目录

NOIP2023 1
B.三值逻辑 1
C.双序列拓展 5
D.天天爱打卡 7

NOIP2023

B.三值逻辑

题目描述

小 L 今天学习了 Kleene 三值逻辑。

在三值逻辑中，一个变量的值可能为：真（True，简写作 T）、假（False，简写作 F）或未确定（Unknown，简写作 U）。

在三值逻辑上也可以定义逻辑运算。由于小 L 学习进度很慢，只掌握了逻辑非运算 $\neg$，其运算法则为：$\neg T = F$，$\neg F = T$，$\neg U = U$。

现在小 L 有 n 个三值逻辑变量 $x_1, ..., x_n$。小 L 想进行一些有趣的尝试，于是他写下了 m 条语句。语句有以下三种类型，其中 $\leftarrow$ 表示赋值：

1. $x_i \leftarrow v$，其中 v 为 T、F、U 的一种；
2. $x_i \leftarrow x_j$；
3. $x_i \leftarrow \neg x_j$。

一开始，小 L 会给这些变量赋初值，然后按顺序运行这 m 条语句。

小 L 希望执行了所有语句后，所有变量的最终值与初值都相等。在此前提下，小 L 希望初值中 Unknown 的变量尽可能少。

在本题中，你需要帮助小 L 找到 Unknown 变量个数最少的赋初值方案，使得执行了所有语句后所有变量的最终值和初始值相等。小 L 保证，至少对于本题的所有测试用例，这样的赋初值方案都必然是存在的。

输入格式

本题的测试点包含有多组测试数据。

输入的第一行包含两个整数 c 和 t，分别表示测试点编号和测试数据组数。对于样例，c 表示该样例与测试点 c 拥有相同的限制条件。

接下来，对于每组测试数据：

- 输入的第一行包含两个整数 n 和 m，分别表示变量个数和语句条数。
- 接下来 m 行，按运行顺序给出每条语句。
  - 输入的第一个字符 v 描述这条语句的类型。保证 v 为 `TFU+-` 的其中一种。
  - 若 v 为 `TFU` 的某一种时，接下来给出一个整数 i，表示该语句为 $x_i \leftarrow v$；
  - 若 v 为 `+`，接下来给出两个整数 i, j，表示该语句为 $x_i \leftarrow x_j$；
  - 若 v 为 `-`，接下来给出两个整数 i, j，表示该语句为 $x_i \leftarrow \neg x_j$。

输出格式

对于每组测试数据输出一行一个整数，表示所有符合条件的赋初值方案中，Unknown 变量个数的最小值。

样例

1.in / 1.out

```
1  1 3
2  3 3
3  - 2 1
4  - 3 2
5  + 1 3
6  3 3
7  - 2 1
8  - 3 2
9  - 1 3
10 2 2
11 T 2
12 U 2
```

```
1 0
2 3
3 1
```

第一组测试数据中，m 行语句依次为 $x_2 \leftarrow \neg x_1$、$x_3 \leftarrow \neg x_2$、$x_1 \leftarrow x_3$。一组合法的赋初值方案为 $x_1 = T$，$x_2 = F$，$x_3 = T$，共有 0 个 Unknown 变量，故输出 0。

第二组测试数据中，m 行语句依次为 $x_2 \leftarrow \neg x_1$、$x_3 \leftarrow \neg x_2$、$x_1 \leftarrow \neg x_3$。唯一的赋初值方案为 $x_1 = x_2 = x_3 = U$，共有 3 个 Unknown 变量，故输出 3。

第三组测试数据中，m 行语句依次为 $x_2 \leftarrow T$、$x_2 \leftarrow U$。一个最小化 Unknown 变量个数的赋初值方案为 $x_1 = T$，$x_2 = U$，共有 1 个 Unknown 变量，故输出 1。

数据范围

对于所有测试数据，保证：

- $1 \leq t \leq 6$，$1 \leq n, m \leq 10^5$；
- 对于每个操作，v 为 `TFU+-` 中的某个字符，$1 \leq i, j \leq n$。

