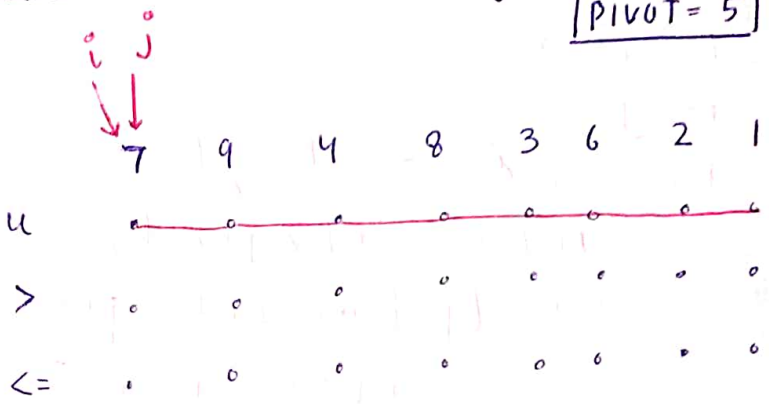
PARTITION AN ARRAY

- Hume ek array dia hai numbers ka! and Hume ek PIVOT bhi diya hai (pivot = 5).
- Toh hume esse partition krna hai taki PIVOT se saare chotey no. ek traf aur usse bade no. ek traf!
- Iss Algorithm me hum (3) regions smaj lety hai! Total array split into (3) regions
  - unknown elements
  - greater elements
  - less than equal to elements
- Aur (2) pointer use krty hai (i) and (j)
- (0) se (j-1) tak saare choty elements rahty hai!
- (j) se (i-1) tak rahty hai saare bade log/elements!
- (i) se (end) tak rahty hai saare unknown elements

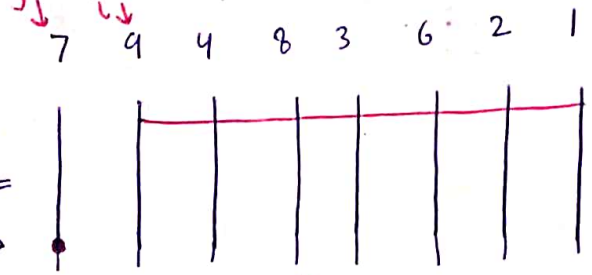
PIVOT = 5



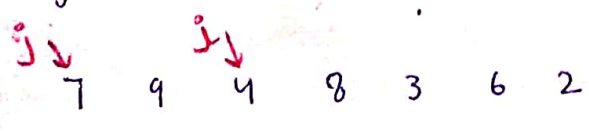
- Agar `arr[i] > p` hai toh (i++);
  - iska mtlb unknown region se ek banda kam! Aur greater wale region me ek bad gaya!
- Agar `arr[i] ≤ p` hai toh `swap(a, i, j)` and (i++) (j++)
  - isska mtlb unknow ka region kam hojata hai, less than equal to walo ka region badta hai, greater waley na toh badty hai na toh ghaty hai, agey shift hojaty hai!

(pivot = 5)

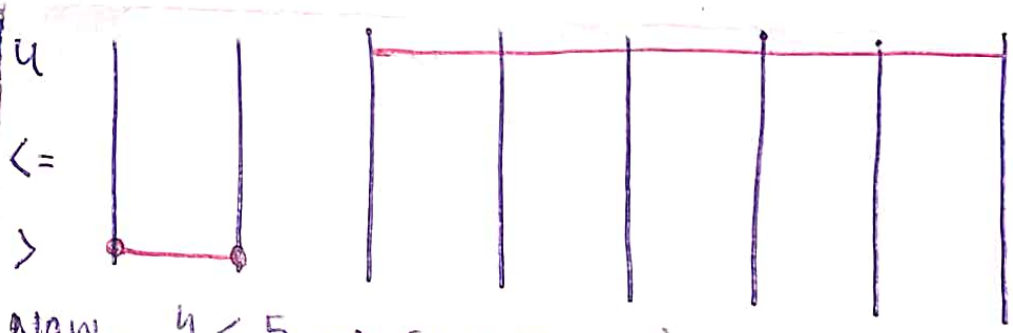
★ phele element (7) check hua
7 > 5 → i++



