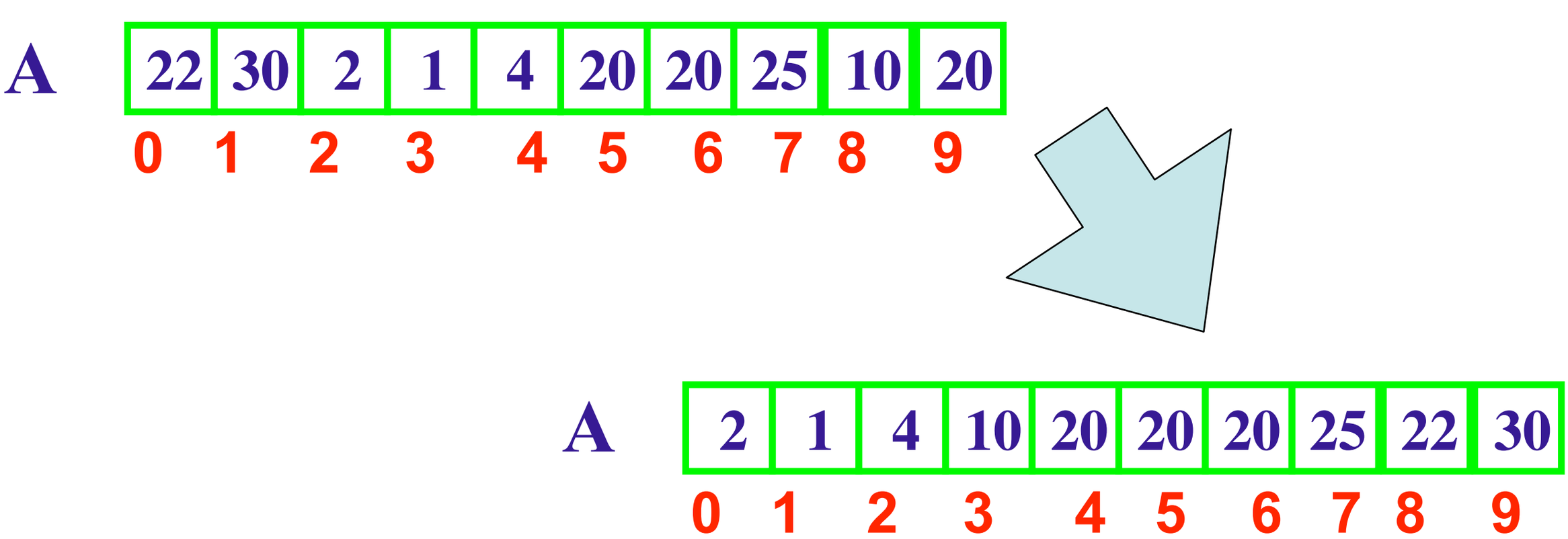
Esercizio

Si progetti un algoritmo che realizza una tripartizione degli elementi dell'array determinata da un pivot.
L'array finale deve avere tutti gli elementi minori del pivot a sinistra, seguiti da tutte le occorrenze del pivot a loro volta seguite dagli elementi più grandi del pivot.
Nell'esempio il pivot è l'ultimo elemento:



Tripartition

Se avessimo un array di elementi con ripetizioni, quale approccio possiamo usare per ottenere un array in cui gli elementi minori del pivot precedono il pivot, essendo seguiti da quelli uguali al pivot e poi da quelli maggiori?

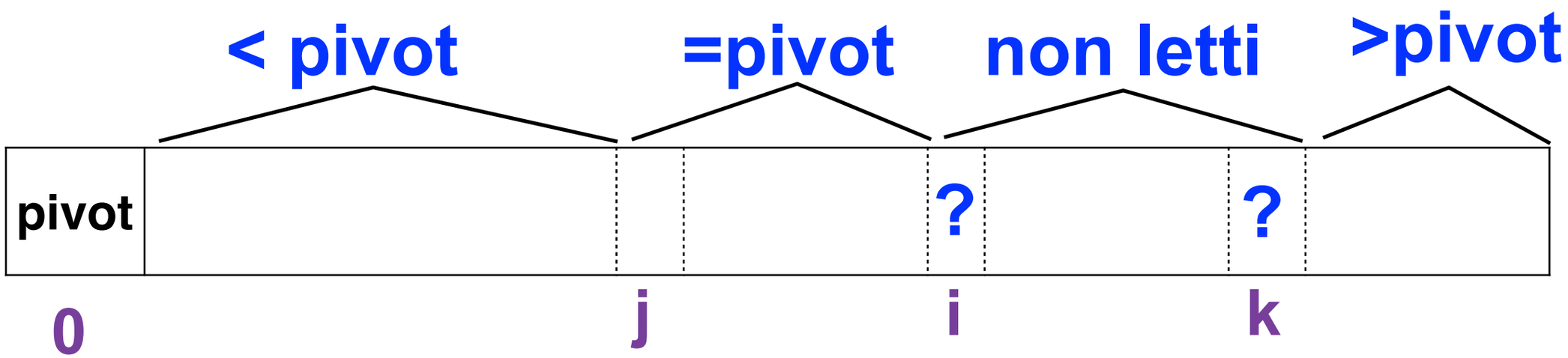
Potremmo utilizzare sia l'idea di Hoare che quella della scansione solo da sinistra.

