

总共有  $n$  个人报名，每个人有一个昵称（ $1 \leq \text{昵称} \leq 10$ ），他们组成了多个团队。需要知道每个团队的  $m$  条消息均被读到的两个人是在同一个团队中，再求：

- 消息构成  $a, b, c$ ，整数  $a, b$  分别代表两个人的昵称，整数  $c$  代表编号
- $c = 0$  代表  $a$  和  $b$  在一个团队内
- $c = 1$  代表读到了  $a$  和  $b$  的某条消息，如果  $a$  和  $b$  是第一个团队，则输一行 `we are a team!`，如果读不到，则输一行 `we are not a team!`
- $c$  为其他值，输出顺序  $a$  和  $b$  超出  $1 \sim 10$  的范围，输出 `da pian zi!`

输入描述

- 第一行包含两个整数  $n, m$  ( $n \leq 100, m \leq 10000$ )，分别表示有  $n$  个人和  $m$  条消息
- 随后行  $m$  行，每行一条消息，消息格式为： $a\ b\ c$  ( $1 \leq a, b \leq 10, 0 \leq c \leq 1$ )

输出描述

- $c = 1$  时，根据  $a$  和  $b$  是否在一个团队中输出一行字符串，在一个团队中输过 `we are a team!`，不在一个团队中输过 `we are not a team!`
- $c$  为其他值，按照顺序行  $a$  和  $b$  的数值小于 1 则按大于  $a$  时，输出字符串 `da pian zi!`
- 如果输一行  $a$  和  $b$  的数值超出范围时，输出字符串 `da pian zi!`

用例

	5 7
	1 2 0
	2 3 0
输入	2 3 1
	2 3 1
	2 3 1
	1 3 1
输出	we are a team!
	we are a team!
	we are a team!
	we are not a team!
说明	无

	4 4
	1 2 0
	1 3 1
输入	1 3 0
	2 3 1
	2 3 1
	1 3 1
输出	we are a team!
	we are not a team!
	we are a team!
	da pian zi!
说明	无

题目解析

本题其实是一道经典 无向图 DFS，并需要无向图确定两个顶点的连接关系，因此可以使用并查集实现。

JavaScript算法源码

```
1 // 并查集实现
2 const readline = require("readline");
3
4 const rl = readline.createInterface({
5   input: process.stdin,
6   output: process.stdout,
7 });
8
9 const lines = [];
10 let n, m;
11 rl.on("line", (line) => {
12   line.split(" ");
13
14   if (line.length === 1) {
15     [n, m] = line[0].split(" ").map(Number);
16   }
17
18   if (n && line.length === n + 1) {
19     lines.push(line);
20     const msg = line.map((line) => line.split(" ").map(Number));
21     print(msg, n, m);
22     lines.length = 0;
23   }
24 });
25
26 function print(msg, n, m) {
27   if (n < 1 || n > 10000 || m < 1 || m > 10000) return console.log("No!!");
28
29   const ufs = new UnionFindSet(n);
30   msg.forEach(([, , c], [a, b, c]) => {
31     if (c === 0) {
32       ufs.union(a, b);
33     } else if (c === 1) {
34       const fa = ufs.find(a);
35       if (fa < 1 || fa > n || b < 1 || b > n) {
36         return console.log("da pian zi!");
37       }
38       if (fa === b) {
39         ufs.union(a, b);
40       } else if (fa !== b) {
41         const fb = ufs.find(b);
42         if (ufs.find(a) === ufs.find(b) ? "we are a team!" : "we are not a team!");
43         ufs.union(fa, fb);
44       } else {
45         console.log("da pian zi!");
46       }
47     }
48   });
49 }
50
51 class UnionFindSet {
52   constructor(n) {
53     this.fa = new Array(n + 1).fill(0).map((_, idx) => idx);
54   }
55
56   find(x) {
57     if (this.fa[x] !== x) {
58       this.fa[x] = this.find(this.fa[x]);
59     }
60     return this.fa[x];
61   }
62
63   union(x, y) {
64     let fa = this.find(x);
65     let fb = this.find(y);
66     if (fa < 1 || fa > n || fb < 1 || fb > n) {
67       this.fa[x] = fa;
68     }
69   }
70 }
71
72 // 并查集实现
73 const readline = require("readline");
74 const rl = readline.createInterface({
75   input: process.stdin,
76   output: process.stdout,
77 });
78
79 const lines = [];
80 let n, m;
81 rl.on("line", (line) => {
82   line.split(" ");
83
84   if (line.length === 1) {
85     [n, m] = line[0].split(" ").map(Number);
86   }
87
88   if (n && line.length === n + 1) {
89     lines.push(line);
90     const msg = line.map((line) => line.split(" ").map(Number));
91     print(msg, n, m);
92     lines.length = 0;
93   }
94 });
95
96 function print(msg, n, m) {
97   if (n < 1 || n > 10000 || m < 1 || m > 10000) return console.log("No!!");
98
99   const ufs = new UnionFindSet(n);
100   msg.forEach(([, , c], [a, b, c]) => {
