信奥算法

动态规划

dynamic programming

一维DP本质是填写一维数组

魔鬼的步伐 - 最短路

优化问题

魔鬼共有n级楼梯要走，魔鬼有他的步伐，每一步他只可以向上走a级楼梯或者b级楼梯，**走到第n级台阶至少要几步？走不到时输出-1。**

输入正整数n，a和b。n，a，b<=50。a不等于b。

输入样例	输出样例
10 2 5	2

输入样例	输出样例
10 6 7	-1

一步一步
从低到高走

一步一步
从低到高
递推求解

状态用i描述

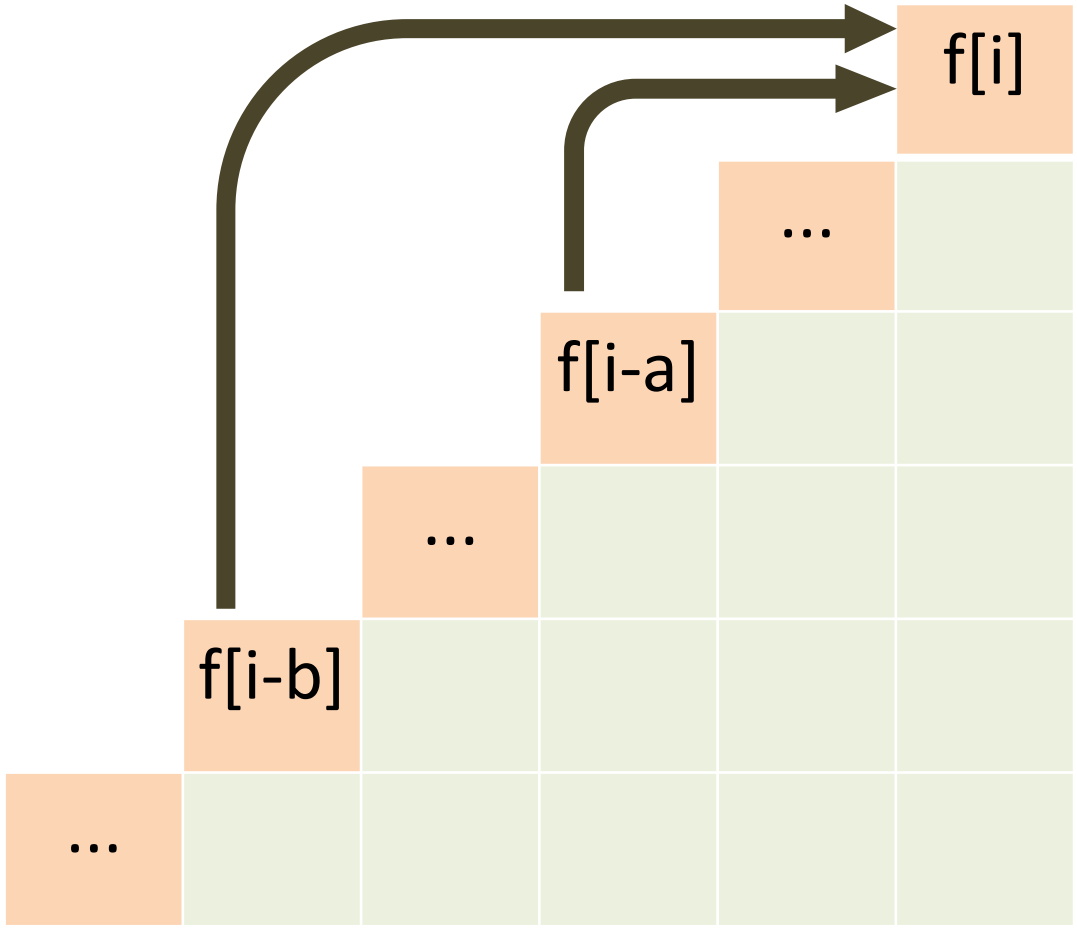
f[i]代表走到第i级至少要几步

若走不到第i级，f[i]=INF

到达第i级的方式有两种
可以从第i-a级走a级来
也可以从第i-b级走b级来

这句话有什么漏洞?