题解

这一道题对我而言意义相当特别。因为我考场上想了很久也没想出来，最后似乎只拿了一点点过程分。

虽然说过了三年，但我还记得很清楚原先想的是拓扑排序，然后用并查集来维护值，被循环赋值卡住了。

后来考完试，看到题解的“并查集按时间开点”，就这一句话让我瞬间想明白了，然后觉得自己很蠢，然后光速退役。

考试是 2023 年 11 月 19 日，今天是 2026 年 11 月 4 日。三年了，也该清算了。

为什么按时间开点？因为事实上每一个时间点的变量都可以认为是独立的。

这些中间值可能也会影响其他值，但即使它们是 U（或被推断为 U），也不一定末尾的变量就是 U（可能被再次赋值）。

所以我们每次变量赋值的时候，被赋值的变量就要新开点。

然后，按照并查集分类（我忘了是不是叫这个名字）的思路，每个点建立虚点，若 u，v 相同，u 连 v，u’ 连 v’；若 u，v 不同，u 连 v’，u’ 连 v。

当然这里 T，F，U 都要单独开点。

然后在检查的时候，如果某个变量的最后一个版本和第一个版本相冲突，或最后一个版本被 U 约束，它的初始值必须是 U。

我在调试代码的时候遇到这样一个问题：

```cpp
if (op == '-') {
    scanf("%d%d", &u, &v);

    int p = gp(v), pi = gpinv(v);
    merge(np(u), pi);
    merge(npinv(u), p);
}
```

我原先直接把函数写进去，然后发现似乎下面的 `npinv` 得到的值被影响了，得到了新点的虚点。

这个问题相当隐蔽，需要注意。

```cpp
#include <stdio.h>
#include <vector>

#define maxn 2000007

std::vector<int> vers[maxn], invs[maxn];

int fas[maxn], n, m;
int cnt;
int vU, vT, vF;
void init() {
    for (int i = 1; i < maxn; i++)
        vers[i].clear(), invs[i].clear(), fas[i] = i;
    cnt = 0;
    vU = ++cnt;
    vT = ++cnt;
    vF = ++cnt;
}

int ffa(int x) {
    if (fas[x] == x)
        return x;
    return fas[x] = ffa(fas[x]);
}

inline void merge(int u, int v) { fas[ffa(u)] = fas[ffa(v)]; }
inline int check(int u, int v) { return ffa(u) == ffa(v); }

char gc() {
    char c = ' ';
    do {
        c = getchar();
```

```
33      } while (c != '+' and c != '-' and c != 'T' and c != 'F' and c != 'U');
34      return c;
35  }
36
37  // last version of x
38  inline int gp(int x) { return *(--vers[x].end()); }
39  inline int gpinv(int x) { return *(--invs[x].end()); }
40  // insert new version of x
41  inline int np(int x) {
42      vers[x].push_back(++cnt);
43      return gp(x);
44  }
45  inline int npinv(int x) {
46      invs[x].push_back(++cnt);
47      return gpinv(x);
48  }
49  // first version of x
50  inline int gf(int x) { return (vers[x][0]); }
51  inline int gfinv(int x) { return (invs[x][0]); }
52
53  int _, T, u, v;
54  char op;
55
56  int main() {
57      scanf("%d%d", &_, &T);
58      while (T--) {
59          scanf("%d%d", &n, &m);
60          init();
61          for (int i = 1; i <= n; i++)
62              np(i), npinv(i);
63          // puts("");
64
65          for (int i = 1; i <= m; i++) {
66              op = gc();
67              if (op == 'T') {
68                  scanf("%d", &u);
69
70                  merge(np(u), vT);
71                  merge(npinv(u), vF);
72              }
73              if (op == 'F') {
74                  scanf("%d", &u);
75
76                  merge(np(u), vF);
77                  merge(npinv(u), vT);
78              }
79              if (op == 'U') {
80                  scanf("%d", &u);
```

```
81
82                  merge(np(u), vU);
83                  merge(npinv(u), vU);
84              }
85
86              if (op == '+') {
87                  scanf("%d%d", &u, &v);
88
89                  int p = gp(v), pi = gpinv(v);
90                  merge(np(u), p);
91                  merge(npinv(u), pi);
92              }
93              if (op == '-') {
94                  scanf("%d%d", &u, &v);
95
96                  int p = gp(v), pi = gpinv(v);
97                  merge(np(u), pi);
98                  merge(npinv(u), p);
99              }
100         }
101
102         int ans = 0;
103
104         for (int i = 1; i <= n; i++)
105             merge(gp(i), gf(i)), merge(gpinv(i), gfinv(i));
106
107         for (int i = 1; i <= n; i++)
108             if (check(gp(i), gfinv(i)))
109                 merge(gp(i), vU), merge(gpinv(i), vU);
110
111         for (int i = 1; i <= n; i++) {
112             if (check(gp(i), vU))
113                 ans++;
114         }
115
116         printf("%d\n", ans);
117     }
118     return 0;
119 }
```

