Merge Step of MergeSort

Input: Sorted lists S_1 and S_2

Output: Sorted list S, combining the elements of S_1 and S_2

```
1 while (S_1 ≠ ∅ AND S_2 ≠ ∅)
2         if (first element of S_1 < first element of S_2 )
3           insert in S the first element of S_1
4           remove from S_1 the first element
5         else
6           insert in S the first element of S_2
7           remove from S_2 the first element
8         endif
9 endwhile
10 while (S_1 ≠ ∅)
11          insert in S the first element of S_1
12          remove from S_1 the first element
13 endwhile
14 while (S_2 ≠ ∅)
15          insert in S the first element of S_2
16          remove from S_2 the first element
17 endwhile
```

PARTITION(A,p,q,r)

Input: Array A, indices p and q with $p \leq q$

Output: By swapping elements of $A[p \ldots q]$, we have that all elements of $A[p \ldots r-1]$ are at most $A[r]$, and all elements of $A[r+1 \ldots q]$ are at least $A[r]$.

```
1  x ← A[p]
2  left ← q + 1
3  right ← p
4  while TRUE
5        /* Invariant: A[right] = x
6        /* Invariant: right ≤ left
7        /* Invariant: A[p] ≤ x, A[p + 1] ≤ x, ..., A[right] ≤ x
8        /* Invariant: A[left] ≥ x, A[left + 1] ≥ x, ... A[q] ≥ x
9        repeat left ← left − 1
10       until A[left] ≤ x
11       if left = right then return( r ← left)
12       else
13             A[right] ← A[left]
14             A[left] ← x
15       endif
16       /* Invariant: A[left] = x
17       /* Invariant: right ≤ left
18       /* Invariant: A[p] ≤ x, A[p + 1] ≤ x, ..., A[right] ≤ x
19       /* Invariant: A[left] ≥ x, A[left + 1] ≥ x, ... A[q] ≥ x
20       repeat right ← right + 1
21       until A[right] ≥ x
22       if left = right then return(r ← left)
23       else
24             A[left] ← A[right]
25             A[right] ← x
26       endif
27 endwhile
```