101     if (c === 0) {
102       ufs.union(a, b);
103     } else if (c === 1) {
104       const fa = ufs.find(a);
105       if (fa < 1 || fa > n || b < 1 || b > n) {
106         return console.log("da pian zi!");
107       }
108       if (fa === b) {
109         ufs.union(a, b);
110       } else if (fa !== b) {
111         const fb = ufs.find(b);
112         if (ufs.find(a) === ufs.find(b) ? "we are a team!" : "we are not a team!");
113         ufs.union(fa, fb);
114       } else {
115         console.log("da pian zi!");
116       }
117     }
118   });
119 }
120
121 class UnionFindSet {
122   constructor(n) {
123     this.fa = new Array(n + 1).fill(0).map((_, idx) => idx);
124   }
125
126   find(x) {
127     if (this.fa[x] !== x) {
128       this.fa[x] = this.find(this.fa[x]);
129     }
130     return this.fa[x];
131   }
132
133   union(x, y) {
134     let fa = this.find(x);
135     let fb = this.find(y);
136     if (fa < 1 || fa > n || fb < 1 || fb > n) {
137       this.fa[x] = fa;
138     }
139   }
140 }
141
142 // 并查集实现
143 const readline = require("readline");
144 const rl = readline.createInterface({
145   input: process.stdin,
146   output: process.stdout,
147 });
148
149 const lines = [];
150 let n, m;
151 rl.on("line", (line) => {
152   line.split(" ");
153
154   if (line.length === 1) {
155     [n, m] = line[0].split(" ").map(Number);
156   }
157
158   if (n && line.length === n + 1) {
159     lines.push(line);
160     const msg = line.map((line) => line.split(" ").map(Number));
161     print(msg, n, m);
162     lines.length = 0;
163   }
164 });
165
166 function print(msg, n, m) {
167   if (n < 1 || n > 10000 || m < 1 || m > 10000) return console.log("No!!");
168
169   const ufs = new UnionFindSet(n);
170   msg.forEach(([, , c], [a, b, c]) => {
171     if (c === 0) {
172       ufs.union(a, b);
173     } else if (c === 1) {
174       const fa = ufs.find(a);
175       if (fa < 1 || fa > n || b < 1 || b > n) {
176         return console.log("da pian zi!");
177       }
178       if (fa === b) {
179         ufs.union(a, b);
180       } else if (fa !== b) {
181         const fb = ufs.find(b);
182         if (ufs.find(a) === ufs.find(b) ? "we are a team!" : "we are not a team!");
183         ufs.union(fa, fb);
184       } else {
185         console.log("da pian zi!");
186       }
187     }
188   });
189 }
190
191 class UnionFindSet {
192   constructor(n) {
193     this.fa = new Array(n + 1).fill(0).map((_, idx) => idx);
194   }
195
196   find(x) {
197     if (this.fa[x] !== x) {
198       this.fa[x] = this.find(this.fa[x]);
199     }
200     return this.fa[x];
201   }
202
203   union(x, y) {
204     let fa = this.find(x);
205     let fb = this.find(y);
206     if (fa < 1 || fa > n || fb < 1 || fb > n) {
207       this.fa[x] = fa;
208     }
209   }
210 }
211
212 // 并查集实现
213 const readline = require("readline");
214 const rl = readline.createInterface({
215   input: process.stdin,
216   output: process.stdout,
217 });
218
219 const lines = [];
220 let n, m;
221 rl.on("line", (line) => {
222   line.split(" ");
223
224   if (line.length === 1) {
225     [n, m] = line[0].split(" ").map(Number);
226   }
227
228   if (n && line.length === n + 1) {
229     lines.push(line);
230     const msg = line.map((line) => line.split(" ").map(Number));
231     print(msg, n, m);
232     lines.length = 0;
233   }
234 });
235
236 function print(msg, n, m) {
237   if (n < 1 || n > 10000 || m < 1 || m > 10000) return console.log("No!!");
238
239   const ufs = new UnionFindSet(n);
240   msg.forEach(([, , c], [a, b, c]) => {
