CODE+5.5 清华大学研究生招生计算机类上机考试复现练习赛 题目讲解

清华大学计算机系

张慧盟

I 面试（INTERVIEW）

太长不看版：二分答案

题意

- 给定n个小朋友的身高，要求从前x个人中选出m个人，他们的最高身高和最低身高之差不能超过k
- 求x的最小值；如果不存在这样的x，输出impossible
- $1 \le m \le n \le 10^5$，$0 \le k \le 10^5$，$1 \le h_i \le 10^5$

测试点编号	n, m	h_i, k
1,2	$1 \le m \le n \le 100$	$k = 0; 1 \le h_i \le 100$
3,4	$1 \le m \le n \le 2 \times 10^3$	$0 \le k \le 50; 1 \le h_i \le 100$
5,6,7,8		$0 \le k \le 100; 1 \le h_i \le 5 \times 10^3$
9,10,11,12		$0 \le k \le 5 \times 10^3; 1 \le h_i \le 5 \times 10^3$
13,14	$1 \le m \le n \le 2 \times 10^3$	$0 \le k \le 10^5; 1 \le h_i \le 10^5$
15,16	$1 \le m \le n \le 10^5$	$0 \le k \le 100; 1 \le h_i \le 10^5$
17,18,19,20	$1 \le m \le n \le 10^5$	$0 \le k \le 10^5; 1 \le h_i \le 10^5$

10分解法：暴力

测试点编号	n, m	h_i, k
1,2	$1 \le m \le n \le 100$	$k = 0; 1 \le h_i \le 100$
3,4	$1 \le m \le n \le 2 \times 10^3$	$0 \le k \le 50; 1 \le h_i \le 100$
5,6,7,8		$0 \le k \le 100; 1 \le h_i \le 5 \times 10^3$
9,10,11,12		$0 \le k \le 5 \times 10^3; 1 \le h_i \le 5 \times 10^3$
13,14	$1 \le m \le n \le 2 \times 10^3$	$0 \le k \le 10^5; 1 \le h_i \le 10^5$
15,16	$1 \le m \le n \le 10^5$	$0 \le k \le 100; 1 \le h_i \le 10^5$
17,18,19,20	$1 \le m \le n \le 10^5$	$0 \le k \le 10^5; 1 \le h_i \le 10^5$

- 用数组维护每个数已经出现的次数
- 每读入一个数，更新数组
- 并判断是否已经有数出现m次
- 算法复杂度为$O(nL)$（L是h_i的范围）

```
cnt[101] = {0}
for i = 1 to n
    input(h[i])
    cnt[h[i]]++
    for j = 1 to 100
        if (cnt[j] >= m)
            output(i)
            exit
output("impossible")
```

60分解法：还是暴力

测试点编号	n, m	h_i, k
1,2	$1 \le m \le n \le 100$	$k = 0; 1 \le h_i \le 100$
3,4	$1 \le m \le n \le 2 \times 10^3$	$0 \le k \le 50; 1 \le h_i \le 100$
5,6,7,8		$0 \le k \le 100; 1 \le h_i \le 5 \times 10^3$
9,10,11,12		$0 \le k \le 5 \times 10^3; 1 \le h_i \le 5 \times 10^3$
13,14	$1 \le m \le n \le 2 \times 10^3$	$0 \le k \le 10^5; 1 \le h_i \le 10^5$
15,16	$1 \le m \le n \le 10^5$	$0 \le k \le 100; 1 \le h_i \le 10^5$
17,18,19,20	$1 \le m \le n \le 10^5$	$0 \le k \le 10^5; 1 \le h_i \le 10^5$

- 用数组维护每个数已经出现的次数
- 每读入一个数，更新数组
- 判断每个长度为$k+1$的区间内的数组和是否超过m
- 前缀和优化

60分解法：还是暴力

- 算法复杂度为$O(nL)$（L是h_i的范围）

测试点编号	n, m	h_i, k
1,2	$1 \le m \le n \le 100$	$k = 0; 1 \le h_i \le 100$
3,4	$1 \le m \le n \le 2 \times 10^3$	$0 \le k \le 50; 1 \le h_i \le 100$
5,6,7,8		$0 \le k \le 100; 1 \le h_i \le 5 \times 10^3$
9,10,11,12		$0 \le k \le 5 \times 10^3; 1 \le h_i \le 5 \times 10^3$
13,14	$1 \le m \le n \le 2 \times 10^3$	$0 \le k \le 10^5; 1 \le h_i \le 10^5$
15,16	$1 \le m \le n \le 10^5$	$0 \le k \le 100; 1 \le h_i \le 10^5$
17,18,19,20	$1 \le m \le n \le 10^5$	$0 \le k \le 10^5; 1 \le h_i \le 10^5$

```
cnt[5001] = {0}
for i = 1 to n
    input(h[i])
    cnt[h[i]]++
    sum = 0
    for j = 0 to k-1
        sum += cnt[j]
    for j = k to 5000
        sum += cnt[j]
        if (sum >= m)
            output(i)
            exit
        sum -= cnt[j - k]
output("impossible")
```

70(80)分解法：对暴力的优化

测试点编号	n, m	h_i, k
1,2	$1 \le m \le n \le 100$	$k = 0; 1 \le h_i \le 100$
3,4	$1 \le m \le n \le 2 \times 10^3$	$0 \le k \le 50; 1 \le h_i \le 100$
5,6,7,8		$0 \le k \le 100; 1 \le h_i \le 5 \times 10^3$
9,10,11,12		$0 \le k \le 5 \times 10^3; 1 \le h_i \le 5 \times 10^3$
13,14	$1 \le m \le n \le 2 \times 10^3$	$0 \le k \le 10^5; 1 \le h_i \le 10^5$
15,16	$1 \le m \le n \le 10^5$	$0 \le k \le 100; 1 \le h_i \le 10^5$
17,18,19,20	$1 \le m \le n \le 10^5$	$0 \le k \le 10^5; 1 \le h_i \le 10^5$

