```cpp
/// Longest Increasing Subsequence(LIS) O(nlogn):
#include<bits/stdc++.h>
using namespace std;
#define ll long long
#define inf 10000000000000000
stack<ll>st;
ll n,L[100005],a[100005],s[100005];
int binarySearch(int v){
    int lo=0, hi=n;
    int ans;
    while(lo<=hi){
        int md = (lo+hi)/2;
        if(s[md]<v){ lo=md+1; ans=md; }
        else { hi = md-1; }
    }
    return ans+1;
}
void path(int lis){
    for(int i=n; i>=1 && lis>0; i--){
        if(L[i]==lis){
            st.push(a[i]);
            lis--;
        }
    }
}
int main(){
    while(scanf("%lld",&n)==1){
        for(int i=1; i<=n; i++){
            scanf("%lld",&a[i]);
        }

        s[0] = -inf;
        for(int i=1; i<=n; i++) s[i]=inf;

        int lis = 0;
        for(int i=1; i<=n; i++){
            ll v = a[i];
            int p = binarySearch(v);
            s[p] = v;   L[i] = p;
            lis = max(lis,p);
        }

        path(lis);

        printf("%d:",lis);
        while(!st.empty()){
            printf(" %lld",st.top());
            st.pop();
        }
        printf("\n");
    }

    return 0;
}
```

```cpp
/// Longest Increasing Subsequence(LIS) O(n^2):
#include<bits/stdc++.h>
using namespace std;
vector<int>ed[1005],vv;
int n,dp[1005],a[1005],w[1005];
int lis(int u){
    if(ed[u].size()==0)return dp[u]=1;
    if(dp[u]!=-1)return dp[u];

    int ret = 0;
    for(int i=0; i<ed[u].size(); i++){
        int v = ed[u][i];
        ret = max(ret,1+lis(v));
    }
    return dp[u] = ret;
}
void path(int u,int x){
    if(x==0)return;

    for(int i=0; i<ed[u].size(); i++){
        int v = ed[u][i]; int ret = 1+lis(v);
        if(ret==x){
            vv.push_back(a[v]);
            path(v,x-1);
            break;
        }
    }
}
int main(){
    while(scanf("%d",&n)==1){
        for(int i=1; i<=n; i++){
            scanf("%d",&a[i]);
        }

        for(int i=1; i<=n; i++){
            for(int j=i+1; j<=n; j++){
                if(a[j]>a[i]) {
                    ed[i].push_back(j);
                }
            }
        }

        for(int i=1; i<=n; i++) ed[0].push_back(i);

        memset(dp,-1,sizeof(dp));
        int ans = lis(0);

        path(0,ans);

        printf("%d:",ans-1);
        for(int i=0; i<vv.size(); i++) printf(" %d",vv[i]);
        printf("\n");

        vv.clear(); for(int i=0; i<=n; i++)ed[i].clear();
    }
    return 0;
}
```