① The pancake problem

2	6	5	2	4	3	1
6	2	3	1	1	2	2
1	1	4	4	2	1	3
4	4	1	3	3	4	4
3	3	2	5	5	5	5
5	5	6	6	6	6	6

Python :

1) find max index
2) flip starting from this max index
3) flip entire stack
   - leave this max alone for the rest
   - resolved_pancakes = 1
4) find the max starting from resolved_pancakes + 1
5) flip starting from this max index
6) flip entire stack
   - resolved_pancakes = 2

Strategy :

1) findmaxidx and start fliping from there
2) start fliping from unresolved pancakes
3) increment resolved pancakes

```
pancakes = [5,3,4,1,6,2]
afisare....
def findmaxidx(pans, start):
    maxval = pans[start]
    maxidx = start
    i = start
    while i < len(pans)
        if pans[i] > maxval :
            maxval = pans[i]
            maxidx = i
        i += 1
    return maxidx
```

---

findmaxidx (pancakes, 0) – example

```
def flip (pancakes, start) :
    i = start
    j = len(pancakes) - 1   # because of 0- based indexing
    while i < j
        temp = pans[i]
        pans[i] = pans[j]
        pans[j] = temp
        i += 1
        j -= 1
```

---

test

```
resolved_pancakes = len(pancakes)   # stop cond.
def pancake_sort (pans) :
    resolved = 0
    while resolved < len(pans):
        print(f"Step: {resolved+1}")
        print(f" Before {pans}")
        flip (pans, findmaxidx (pans, resolved))
        flip (pans, resolved)
        resolved += 1
```