

Fibonacci Numbers

$$\textcircled{1} \sum_{i=0}^n F_i = F_{n+2} - 1$$

$$\textcircled{2} \sum_{i=0}^n F_i^2 = F_n \cdot F_{n+1}$$

$$\textcircled{3} \gcd(F_n, F_m) = F_{\gcd(n, m)}$$

Inversions using merge sort

Modified Merge Sort to find the Inversions !!

stable = sort

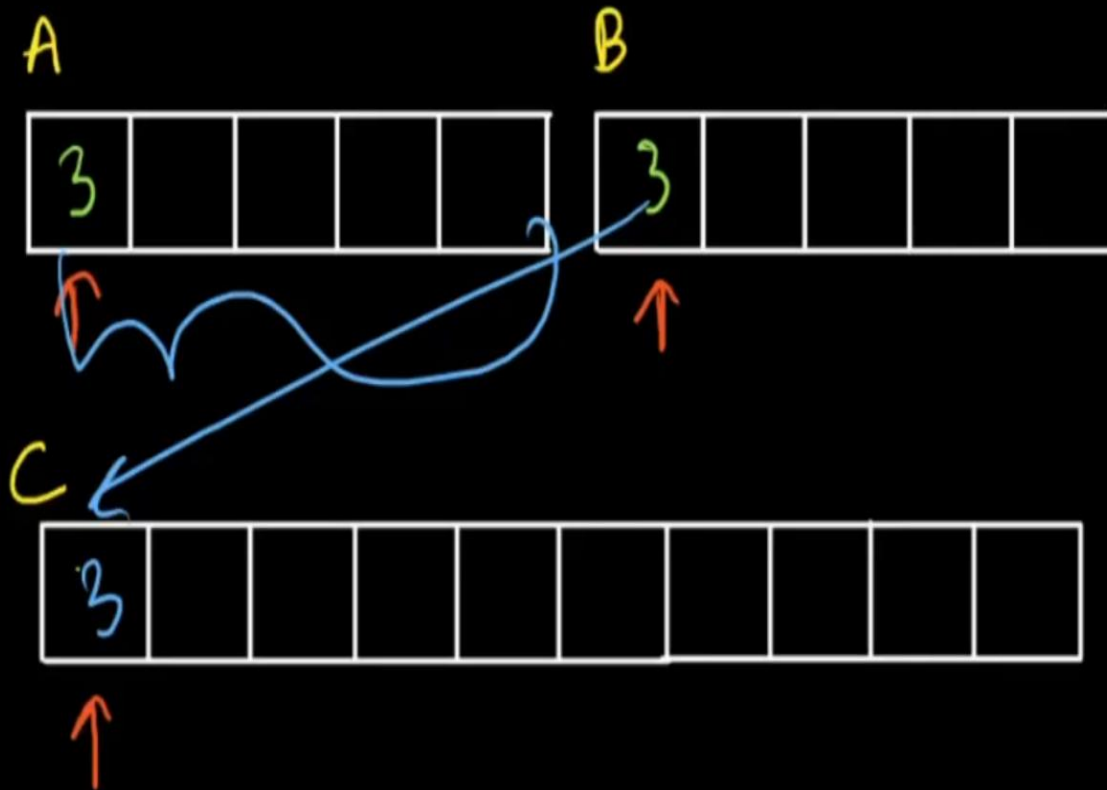
a_i a_j

```
int i = 0, j = 0, k = 0;  
int ans = 0;
```

```
while(i < A.size() && j < B.size()){  
    if( A[i] <= B[j] ){  
        C[k++] = A[i++];  
    }else{  
        ans += A.size() - i;  
        C[k++] = B[j++];  
    }  
}
```

```
while(i < A.size()){  
    C[k++] = A[i++];  
}
```

```
while(j < B.size()){  
    C[k++] = B[j++];  
}
```



Code for recursive Merge Sort

<https://drive.google.com/file/d/1G2eJiTZrms5XBplO2Kcp4LM2Usc9opHO/view>

Inversion:
 $a[i] > a[j]$ for $i < j$

$$① (a+b)^n = {}^nC_0 a^0 b^n + {}^nC_1 a^1 b^{n-1} + \dots + {}^nC_n a^n b^0$$

② Calculating nC_r

- Linearly \rightarrow $Fact[0] = 1$
 $(n \leq 10^6)$ $Fact[i] = (Fact[i-1] * i) \% \text{prime}$
- Multiplication $\rightarrow \prod_{i=0}^r \frac{(n-i)}{(i+1)}$
 $(n \leq 10^{18})$
 $(n-r, r \leq 10^6)$

$$③ \sum_{i=0}^n {}^nC_i = 2^n \quad \sum_{i=0}^n i \cdot {}^nC_i = n \cdot 2^{n-1}$$

$$④ \sum_{m=k}^n m C_k = {}^{n+1}C_{k+1} \quad \text{eg. } {}^1C_1 + {}^2C_1 + {}^3C_1 = {}^4C_2 = 6$$

