

# QUICK SORT

```
QuickSort (array, first, last)
    int pivotIndex
    if (first < last) {
        pivotIndex = partition(array, first, last);
        QuickSort (array, first, pivotIndex - 1);
        QuickSort (array, pivotIndex + 1, last);
    }
}
```

Diagram illustrating the partitioning step in Quick Sort:

- The initial range is from index 0 to 9.
- The pivot is chosen at index 6.
- The partitioning process results in two sub-arrays: one from index 0 to 5 (pivotIndex - 1) and another from index 7 to 9 (pivotIndex + 1).

```

int partition (array, first, last) {
    int pivot = array[first];
    int lastSI = first;
    for (firstUnknow = first + 1; firstUnknow <= last; firstUnknow++) {
        if (array[firstUnknow] < pivot) {
            ++lastSI;
            temp = array[firstUnknow];
            array[firstUnknow] = array[lastSI];
            array[lastSI] = temp;
        }
    }
    temp = array[first];
    array[first] = array[lastSI];
    array[lastSI] = temp;
    return lastSI;
}

```

0	1	2	3	4	5	6	7	8	9
8	33	18	15	22	10	42	75	72	55