- 用数组维护每个数已经出现的次数
- 每读入一个数，更新数组
- 判断包含这个数的长度为$k+1$的区间内的数组和是否超过m

70(80)分解法：对暴力的优化

- 复杂度从$O(nL)$下降到$O(nk)$

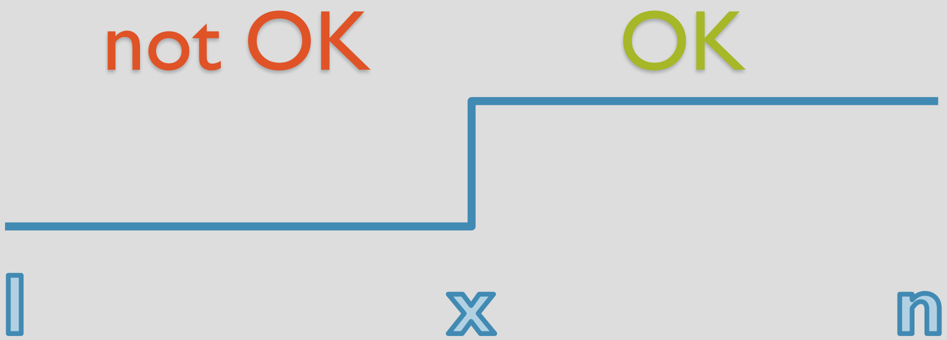
测试点编号	n, m	h_i, k
1,2	$1 \le m \le n \le 100$	$k = 0; 1 \le h_i \le 100$
3,4		$0 \le k \le 50; 1 \le h_i \le 100$
5,6,7,8	$1 \le m \le n \le 2 \times 10^3$	$0 \le k \le 100; 1 \le h_i \le 5 \times 10^3$
9,10,11,12		$0 \le k \le 5 \times 10^3; 1 \le h_i \le 5 \times 10^3$
13,14	$1 \le m \le n \le 2 \times 10^3$	$0 \le k \le 10^5; 1 \le h_i \le 10^5$
15,16	$1 \le m \le n \le 10^5$	$0 \le k \le 100; 1 \le h_i \le 10^5$
17,18,19,20	$1 \le m \le n \le 10^5$	$0 \le k \le 10^5; 1 \le h_i \le 10^5$

```
cnt[100001] = {0}
for i = 1 to n
    input(h[i])
    cnt[h[i]]++
    sum = 0
    start = max(0, h[i] - k)
    for j = 0 to k-1
        sum += cnt[start + j]
    for j = k to min(2*k, 100000-start)
        sum += cnt[start + j]
        if (sum >= m)
            output(i)
            exit
        sum -= cnt[start + j - k]
output("impossible")
```

100分解法：二分答案

测试点编号	n,m	h_i,k
1,2	$1 \le m \le n \le 100$	$k = 0; 1 \le h_i \le 100$
3,4	$1 \le m \le n \le 2 \times 10^3$	$0 \le k \le 50; 1 \le h_i \le 100$
5,6,7,8		$0 \le k \le 100; 1 \le h_i \le 5 \times 10^3$
9,10,11,12		$0 \le k \le 5 \times 10^3; 1 \le h_i \le 5 \times 10^3$
13,14	$1 \le m \le n \le 2 \times 10^3$	$0 \le k \le 10^5; 1 \le h_i \le 10^5$
15,16	$1 \le m \le n \le 10^5$	$0 \le k \le 100; 1 \le h_i \le 10^5$
17,18,19,20	$1 \le m \le n \le 10^5$	$0 \le k \le 10^5; 1 \le h_i \le 10^5$

- 显然x是否符合要求满足单调性：多加几个人后仍然能找到符合要求的人
- 因此可以对答案x进行二分查找



100分解法：二分答案

- 复杂度为$O((n+L)\log n)$

```
bool isOk(int x)
    cnt[100001] = {0}
    for i = 1 to x
        cnt[h[i]]++
    sum = 0
    for i = 0 to k-1
        sum += cnt[i]
    for i = k to 100000
        sum += cnt[i]
        if (sum >= m)
            return true
        sum -= cnt[i-k]
    return false
```

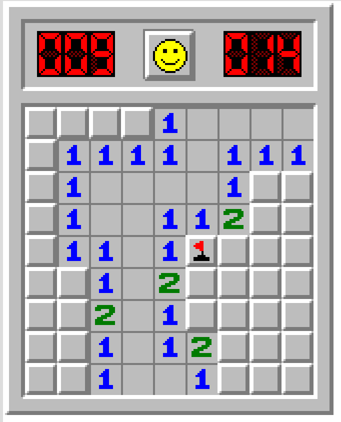
```
l = 1
r = n + 1
while (l < r)
    mid = (l + r) / 2
    if (isOk(mid))
        r = mid
    else l = mid + 1
if (l == n + 1)
    output("impossible")
else
    output(l)
```

2 扫雷（MINE）

太长不看版：大模拟，没别的了

题意

- 写一个扫雷模拟程序
- 游戏（未失败）过程中方块可能的3种状态：
  - 未探明
  - 已插旗
  - 已探明



题意

- 实现以下四种操作：
  - Flag：插上/撤销一面旗帜，可能输出结果为swept/success/cancelled
  - Sweep：对方块进行**扫雷**，可能输出结果为swept/flagged/**扫雷结果**
  - DSweep：对已探明方块八连通的未探明方块进行**扫雷**，可能输出结果为not swept/failed/**扫雷结果**
  - Quit：放弃并退出

（下一页再讲扫雷和扫雷结果）

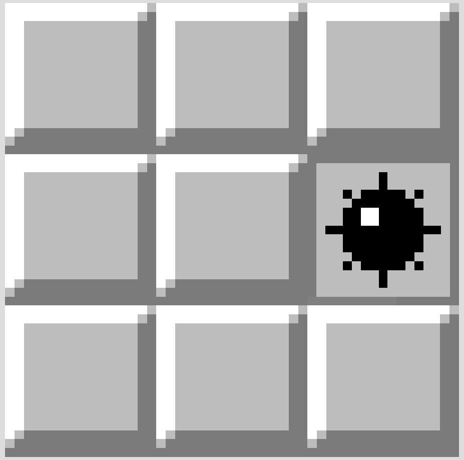
题意

- 对(x, y)进行扫雷的步骤：
  - 首先判断(x, y)是否为地雷，如果是，则扫雷失败，输出boom，并结束游戏
  - 否则标记(x, y)为已探明，令它显示相邻方块的地雷总数
  - 如果相邻方块中没有地雷，则自动对相邻的未探明和已插旗方块进行扫雷（先清除旗帜信息）
  - 扫雷（全部）结束后输出新探明的方块总数，以及它们的信息

题意

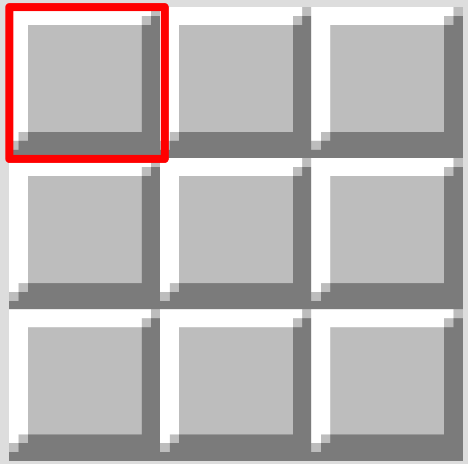
- 游戏结束
  - （除Quit操作外的）所有操作结束时，都应检查游戏是否结束
  - 一旦发现游戏结束，就立刻忽略之后的所有输入，并输出结果信息
- 游戏可能结果
  - 游戏胜利：所有没有地雷的方块均被探明，输出finish
  - 扫雷失败：扫雷过程中踩到雷，输出game over
  - 退出游戏：因为Quit操作而退出，输出give up
- 最后输出行动次数

