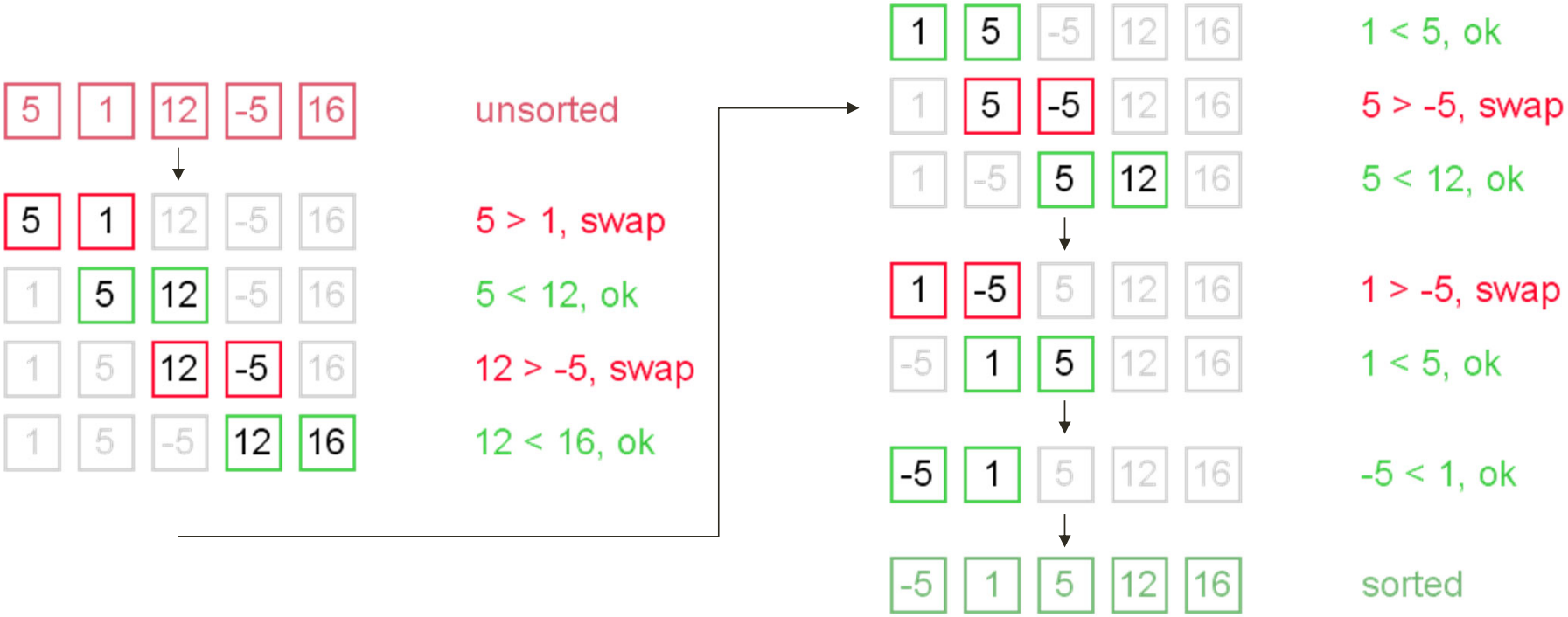
BUBBLE SORT


5 1 12 -5 16
unsorted
5 1 12 -5 16
5 > 1, swap
1 5 12 -5 16
5 < 12, ok
1 5 12 -5 16
12 > -5, swap
1 5 -5 12 16
12 < 16, ok
1 5 -5 12 16
1 < 5, ok
1 5 -5 12 16
5 > -5, swap
1 -5 5 12 16
5 < 12, ok
1 -5 5 12 16
1 > -5, swap
-5 1 5 12 16
1 < 5, ok
-5 1 5 12 16
-5 < 1, ok
-5 1 5 12 16
sorted

BUBBLE SORT ALGORITHM

This algorithm can do the job but not effective

```
number = [5,1,12,-5,16]
for i in range(len(number)-1): #outer loop, repeat at most (numbers of data - 1) times
    for j in range(len(number)-1):  #inner loop,
        if number[j] > number[j+1]: #swap data if the first data is greater than the second data
            temp = number[j]
            number[j] = number[j+1]
            number[j+1] = temp
    print('At the end of loop',(i+1),':', number)
```

BUBBLE SORT ALGORITHM

- This algorithm is effective because no unnecessary processes

```
number = [5, 1, 12, -5, 16]

swap = True  # initialise flag
while swap == True:  # this means if there was a swap in the previous loop
                     # the number is not in order yet.
    swap = False     # before checking the number reset swap flag to False
                     # it means there is no swap yet
    for i in range(len(number)-1):  # inner loop, this will check (number of data -1) times
        if number[i] > number[i+1]:  # if the left number is greater than the right number
            temp = number[i]         # swap them
            number[i] = number[i+1]
            number[i+1] = temp
            swap = True  # this means, there was a swap during the check

print(number)
```