241     if (c === 0) {
242       ufs.union(a, b);
243     } else if (c === 1) {
244       const fa = ufs.find(a);
245       if (fa < 1 || fa > n || b < 1 || b > n) {
246         return console.log("da pian zi!");
247       }
248       if (fa === b) {
249         ufs.union(a, b);
250       } else if (fa !== b) {
251         const fb = ufs.find(b);
252         if (ufs.find(a) === ufs.find(b) ? "we are a team!" : "we are not a team!");
253         ufs.union(fa, fb);
254       } else {
255         console.log("da pian zi!");
256       }
257     }
258   });
259 }
260
261 class UnionFindSet {
262   constructor(n) {
263     this.fa = new Array(n + 1).fill(0).map((_, idx) => idx);
264   }
265
266   find(x) {
267     if (this.fa[x] !== x) {
268       this.fa[x] = this.find(this.fa[x]);
269     }
270     return this.fa[x];
271   }
272
273   union(x, y) {
274     let fa = this.find(x);
275     let fb = this.find(y);
276     if (fa < 1 || fa > n || fb < 1 || fb > n) {
277       this.fa[x] = fa;
278     }
279   }
280 }
281
282 // 并查集实现
283 const readline = require("readline");
284 const rl = readline.createInterface({
285   input: process.stdin,
286   output: process.stdout,
287 });
288
289 const lines = [];
290 let n, m;
291 rl.on("line", (line) => {
292   line.split(" ");
293
294   if (line.length === 1) {
295     [n, m] = line[0].split(" ").map(Number);
296   }
297
298   if (n && line.length === n + 1) {
299     lines.push(line);
300     const msg = line.map((line) => line.split(" ").map(Number));
301     print(msg, n, m);
302     lines.length = 0;
303   }
304 });
305
306 function print(msg, n, m) {
307   if (n < 1 || n > 10000 || m < 1 || m > 10000) return console.log("No!!");
308
309   const ufs = new UnionFindSet(n);
310   msg.forEach(([, , c], [a, b, c]) => {
311     if (c === 0) {
312       ufs.union(a, b);
313     } else if (c === 1) {
314       const fa = ufs.find(a);
315       if (fa < 1 || fa > n || b < 1 || b > n) {
316         return console.log("da pian zi!");
317       }
318       if (fa === b) {
319         ufs.union(a, b);
320       } else if (fa !== b) {
321         const fb = ufs.find(b);
322         if (ufs.find(a) === ufs.find(b) ? "we are a team!" : "we are not a team!");
323         ufs.union(fa, fb);
324       } else {
325         console.log("da pian zi!");
326       }
327     }
328   });
329 }
330
331 class UnionFindSet {
332   constructor(n) {
333     this.fa = new Array(n + 1).fill(0).map((_, idx) => idx);
334   }
335
336   find(x) {
337     if (this.fa[x] !== x) {
338       this.fa[x] = this.find(this.fa[x]);
339     }
340     return this.fa[x];
341   }
342
343   union(x, y) {
344     let fa = this.find(x);
345     let fb = this.find(y);
346     if (fa < 1 || fa > n || fb < 1 || fb > n) {
347       this.fa[x] = fa;
348     }
349   }
350 }
351
352 // 并查集实现
353 const readline = require("readline");
354 const rl = readline.createInterface({
355   input: process.stdin,
356   output: process.stdout,
357 });
358
359 const lines = [];
360 let n, m;
361 rl.on("line", (line) => {
362   line.split(" ");
363
364   if (line.length === 1) {
365     [n, m] = line[0].split(" ").map(Number);
366   }
367
368   if (n && line.length === n + 1) {
369     lines.push(line);
370     const msg = line.map((line) => line.split(" ").map(Number));
371     print(msg, n, m);
372     lines.length = 0;
373   }
374 });
375
376 function print(msg, n, m) {
377   if (n < 1 || n > 10000 || m < 1 || m > 10000) return console.log("No!!");
378
379   const ufs = new UnionFindSet(n);
380   msg.forEach(([, , c], [a, b, c]) => {