样例 I



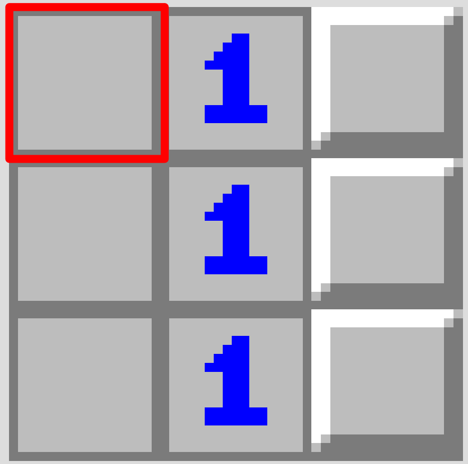
- $n = m = 3$
- 只有一颗雷

样例1 – 操作1



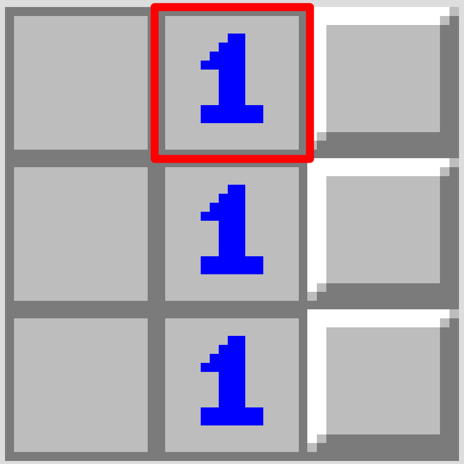
- 输入：Sweep 1 1

样例1 – 操作1



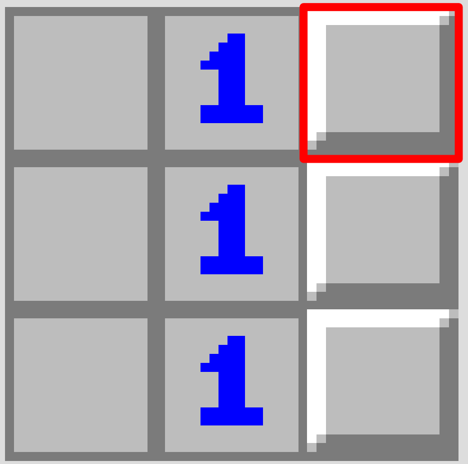
- 输入：Sweep 1 1
- 输出
  - 6 cell(s) detected
  - 1 1 0
  - 1 2 1
  - 2 1 0
  - 2 2 1
  - 3 1 0
  - 3 2 1

