2025~26 算法入门 C++

KGOI-S（高级组）| PCOI 预热赛

时间：2026 年 02 月 27 日 17:00 ~ 20:00

题目名称	翻转硬币	分配糖果	迷宫游戏	树的路径
题目类型	传统型	传统型	传统型	传统型
目录	coin	candy	game	tree
输入文件名	stdin	stdin	stdin	stdin
输出文件名	stdout	stdout	stdout	stdout
每个测试点时限	1.0 秒	3.0 秒	2.0 秒	3.0 秒
内存限制	512MiB	512MiB	512MiB	512MiB
测试点数目	20	20	25	25
测试点是否等分	是	是	是	是

提交源程序文件名

对于 C++ 语言	coin.cpp	candy.cpp	game.cpp	tree.cpp

注意事项：

1. 本比赛使用 OI 赛制，选手提交后不会提供实时反馈，且对于每题，只会对该题的最后一次提交评分。
2. 时限为 180 分钟。每题满分为 100 分，总分 400 分。
3. C++ 中函数 main() 的返回值类型必须是 int，程序正常结束时的返回值必须是 0。
4. 若无特殊说明，结果的比较方式为全文比较 (即过滤文末空格及文末行)。
5. 选手需要把源程序提交到 Google Classroom 的指定教室中。
6. 采用洛谷的配置作为评测配置。上述时限以此配置为准。
7. **注意：由于洛谷环境下的换行符为 \r\n，如需读取换行符应连续读取两个字符，例如执行两次 getchar()。**
8. 结束后，在正常情况下系统会显示 0 分，这不是选手的成绩。
9. 选手的成绩、题面、样例数据及真实数据将于比赛结束一周内公布于 Google Classroom，届时在 zero1 平台会有通知。同意公开自己成绩的选手的成绩会发布于 zero1。

翻转硬币（coin）

【题目描述】

小 L 有 n 枚相同的硬币排成一列。

对于第 i 枚硬币，它的状态可以用一个整数 $a[i]$ 表示：

- 若正面朝上，则 $a[i]=1$。
- 若背面朝上，则 $a[i]=0$。

若把一枚硬币翻转一次，如果它原来是正面朝上的，则它会变成背面朝上；如果它原来是背面朝上的，则它会变成正面朝上。

接下来，小 L 会对硬币进行 m 次操作，对于每次操作：

- 若当前正面朝上的硬币个数为 p，则小 L 会把第 p 枚硬币翻转一次。
- 特别地，若 $p=0$，则他不会翻转任何硬币。

操作完成后，小 L 想知道有多少枚硬币是正面朝上的。于是他找到了你，并邀请你帮助他解决这个问题。

【输入格式】

從标准输入 (stdin) 读入数据。

输入的第一行包含两个正整数 n,m，以空格分隔。

第二行包含 n 个非负整数 $a[1],a[2],a[3],\ldots,a[n]$，以空格分隔。

【输出格式】

输出到标准输出 (stdout) 中。

输出一行，一个整数表示答案。

【样例 1 输入】

```
4 2
1 0 0 1
```

【样例 1 输出】

```
4
```

【样例 1 解释】

操作前，四枚硬币的状态分别为 $1,0,0,1$。

对于第一次操作，正面朝上的硬币个数为 2，于是小 L 把第 2 枚硬币翻转。

此时，四枚硬币的状态分别为 $1,1,0,1$。

对于第二次操作，正面朝上的硬币个数为 3，于是小 L 把第 3 枚硬币翻转。

最后，四枚硬币的状态分别为 $1,1,1,1$。因此，应输出 4。

【样例 2】

见 coin/coin2.in 与 coin/coin2.ans。

这个样例满足测试点 1 ~ 3 的约束条件。

【样例 3】

见 coin/coin3.in 与 coin/coin3.ans。

这个样例满足测试点 6 ~ 10 的约束条件。

【样例 4】

见 coin/coin4.in 与 coin/coin4.ans。

这个样例满足测试点 16 ~ 20 的约束条件。

【数据范围】

对于所有测试数据，保证：

- $n \leq 10^6$
- $m \leq 10^6$
- 对于所有满足 $1 \leq i \leq n$ 的整数 i，$a[i] \leq 1$。

测试点	$n \leq$	$m \leq$	特殊性质
1 ~ 3	1000	10	无
4 ~ 5		1000	B
6 ~ 10			无
11 ~ 12	10^6	10^6	A
13 ~ 15			B
16 ~ 20			无