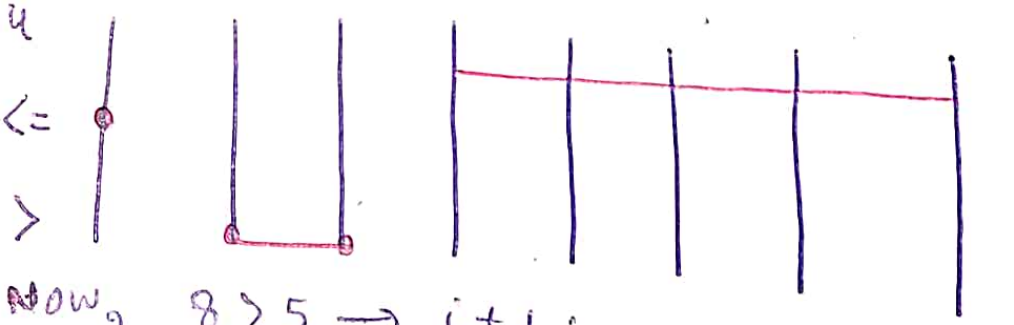
★ NOW, 9 > 5 → i++
unknown ne element khodiya greater walo ko element mila!



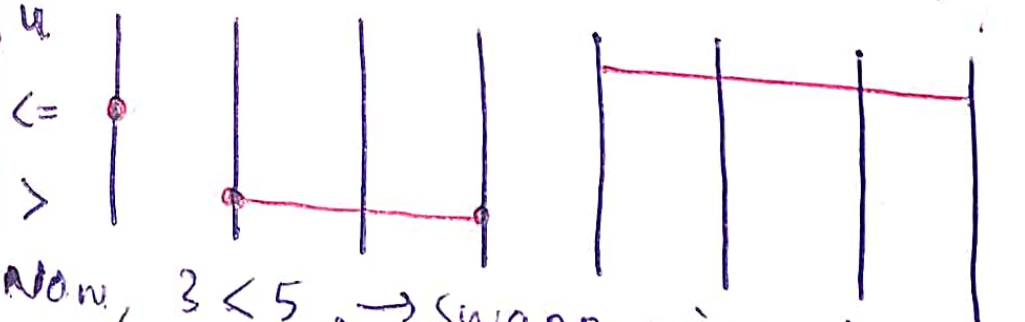

```
int i = 0;
int j = 0;
while (i < arr.length){
 if(arr[i] > pivot) {
      i++;
 }else {
      swap(arr, i, j);
      i++;
      j++;
 }
```



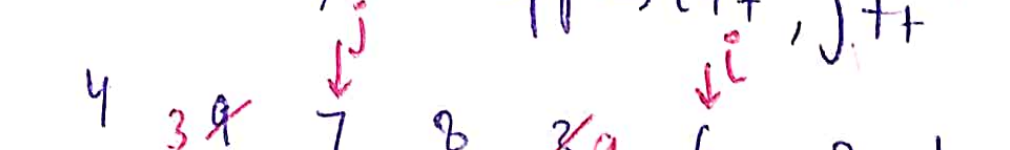
Now, $4 < 5 \rightarrow$ swap $\rightarrow i++ \rightarrow j++$