样例1 – 操作2



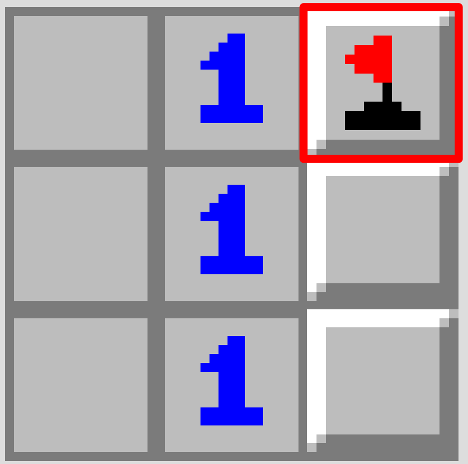
- 输入： DSweep 1 2
- 输出： failed

样例1 – 操作3



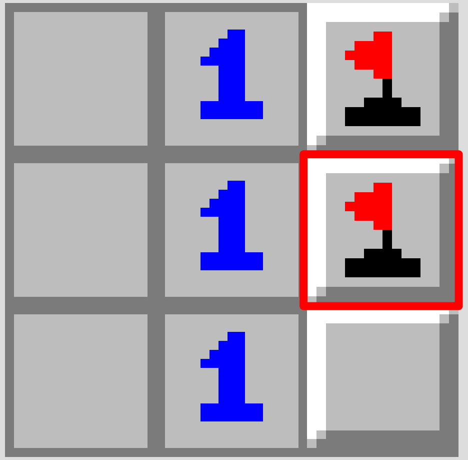
- 输入：Flag 1 3

样例1 – 操作3



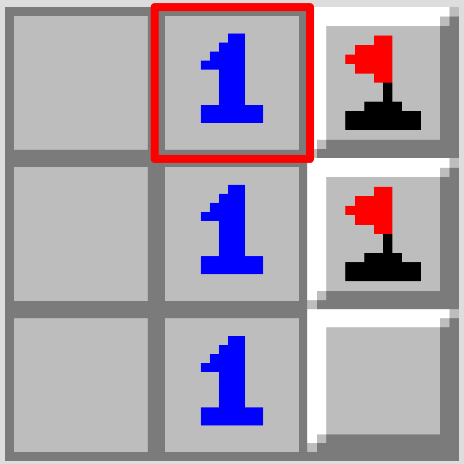
- 输入： Flag 1 3
- 输出： success

样例1 – 操作4



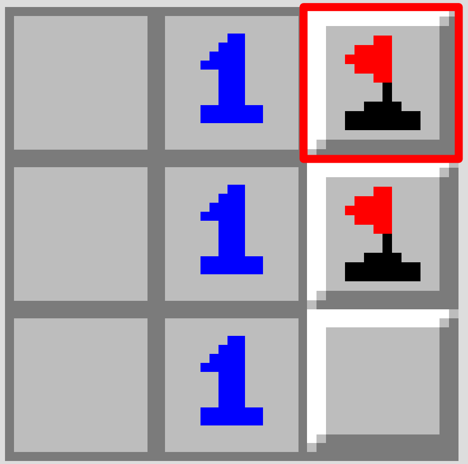
- 输入： Flag 2 3
- 输出： success

样例1 – 操作5



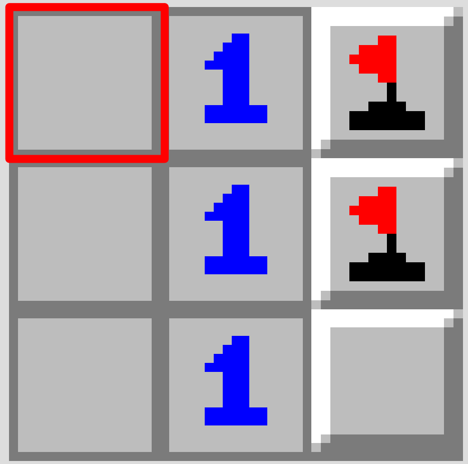
- 输入： DSweep 1 2
- 输出： failed

样例1 – 操作6



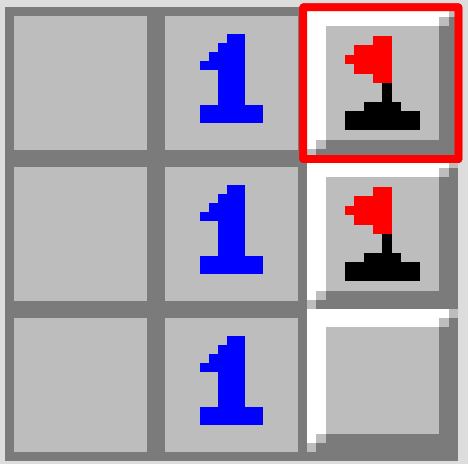
- 输入： Sweep 1 3
- 输出： flagged

样例1 – 操作7



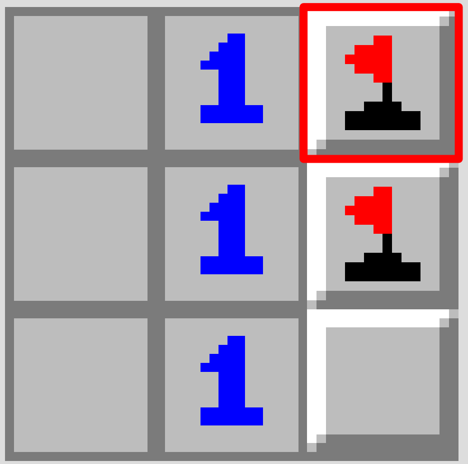
- 输入：Flag 1 1
- 输出：swept

样例1 – 操作8



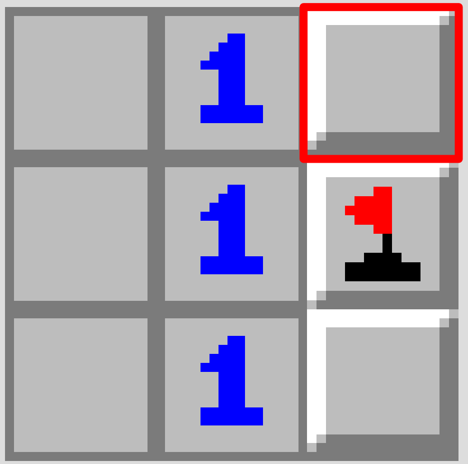
- 输入：DSweep 1 3
- 输出：not swept

样例1 – 操作9



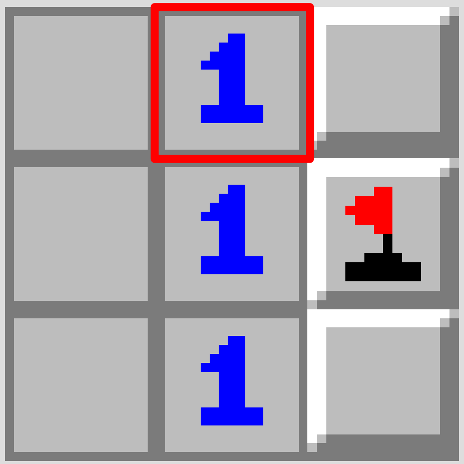
- 输入：Flag 1 3

样例1 – 操作9



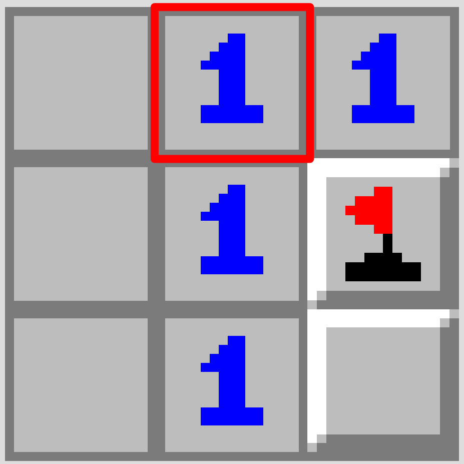
- 输入： Flag 1 3
- 输出： cancelled

样例1 – 操作10



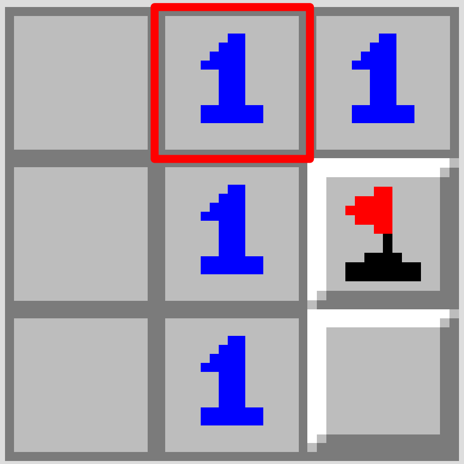
- 输入：DSweep 1 2

样例1 – 操作10



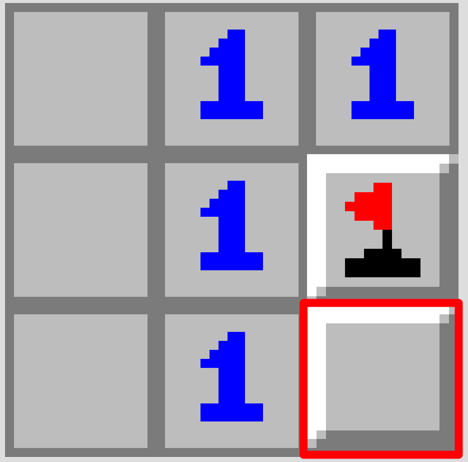
- 输入： DSweep 1 2
- 输出：
  - 1 cell(s) detected
  - 1 3 1

样例1 – 操作11



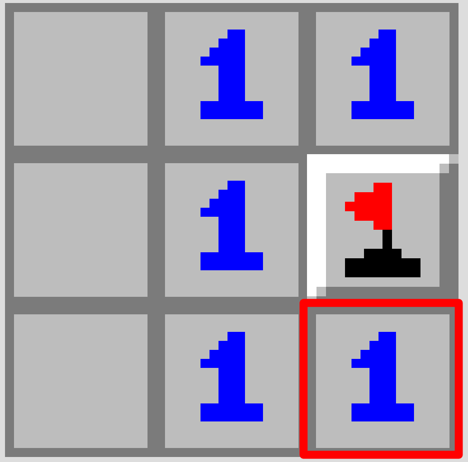
- 输入： DSweep 1 2
- 输出： no cell detected

样例1 – 操作12



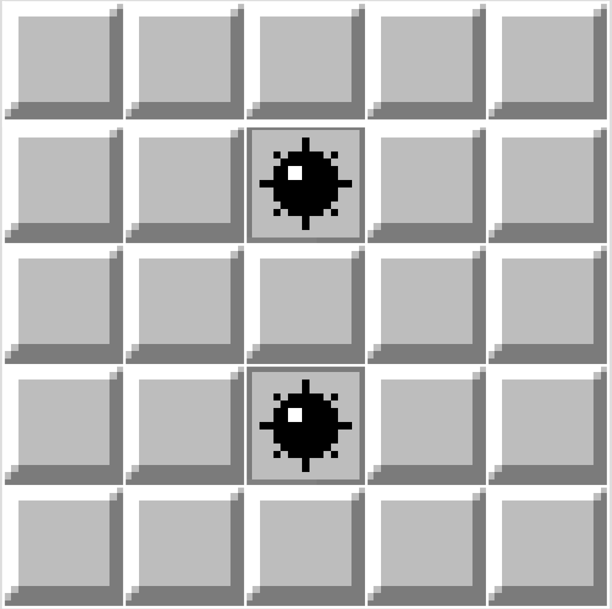
- 输入：Sweep 3 3

样例1 – 操作12



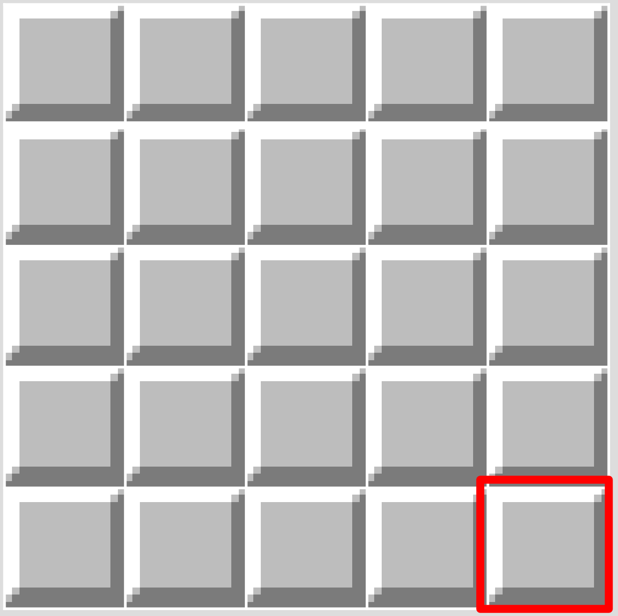
- 输入：Sweep 3 3
- 输出：
  - 1 cell(s) detected
  - 3 3 1
  - finish
  - total step 12
- ~~输入：Quit~~