特殊性质 A：对于所有满足 $1 \leq i \leq n$ 的整数 i，$a[i] = 0$。

特殊性质 B：对于所有满足 $1 \leq i \leq n$ 的整数 i，$a[i] = 1$。

分配糖果（candy）

【题目描述】

小 K 有 n 颗糖果，他想把这些糖果分配给若干位学生。

为了学生的健康，每位学生不得吃超过 5 颗糖果。于是，他把这 n 颗糖果放成一堆，并进行以下操作：

- 对每堆糖果都进行判断和操作。
- 假设其中一堆糖果内有 k 颗糖果，小 K 会对这堆糖果进行以下判断和操作：若 $k > 5$，小 K 会把这堆糖果分成左右两堆，把其中 $\lfloor k/3 \rfloor$ 颗糖果放到左边的新堆内，而原来的堆内其余的糖果会被放到右边的新堆内；否则，他不会对这堆糖果进行操作。

小 K 会不断重复操作直到所有的糖果堆内的糖果数量都不大于 5。

操作完成后，记刚好有 i 颗糖果的糖果堆的数量为 $a[i]$。小 K 想考考你的计算能力，于是他让你计算 $a[1] \oplus a[2] \oplus a[3] \oplus a[4] \oplus a[5]$ 的值。其中 $\oplus$ 表示二进制按位异或运算。

【输入格式】

從标准输入 (stdin) 读入数据。

本题有多组测试数据。

输入的第一行包含一个正整数 T，表示测试数据组数。

对于每组测试数据：

- 仅含一行，包含一个正整数 n。

【输出格式】

输出到标准输出 (stdout) 中。

对于每组测试数据：

- 输出一行，一个整数表示 $a[1] \oplus a[2] \oplus a[3] \oplus a[4] \oplus a[5]$ 的值。

【样例 1 输入】

```
1
10
```

【样例 1 输出】

```
1
```

【样例 1 解释】

操作前，只有一堆糖果，其糖果数量为 10。

小 K 会把这堆糖果分成左右两堆。他会把 $\lfloor 10/3 \rfloor = 3$ 颗糖果放到左边的新堆内，原来的堆内其余的 $10 - 3 = 7$ 颗糖果会被放到右边的新堆内。

现在，有两堆糖果，其糖果数量分别为 $3, 7$。

由于第一堆糖果的糖果数量不大于 5，因此不进行操作。

小 K 会把第二堆糖果分成左右两堆。他会把 $\lfloor 7/3 \rfloor = 2$ 颗糖果放到左边的新堆内，原来的堆内其余的 $7 - 2 = 5$ 颗糖果会被放到右边的新堆内。

现在，有三堆糖果，其糖果数量分别为 $3, 2, 5$。由于没有一堆糖果的糖果数量大于 5，操作结束。

因此，$a = [0, 1, 1, 0, 1]$，则 $a[1] \oplus a[2] \oplus a[3] \oplus a[4] \oplus a[5] = 1$。

【样例 2】

见 `candy/candy2.in` 与 `candy/candy2.ans`。

这个样例满足测试点 `5 ~ 8` 的约束条件。

【样例 3】

见 `candy/candy3.in` 与 `candy/candy3.ans`。

这个样例满足测试点 `9 ~ 14` 的约束条件。

【样例 4】

见 `candy/candy4.in` 与 `candy/candy4.ans`。

这个样例满足测试点 `15 ~ 16` 的约束条件。

【样例 5】

见 `candy/candy5.in` 与 `candy/candy5.ans`。

这个样例满足测试点 `19 ~ 20` 的约束条件。

【数据范围】

对于所有测试数据，保证：

- $T \leq 5 \times 10^4$
- $n \leq 10^9$

测试点	$T \leq$	$n \leq$	特殊性质
1 ~ 2	10	20	无
3 ~ 4		10^6	
5 ~ 8		10^9	
9 ~ 14	10^4	10^6	
15 ~ 16		10^8	
17 ~ 18		10^9	
19 ~ 20	5×10^4		

