动态规划 常用

```
#include <iostream>
#include <vector>
#include <string>
#include <cstring>
#include <set>
#include <map>
#include <queue>
#include <ctime>
#include <random>
#include <sstream>
#include <numeric>
#include <stack>
#include <stdio.h>
#include <algorithm>
using namespace std;

#define Multiple_groups_of_examples
#define rep(i,x,n) for(int i = x; i <= n; i++)
#define vf first
#define vs second

typedef long long LL;
typedef pair<int,int> PII;

const int INF = 0x3f3f3f3f;
const int N = 1e2 + 21;
namespace golitter {
namespace linear {
const int N = 1e5 + 21;
int a[N];
// 最长上升子序列
int LIS_hd()
{
    int n; cin>>n;
    for(int i = 1; i <= n; ++i) cin>>a[i];
    vector<int> f;
    for(int i = 1; i <= n; ++i)  {
        auto pos = lower_bound(f.begin(), f.end(), a[i]);
        if(pos == f.end()) {
            f.push_back(a[i]);
        } else *pos = a[i];
    }
    cout<<f.size();

    return 0;
}
// 最长公共子序列
void LCS() { // O(n ** 2)
    int n; cin>>n;
    vector<int> A(n+1);
    auto B(A);
    vector<vector<int>> f(n+1, vector<int>(n+1));
    rep(i,1,n) cin>>A[i];
    rep(i,1,n) cin>>B[i];
    rep(i,1,n) {
        rep(j,1,n) {
            f[i][j] = max(f[i-1][j], f[i][j-1]);
            if(A[i] == B[j]) f[i][j] = max(f[i][j], f[i-1][j-1] + 1);
        }
    }
    cout<<f[n][n];
}
void LCS_hd() { // 将其转为LIS做
    // https://www.luogu.com.cn/problem/solution/P1439
    int n; cin>>n;
    vector<int> b(n);
    for(int i = 0; i < n; ++i) {
        int t; cin>>t;
        b[t-1] = i;
    }
    vector<int> f;
    for(int i = 0; i < n; ++i) {
        int a; cin>>a;
        auto pos = lower_bound(all(f), b[a-1]);
        if(pos == f.end()) f.push_back(b[a-1]);
        else *pos = b[a-1];
    }
    cout<<f.size();
}

}}
```