样例2



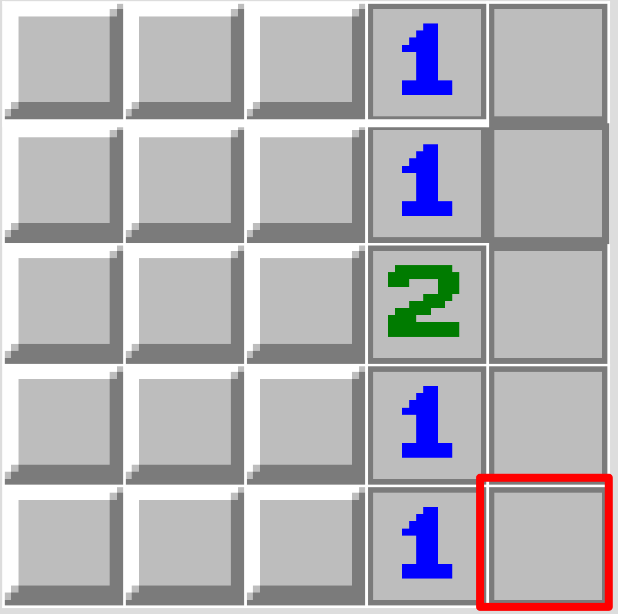
- $n = m = 5$
- 两颗雷

样例2 – 操作1



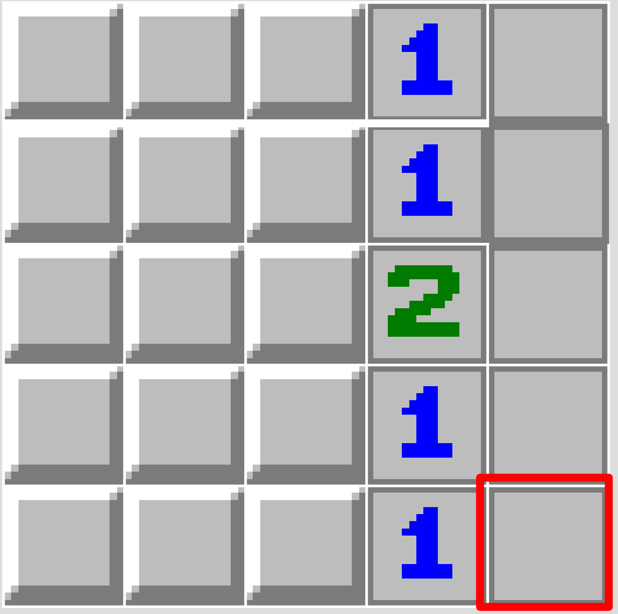
- 输入：Sweep 5 5

样例2 – 操作1



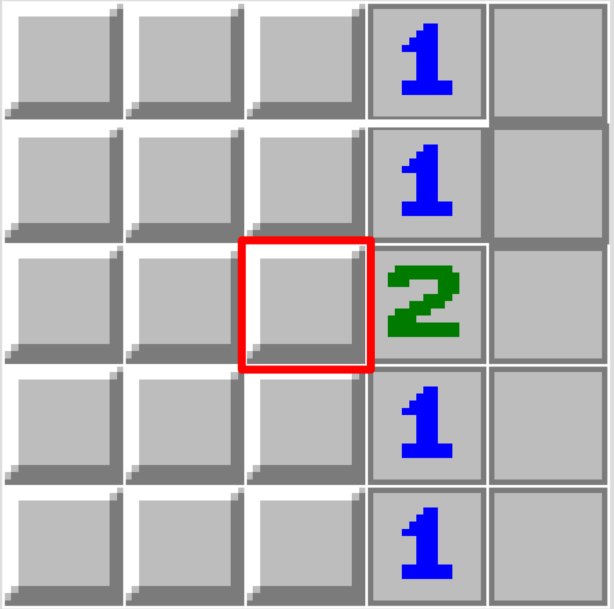
- 输入： Sweep 5 5
- 输出：
  - 10 cell(s) detected
  - 1 4 1
  - 1 5 0
  - 2 4 1
  - 2 5 0
  - 3 4 2
- 3 5 0
- 4 4 1
- 4 5 0
- 5 4 1
- 5 5 0