迷宫游戏（game）

【题目描述】

小 K 在游玩一款关于迷宫的游戏。

这个迷宫可被看成一个 n 行 m 列的方格图。其中，$a[i][j]=0$ 表示第 i 行第 j 列的方格是通道的一部分，可以行走；$a[i][j]=1$ 则表示该方格是墙壁，不能行走。

游戏会进行若干个回合。对于每一个回合，小 K 会被随机传送到一个位置，同时游戏会生成一颗宝石，同样地随机放到一个位置。这两个位置都不会是墙壁。

小 K 可以向上、下、左、右其中一个方向走，他有无限的时间行走。

对于每个回合，小 K 希望知道，他是否可能到达宝石的位置并获得它。于是，他会询问你 q 次，对于第 i 次询问：

- 他被传送到第 $x_1[i]$ 行第 $y_1[i]$ 列的位置中。
- 宝石被放到第 $x_2[i]$ 行第 $y_2[i]$ 列的位置中。
- 如果经过一定的时间后，他能到达宝石的位置，你需要回答「可以到达」。
- 如果他永远拿不到宝石，你需要回答「无法到达」。

你希望知道你需要回答多少次「可以到达」。

【输入格式】

从标准输入 (stdin) 读入数据。

输入的第一行包含一个整数 c，表示测试点编号。$c=0$ 表示该测试点为样例。

第二行包含三个正整数 n,m,q，以空格分隔。

接下来 n 行，第 i 行包含 m 个非负整数 $a[i][1],a[i][2],a[i][3],\ldots,a[i][m]$，没有任何符号分隔。

接下来 q 行，第 i 行表示第 i 次询问：

- 包含四个正整数 $x_1[i],y_1[i],x_2[i],y_2[i]$，以空格分隔。

【输出格式】

输出到标准输出 (stdout) 中。

输出一行，一个整数表示你需要回答「可以到达」的次数。

【样例 1 输入】

```
0
5 6 2
011001
```

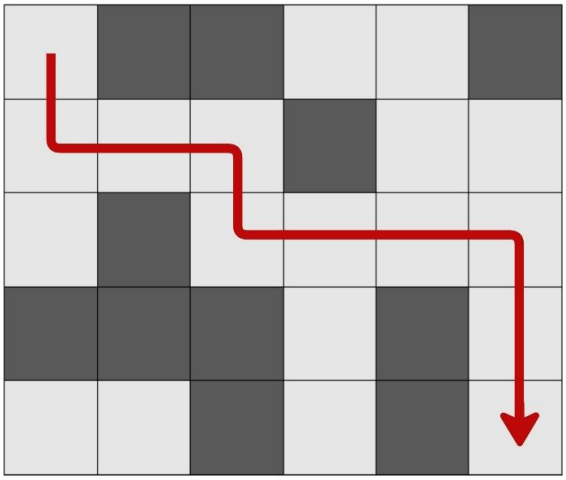
```
000100
010000
111010
001010
1 1 5 6
3 3 5 2
```

【样例 1 输出】

```
1
```

【样例 1 解释】

对于第一次询问，由小 K 的位置走到宝石的位置的路线如下图所示，其中浅灰色的格子是通道，而深灰色的格子是墙壁。因此，你需要回答「可以到达」。



对于第二次询问，可以证明小 K 无论如何行走也无法抵达宝石的位置。因此，你需要回答「无法到达」。

综上，需要回答「可以到达」的次数为 1。

【样例 2】

见 game/game2.in 与 game/game2.ans。

这个样例满足测试点 1 ~ 2 的约束条件。

【样例 3】

见 game/game3.in 与 game/game3.ans。

这个样例满足测试点 3 ~ 5 的约束条件。

【样例 4】

见 game/game4.in 与 game/game4.ans。

这个样例满足测试点 6 ~ 10 的约束条件。

【样例 5】

见 game/game5.in 与 game/game5.ans。

这个样例满足测试点 14 ~ 16 的约束条件。

【样例 6】

见 game/game6.in 与 game/game6.ans。

这个样例满足测试点 17 ~ 20 的约束条件。

【数据范围】

对于所有测试数据，保证：

- $n, m \leq 2000$
- $q \leq 10^6$
- 对于所有满足 $1 \leq i \leq n, 1 \leq j \leq m$ 的整数 i, j，$a[i][j] \leq 1$。
- 对于所有满足 $1 \leq i \leq q$ 的整数 i，$x_1[i], x_2[i] \leq n,\ y_1[i], y_2[i] \leq m$。
- 对于所有满足 $1 \leq i \leq q$ 的整数 i，$a[x_1[i]][y_1[i]] = a[x_2[i]][y_2[i]] = 0$。

