小蝌蚪

破题

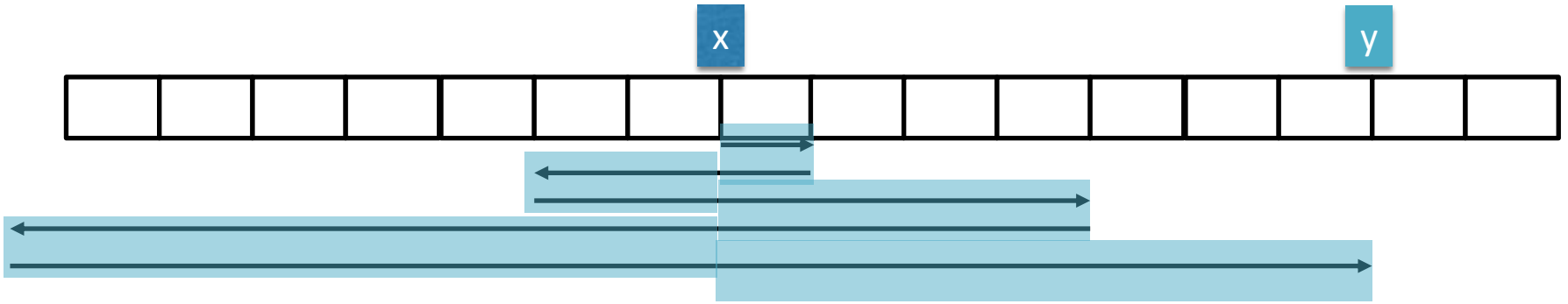
请同学简述题意
突出核心要点

经过多次来回走，小蝌蚪可以走到y

模拟

数据范围：
100%数据，0<=x,y<=1000

演示



首先游到x+1的位置，然后反向游到x-2的位置，再反向游到x+4的位置，在之后就是游到x-8。以此类推

在到达y之前，看成一系列的往返走

代码

模拟

```
ll x, y;
cin >> x >> y;
ll ans = 0; ll by = 1; ll dir = 1;
while(true) {
    if((dir==1 && x<=y && y<=x+by) ||
        (dir==-1 && x-by<=y && y<=x)) {
        ans += abs(y-x);
        cout << ans << endl;
        break;
    } else {
        ans += by*2;
        by *= 2;
        dir *= -1;
    }
}
```

by表示当前移动的最大距离

dir表示方向（1为向右）

判断是否能够回家

最后一步不用走完by距离

相当于蝌蚪往一个方向走by距离，然后回到原点

自来水供给

请同学简述题意
突出核心要点

N个在一条线上的村庄，安装粗管和细管，粗管每公里8000元，细管每公里2000元，如何搭配使得总费用最小？

问题分析

这是一个计算的问题，找最小花费问题

可以考虑，把粗水管埋到第i个村，剩下的村庄从第i+1个村庄开始使用细水管。

注意题目要求必须有粗水管！

问题分析

如何计算前i个村庄装粗水管的花费？

d[i] * 8000, d[i]表示第i个村庄的距离

后n-i个村庄装细水管需要多少花费？

一项项去计算吗？！

可以考虑先计算前缀和

复习

连续和
转换为
前缀和的差

当遇到多次计算"连续和"问题

可以考虑**预计算**前缀和数组

```
9     int n;
10    cin >> n;
11    for(int i=1;i<=n;i++){
12        cin>>d[i];
13        d[i]+=d[i-1];
14        sum[i]=sum[i-1]+d[i];
15    }
16    int ans = INF;
17    for(int i=1; i<=n; i++){
18        int p = 8000*d[i] + 2000*(sum[n]-sum[i]-(n-i)*d[i]);
19        ans = min(ans, p);
20    }
21    cout<<ans<<endl;
```

