// ========== LIS ==========

// A[1] ~ A[N]

for (int i=2;i<=N+1;i++) g[i]=INF;

for (int i=1;i<=N;i++) {

int k=lower\_bound(g+2,g+N+2,A[i]) - g;

// non-decreasing: 用upper\_bound

dp[i] = k-1; // 以A[i]结尾的LIS长度

Ans = max(Ans, dp[i]);

g[k] = A[i];

}

// ========== 状态压缩 ==========

O(3^n)

for (int S = 1; S < (1<<n); S++)

for (int S0 = (S-1)&S; S0; S0 = (S0-1)&S)

int S1 = S-S0; //S 被拆成两个集合 S0 & S1

// ========== 四边形优化 ==========

dp(i,j)=min{dp(i,k-1),dp(k,j)}+w(i,j)(i≤k≤j)

(min也可以改为max) 如果满足：

1.区间包含的单调性: if (i≤i'<j≤j') then w(i',j)≤w(i,j')

2. 四边形不等式: if (i≤i'<j≤j') then w(i,j)+w(i',j')≤w(i',j)+w(i,j')

// 再定义 s(i,j) 表示 m(i,j) 取得最优值时对应的下标(即 i≤k≤j 时,k 处的 w 值最大,则 s(i,j)=k) 我们有s(i,j)单调 ie.

s(i,j-1)≤s(i,j)≤s(i+1,j)

故而dp(i,j)=min{dp(i,k-1)+dp(k,j)}+w(i,j) 且s(i,j-1)≤k≤s(i+1,j) (min可以改为max)