

Dynamic Programming 动态规划

【举例】

7

3 8

8 1 0

2 7 4 4

4 5 2 6 5

若从倒数第二排的‘2’开始走，只有2个选择，往左下方和右下方。

往左下方是‘4’，得到的最终值为6，往右下方是‘5’，得到的最终值是7.这时当然选择右下方。

我们就将‘2’改写成2+5=7。 再次考虑倒数第二排的7，

同理，应选择左下，得到最终值是12。还是将‘7’改写成5+7=12。

依次类推则倒数第二排变为：

7 12 10 10

原数字三角形变为：

7

3 8

8 1 0

7 12 10 10

4 5 2 6 5

这时再考虑第三行第一个。有两种选择：左下和右下。

假设走左下方，由于这时左下的值已经是从左下开始走到底的最优值，我们不需要在选择下一步怎么走，直接加上左下的值即可。

同理，走右下时，直接加上右下的值即可。因为此时右下的值已经是从右下走到底的最优值，不需要选择了。

再比较走两条路的值，右边的值更大，选择右边的值。则第三行的第一个值更新为8+12=20。

以此类推，得到下面的数字三角形：

7

3 8

20 13 10

7 12 10 10

4 5 2 6 5

同理，更新第二排，有：

7

23 21

20 13 10

7 12 10 10

4 5 2 6 5

最后一个了，有：

30

23 21

20 13 10

7 12 10 10

4 5 2 6 5

贝尔曼方程：f[x][y]=max(f[x+1][y],f[x+1][y+1])+a[x][y];

#include <bits/stdc++.h>

#define fp(i,l,r) for(register int i=(l);i<=(r);i++)

#define fd(i,r,l) for(register int i=(r);i>=(l);i--)

using namespace std;

inline int \_max(int a,int b){

if(a>b){

return a;

}

else{

return b;

}

}

int a[1000][1000],f[1000][1000];

int main(){

int n;

scanf("%d",&n);

fp(i,1,n){

fp(j,1,i){

scanf("%d",&a[i][j]);

f[i][j]=a[i][j];

}

}

fd(i,n-1,1){

fp(j,1,i){

f[i][j]=\_max(f[i+1][j],f[i+1][j+1])+a[i][j];

}

}

printf("%d",f[1][1]);

return 0;

}//...

【题目】

2. Let x be a positive integer variable with an initial value of 1. Consider the following computational procedure: At each step, x can be increased by two ways: (1) x = x + 4, or, (2) x = 3x.

Your problem is: How can x reach the target value n(n≥5) in a minimum number of computation steps?

Design a dynamic programming algorithm to solve this problem. Requirements: 1) Give the relevant Bellman equation and 2) Give the pseudo-code of the algorithm.

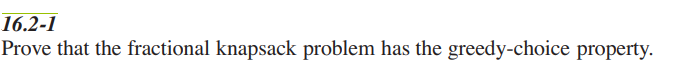
设 x 是一个初始值为 1 的正整数变量。考虑以下计算过程： 在每一步，x 可以通过两种方式增加：(1) x = x + 4，或者，(2) x = 3x。

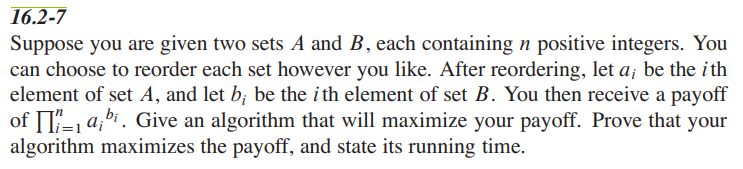
您的问题是：x 如何在最少的计算步骤中达到目标值 n(n≥5)？

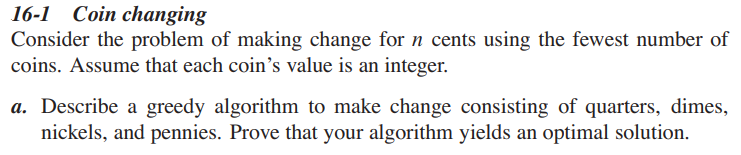
设计一个动态规划算法来解决这个问题。 要求：1）给出相关的贝尔曼方程，2）给出算法的伪代码。

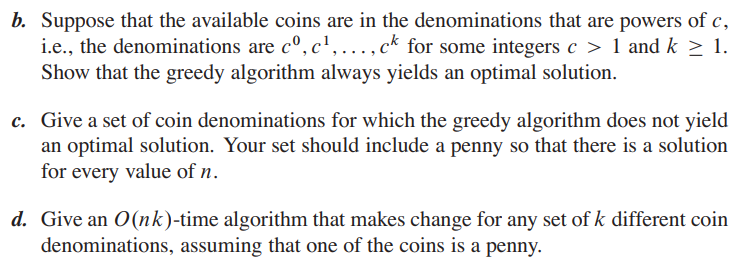
（伪代码用自底向上写）

Greedy Algorithm









（a和d写伪代码，d用自底向上写，时间复杂度不做要求；b给出证明，c举例即可）

GRAPH BFS & DFS

