

证明:

首先, 可以证明一个前提结论: 对于一个遍历顺序 (逆时针或顺时针) 已经确定的奇数长的环状排列, 改变起点的位置 (当然, 终点也会随之变化), 并不会改变该排列逆序数的奇偶性。可以假设原排列为 $a_1, a_2, \dots, a_{n-1}, a_n$, 先经过一次的变化, 变为 $a_2, a_3, \dots, a_{n-1}, a_n, a_1$, 假设原本与 a_1 组合的逆序对有 x 个, 那么之后的即为 $n - 1 - x$, 那么变化的逆序对为 $n - 1 - 2x$, 那么其奇偶性未变。得证。

另外, 操作一次后, 逆序对的改变量是 $n - 2x$, 为偶数

那么, 当 n 为奇数, 并且逆序对是奇数时, 不可能通过操作使得其逆序数变为 0;

如果 n 为偶数, 由上面的分析可知, 必然存在着 $n/2$ 个起点, 其逆序对为偶数。

现在证明如果逆序对为偶数, 必有解。

假设位置 i 左边已经拍好, 那么数 i 所在的位置 j 大于 i ,

如果:

- $a_i, i, ?, ?, a_{i+4}$
- $a_i, ?, i, ?, a_{i+4}$
- $a_i, ?, ?, i, a_{i+4}$

容易得到, 在上面情况下, 我们仅需要通过操作下标 $i - 1$ 到 $i + 4$ 的这 5 个数来让数 i 到下标 i 处。

而如果 i 的位置大于等于 $i + 4$, 设 1 它为 $pos[i]$, 我们也可以通过仅仅操作下标为 $i + 1$ 到 $pos[i]$ 的数, (不断选取以它为右端点的区间操作即可) 到达上面的情况。

所以, 我们可以不断延展这个序列, 直到 $i + 4 > n$, 就无法拓展了, 而此时, 只有 $n - 3, n - 2, n - 1, n$ 这 4 个数未被排序了, $4! = 24$, 那么我们就可以通过暴力枚举 $n = 8$, 排列为 $1, 2, 3, 4, ?, ?, ?, ?$ 的 24/2 种情况即可

。于是, 就得以证明了。

下面是测试的程序:

```
1  #include<cstdio>
2  #include<cstring>
3  #include<algorithm>
4  using namespace std;
5
6  typedef int State[8];
7  State st[40400], goal, start[30];
8  bool vis1[10], vis2[40400];
9  int fact[8];
10 int cnt = 0;
11
12 void init_fact() {
13     fact[0] = 1;
14     for (int i = 1; i < 8; ++i) fact[i] = fact[i - 1] * i;
15 }
16 bool try_insert(int s) {
17     int code = 0;
18     for (int i = 0; i < 8; ++i) {
19         int count = 0;
```

```

20     for (int j = i + 1; j < 8; ++j) if (st[s][j] < st[s][i])
count++;
21     code += fact[7 - i] * count;
22 }
23 if (vis2[code]) return 0;
24 return vis2[code] = 1;
25 }
26 void to_start(int k) {
27     if (k == 8) {
28         start[cnt][0] = 1; start[cnt][1] = 2; start[cnt][2] = 3;
start[cnt][3] = 4;
29         for (int i = 4; i < 8; ++i) {
30             if (start[cnt][i] == 0) start[cnt][i] = start[cnt - 1][i];
31             else break;
32         }
33         cnt++;
34         return;
35     }
36     for (int i = 5; i <= 8; ++i)
37         if (vis1[i] == 0) {
38             start[cnt][k] = i;
39             vis1[i] = 1;
40             to_start(k + 1);
41             vis1[i] = 0;
42         }
43 }
44
45 bool judge(int x) {
46     int t = 0;
47     for (int i = 0; i < 8; ++i)
48         for (int j = i + 1; j < 8; ++j)
49             if (start[x][i] > start[x][j]) t++;
50     return !(t & 1);
51 }
52 void caozuo(int i, int j) {
53     swap(st[i][j], st[i][j + 3]);
54     swap(st[i][j + 1], st[i][j + 2]);
55 }
56
57 int main() {
58     to_start(4);
59     init_fact();
60     goal[0] = 1; goal[1] = 2; goal[2] = 3; goal[3] = 4; goal[4] = 5;
goal[5] = 6; goal[6] = 7; goal[7] = 8;
61     bool flag1 = 1, flag2 = 0;
62     for (int i = 0; i < cnt; ++i) if (judge(i)) {
63         printf("%d ", i);
64         memset(vis2, 0, sizeof(vis2));
65         int front = 0, rear = 1;
66         memcpy(st[0], start[i], sizeof(st[0]));
67         while (front < rear) {
68             State& s = st[front];
69             if (memcmp(goal, s, sizeof(s)) == 0) {flag2 = 1; break;}
70             for (int j = 0; j < 5; ++j) {
71                 State& t = st[rear];

```

```

72         memcpy(t, s, sizeof(s));
73         caozuo(rear, j);
74         if (try_insert(rear)) rear++;
75     }
76     front++;
77 }
78 if (flag2) {flag2 = 0; printf("yes\n") ; continue;}
79 else {flag1 = 0; printf("no\n"); break;}
80 }
81 printf("%d", flag1);
82 return 0;
83 }

```

得到的结果是：

```

1  0 yes
2  3 yes
3  4 yes
4  7 yes
5  8 yes
6  11 yes
7  12 yes
8  15 yes
9  16 yes
10 19 yes
11 20 yes
12 23 yes
13 1

```

每一种情况都是可以找到解的。

证明完毕。