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$a_1, a_2, \ldots a_{n-1}, a_n$，先经过一次的变化，变为$a_2, a_3, . a_{n-1}, a_n, a_1$,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$6-2x$，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$n/2$个起点是其逆序对为偶数。

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$i--i+4$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$i+1--pos[i]$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：

```cpp
#include<cstdio>
#include<cstring>
#include<algorithm>
using namespace std;

typedef int State[8];
State st[40400], goal, start[30];
bool vis1[10], vis2[40400];
int fact[8];
int cnt = 0;

void init_fact() {
    fact[0] = 1;
    for (int i = 1; i < 8; ++i) fact[i] = fact[i - 1] * i;
}
bool try_insert(int s) {
    int code = 0;
    for (int i = 0; i < 8; ++i) {
        int count = 0;
```

```
        for (int j = i + 1; j < 8; ++j) if (st[s][j] < st[s][i])
count++;
        code += fact[7 - i] * count;
    }
    if (vis2[code]) return 0;
    return vis2[code] = 1;
}
void to_start(int k) {
    if (k == 8) {
        start[cnt][0] = 1; start[cnt][1] = 2; start[cnt][2] = 3;
start[cnt][3] = 4;
        for (int i = 4; i < 8; ++i) {
            if (start[cnt][i] == 0) start[cnt][i] = start[cnt - 1][i];
            else break;
        }
        cnt++;
        return;
    }
    for (int i = 5; i <= 8; ++i)
        if (vis1[i] == 0) {
            start[cnt][k] = i;
            vis1[i] = 1;
            to_start(k + 1);
            vis1[i] = 0;
        }
}

bool judge(int x) {
    int t = 0;
    for (int i = 0; i < 8; ++i)
        for (int j = i + 1; j < 8; ++j)
            if (start[x][i] > start[x][j]) t++;
    return !(t & 1);
}
void caozuo(int i, int j) {
    swap(st[i][j], st[i][j + 3]);
    swap(st[i][j + 1], st[i][j + 2]);
}

int main() {
    to_start(4);
    init_fact();
    goal[0] = 1; goal[1] = 2; goal[2] = 3; goal[3] = 4; goal[4] = 5;
goal[5] = 6; goal[6] = 7; goal[7] = 8;
    bool flag1 = 1, flag2 = 0;
    for (int i = 0; i < cnt; ++i) if (judge(i)) {
        printf("%d ", i);
        memset(vis2, 0, sizeof(vis2));
        int front = 0, rear = 1;
        memcpy(st[0], start[i], sizeof(st[0]));
        while (front < rear) {
            State& s = st[front];
            if (memcmp(goal, s, sizeof(s)) == 0) {flag2 = 1; break;}
            for (int j = 0; j < 5; ++j) {
                State& t = st[rear];
```

```
                memcpy(t, s, sizeof(s));
                caozuo(rear, j);
                if (try_insert(rear)) rear++;
            }
            front++;
        }
        if (flag2) {flag2 = 0; printf("yes\n") ; continue;}
        else {flag1 = 0; printf("no\n"); break;}
    }
    printf("%d", flag1);
    return 0;
}
```

得到的结果是：

```
0 yes
3 yes
4 yes
7 yes
8 yes
11 yes
12 yes
15 yes
16 yes
19 yes
20 yes
23 yes
1
```

每一种情况都是可以找到解的。

证明完毕。