381     if (c === 0) {
382       ufs.union(a, b);
383     } else if (c === 1) {
384       const fa = ufs.find(a);
385       if (fa < 1 || fa > n || b < 1 || b > n) {
386         return console.log("da pian zi!");
387       }
388       if (fa === b) {
389         ufs.union(a, b);
390       } else if (fa !== b) {
391         const fb = ufs.find(b);
392         if (ufs.find(a) === ufs.find(b) ? "we are a team!" : "we are not a team!");
393         ufs.union(fa, fb);
394       } else {
395         console.log("da pian zi!");
396       }
397     }
398   });
399 }
400
401 class UnionFindSet {
402   constructor(n) {
403     this.fa = new Array(n + 1).fill(0).map((_, idx) => idx);
404   }
405
406   find(x) {
407     if (this.fa[x] !== x) {
408       this.fa[x] = this.find(this.fa[x]);
409     }
410     return this.fa[x];
411   }
412
413   union(x, y) {
414     let fa = this.find(x);
415     let fb = this.find(y);
416     if (fa < 1 || fa > n || fb < 1 || fb > n) {
417       this.fa[x] = fa;
418     }
419   }
420 }
421
422 // 并查集实现
423 const readline = require("readline");
424 const rl = readline.createInterface({
425   input: process.stdin,
426   output: process.stdout,
427 });
428
429 const lines = [];
430 let n, m;
431 rl.on("line", (line) => {
432   line.split(" ");
433
434   if (line.length === 1) {
435     [n, m] = line[0].split(" ").map(Number);
436   }
437
438   if (n && line.length === n + 1) {
439     lines.push(line);
440     const msg = line.map((line) => line.split(" ").map(Number));
441     print(msg, n, m);
442     lines.length = 0;
443   }
444 });
445
446 function print(msg, n, m) {
447   if (n < 1 || n > 10000 || m < 1 || m > 10000) return console.log("No!!");
448
449   const ufs = new UnionFindSet(n);
450   msg.forEach(([, , c], [a, b, c]) => {
451     if (c === 0) {
452       ufs.union(a, b);
453     } else if (c === 1) {
454       const fa = ufs.find(a);
455       if (fa < 1 || fa > n || b < 1 || b > n) {
456         return console.log("da pian zi!");
457       }
458       if (fa === b) {
459         ufs.union(a, b);
460       } else if (fa !== b) {
461         const fb = ufs.find(b);
462         if (ufs.find(a) === ufs.find(b) ? "we are a team!" : "we are not a team!");
463         ufs.union(fa, fb);
464       } else {
465         console.log("da pian zi!");
466       }
467     }
468   });
469 }
470
471 class UnionFindSet {
472   constructor(n) {
473     this.fa = new Array(n + 1).fill(0).map((_, idx) => idx);
474   }
475
476   find(x) {
477     if (this.fa[x] !== x) {
478       this.fa[x] = this.find(this.fa[x]);
479     }
480     return this.fa[x];
481   }
482
483   union(x, y) {
484     let fa = this.find(x);
485     let fb = this.find(y);
486     if (fa < 1 || fa > n || fb < 1 || fb > n) {
487       this.fa[x] = fa;
488     }
489   }
490 }
491
492 // 并查集实现
493 const readline = require("readline");
494 const rl = readline.createInterface({
495   input: process.stdin,
496   output: process.stdout,
497 });
498
499 const lines = [];
500 let n, m;
501 rl.on("line", (line) => {
502   line.split(" ");
503
504   if (line.length === 1) {
505     [n, m] = line[0].split(" ").map(Number);
506   }
507
508   if (n && line.length === n + 1) {
509     lines.push(line);
510     const msg = line.map((line) => line.split(" ").map(Number));
511     print(msg, n, m);
512     lines.length = 0;
513   }
514 });
515
516 function print(msg, n, m) {
517   if (n < 1 || n > 10000 || m < 1 || m > 10000) return console.log("No!!");
518
519   const ufs = new UnionFindSet(n);
520   msg.forEach(([, , c], [a, b, c]) => {