从左到右依次
查看每个村庄

找出费用最小值

易错点总结

ans定义的不够大！

前缀和数组越界

未想到用前缀和数组

```
9       int n;
10      cin >> n;
11      for(int i=1;i<=n;i++){
12          cin>>d[i];
13          d[i]+=d[i-1];
14          sum[i]=sum[i-1]+d[i];
15      }
16      int ans = INF;
17      for(int i=1; i<=n; i++){
18          int p = 8000*d[i] + 2000*(sum[n]-sum[i]-(n-i)*d[i]);
19          ans = min(ans, p);
20      }
21      cout<<ans<<endl;
```

从左到右依次
查看每个村庄

找出费用最小值

动物园

请同学写出题目大意
已知什么求什么

对整数数组h
每次可将相邻两点h_i和h_{i+1}减1
求最少下降多少值(每次开销为2)
可将各点都降至某个相同非负值(做不到则得-1)

请同学阅读[数据规模和约定]
识别部分得分点

【数据规模与约定】
测试点 2 的所有子测试用例满足N≤3以及h_i≤100。
测试点 3-8 的所有子测试用例满足N≤100以及h_i≤100。
测试点 9-14 的所有子测试用例满足N≤100。
测试点 15 没有额外限制。
此外，测试点 3-5 和 9-11 中的N均为偶数，测试点 6-8 和 12-14 中的N均为奇数。
本题每个测试用例输入包含T（1≤T≤100）个子测试用例。
输入保证所有子测试用例的N之和不超过10^5

子测试用例1

输入样例
3
8 10 5

输出样例

14

h_2和h_3先联动下降2,共花销4
h_1和h_2再联动下降5,共花销10

输入样例

6
4 6 4 4 6 4

输出样例

16

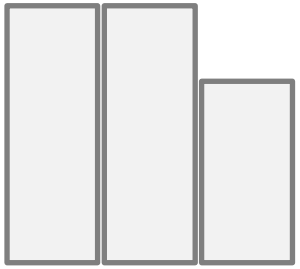
h_1和h_2先联动下降2,共花销4
[2 4 4 4 6 4]
h_2和h_3再联动下降2,共花销4
[2 2 2 4 6 4]
h_4和h_5再联动下降2,共花销4
[2 2 2 2 4 4]
h_5和h_6再联动下降2,共花销4
[2 2 2 2 2 2]

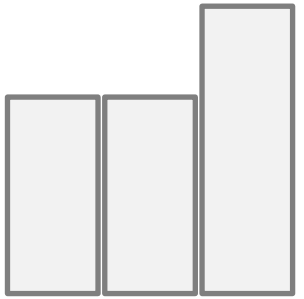
从1和2开始

若$h_1==h_2$:
不需要调整
若$h_1>h_2$:
无法高度一致
若$h_1<h_2$:
h_2至少须被拉平到h_1,
h_2只能跟h_3联动下降(h_2-h_1)格

若h_3下降(h_2-h_1)后变负数,则终止

h_1和h_2等高之后



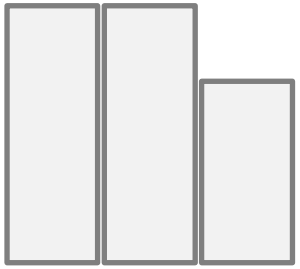


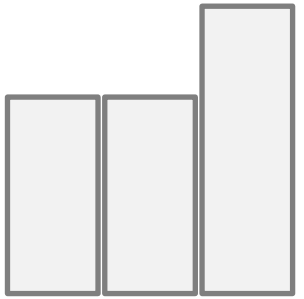
若h3==h2:不需要调整

若$h_3>h_2$:
1和2联动/2和3联动,都无法抹平2和3
只能3和4联动下降(h_3-h_2)格
使得2和3平级

若$h_3<h_2$:
不管2和3(到时再联动下降1和2就行)
继续往后,找到下一个比前邻居更高点
$h_p>h_{p-1}$,将p和p+1联动降(h_p-h_{p-1})格

