Noora Number

Limits: 2s, 512 MB

A number is a “Noora” number if the count of distinct digits of the number is equal to the maximum digit of the number. So, 123 is a Noora number but 124 is not. Some of the first Noora number are 1,11,12,20,21,103,111,112, …. .

Given n, you have to tell how many Noora number ≤ n

Input

Input starts with an integer T(1 ≤ T ≤ 100000), denoting the number of test cases. Each case contains an integer n (1 ≤ n ≤ 10^18).

Output

For each test case, output the number of Noora number ≤ n

Samples

Input	Output
5	1
10	5
100	29
1000	213
10000	1893
100000	

```cpp
#include<bits/stdc++.h>
using namespace std;
#define LL long long
const int MAXN = 19;
const LL INF = 1000000000000000005LL;
LL dp[20][1030][2];
string s;
LL fun(int pos,int mask,int suru){
    if(pos==MAXN){
        if(suru==0) return 0LL;
        int bit=0,MAXX=0;
        for(int i=0; i<10; i++){
            if(mask&(1<<i)){
                MAXX = max(MAXX,i);
                bit++;
            }
        }
        if(bit==MAXX) return 1LL;
        return 0LL;
    }
    if(dp[pos][mask][suru]!=-1)return dp[pos][mask][suru];
    LL ret = 0LL;
    for(int i=0; i<10; i++){
        if(i==0 && suru==0) ret += fun(pos+1,mask,suru|(i>0));
        else ret += fun(pos+1,mask|(1<<i),suru|(i>0));
    }
    return dp[pos][mask][suru] = ret;
}
void path(int pos,int mask,int suru,LL nth){
    if(pos==MAXN)return;
    LL ret = 0,res=0;
    for(int i=0; i<10; i++){
        if(i==0 && suru==0) ret += fun(pos+1,mask,suru|(i>0));
        else ret += fun(pos+1,mask|(1<<i),suru|(i>0));

        if(ret>=nth){
            s += i+'0';
            if(i==0 && suru==0) path(pos+1,mask,suru|(i>0),nth-res);
            else path(pos+1,mask|(1<<i),suru|(i>0),nth-res);
            break;
        }
        res = ret;
    }
}
int main(){
    memset(dp,-1,sizeof(dp));
    fun(0,0,0);

    int tt; scanf("%d",&tt);
    LL ans = 0;

    for(int ks=1; ks<=tt; ks++){
        LL n; scanf("%lld",&n);
        LL lo = 1, hi = n;
        while(lo<=hi){
            LL md = (lo+hi)/2;

            s = "";
            path(0,0,0,md);
            LL v = 0;
            for(int i=0; i<s.size(); i++){
                LL d = s[i]-'0';
                v = (v*10)+d;
            }

            if(v<=n){
                ans = md;
                lo = md+1;
            }else{
                hi = md-1;
            }
        }
        printf("%lld\n",ans);
    }
    return 0;
}
```