数组`arr`是`[0, 1, ..., arr.length - 1]`的一种排列，我们将这个数组分割成几个“块”，并将这些块分别进行排序。之后再连接起来，使得连接的结果和按升序排序后的原数组相同。我们最多能将数组分成多少块？

示例 1:

输入: `arr = [4,3,2,1,0]`
输出: `1`
解释:
将数组分成2块或者更多块，都无法得到所需的结果。
例如，分成 `[4, 3]`, `[2, 1, 0]` 的结果是 `[3, 4, 0, 1, 2]`，这不是有序的数组。

示例 2:

输入: `arr = [1,0,2,3,4]`
输出: `4`
解释:
我们可以把它分成两块，例如 `[1, 0]`, `[2, 3, 4]`。
然而，分成 `[1, 0]`, `[2]`, `[3]`, `[4]` 可以得到最多的块数。

注意:

- `arr` 的长度在 `[1, 10]` 之间。
- `arr[i]`是 `[0, 1, ..., arr.length - 1]`的一种排列。

思路：该题其实有种规律，因为其数字组成为0~arr.length-1那么其中的数字为连续不重复的，可以重排序为升序的段有种特点，就是其最大的数字刚好为其升序完后该段最后一个数字，按这个规律找，若是该段中最大数字刚好等于其位置，分段+1，不等于，继续向下一个找

代码

```
class Solution {
public:
    int maxChunksToSorted(vector<int>& arr) {
        int num=0;
        int max=-1;
        for(int i=0;i<arr.size();i++){
            if(arr[i]>max){
                max=arr[i];
            }
            if(max==i){
                num++;
```

```
            }
        }
        return num;
    }
};
```