测试点	$n, m \leq$	$q \leq$	特殊性质
1 ~ 2	100	100	A
3 ~ 5			B
6 ~ 10			无
11 ~ 13	1000	10^6	A
14 ~ 16			B
17 ~ 20			无
21 ~ 25	2000		

特殊性质 A：$\min\{n, m\} = 1$。

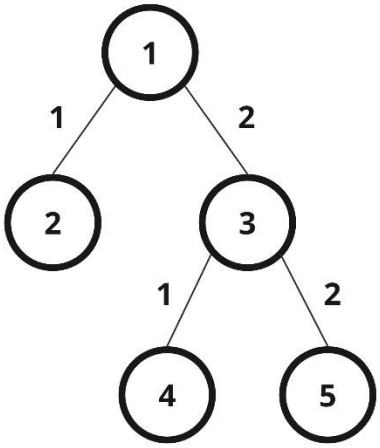
特殊性质 B：$\min\{n, m\} = 2$。

树的路径（tree）

【题目描述】

小 P 有一棵包含 n 个节点的有根树，这些节点的编号分别为 $1, 2, 3, \ldots, n$。其中，节点 1 为根节点，且对于所有满足 $2 \le i \le n$ 的整数 i，节点 i 的父亲节点为节点 $p[i]$。特别地，$p[1] = 1$。

若节点 i 有 $s[i]$ 个儿子节点，则按儿子节点的编号顺序，节点 i 与儿子节点之间的边的权值依次为 $1, 2, 3, \ldots, s[i]$。以下是对于 $p = [1, 1, 1, 3, 3]$ 的一个例子：



小 P 定义一棵树对于一个节点 i 的路径可以用一个由根节点到节点 i 的简单路径所经过的边的权值所组成的序列。例如，在上图中节点 4 的路径就是序列 $[2, 1]$。

小 P 找到了他的朋友小 Q，并打算用这棵树和他玩一个游戏。

游戏会进行 m 轮，对于第 i 轮：

- 小 P 会在他的树中找出根节点为 $r[i]$ 的一棵子树进行游戏。对于那棵子树，他会挑选一个节点 $a[i]$，并告诉小 Q 它的路径。记它的路径的长度是 $l[i]$。在这里，路径是对于该子树而言，与原来的树无关。
- 小 Q 会根据这条路径从节点 $r[i]$ 开始搜索。经过 j 条边抵达一个节点时，他都会根据路径的第 $j + 1$ 个元素选择边权与它相等的一条边，并走到那条边所对应的另一个节点。但小 Q 可能会选错边，在一轮游戏中，他最多可能选错 k 条边。
- 如果小 Q 无法继续搜索（即到达了子树的叶子节点、走过的边数超过 $l[i]$ 或没有边权与对应的路径元素相等的一条边），他会回答当前的节点编号。
- 现在，小 P 想知道，对于这轮游戏，小 Q 会有多少种不同的回答。记这个数量为 $b[i]$，则你需要回答 $b[i]$ 的值。

【输入格式】

从标准输入（stdin）读入数据。

本题有多组测试数据。

输入的第一行包含一个整数 c，表示测试点编号。$c = 0$ 表示该测试点为样例。

第二行包含一个正整数 T，表示测试数据组数。

对于每组测试数据：

- 第一行包含三个正整数 n, m, k，以空格分隔。
- 第二行包含 n 个正整数 $p[1], p[2], p[3], \ldots, p[n]$，以空格分隔。
- 第三行包含 m 个正整数 $r[1], r[2], r[3], \ldots, r[m]$，以空格分隔。
- 第四行包含 m 个正整数 $a[1], a[2], a[3], \ldots, a[m]$，以空格分隔。