总算是解开了从前的心结，希望你不要走从前的老路。

C.双序列拓展

题目描述

称某个序列 $B=\{b_1,b_2,...,b_n\}$ 是另一个序列 $A=\{a_1,a_2,...,a_m\}$ 的拓展，当且仅当存在正整数序列 $L=\{l_1,l_2,...,l_m\}$，将 a_i 替换为 l_i 个 a_i 后得到序列 B。例如：

- $\{1,3,3,3,2,2,2\}$ 是 $\{1,3,3,2\}$ 的拓展，取 $L=\{1,1,2,3\}$ 或 $\{1,2,1,3\}$；
- 而 $\{1,3,3,2\}$ 不是 $\{1,3,3,3,2\}$ 的拓展，$\{1,2,3\}$ 不是 $\{1,3,2\}$ 的拓展。

小 R 给了你两个序列 X 和 Y，他希望你找到 X 的一个长度为 $l_0 = 10^{100}$ 的拓展 $F = \{f_i\}$ 以及 Y 的一个长度为 l_0 的拓展 $G = \{g_i\}$，使得任意 $1 \le i, j \le l_0$ 都有 $(f_i - g_i)\big(f_j - g_j\big) > 0$。由于序列太长，你只需要告诉小 R 是否存在这样的两个序列即可。

为了避免你扔硬币蒙混过关，小 R 还给了 q 次额外询问，每次额外询问中小 R 会修改 X 和 Y 中若干元素的值。你需要对每次得到的新的 X 和 Y 都进行上述的判断。

询问之间是独立的，每次询问中涉及的修改均在原始序列上完成。

输入格式

输入的第一行包含四个整数 c, n, m, q，分别表示测试点编号、序列 X 的长度、序列 Y 的长度和额外询问的个数。对于样例，c 表示该样例与测试点 c 拥有相同的限制条件。

输入的第二行包含 n 个整数 $x_1, x_2, ..., x_n$，描述序列 X。

输入的第三行包含 m 个整数 $y_1, y_2, ..., y_m$，描述序列 Y。

接下来依次描述 q 组额外询问。对于每组额外询问：

- 输入的第一行包含两个整数 k_x 和 k_y，分别表示对序列 X 和 Y 产生的修改个数。
- 接下来 k_x 行每行包含两个整数 p_x, v_x，表示将 x_{p_x} 修改为 v_x。
- 接下来 k_y 行每行包含两个整数 p_y, v_y，表示将 y_{p_y} 修改为 v_y。

输出格式

输出一行，其中包含一个长度为 $(q+1)$ 的 `01` 序列，序列的第一个元素表示初始询问的答案，之后 q 个元素依次表示每组额外询问的答案。对于每个询问，如果存在满足题目条件的序列 F 和 G，输出 `1`，否则输出 `0`。

样例

1.in / 1.out

```
3 3 3 3
8 6 9
1 7 4
1 0
3 0
0 2
1 8
3 5
1 1
2 8
1 7
```

```
1001
```