Mettiamo nella parte iniziale dell'array gli elementi minori e uguali e nella parte finale i maggiori, come nella partizione di Hoare, ma teniamo separati e gestiamo i minori e gli uguali come nel primo algoritmo per la partizione.

Tripartition

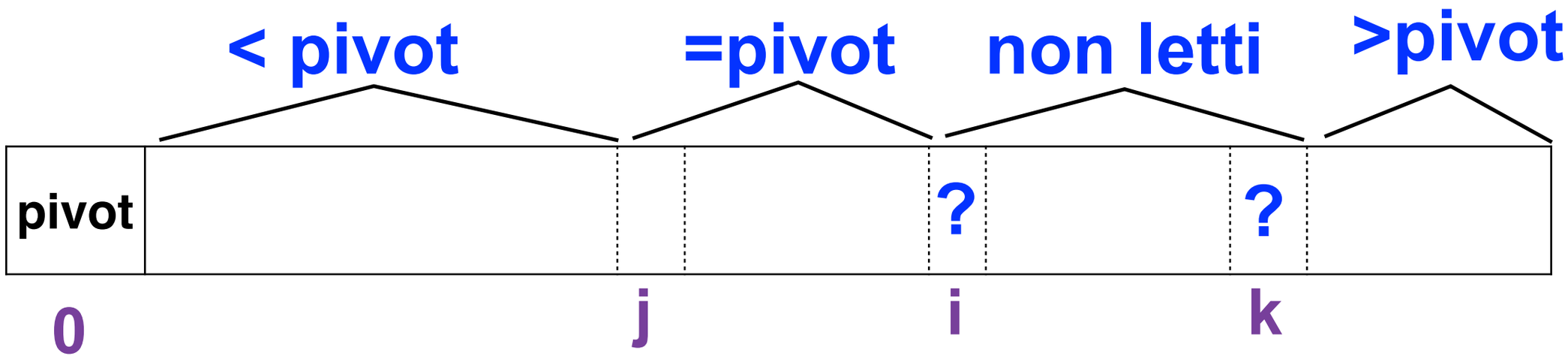
Immaginiamo di trovarci in un momento intermedio del calcolo, in cui una parte dell'array è stato già esaminato e i cui elementi quindi sono stati già correttamente sistemati.

Per descrivere questa situazione si devono usare tre indici, uno, i per scorrere gli elementi dell'array, uno che individui il confine tra i minori e gli uguali, j, e un altro tra i maggiori e la parte ancora da esaminare, k.



La tripartizione di un array

A[i] è l'elemento da considerare.

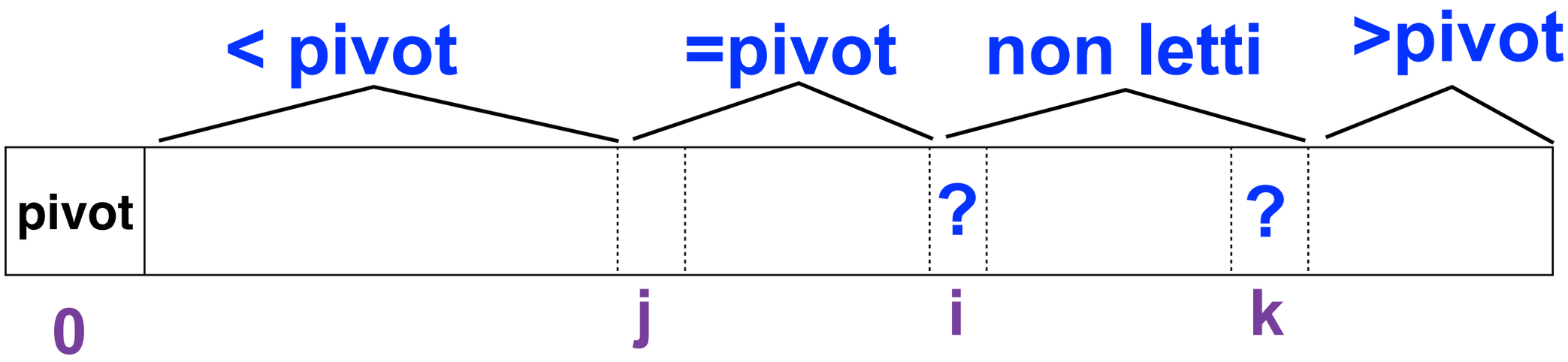


Se A[i] == pivot incrementa i

Se A[i] < pivot scambia A[j] con A[i] e incrementa i e j

La tripartizione di un array

A[i] è l'elemento da considerare.



Se A[i] > pivot scambia A[i] con A[k]

Si decrementa k, in modo da ripristinare la situazione di partenza e poter verificare A[i]

TriPartizione(A)

input: A è un array di n interi

output: dà in output due indici, i e j, tali che nell'array A i minori del pivot, che è A[0], sono in A[0:j-1], gli uguali in A[j:i-1] e i maggiori del pivot in A[i:A.length -1]

```
n = A.length, j = i = 1, k = n-1; pivot = A[0]
while(i <= k):
      { if (A[i] < pivot):
           scambia A[j] e A[i]
           j++
           i++
        else if (A[i] > pivot):
           scambia A[i] e A[k]
           k - -;
               else  i++
        }
scambia A[0] e A[j-1]
return i,j
```

Durante il ciclo l'indice di scorrimento dell'array è i, tutti gli elementi di indice da 1 a j-1 sono minori del pivot e quelli da j a i-1 sono uguali al pivot, mentre i maggiori sono tra k+1 e n-1.