样例2 – 操作2



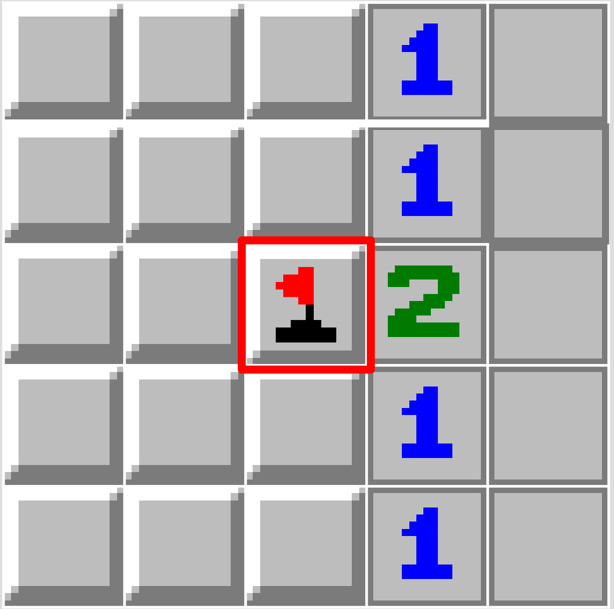
- 输入：Sweep 5 5
- 输出：swept

样例2 – 操作3



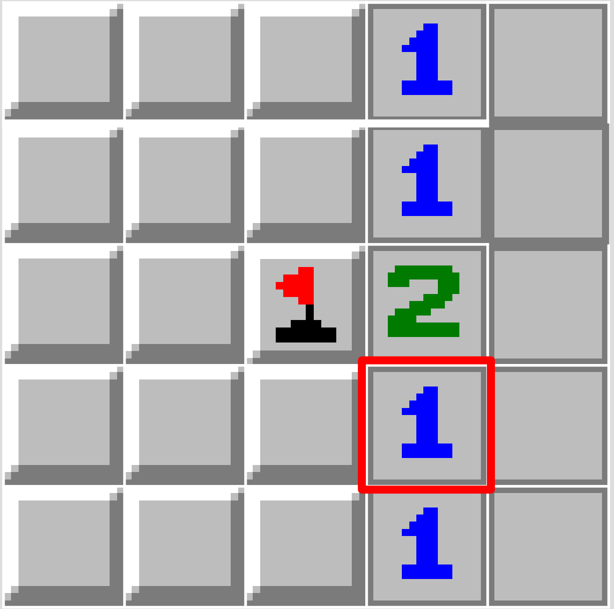
- 输入：Flag 3 3

样例2 – 操作3



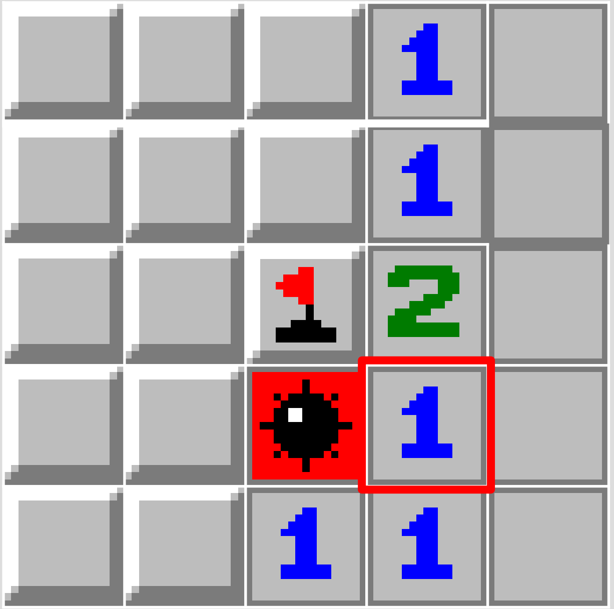
- 输入： Flag 3 3
- 输出： success

样例2 – 操作4



- 输入：DSweep 4 4

样例2 – 操作4

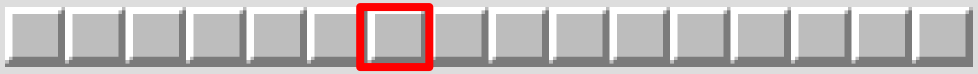


- 输入： DSweep 4 4
- 输出：
  - boom
  - game over
  - total step 4
- ~~输入： Quit~~

样例3

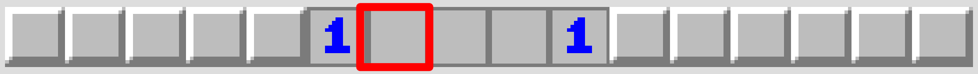
- $n = 1$，$m = 16$
- 两颗雷

样例3 – 操作1



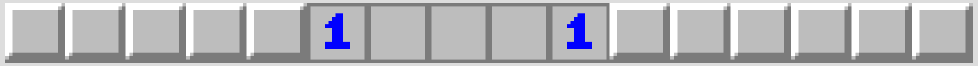
- 输入：Sweep 1 7

样例3 – 操作1



- 输入： Sweep 1 7
- 输出：
  - 5 cell(s) detected
  - 1 6 1
  - 1 7 0
  - 1 8 0
  - 1 9 0
  - 1 10 1

样例3 – 操作2



- 输入： Quit
- 输出：
  - give up
  - total step 1
- ~~输入： Sweep 1 1~~

数据范围

- $n, m \le 1000$，$q \le 60000$
- 性质A：保证只有Sweep操作和Quit操作
- 性质B：保证没有DSweep操作
- 请不要使用过于缓慢的输出方式

测试点	n	m	q	性质
$1 \sim 2$	≤ 10	≤ 10	≤ 60	A
$3 \sim 4$	≤ 10	≤ 10	≤ 60	B
$5 \sim 6$	≤ 10	≤ 10	≤ 60	无
$7 \sim 8$	$= 1$	≤ 1000	≤ 1000	A
$9 \sim 10$	$= 1$	≤ 1000	≤ 1000	B
$11 \sim 12$	$= 1$	≤ 1000	≤ 1000	无
$13 \sim 14$	≤ 300	≤ 300	≤ 8000	A
$15 \sim 16$	≤ 300	≤ 300	≤ 8000	B
$17 \sim 19$	≤ 300	≤ 300	≤ 8000	无
20	≤ 1000	≤ 1000	≤ 60000	无

100分解法：模拟

- 记录：
  - 每个方格的状态
  - 剩余未探明空白方格数量
- 更新信息
- 扫雷过程：BFS
- 游戏结束条件：
  - Quit
  - 扫雷过程中踩到雷
  - 探明所有未探明方格

- **isMine：是否为地雷**
- **swept：是否已探明**
- **flagged：是否已插旗**
- **surroundMines：周围地雷数量**
- **surroundFlags：周围旗帜数量**

100分解法：模拟

- 需要注意的几点：
  - 输入输出的细节
  - 扫雷时如果发现周围没有雷，则除了对相邻未探明方块扫雷外，已插旗（但是插错了）的方块也需要进行扫雷（先把旗子拿掉）
  - 如果扫雷的过程写成递归DFS，会爆栈（过不了最后一个点）
  - 使用速度>=scanf/printf的方法进行输入输出

3 多项式求和（POLYNOMIAL）

太长不看版：杨辉三角+矩阵快速幂递推

题意

- 求
$$S(n)=\sum_{k=0}^{n}a^k\sum_{i=0}^{m}b_ik^i$$
模10^9+7的结果
- $1\le n,a,b_i\le 10^9$，$1\le m\le 100$

测试点	n	m	a
$1\sim 2$	≤ 1000	≤ 10	$\le 10^9$
3	$\le 10^9$	$=1$	$=1$
4	$\le 10^9$	$=2$	$=1$
5	$\le 10^9$	$=3$	$\le 10^9$
6	$\le 10^9$	$=5$	$=1$
$7\sim 8$	$\le 10^9$	≤ 20	$=1$
9	$\le 10^9$	≤ 50	$\le 10^9$
10	$\le 10^9$	≤ 100	$\le 10^9$

20分解法：暴力

测试点	n	m	a
$1\sim2$	≤ 1000	≤ 10	$\le 10^9$
3	$\le 10^9$	$=1$	$=1$
4	$\le 10^9$	$=2$	$=1$
5	$\le 10^9$	$=3$	$\le 10^9$
6	$\le 10^9$	$=5$	$=1$
$7\sim8$	$\le 10^9$	≤ 20	$=1$
9	$\le 10^9$	≤ 50	$\le 10^9$
10	$\le 10^9$	≤ 100	$\le 10^9$

- 直接按$S(n)$的公式进行求和

$$S(n)=\sum_{k=0}^{n}a^k\sum_{i=0}^{m}b_ik^i$$

20分解法：暴力

测试点	n	m	a
$1\sim2$	≤ 1000	≤ 10	$\le 10^9$
3	$\le 10^9$	$=1$	$=1$
4	$\le 10^9$	$=2$	$=1$
5	$\le 10^9$	$=3$	$\le 10^9$
6	$\le 10^9$	$=5$	$=1$
$7\sim8$	$\le 10^9$	≤ 20	$=1$
9	$\le 10^9$	≤ 50	$\le 10^9$
10	$\le 10^9$	≤ 100	$\le 10^9$

