0 1 2 3 4 5

[4, 9, -3, 4, 2, 1]

partition(smallArr, 0, smallArr.length)

low = 0
high = 6

pivotIndex = 5
pivot = 1

SBI = 0
LAI = 4

```
5
6  public class ClassPartitioner implements Partitioner{
7      public static String[] makeRandomArr(int size){
8          String[] randomArr = new String[size];
9          for(int i = 0; i < size; i++){
10             randomArr[i] = Integer.toString(ThreadLocalRandom.current().nextInt(-10, 11));
11         }
12         return randomArr;
13     }
14
15     public static void swap(String[] array, int i1, int i2) {
16         String temp = array[i1];
17             array[i1] = array[i2];
18             array[i2] = temp;
19     }
20     public int partition(String[] array, int low, int high) {
21         if(low == high){
22                 return low;
23                 }
24                 int pivotIndex = high - 1;
25         String pivot = array[pivotIndex];
26         int smallerBeforeIndex = low;
27         int largerAfterIndex = high - 2;
28         while(largerAfterIndex >= smallerBeforeIndex) {
29             if(Integer.parseInt(array[smallerBeforeIndex]) > Integer.parseInt(pivot)) {
30                 swap(array, smallerBeforeIndex, largerAfterIndex);
31                         largerAfterIndex -= 1;
32                 }
33             else {
34                     smallerBeforeIndex += 1;
35                 }
36         }
37
38         if(Integer.parseInt(array[smallerBeforeIndex]) < Integer.parseInt(pivot)){
```

LAI >= SBI

array[0] = 4 > 1

[2, 9, -3, 4, 4, 1] ⇒

SBI = 0
LAI = 3

LAI >= SBI

array[0] = 2 > 1

[4, 9, -3, 2, 4, 1]

SBI = 0
LAI = 2

LAI >= SBI

array[0] = 4 > 1

[-3, 9, 4, 2, 4, 1]

SBI = 0
LAI = 1

LAI >= SBI

array[0] = -3 < 1

[-3, 9, 4, 2, 4, 1] (no swap)

SBI = 1
LAI = 1

LAI >= SBI

array[1] = 9 > 1

[-3, 9, 4, 2, 4, 1]

SBI = 1
LAI = 0

~~LAI >= SBI~~

while loop is finished

if(array[SBI] < pivot)
9 < 1)

else: swap(SBI, pivotIndex)

[-3, 1, 4, 2, 4, 9]

1> 1<