i-a或i-b可能是负数



f[i]代表走到第i级至少要几步

若走不到第i级，f[i]=INF

初始条件

$$f[0]=0$$

状态转移方程

当i大于0时

$$f[i]=min\{\,f[i-a]+1 \mid i\geq a,\ f[i-b]+1 \mid i\geq b\,\}$$

若$i\geq a$ 可走a级来

若$i\geq b$ 可走b级来

易错点	第i格走不到时f[i]设置INF

|竖线后代表条件

状态用i描述

f[i]代表走到第i级至少要几步

若走不到第i级，f[i]=INF

无限大infinity
简写INF
但程序内实际用
较大数代表如1e9
表示10的9次方

n=9 a=2 b=5

i=	0	1	2	3	4	5	6	7	8	9
f[i]=	0	?	?	?	?	?	?	?	?	?

补全程序

```
10  const int N=1009;
11  const int INF=1e9;
12  int n,a,b,f[N];

16      cin>>n>>a>>b;
17      f[0]=0;
18      for(int i=1;i<=n;i++){
19          f[i]=INF;
20          if(i>=a) f[i]=min(f[i],f[i-a]+1);
21          if(i>=b) f[i]=
22      }
23      if(                )cout<<-1<<endl;
24      else cout<<f[n]<<endl;
```

记笔记

最小化问题的答案初始化INF 计算时会逐渐变小
最大化问题的答案初始化-INF 计算时会逐渐变大

原题说若走不到就输出-1 请问f[]不填INF填-1可以吗?	高频错误	无解情况比最小值更小 会破坏答案计算

DP状态定义

原题求:走到第n级至少要几步
若走不到第n级，输出特殊值-1

状态用i描述

f[i]代表走到第i级至少要几步

若走不到第i级，f[i]=INF

以上定义状态的方法叫
"抄原题"大法

DP的三类问题

计数问题

魔鬼共有n级楼梯要走，魔鬼有他的步伐，每一步他只可以向上走a级楼梯或者b级楼梯，请问

共有多少种不同的走法正好走完n级台阶？

可行性问题

魔鬼共有n级楼梯要走，魔鬼有他的步伐，每一步他只可以向上走a级楼梯或者b级楼梯，请问

能否走到第n级台阶？

最优化问题

魔鬼共有n级楼梯要走，魔鬼有他的步伐，每一步他只可以向上走a级楼梯或者b级楼梯，请问

走到第n级台阶至少要几步？走不到时输出-1

最大连续子序列和

也叫最大子段和

最大连续子序列和

输入n，再依次输入n个整数组成的数组,第i个数为x[i],求数组中最大连续子序列和(至少一个数字),n<=100000

输入样例	输出样例
5 1 3 -2 4 -5	6

输入样例	输出样例
3 -3 -2 -1	-1

最大连续子序列和

输入n，再依次输入n个整数组成的数组,第i个数为x[i],
求数组中最大连续子序列和(至少一个数字),n<=100000

算法1	枚举端点+前缀和

枚举连续段左端编号i	从1到n

枚举连续段右端编号j	从i到n

计算i号到j号连续和
尝试更新最优解答案

算法1	枚举端点+前缀和

```cpp
#include<bits/stdc++.h>
using namespace std;
const int N=100009;
const int INF=1e9;
int n,x[N],s[N];
int main(){
    cin>>n;
    for(int i=1;i<=n;++i) cin>>x[i];
    for(int i=1;i<=n;++i) s[i]=s[i-1]+x[i];
    int ans=-INF;
    for(int i=1;i<=n;++i)
        for(int j=i;j<=n;++j){
            int sum=s[j]-s[i-1];
            ans=max(ans,sum);
        }
    cout<<ans<<endl;
    return 0;
}
```

s[i]代表前i个数字和

时间复杂度?	$O(n^2)$

最大连续子序列和

输入n，再依次输入n个整数组成的数组,第i个数为x[i],
求数组中最大连续子序列和(至少一个数字),n<=100000

算法2	动态规划

如何设计数组元素的含义?