- $S(n)=\sum_{k=0}^{n} a^k \sum_{i=0}^{m} b_i k^i$
- 复杂度为$O(nm^2)$

```
S = 0
for k = 0 to n
    S1 = 0
    for i = 0 to m
        S1 += b[i] * k^i
    S += a^k * S1
```

40分解法：暴力+等幂求和公式

测试点	n	m	a
$1 \sim 2$	≤ 1000	≤ 10	$\le 10^9$
3	$\le 10^9$	$= 1$	$= 1$
4	$\le 10^9$	$= 2$	$= 1$
5	$\le 10^9$	$= 3$	$\le 10^9$
6	$\le 10^9$	$= 5$	$= 1$
$7 \sim 8$	$\le 10^9$	≤ 20	$= 1$
9	$\le 10^9$	≤ 50	$\le 10^9$
10	$\le 10^9$	≤ 100	$\le 10^9$

- 测试点3-4满足$a = 1$，$m \le 2$
- 此时

$$S(n) = \sum_{k=0}^{n} a^k \sum_{i=0}^{m} b_i k^i = \sum_{k=0}^{n} \sum_{i=0}^{m} b_i k^i$$

$$= \sum_{i=0}^{m} b_i \sum_{k=0}^{n} k^i$$

$$= b_0 \sum_{k=0}^{n} k^0 + b_1 \sum_{k=0}^{n} k^1 + b_2 \sum_{k=0}^{n} k^2$$

40分解法：暴力+等幂求和公式

- 因此可以推导出计算公式：

$$S(n) = b_0(n+1) + b_1\frac{n(n+1)}{2} + b_2\frac{n(n+1)(2n+1)}{6}$$

- 注意：取模和除法的顺序
- 乘法逆元：费马小定理/扩展欧几里得算法

```
S = 0
S += b[0] * (n + 1)
S += b[1] * n * (n + 1) / 2
if (2 <= m)
    S += b[2] * n * (n + 1) (2*n + 1) / 6
```

70分解法：暴力+等幂求和递推

测试点	n	m	a
$1 \sim 2$	≤ 1000	≤ 10	$\le 10^9$
3	$\le 10^9$	$= 1$	$= 1$
4	$\le 10^9$	$= 2$	$= 1$
5	$\le 10^9$	$= 3$	$\le 10^9$
6	$\le 10^9$	$= 5$	$= 1$
$7 \sim 8$	$\le 10^9$	≤ 20	$= 1$
9	$\le 10^9$	≤ 50	$\le 10^9$
10	$\le 10^9$	≤ 100	$\le 10^9$

- 测试点3-4，6-8满足$a = 1$，$m \le 20$
- 此时仍有

$$S(n) = \sum_{k=0}^{n} a^k \sum_{i=0}^{m} b_i k^i$$

$$= \sum_{k=0}^{n} \sum_{i=0}^{m} b_i k^i$$

$$= \sum_{i=0}^{m} b_i \sum_{k=0}^{n} k^i$$

70分解法：暴力+等幂求和递推

- 平方和公式的一种推导：

$$
\begin{array}{cccccc}
(n+1)^3 & -n^3 & = & 3n^2 & +3n & +1 \\
\big((n-1)+1\big)^3 & -(n-1)^3 & = & 3(n-1)^2 & +3(n-1) & +1 \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
(2+1)^3 & -2^3 & = & 3\cdot 2^2 & +3\cdot 2 & +1 \\
(1+1)^3 & -1^3 & = & 3\cdot 1^2 & +3\cdot 1 & +1
\end{array}
$$

- 求和可得：

$$(n+1)^3 - 1^3 = 3(1^2+2^2+\cdots+n^2) + 3(1+2+\cdots+n) + n$$

- 记$T(m) = \sum_{i=0}^{n} i^m$，则$T(2) = \frac{(n+1)^3 - 3T(1) - T(0)}{3}$

70分解法：暴力+等幂求和递推

- 将上述推导过程一般化：

$$
\begin{array}{cccccccc}
(n+1)^{m+1} & -n^{m+1} & = & \binom{m+1}{1} n^m & + \binom{m+1}{2} n^{m-1} & + \cdots & +1 \\
\left((n-1)+1\right)^{m+1} & -(n-1)^{m+1} & = & \binom{m+1}{1} (n-1)^m & + \binom{m+1}{2} (n-1)^{m-1} & + \cdots & +1 \\
\vdots & \vdots & \vdots & \vdots & \vdots & + \cdots & \vdots \\
(1+1)^{m+1} & -1^{m+1} & = & \binom{m+1}{1} 1^m & + \binom{m+1}{2} 1^{m-1} & + \cdots & +1
\end{array}
$$

- 求和可得：

$$T(m) = \sum_{i=0}^{n} i^m$$

$$(n+1)^m - 1^m = \binom{m+1}{1} T(m) + \binom{m+1}{2} T(m-1) + \cdots + (T(0) - 1)$$

70分解法：暴力+等幂求和递推

$$T(m)=\sum_{i=0}^{n} i^m$$

- 得到$T(m)$的递推公式：

$$T(m)=\frac{(n+1)^m-\binom{m+1}{2}T(m-1)-\cdots-\binom{m+1}{m+1}T(0)}{m+1}$$

- 代入$S(n)$公式中：

$$S(n)=\sum_{k=0}^{n} a^k\sum_{i=0}^{m} b_i k^i=\sum_{k=0}^{n}\sum_{i=0}^{m} b_i k^i=\sum_{i=0}^{m} b_i\sum_{k=0}^{n} k^i=\sum_{i=0}^{m} b_i T(i)$$

- 求组合数：利用组合数性质递推

$$\binom{n}{m}=\binom{n-1}{m}+\binom{n-1}{m-1}$$

70分解法：暴力+等幂求和递推

- 复杂度：$O(m^2)$

```
T[0] = n + 1
for i = 1 to m
    T[i] = (n + 1)^i
    for j = 0 to i-1
        T[i] -= c[i+1][j] * T[j]
    T[i] /= i + 1
S = 0
for i = 0 to m
    S += b[i] * T[i]
```

80分解法：矩阵快速幂

测试点	n	m	a
$1 \sim 2$	≤ 1000	≤ 10	$\le 10^9$
3	$\le 10^9$	$= 1$	$= 1$
4	$\le 10^9$	$= 2$	$= 1$
5	$\le 10^9$	$= 3$	$\le 10^9$
6	$\le 10^9$	$= 5$	$= 1$
$7 \sim 8$	$\le 10^9$	≤ 20	$= 1$
9	$\le 10^9$	≤ 50	$\le 10^9$
10	$\le 10^9$	≤ 100	$\le 10^9$

- 改写$S(n)$

$$S(n) = \sum_{k=0}^{n} a^k \sum_{i=0}^{m} b_i k^i$$

$$= \sum_{k=0}^{n} \sum_{i=0}^{m} a^k b_i k^i = \sum_{i=0}^{m} b_i \sum_{k=0}^{n} a^k k^i$$