521     if (c === 0) {
522       ufs.union(a, b);
523     } else if (c === 1) {
524       const fa = ufs.find(a);
525       if (fa < 1 || fa > n || b < 1 || b > n) {
526         return console.log("da pian zi!");
527       }
528       if (fa === b) {
529         ufs.union(a, b);
530       } else if (fa !== b) {
531         const fb = ufs.find(b);
532         if (ufs.find(a) === ufs.find(b) ? "we are a team!" : "we are not a team!");
533         ufs.union(fa, fb);
534       } else {
535         console.log("da pian zi!");
536       }
537     }
538   });
539 }
540
541 class UnionFindSet {
542   constructor(n) {
543     this.fa = new Array(n + 1).fill(0).map((_, idx) => idx);
544   }
545
546   find(x) {
547     if (this.fa[x] !== x) {
548       this.fa[x] = this.find(this.fa[x]);
549     }
550     return this.fa[x];
551   }
552
553   union(x, y) {
554     let fa = this.find(x);
555     let fb = this.find(y);
556     if (fa < 1 || fa > n || fb < 1 || fb > n) {
557       this.fa[x] = fa;
558     }
559   }
560 }
561
562 // 并查集实现
563 const readline = require("readline");
564 const rl = readline.createInterface({
565   input: process.stdin,
566   output: process.stdout,
567 });
568
569 const lines = [];
570 let n, m;
571 rl.on("line", (line) => {
572   line.split(" ");
573
574   if (line.length === 1) {
575     [n, m] = line[0].split(" ").map(Number);
576   }
577
578   if (n && line.length === n + 1) {
579     lines.push(line);
580     const msg = line.map((line) => line.split(" ").map(Number));
581     print(msg, n, m);
582     lines.length = 0;
583   }
584 });
585
586 function print(msg, n, m) {
587   if (n < 1 || n > 10000 || m < 1 || m > 10000) return console.log("No!!");
588
589   const ufs = new UnionFindSet(n);
590   msg.forEach(([, , c], [a, b, c]) => {
591     if (c === 0) {
592       ufs.union(a, b);
593     } else if (c === 1) {
594       const fa = ufs.find(a);
595       if (fa < 1 || fa > n || b < 1 || b > n) {
596         return console.log("da pian zi!");
597       }
598       if (fa === b) {
599         ufs.union(a, b);
600       } else if (fa !== b) {
601         const fb = ufs.find(b);
602         if (ufs.find(a) === ufs.find(b) ? "we are a team!" : "we are not a team!");
603         ufs.union(fa, fb);
604       } else {
605         console.log("da pian zi!");
606       }
607     }
608   });
609 }
610
611 class UnionFindSet {
612   constructor(n) {
613     this.fa = new Array(n + 1).fill(0).map((_, idx) => idx);
614   }
615
616   find(x) {
617     if (this.fa[x] !== x) {
618       this.fa[x] = this.find(this.fa[x]);
619     }
620     return this.fa[x];
621   }
622
623   union(x, y) {
624     let fa = this.find(x);
625     let fb = this.find(y);
626     if (fa < 1 || fa > n || fb < 1 || fb > n) {
627       this.fa[x] = fa;
628     }
629   }
630 }
631
632 // 并查集实现
633 const readline = require("readline");
634 const rl = readline.createInterface({
635   input: process.stdin,
636   output: process.stdout,
637 });
638
639 const lines = [];
640 let n, m;
641 rl.on("line", (line) => {
642   line.split(" ");
643
644   if (line.length === 1) {
645     [n, m] = line[0].split(" ").map(Number);
646   }
647
648   if (n && line.length === n + 1) {
649     lines.push(line);
650     const msg = line.map((line) => line.split(" ").map(Number));
651     print(msg, n, m);
652     lines.length = 0;
653   }
654 });
655
656 function print(msg, n, m) {
657   if (n < 1 || n > 10000 || m < 1 || m > 10000) return console.log("No!!");
658
659   const ufs = new UnionFindSet(n);
660   msg.forEach(([, , c], [a, b, c]) => {