调整持续到n-2和n-1号
结束后整个序列是不升的





h_1和h_2等高之后

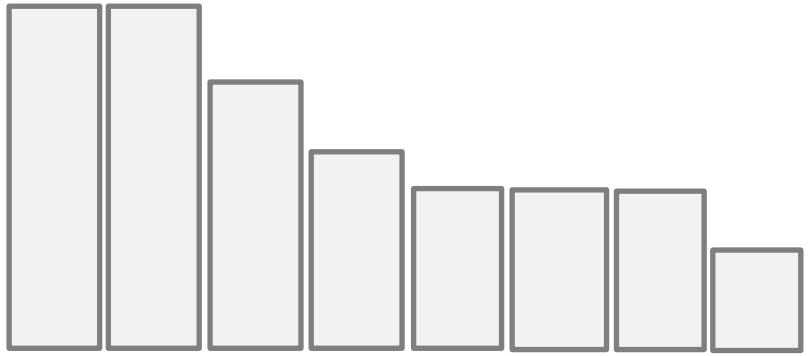
若h3==h2:不需要调整

若$h_3>h_2$:
1和2联动/2和3联动,都无法抹平2和3
只能3和4联动下降(h_3-h_2)格
使得2和3平级

若$h_3<h_2$:
不管2和3(到时再联动下降1和2就行)
继续往后,找到下一个比前邻居更高点
$h_p>h_{p-1}$,将p和p+1联动降(h_p-h_{p-1})格

特判:若中途p+1下降(h_p-h_{p-1})后
变负数,则失败

特判:若最后$h_n>h_{n-1}$,则失败



对不升序列再倒序遍历

从n号到3号(1和2一定等高)

若$h_{i-1}==h_i$:
不需要调整
若$h_{i-1}>h_i$:
则将i-1和i-2号
联动下降($h_{i-1}-h_i$)格

特判:若最终导致$h_1>h_2$,则失败

solve函数

```
6  ll solve(){
7      ll cnt=0;
8      if(n==1) return 0;
9      if(n==2){
10         if(h[1]==h[2]) return 0;
11         else return -1;
12     }
13     if(h[1]>h[2]) return -1;
```

cnt统计总下降量

若n==1

n==2时:1和2必须同高

1不能高于2

solve函数

```
14  for(ll i=1;i<=n-2;i++){
15      diff=h[i+1]-h[i];
16      if(diff<=0) continue;
17      h[i+1]-=diff;
18      h[i+2]-=diff;
19
20      cnt+=2*diff;
21  }
22  if(h[n-1]<h[n]) return -1;
```

h_{i+1}<=h_i无需操作

当h_{i+1}>h_i，
将i+2和i+1联动下降diff

特判

特判

请描述两种特判

```
23  for(ll i=n;i>=3;i--){
24      diff=h[i-1]-h[i];
25      if(!diff) continue;
26      h[i-1]-=diff;
27      h[i-2]-=diff;
28
29  }
30  if(h[1]>h[2]) return -1;
31  return cnt;
```

当$h_{i-1}>h_i$，
将i-1和i-2联动下降diff

特判

请尝试构造:1和2号最终不等高的一种可能情形

回老家

请同学写出题目大意
已知什么求什么

在N×N方阵中
绕过障碍点,向下或向右走,至多转向M次
求从左上点到右下点的总路线数

请同学阅读[数据规模和约定]
识别部分得分点

【数据规模与约定】
2≤N≤50。
2号测试点满足M=1。
3-5号测试点满足M=2。
6-10号测试点满足M=3。

输入样例

3 1

...

...

...

输出样例

2

两条路线DDRR, RRDD

(D和R分别表示down和right)

多手算几组

输入样例

3 2

...

...

...

输出样例

4

四条路线DDRR,DRRD,RDDR,RRDD

(转向次数k<=2)

多手算几组

输入样例

3 3

...

...

...