- 记$t(i,k) = a^k k^i$
- 记$T(i) = \sum_{k=0}^{n} a^k k^i = \sum_{k=0}^{n} t(i,k)$
- $S(n) = \sum_{i=0}^{m} b_i T(i)$

- $t(i,k) = a^k k^i$
- $T(i) = \sum_{k=0}^{n} t(i,k)$
- $S(n) = \sum_{i=0}^{m} b_i T(i)$

80分解法：矩阵快速幂

- $$\begin{aligned} t(i,k+1) &= a^{k+1}(k+1)^i \\ &= a^{k+1}\left[\binom{i}{i}k^i + \binom{i}{i-1}k^{i-1} + \cdots + \binom{i}{0}1\right] \\ &= a\left[\binom{i}{i}a^k k^i + \binom{i}{i-1}a^k k^{i-1} + \cdots + \binom{i}{0}a^k \cdot 1\right] \\ &= a\left[\binom{i}{i}t(i,k) + \binom{i}{i-1}t(i-1,k) + \cdots + \binom{i}{0}t(0,k)\right] \end{aligned}$$

- $t(i,k) = a^k k^i$
- $T(i) = \sum_{k=0}^{n} t(i,k)$
- $S(n) = \sum_{i=0}^{m} b_i T(i)$

80分解法：矩阵快速幂

- $t(i,k+1) = a\left[\binom{i}{i}t(i,k) + \binom{i}{i-1}t(i-1,k) + \cdots + \binom{i}{0}t(0,k)\right]$

$$\begin{bmatrix} t(0,k+1) \\ t(1,k+1) \\ \vdots \\ t(i,k+1) \end{bmatrix} = \begin{bmatrix} \binom{0}{0}a & 0 & \cdots & 0 \\ \binom{1}{0}a & \binom{1}{1}a & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ \binom{i}{0}a & \binom{i}{1}a & \cdots & \binom{i}{i}a \end{bmatrix} \begin{bmatrix} t(0,k) \\ t(1,k) \\ \vdots \\ t(i,k) \end{bmatrix}$$

- 矩阵快速幂求$t(i,k)$

- $t(i,k) = a^k k^i$
- $T(i) = \sum_{k=0}^{n} t(i,k)$
- $S(n) = \sum_{i=0}^{m} b_i T(i)$

80分解法：矩阵快速幂

- 把矩阵改造一下

$$\begin{bmatrix} t(0,k+1) \\ t(1,k+1) \\ \vdots \\ t(i,k+1) \\ \boxed{\displaystyle\sum_{k'=0}^{k} t(i,k')} \end{bmatrix} = \begin{bmatrix} \binom{0}{0}a & 0 & \cdots & 0 & 0 \\ \binom{1}{0}a & \binom{1}{1}a & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & 0 \\ \binom{i}{0}a & \binom{i}{1}a & \cdots & \binom{i}{i}a & 0 \\ 0 & 0 & \cdots & 1 & 1 \end{bmatrix} \begin{bmatrix} t(0,k) \\ t(1,k) \\ \vdots \\ t(i,k) \\ \boxed{\displaystyle\sum_{k'=0}^{k-1} t(i,k')} \end{bmatrix}$$

$T(i) =$

- $t(i,k) = a^k k^i$
- $T(i) = \sum_{k=0}^{n} t(i,k)$
- $S(n) = \sum_{i=0}^{m} b_i T(i)$

80分解法：矩阵快速幂

- 复杂度：$O(m^4 \log n)$

```
S = 0
for i = 0 to m
    A = [c[0][0]*a,         0, ...,         0, 0;
         c[1][0]*a, c[1][1]*a, ...,         0, 0;
              ...,       ..., ...,          0, 0;
         c[i][0]*a, c[i][1]*a, ..., c[i][i]*a, 0;
                0,          0, ...,         1, 1]
    t = [1; 0; ...; 0; 0]
    A = quick_pow(A, n+1)
    t = A * t
    S += b[i] * t[i+1]
```

- $t(i,k) = a^k k^i$
- $T(i) = \sum_{k=0}^{n} t(i,k)$
- $S(n) = \sum_{i=0}^{m} b_i T(i)$

100分解法：改进的矩阵快速幂

测试点	n	m	a
$1 \sim 2$	≤ 1000	≤ 10	$\leq 10^9$
3	$\leq 10^9$	$= 1$	$= 1$
4	$\leq 10^9$	$= 2$	$= 1$
5	$\leq 10^9$	$= 3$	$\leq 10^9$
6	$\leq 10^9$	$= 5$	$= 1$
$7 \sim 8$	$\leq 10^9$	≤ 20	$= 1$
9	$\leq 10^9$	≤ 50	$\leq 10^9$
10	$\leq 10^9$	≤ 100	$\leq 10^9$

- $t(0,0), t(1,0), \cdots, t(i,0)$
- $t(0,1), t(1,1), \cdots, t(i,1)$
- ……
- $t(0,k), t(1,k), \cdots, t(i,k)$

$$S(n) = \sum_{i=0}^{m} b_i T(i) = \sum_{i=0}^{m} b_i \sum_{k=0}^{n} t(i,k)$$

$$= \sum_{i=0}^{m} \sum_{k=0}^{n} b_i t(i,k)$$

- $t(i,k) = a^k k^i$
- $S(n) = \sum_{i=0}^{m} \sum_{k=0}^{n} b_i t(i,k)$

100分解法：改进的矩阵快速幂

- 把矩阵再改造一下

$$\begin{bmatrix} t(0,k+1) \\ t(1,k+1) \\ \vdots \\ t(m,k+1) \\ \boxed{\sum_{i=0}^{m} \sum_{k'=0}^{k} b_i t(i,k')} \end{bmatrix} = \begin{bmatrix} \binom{0}{0}a & 0 & \cdots & 0 & 0 \\ \binom{1}{0}a & \binom{1}{1}a & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & 0 \\ \binom{m}{0}a & \binom{m}{1}a & \cdots & \binom{m}{m}a & 0 \\ \boxed{b_0} & \boxed{b_1} & \boxed{\cdots} & \boxed{b_m} & 1 \end{bmatrix} \begin{bmatrix} t(0,k) \\ t(1,k) \\ \vdots \\ t(m,k) \\ \boxed{\sum_{i=0}^{m} \sum_{k'=0}^{k-1} b_i t(i,k')} \end{bmatrix}$$

- $t(i,k) = a^k k^i$
- $S(n) = \sum_{i=0}^{m} \sum_{k=0}^{n} b_i t(i,k)$

100分解法：改进的矩阵快速幂

- 复杂度：$O(m^3 \log n)$

```
A = [c[0][0]*a,          0, ...,          0, 0;
     c[1][0]*a, c[1][1]*a, ...,          0, 0;
          ...,       ..., ...,          0, 0;
     c[m][0]*a, c[m][1]*a, ..., c[m][m]*a, 0;
          b[0],      b[1], ...,       b[m], 1]
t = [1; 0; ...; 0; 0]
A = quick_pow(A, n+1)
t = A * t
S = t[m+1]
```

参考文献

- 扫雷截图来自http://minesweeperonline.com/