661     if (c === 0) {
662       ufs.union(a, b);
663     } else if (c === 1) {
664       const fa = ufs.find(a);
665       if (fa < 1 || fa > n || b < 1 || b > n) {
666         return console.log("da pian zi!");
667       }
668       if (fa === b) {
669         ufs.union(a, b);
670       } else if (fa !== b) {
671         const fb = ufs.find(b);
672         if (ufs.find(a) === ufs.find(b) ? "we are a team!" : "we are not a team!");
673         ufs.union(fa, fb);
674       } else {
675         console.log("da pian zi!");
676       }
677     }
678   });
679 }
680
681 class UnionFindSet {
682   constructor(n) {
683     this.fa = new Array(n + 1).fill(0).map((_, idx) => idx);
684   }
685
686   find(x) {
687     if (this.fa[x] !== x) {
688       this.fa[x] = this.find(this.fa[x]);
689     }
690     return this.fa[x];
691   }
692
693   union(x, y) {
694     let fa = this.find(x);
695     let fb = this.find(y);
696     if (fa < 1 || fa > n || fb < 1 || fb > n) {
697       this.fa[x] = fa;
698     }
699   }
700 }
701
702 // 并查集实现
703 const readline = require("readline");
704 const rl = readline.createInterface({
705   input: process.stdin,
706   output: process.stdout,
707 });
708
709 const lines = [];
710 let n, m;
711 rl.on("line", (line) => {
712   line.split(" ");
713
714   if (line.length === 1) {
715     [n, m] = line[0].split(" ").map(Number);
716   }
717
718   if (n && line.length === n + 1) {
719     lines.push(line);
720     const msg = line.map((line) => line.split(" ").map(Number));
721     print(msg, n, m);
722     lines.length = 0;
723   }
724 });
725
726 function print(msg, n, m) {
727   if (n < 1 || n > 10000 || m < 1 || m > 10000) return console.log("No!!");
728
729   const ufs = new UnionFindSet(n);
730   msg.forEach(([, , c], [a, b, c]) => {
731     if (c === 0) {
732       ufs.union(a, b);
733     } else if (c === 1) {
734       const fa = ufs.find(a);
735       if (fa < 1 || fa > n || b < 1 || b > n) {
736         return console.log("da pian zi!");
737       }
738       if (fa === b) {
739         ufs.union(a, b);
740       } else if (fa !== b) {
741         const fb = ufs.find(b);
742         if (ufs.find(a) === ufs.find(b) ? "we are a team!" : "we are not a team!");
743         ufs.union(fa, fb);
744       } else {
745         console.log("da pian zi!");
746       }
747     }
748   });
749 }
750
751 class UnionFindSet {
752   constructor(n) {
753     this.fa = new Array(n + 1).fill(0).map((_, idx) => idx);
754   }
755
756   find(x) {
757     if (this.fa[x] !== x) {
758       this.fa[x] = this.find(this.fa[x]);
759     }
760     return this.fa[x];
761   }
762
763   union(x, y) {
764     let fa = this.find(x);
765     let fb = this.find(y);
766     if (fa < 1 || fa > n || fb < 1 || fb > n) {
767       this.fa[x] = fa;
768     }
769   }
770 }
771
772 // 并查集实现
773 const readline = require("readline");
774 const rl = readline.createInterface({
775   input: process.stdin,
776   output: process.stdout,
777 });
778
779 const lines = [];
780 let n, m;
781 rl.on("line", (line) => {
782   line.split(" ");
783
784   if (line.length === 1) {
785     [n, m] = line[0].split(" ").map(Number);
786   }
787
788   if (n && line.length === n + 1) {
789     lines.push(line);
790     const msg = line.map((line) => line.split(" ").map(Number));
791     print(msg, n, m);
792     lines.length = 0;
793   }
794 });
795
796 function print(msg, n, m) {
797   if (n < 1 || n > 10000 || m < 1 || m > 10000) return console.log("No!!");
798
799   const ufs = new UnionFindSet(n);