【输出格式】

输出到标准输出（stdout）中。

对于每组测试数据：

- 输出一行，包含 m 个整数 $b[1], b[2], b[3], \ldots, b[m]$，以空格分隔。

【样例 1 输入】

```
0
1
5 4 1
1 1 1 3 3
1 1 3 3
2 4 4 5
```

【样例 1 输出】

```
2 3 2 2
```

【样例 1 解释】

在第一轮游戏中，小 Q 可能会回答的节点有 2 个，编号分别为 $2, 3$。

在第二轮游戏中，小 Q 可能会回答的节点有 3 个，编号分别为 $2, 4, 5$。

在第三轮游戏中，小 Q 可能会回答的节点有 2 个，编号分别为 $4, 5$。

在第四轮游戏中，小 Q 可能会回答的节点有 2 个，编号分别为 $4, 5$。

【样例 2】

见 tree/tree2.in 与 tree/tree2.ans。

这个样例满足测试点 2 ~ 3 的约束条件。

【样例 3】

见 tree/tree3.in 与 tree/tree3.ans。

这个样例满足测试点 5 ~ 6 的约束条件。

【样例 4】

见 tree/tree4.in 与 tree/tree4.ans。

这个样例满足测试点 8 ~ 10 的约束条件。

【样例 5】

见 tree/tree5.in 与 tree/tree5.ans。

这个样例满足测试点 16 ~ 17 的约束条件。

【样例 6】

见 tree/tree6.in 与 tree/tree6.ans。

这个样例满足测试点 18 ~ 20 的约束条件。

【样例 7】

见 tree/tree7.in 与 tree/tree7.ans。

这个样例满足测试点 21 ~ 22 的约束条件。

【样例 8】

见 tree/tree8.in 与 tree/tree8.ans。

这个样例满足测试点 23 ~ 25 的约束条件。

【数据范围】

定义节点 i 的深度 $d[i]$ 为根节点到节点 i 的简单路径的边数。即 $d[1]=0$，且对于所有满足 $2\le i\le n$ 的整数 i，$d[i]=d\left[p[i]\right]+1$。定义有根树的高度 h 为所有节点的深度的最大值，即 $h=\max_{i=1}^{n} d[i]$。

对于所有测试数据，保证：

- $T\le 5$
- $n\le 3000$

- $m \leq 8 \times 10^4$
- $h \leq n-1$
- $k \leq 10^9$
- $p[1]=1$
- 对于所有满足 $2 \leq i \leq n$ 的整数 i，$p[i] \leq n$。
- 序列 p 所表示的是一棵树。
- 对于所有满足 $1 \leq i \leq m$ 的整数 i，$r[i] \leq n, a[i] \leq n$，且 $a[i]$ 是根节点为 $r[i]$ 的子树中的一个节点。

<table>
<tr><th>测试点</th><th>$n \leq$</th><th>$m \leq$</th><th>k</th><th>$h \leq$</th><th>$p[i] \leq$</th><th>特殊性质</th></tr>
<tr><td>1</td><td>3</td><td>10</td><td rowspan="14">$\leq 10^9$</td><td rowspan="8">$n-1$</td><td rowspan="3">n</td><td rowspan="4">无</td></tr>
<tr><td>2 ~ 3</td><td>150</td><td>400</td></tr>
<tr><td>4</td><td rowspan="2">500</td><td rowspan="2">10^4</td></tr>
<tr><td>5 ~ 6</td><td rowspan="2">$i-1$</td></tr>
<tr><td>7</td><td rowspan="12">3000</td><td rowspan="12">8×10^4</td><td>A</td></tr>
<tr><td>8 ~ 10</td><td rowspan="10">n</td><td>B</td></tr>
<tr><td>11</td><td>1</td><td>无</td></tr>
<tr><td>12</td><td rowspan="2">2</td><td>B</td></tr>
<tr><td>13 ~ 14</td><td rowspan="9">无</td></tr>
<tr><td>15</td><td>$=10^9$</td><td rowspan="5">$n-1$</td></tr>
<tr><td>16 ~ 17</td><td>$=1$</td></tr>
<tr><td>18 ~ 20</td><td>≤ 5</td></tr>
<tr><td>21 ~ 22</td><td rowspan="2">$\leq 10^9$</td><td>$i-1$</td></tr>
<tr><td>23 ~ 25</td><td>n</td></tr>
</table>

特殊性质 A：对于所有满足 $2 \leq i \leq n$ 的整数 i，$p[i]=i-1$。

特殊性质 B：$r[1]=r[2]=r[3]=\cdots=r[m]$。