A　f[i]代表前i个数字里的最大连续子序列和

B　f[i]代表以i号数字结尾的连续子序列的最大和

算法2 动态规划

x[i]代表输入的第i号数字

f[i]代表以i号数字结尾的连续子序列的最大和

输入
8
-2 11 -4 13 -5 -2 8 -1

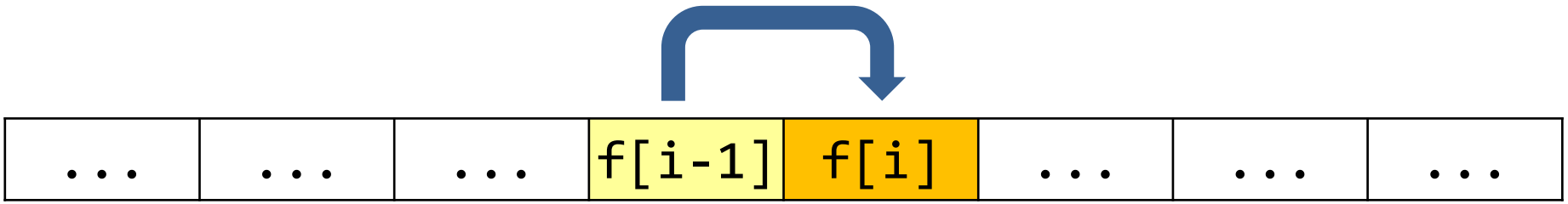
i=	0	1	2	3	4	5	6	7	8
x[i]=	0	-2	11	-4	13	-5	-2	8	-1
f[i]=	0	-2	11	7	20	15	13	21	20

缓冲格

答案是f[]
数组里最大值

算法2	动态规划

x[i]代表输入的第i号数字
f[i]代表以i号数字结尾的连续子序列的最大和



计算f[i]时考虑两种可能：

1.延用f[i-1]结果

2.舍弃f[i-1]结果

f[i]代表以i号数字结尾的连续子序列的最大和

初始条件

$$f[0] = 0$$

状态转移方程

当i大于0时

$$f[i] = max(f[i-1], 0) + x[i]$$

如果f[i-1]<0，舍弃f[i-1]结果

如果f[i-1]>0，延用f[i-1]结果

答案

$$max_i\{f[i]\}$$

答案是f[]数组里最大值

"跑路"算法

有1位老板每天存款变化为x[i]
一旦他发现总存款为负数
马上逃跑宣布破产从零开始

算法2	动态规划

```
14      cin>>n;
15      for(int i=1;i<=n;++i) cin>>x[i];

16      f[0]=0;
17      for(int i=1;i<=n;++i)
18          f[i]=

19      int ans=
20      cout<<ans<<endl;

21      for(int i=0;i<=n;++i)
22          cout<<i<<":"<<f[i]<<endl;
```

打印表格核对每一格是否正确

时间复杂度?	O(n)

DP状态定义

原题求:前n个数里的最大连续子序列和

A

f[i]代表前i个数字里的最大连续子序列和

B

f[i]代表以i号数字结尾的连续子序列的最大和

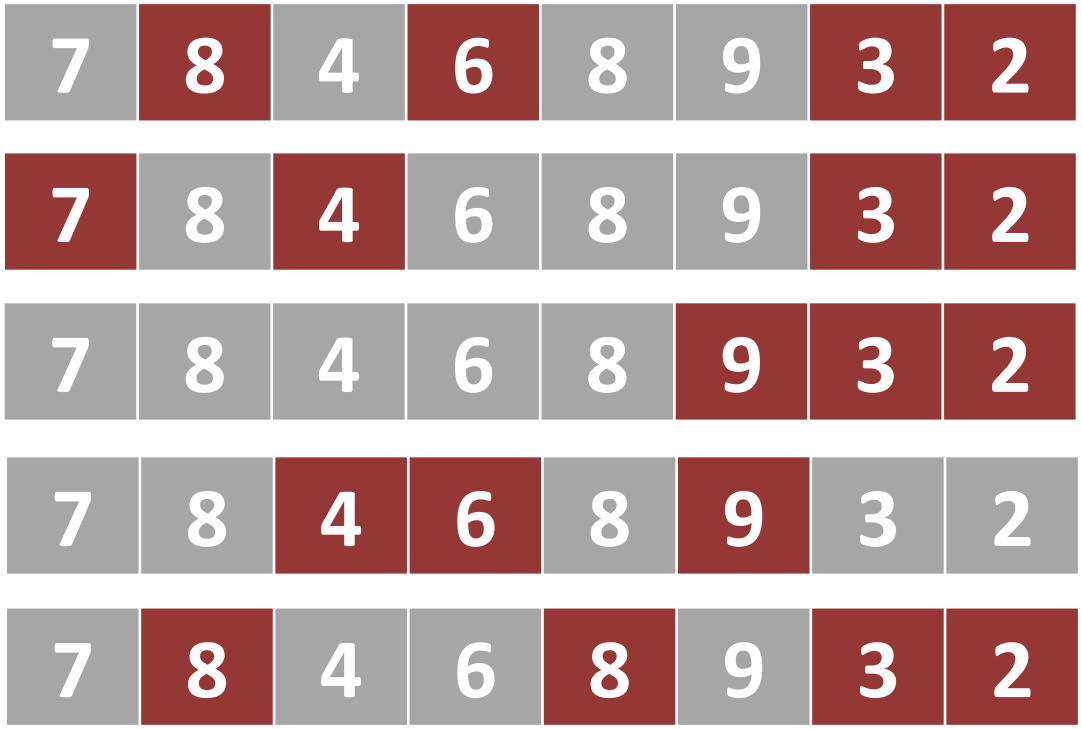
A是"抄原题"大法	f[i]无法利用f[i-1]
B是"状态具体化"	增加了转移所需信息