Now, $8 > 5 \rightarrow i++$

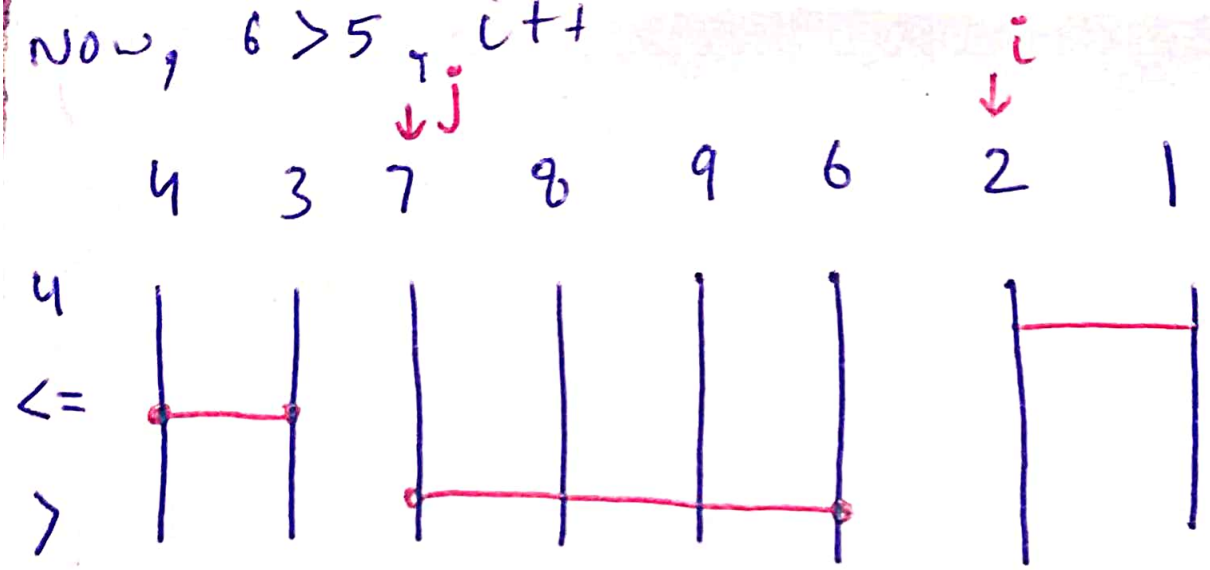


Now, $3 < 5 \rightarrow$ swapp $\rightarrow i++, j++$

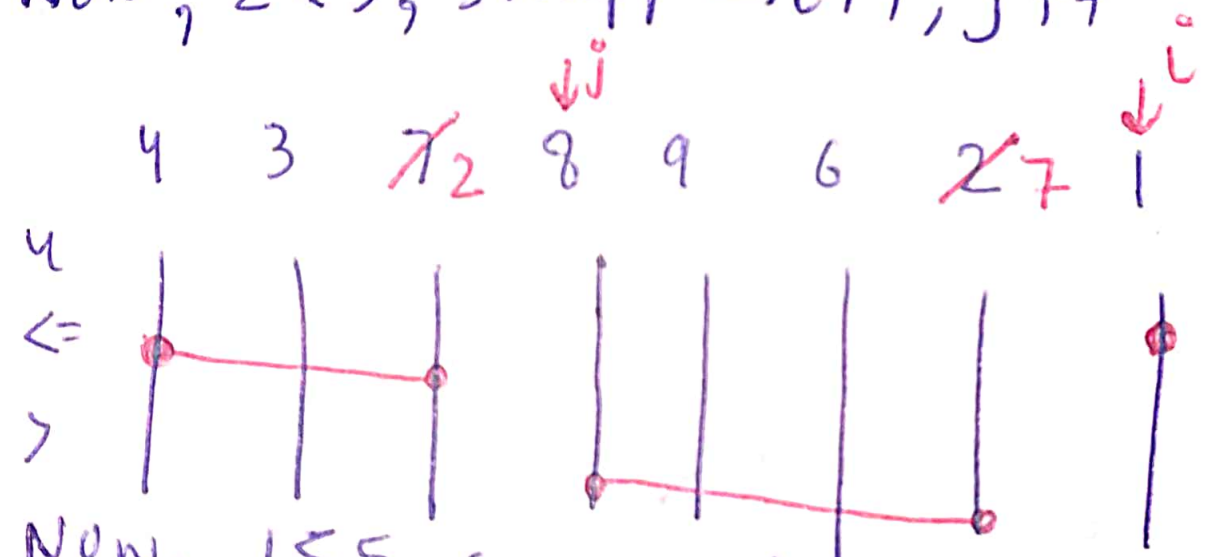


Now, $6 > 5$, $i++$

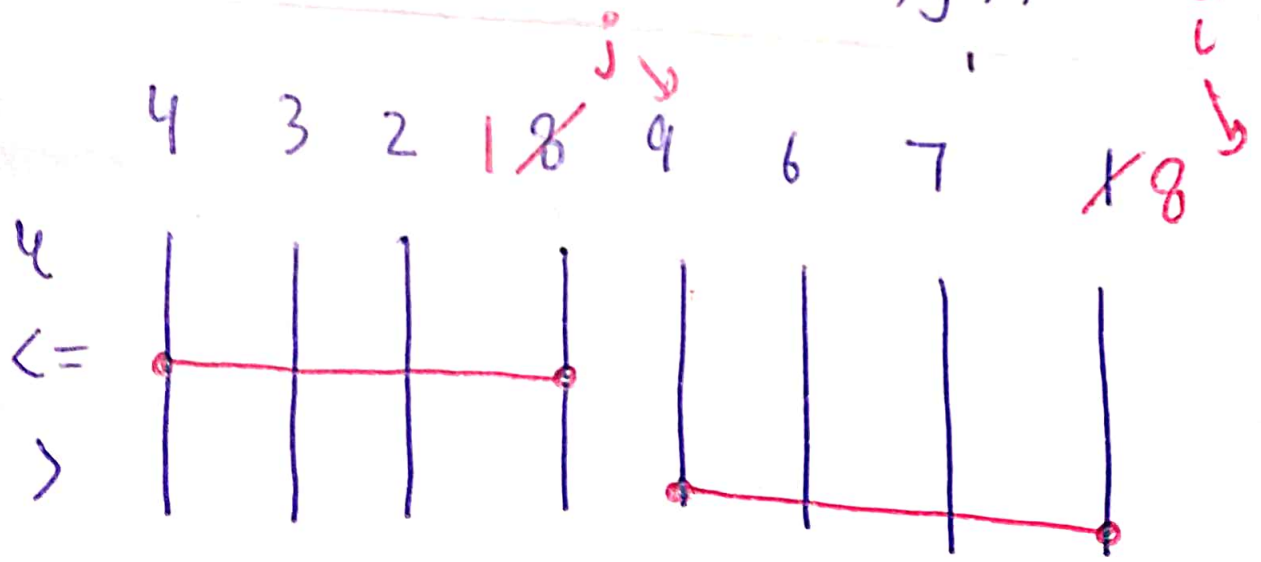
Now, 6 > 5, i++



Now, 2 < 5, swapp → i++, j++



Now, 1 < 5, swap → i++, j++



4	3	2	1	9	6	7	8

Partitioning Algorithms used for
- ≤ pivot, ≥ pivot
- odd - even
- 0-1 separate
- 0-non-zero seperate

```
public static void partition (int []arr, int pivot) {
  int i=0;
  int j=0;
  while ( i< arr.length) {
      if(arr [i] > pivot) { i++; }
      else(if ( arr[i] ≤ pivot) {
         swap(arr, i ,j);
          i++;
          j++;
      }
    }
  }
public static void swap (int []arr, int i, int j) {
 System.out.println("swapping "+ arr[i] + "and" + arr[j]);
    int temp = arr[i];
    arr[i] = arr[j];
      arr[j] = temp;
  }
public static void print (int []arr) {
 for(int i=0; i<arr.length; i++) {
        System.out.print (arr[i] + " ");
  }
  System.out.println();
}
public static void main (String [] args) {
  Scanner s= new Scanner (System.in);
   int n= s.nextInt();
  for(int i=0; i<n; i++) {
     arr[i] = s.nextInt();
  }
  int pivot = s.nextInt();
  partition (arr, pivot);
  print (arr);
  }
```

Time complexity
O(n)

Space complexity
O(1)