输出样例

6

六条路线

DDRR,DRDR,DRRD,RDDR,RDRD,RRDD

(以上三组子样例都是无障碍点)

多手算几组

输入样例

```
4 3
...H
.H..
....
H...
```

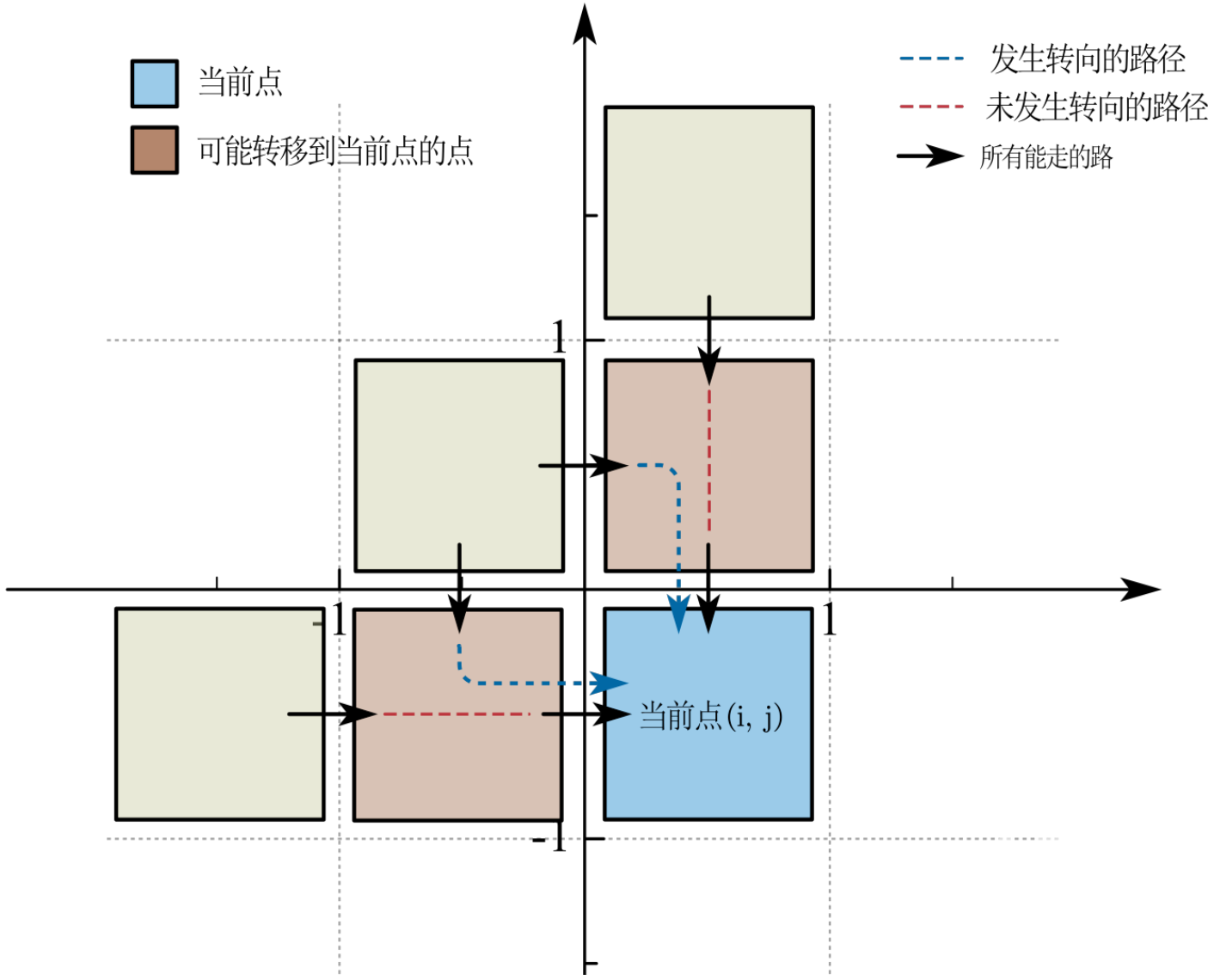
输出样例

```
6
```

六条路线

先DD,然后: RDRR,RRDR,RRRD,

先RR,然后: DDDR,DDRD,DRDD



DP和记忆化DFS都可以尝试

记忆化搜索

f[i][j][c][x]记住
从(i,j)到(n,n)的总路径数的结果
额外状态要求:已转向c次
且是通过x方向的移动而到达(i,j)