800   msg.forEach(([, , c], [a, b, c]) => {
801     if (c === 0) {
802       ufs.union(a, b);
803     } else if (c === 1) {
804       const fa = ufs.find(a);
805       if (fa < 1 || fa > n || b < 1 || b > n) {
806         return console.log("da pian zi!");
807       }
808       if (fa === b) {
809         ufs.union(a, b);
810       } else if (fa !== b) {
811         const fb = ufs.find(b);
812         if (ufs.find(a) === ufs.find(b) ? "we are a team!" : "we are not a team!");
813         ufs.union(fa, fb);
814       } else {
815         console.log("da pian zi!");
816       }
817     }
818   });
819 }
820
821 class UnionFindSet {
822   constructor(n) {
823     this.fa = new Array(n + 1).fill(0).map((_, idx) => idx);
824   }
825
826   find(x) {
827     if (this.fa[x] !== x) {
828       this.fa[x] = this.find(this.fa[x]);
829     }
830     return this.fa[x];
831   }
832
833   union(x, y) {
834     let fa = this.find(x);
835     let fb = this.find(y);
836     if (fa < 1 || fa > n || fb < 1 || fb > n) {
837       this.fa[x] = fa;
838     }
839   }
840 }
841
842 // 并查集实现
843 const readline = require("readline");
844 const rl = readline.createInterface({
845   input: process.stdin,
846   output: process.stdout,
847 });
848
849 const lines = [];
850 let n, m;
851 rl.on("line", (line) => {
852   line.split(" ");
853
854   if (line.length === 1) {
855     [n, m] = line[0].split(" ").map(Number);
856   }
857
858   if (n && line.length === n + 1) {
859     lines.push(line);
860     const msg = line.map((line) => line.split(" ").map(Number));
861     print(msg, n, m);
862     lines.length = 0;
863   }
864 });
865
866 function print(msg, n, m) {
867   if (n < 1 || n > 10000 || m < 1 || m > 10000) return console.log("No!!");
868
869   const ufs = new UnionFindSet(n);
870   msg.forEach(([, , c], [a, b, c]) => {
871     if (c === 0) {
872       ufs.union(a, b);
873     } else if (c === 1) {
874       const fa = ufs.find(a);
875       if (fa < 1 || fa > n || b < 1 || b > n) {
876         return console.log("da pian zi!");
877       }
878       if (fa === b) {
879         ufs.union(a, b);
880       } else if (fa !== b) {
881         const fb = ufs.find(b);
882         if (ufs.find(a) === ufs.find(b) ? "we are a team!" : "we are not a team!");
883         ufs.union(fa, fb);
884       } else {
885         console.log("da pian zi!");
886       }
887     }
888   });
889 }
890
891 class UnionFindSet {
892   constructor(n) {
893     this.fa = new Array(n + 1).fill(0).map((_, idx) => idx);
894   }
895
896   find(x) {
897     if (this.fa[x] !== x) {
898       this.fa[x] = this.find(this.fa[x]);
899     }
900     return this.fa[x];
901   }
902
903   union(x, y) {
904     let fa = this.find(x);
905     let fb = this.find(y);
906     if (fa < 1 || fa > n || fb < 1 || fb > n) {
907       this.fa[x] = fa;
908     }
909   }
910 }
911
912 // 并查集实现
913 const readline = require("readline");
914 const rl = readline.createInterface({
915   input: process.stdin,
916   output: process.stdout,
917 });
918
919 const lines = [];
920 let n, m;
921 rl.on("line", (line) => {
922   line.split(" ");
923
924   if (line.length === 1) {
925     [n, m] = line[0].split(" ").map(Number);
926   }
927
928   if (n && line.length === n + 1) {
929     lines.push(line);
930     const msg = line.map((line) => line.split(" ").map(Number));
931     print(msg, n, m);
932     lines.length = 0;
933   }
934 });
935
936 function print(msg, n, m) {
937   if (n < 1 || n > 10000 || m < 1 || m > 10000) return console.log("No!!");
938
939   const ufs = new UnionFindSet(n);
94
```