$$⑤ \sum_{i=0}^k {}^nC_i \cdot m C_{k-i} = {}^{n+m}C_k$$

$$⑥ \sum_{i=0}^n ({}^nC_i)^2 = 2^n C_n$$

$$⑦ x_1 + x_2 + x_3 + \dots + x_n = r \quad \text{where } x_i \geq 0$$

no. of ways = ${}^{n+r-1}C_{n-1}$

$$⑧ \sum_{k=0}^{\lfloor \frac{n}{2} \rfloor} \binom{n-k}{k} = \text{Fib}(n+1)$$

Catalan Numbers $\rightarrow 1, 1, 2, 5, 14, 42$

$$① C_n = \frac{1}{n+1} \binom{2n}{n} = \binom{2n}{n} - {}^{2n}C_{n-1}$$

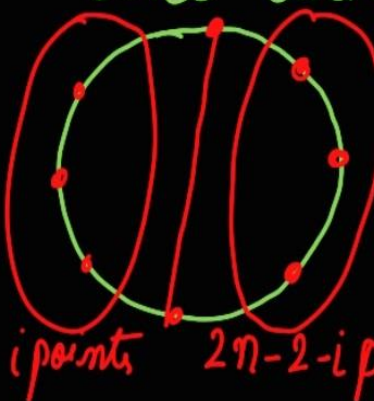
$$② \text{Recurrence } C_{n+1} = \sum_{i=0}^n C_i C_{n-i}, \quad C_0 = 1$$

eg. no. of triangulation ways for convex polygon of $(n+2)$ sides $\rightarrow C_n$



$$f(n+1) = \sum_i f(i) f(n-i)$$

eg. no. of ways to connect $2n$ points on circle such that no 2 intersect



$$f(2n) = \sum f(i) * f(2n-2-i)$$

i points $2n-2-i$ points

SEGMENT TREE

```
1 void build(int index, int l, int r)
2 {
3     if(l==r)
4     {
5         t[index]=a[l];
6         return;
7     }
8     int mid=(l+r)/2;
9     build(index*2,l,mid);
10    build(index*2+1,mid+1,r);
11    t[index]=t[2*index]+t[2*index+1];
12 }
```

```
1 void update(int index, int l, int r, int pos, int val)
2 {
3     if(pos<l || pos>r) return;
4     if(l==r)
5     {
6         t[index]=val;
7         a[l]=val;
8         return;
9     }
10    int mid=(l+r)/2;
11    update(2*index,l,mid,pos,val);
12    update(2*index+1,mid+1,r,pos,val);
13    t[index]=t[2*index]+t[2*index+1];
14 }
```

```
1 int query(int index, int l, int r, int lq, int rq)
2 {
3     if(l>rq || lq>r) return 0;
4     if(lq<=l && r<=rq) return t[index];
5     int mid=(l+r)/2;
6     return query(2*index,l,mid,lq,rq)+ query(2*index+1,mid+1,r,lq,rq);
7 }
```

N QUEENS

```
void rec(int row){  
    if(row==n){  
        ans++;  
        return;  
    }  
    for(int col=0; col<n; col++){  
        bool safe = 1;  
        for(int pRow=0; pRow<row; pRow++){  
            int pCol = placed[pRow];  
            if(pCol==col || (abs(row-pRow)==abs(col-pCol))){  
                safe = 0;  
            }  
        }  
        if(safe){  
            placed.push_back(col);  
            rec(row+1);  
            placed.pop_back();  
        }  
    }  
}
```

} → check / inc .

→ enumerate on moves
is - possible (move)

Do the move
move to next if ch
remove the move

We start from row 0. We try placing in each column. If we can place in that column, we store that column, call for the next row and remove that column. The number of times we reach row n will be the answer.

KMP Algorithm

KMP's algorithm

$lps[i] \rightarrow s[0..i] \rightarrow$ largest suffix which is also a prefix

$s \rightarrow \overbrace{a} \overbrace{b} \overbrace{a} \overbrace{c} \overbrace{a} \overbrace{b} \overbrace{a} \overbrace{d}$
 $lps \rightarrow \boxed{0 \ 0 \ 1 \ 0 \ 1 \ 2 \ 3 \ 0}$

$s \rightarrow \boxed{\text{KMP}} \rightarrow lps$

Q Find pattern P in T. Give no. of occurrences

$S = P + \# + T \rightarrow \boxed{\text{KMP}} \rightarrow LPS$

eg. aba in abaabcaba

$\Rightarrow aba\#abaabcaba$
1 2 3 1 2 0 1 2 3
 ↙ ↖
 occurrence of P

```
void solve(){
    string s;
    cin >> s;
    int n = s.length();

    int lps[n+1];
    int i=0, j=-1; lps[0]=-1;
    while(i<n){
        while(j!=-1 && s[j]!=s[i]) j=lps[j];
        i++; j++; lps[i]=j;
    }
}
```

To find the period in a string $\rightarrow N - lps[N]$