由于 F 和 G 太长，用省略号表示重复最后一个元素直到序列长度为 l_0。如 $\{1, 2, 3, 3, ...\}$ 表示序列从第三个元素之后都是 3。

以下依次描述四次询问，其中第一次询问为初始询问，之后的三次为额外询问：

1. $A = \{8, 6, 9\}$，$B = \{1, 7, 4\}$，取 $F = \{8, 8, 6, 9, ...\}$，$G = \{1, 7, 4, 4, ...\}$；
2. $A = \{8, 6, 0\}$，$B = \{1, 7, 4\}$，可以证明不存在满足要求的方案；
3. $A = \{8, 6, 9\}$，$B = \{8, 7, 5\}$，可以证明不存在满足要求的方案；
4. $A = \{8, 8, 9\}$，$B = \{7, 7, 4\}$，取 $F = \{8, 8, 9, ...\}$，$G = \{7, 7, 4, ...\}$。

数据范围

对于所有测试数据，保证：

- $1 \le n, m \le 5 \times 10^5$；
- $0 \le q \le 60$；
- $0 \le x_i, y_i < 10^9$；
- $0 \le k_x, k_y \le 5 \times 10^5$，且所有额外询问的 $\left(k_x + k_y\right)$ 的和不超过 5×10^5；
- $1 \le p_x \le n$，$1 \le p_y \le m$，$0 \le v_x, v_y < 10^9$；
- 对于每组额外询问，p_x 两两不同，p_y 两两不同。

题解

TODO

D.天天爱打卡

题目描述

小 T 同学非常热衷于跑步。为了让跑步更加有趣，他决定制作一款叫做《天天爱打卡》的软件，使得用户每天都可以进行跑步打卡。

开发完成后，小 T 同学计划进行试运行，他找了大 Y 同学来帮忙。试运行共 n 天，编号为从 1 到 n。

对大 Y 同学来说，如果某天他选择跑步打卡，那么他的能量值会减少 d。初始时，他的能量值是 0，并且试运行期间他的能量值可以是负数。

而且大 Y 不会连续跑步打卡超过 k 天；即不能存在 $1 \le x \le n - k$，使得他在第 x 到第 $x + k$ 天均进行了跑步打卡。

小 T 同学在软件中设计了 m 个挑战，第 i（$1 \le i \le m$）个挑战可以用三个正整数 (x_i, y_i, v_i) 描述，表示如果在第 x_i 天时，用户已经连续跑步打卡至少 y_i 天（即第 $x_i - y_i + 1$ 到第 x_i 天均完成了跑步打卡），那么小 T 同学就会请用户吃饭，从而使用户的能量值提高 v_i。

现在大 Y 想知道，在软件试运行的 n 天结束后，他的能量值最高可以达到多少？

输入格式

本题的测试点包含有多组测试数据。

输入的第一行包含两个整数 c 和 t，分别表示测试点编号和测试数据组数。对于样例，c 表示该样例与测试点 c 拥有相同的限制条件。

接下来，对于每组测试数据：

- 输入的第一行包含四个正整数 n, m, k, d，分别表示试运行的天数、挑战的个数、大 Y 单次跑步打卡的连续天数限制以及大 Y 跑步打卡减少的能量值。
- 接下来 m 行，每行包含三个正整数 x_i, y_i, v_i，表示一次挑战。

输出格式

输出一行一个整数表示对应的答案。

样例

1.in / 1.out

```
1 1 1
2 3 2 2 1
3 2 2 4
```

```
4 3 2 3
```

```
1 2
```

在第 $1,2$ 天跑步打卡，第 3 天不跑步打卡，最终会获得 $(-1)+(-1)+4=2$ 的能量值。

数据范围

记 $l_i=x_i-y_i+1$，$r_i=x_i$。

对于所有测试数据，保证：$1\le t\le 10$，$1\le k\le n\le 10^9$，$1\le m\le 10^5$，$1\le l_i\le r_i\le n$，$1\le d,v_i\le 10^9$。

题解

TODO