最长下降子序列

longest decreasing subsequence

最长下降子序列

The **longest decreasing subsequence (LDS)** problem is to find a subsequence of a given sequence in which the subsequence's elements are in sorted order, highest to lowest, and in which the subsequence is as long as possible.



下降子序列 最长

下降子序列 最长

下降子序列

上升子序列

不升子序列

最长下降子序列

给定长度为n的整数序列：x[1],x[2],...,x[n]
输出下降子序列最长的长度

动态规划

如何设计数组元素的含义?

A f[i]代表前i个数字里的最长下降子序列

B f[i]代表以i号数字结尾的最长下降子序列

```
1 /*
2 x[i]代表输入的i 号数字
3 f[i]代表以i 号结尾的最长下降子序列
4       i=0,1,2,3,4,5,6,7,8
5  x[i]=0,3,6,3,5,2,5,1,2
6  f[i]=0,1,1,2,2,3,2,4,3
7 */
```

定义状态

手算样例

请现场完成
第1-7行

2分钟后
老师检查

f[i]代表以i号数字结尾的最长下降子序列

初始条件

$$f[0] = 0$$

状态转移方程

当i大于0时

$$f[i] = max_{0 \le j \le i-1} \{f[j] | x[j] > x[i]\} + 1$$

满足x[j]>x[i]的数才有资格用f[j]打擂台

答案

$$max_i\{f[i]\}$$

答案是f[]数组里最大值

易错点：答案不是f[n]

```
13      cin>>n;
14      for(int i=1;i<=n;i++)cin>>x[i];
15      f[0]=0;
16      for(int i=1;i<=n;i++){
17          f[i]=1;
18          for(int j=1;j<i;j++)
19              if(                    )
20                  f[i]=max(f[i],f[j]+1);
21      }
22      int ans=                                    
23      cout<<ans<<endl;
24      for(int i=0;i<=n;++i)
25          cout<<i<<":"<<f[i]<<endl;
```

打印表格核对每一格是否正确

时间复杂度?	$O(n^2)$

DP状态定义

原题求:前n个数里的最长下降子序列

A

f[i]代表前i个数字里的最长下降子序列

B

f[i]代表以i号数字结尾的最长下降子序列

A是"抄原题"大法	f[i]无法利用前序状态
B是"状态具体化"	增加了转移所需信息

动态规划

dynamic programming

DP本质是填写数组表格

每个数组元素代表什么含义

最终答案信息在哪些格子里

每格依赖其他哪些格

表格填写顺序

表格初始化

DP实战步骤

定义状态

优化状态

手动填写数组所有格子

再总结状态转移方程

再总结边界状态

优化决策

作业要求

写程序前请写明：
1.数组每一格的含义
2.手算样例对应表格

查错方法：
1.打印数组
2.和手算表格对比找不同

```
1 /*
2 f[i]代表
3 输入样例:
4     i=0,1,2,3,...
5 f[i]=
6 */
```

模版格式
供参考

快快编程作业

452

190

1796

拓展题	450,451,639,664,65