while(divi>=temp)
            {
                divi-=temp;
                quot+=c;
                c<<=1;
                temp<<=1;

            }
        }

        if(sign==1)
        quot=-quot;

        return min(max(quot,(long)INT_MIN),(long)INT_MAX);
    }
};
```

2. Power set

```
class Solution{
       public:
       vector<string>ans;
   void solve(int i,int n,string s,string ans1)
   {
      if(i==n)
      {
        if(ans1.length()>0)
        ans.push_back(ans1) ;
        return;
      }
      else if(i<n)
      {
       solve(i+1,n,s,ans1);
       ans1.push_back(s[i]);
       solve(i+1,n,s,ans1);
       return;
      }

   }
                vector<string> AllPossibleStrings(string s)
                {
                    int i=0;
                   int n=s.length();

                   solve(0,n,s,"");
                   sort(ans.begin(),ans.end());
                   return ans;
                }

};
```

3. Kth symbol in grammar

```
class Solution {
public:
   int solve(int n,int k)
   {
      if(n==1 && k==1)
      return 0;

      int mid = pow(2,n-1)/2;

      if(k<=mid)
```

```
        return solve(n-1,k);
        else
        return !solve(n-1,k-mid);

    }
    int kthGrammar(int n, int k)
    {
        if(n==1)
        return 0;
        return solve(n,k);
    }
};
```

4. Sum of two number

```
class Solution {
public:
    int getSum(int a, int b) {

        int sum=a;
        long long mask=(long long)INT_MAX-INT_MIN;        //0xFFFFFFFF
        // mask == 4294967295
        while(b!=0)
        {
            sum=(a^b);
            b=((a&b)&mask)<<1;
            a=sum;
        }

        return sum;
    }
};
```

5. Number of Good Ways to Split a String

```
class Solution {
public:
    int numSplits(string s)
    {
        unordered_map<char, int> left;
        unordered_map<char, int> right;

        for(int i=0; i<s.length(); i++)
        right[s[i]]++;

        int rightS = right.size();
        int ans=0;
        for(int i=0; i<s.length(); i++)
        {
            left[s[i]]++;
            right[s[i]]--;

            if(right[s[i]] == 0)
            rightS--;

            if(left.size() == rightS)
            ans++;
        }
        return ans;
    }
};
```

HARD :