F(i,j,c,x)函数计算总路径数

约定x:0代表横向,1代表纵向

主函数

```
18      cin>>t;
19      for(ll i=1;i<=t;i++){
20          memset(a,0,sizeof(a));
21          memset(f,0,sizeof(f));
22          memset(ok,0,sizeof(ok));
23          cin>>n>>k;
24          for(ll i=1;i<=n;i++)
25              for(ll j=1;j<=n;j++){
26                  cin>>ch;
27                  a[i][j]=(ch=='.'?1:0);
28              }
29          cout<<                          <<endl;
30      }
```

f[i][j][c][x]记忆
从(i,j)到(n,n)的
总路径数的结果
要求:已转向c次,
且是通过x方向的移
动而到达(i,j)

最多转k次方向!

障碍点赋值为0
四周缓冲带默认是0

```
8  ll a[N][N],f[N][N][K][2],ok[N][N][K][2];
9  ll F(ll x,ll y,ll cnt,bool dir){
10     if(cnt>k||!a[x][y])
11         return 0;
12     if(ok[x][y][cnt][dir])
13         return f[x][y][cnt][dir];
14     ok[x][y][cnt][dir]=1;
15     f[x][y][cnt][dir]=0;
16     if(x==n&&y==n)
17         return
18     if(a[x+1][y])
19         f[x][y][cnt][dir]+=
20     if(a[x][y+1])
21         f[x][y][cnt][dir]+=F(x,y+1,(dir?cnt+1:cnt),0);
22     return f[x][y][cnt][dir];
23 }
```

f[i][j][c][x]记忆从(i,j)到(n,n)的总路径数的结果
要求:已转向c次,且是通过x方向的移动而到达(i,j)

dir:0横向,1纵向

请解释这个三元表达式

思考:第17行能否直接返回1	
不能!	请解释原因

DP

f[i][j][k][R]从起点到第i行第j列
已转折恰好k次
最后一步向右到达的方案数

f[i][j][k][D]从起点到第i行第j列
已转折恰好k次
最后一步向下到达的方案数

```
4  const ll N=509;
5  const ll M=5;
6  ll n,m,f[N][N][M][2];
7  char d[N][N];
8  #define R 0
9  #define D 1
```

```
24          for(ll i=1;i<=n;++i)
25              for(ll j=1;j<=n;++j){
26                  if(d[i][j]=='H'){
27                      for(ll k=0;k<=m;++k)
28                          f[i][j][k][R]=f[i][j][k][D]=0;
29                      continue;
30                  }
31                  if(i==1&&j==1)continue;
32                  if(i==1&&j==2){
33                      
34                      continue;
35                  }
36                  if(i==2&&j==1){
37                      
38                      continue;
39                  }
```

```
24          for(ll i=1;i<=n;++i)
25              for(ll j=1;j<=n;++j){
26                  if(d[i][j]=='H'){
31                  if(i==1&&j==1)continue;
32                  if(i==1&&j==2){
36                  if(i==2&&j==1){
40                  f[i][j][0][R]=f[i][j-1][0][R];
41                  f[i][j][0][D]=f[i-1][j][0][D];
42                  for(ll k=1;k<=m;++k){
43                      f[i][j][k][R]=
44                      f[i][j][k][D]=
45                  }
46              }
```

```
47          ll ans=0;
48          for(ll k=1;k<=m;++k)
49              ans+=
```

3081