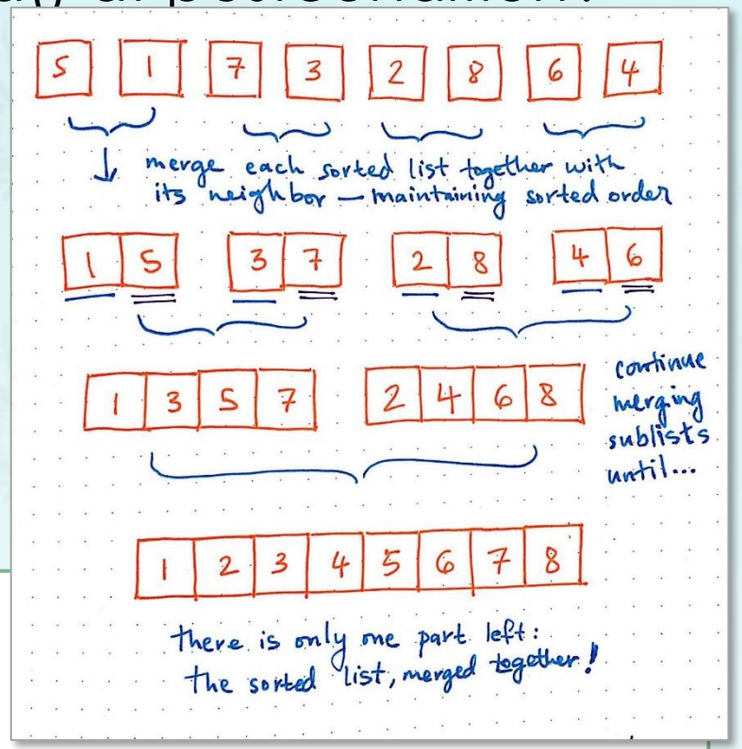
Mergesort: Quiz 1

1. Improvement by reducing the number of `merge()` function call.
   A hint and a solution for this problem are provided in the following pages.
2. How many times did you spare `merge()` calls for "MERGESORTEXAMPLE" case?
   - Total number of **merge()** calls without your improvement: _____
   - The number of **merge()** calls spared with your improvement: _____
3. Identify those sets of char array groups that `merge()` call was unnecessary.

Mergesort: Quiz 2

- In the figure, which elements are compared in isSorted() at postcondition?
- Why isSorted() checks only two elements?
  Is this enough?

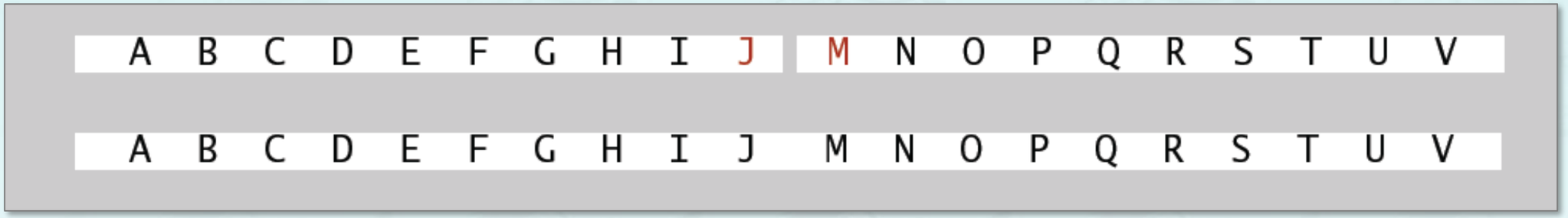



```
int isSorted(int *a, int i, int j){return a[i] <= a[j];}

void merge(int *a, char *aux, int lo, int mi, int hi) {
    assert(isSorted(a, lo,   mi));     // precondition: a[lo..mi]   sorted
    assert(isSorted(a, mi+1, hi));     // precondition: a[mi+1..hi] sorted
    for (int k = lo; k <= hi; k++) aux[k] = a[k];
     ……
    assert(isSorted(a, lo, hi));       // postcondition: a[lo..hi] sorted
}
```

Mergesort: Quiz Reference

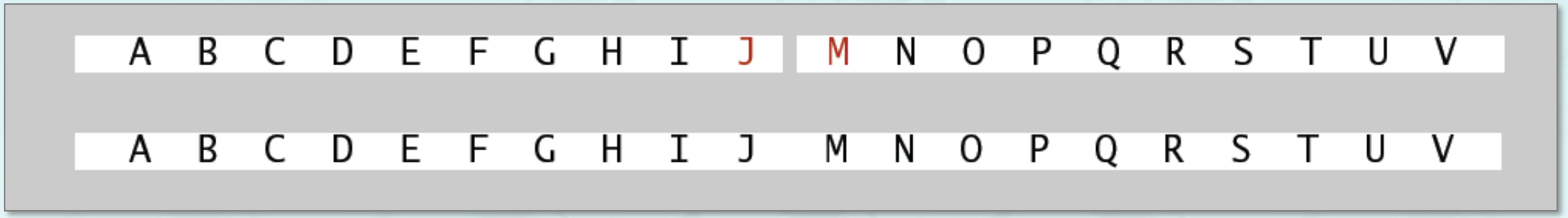
- **Hint:** Do not invoke "merge()" function **if two halves are already sorted..**
  - Is the biggest item in first half ≤ the smallest item in second half?
  - For example, the following case should not call merge() since `J <= M`.



정렬들에 관한 좋은 자료를 읽어 보길 적극 추천합니다.
영어: https://medium.com/basecs/making-sense-of-merge-sort-part-1-49649a143478
한글: https://gmlwjd9405.github.io/2018/05/08/algorithm-merge-sort.html

Mergesort: Quiz Reference

- **Hint:** Do not invoke "merge()" function **if two halves are already sorted..**
  - Is the biggest item in first half ≤ the smallest item in second half?
  - For example, the following case should not call merge() since `J <= M`.




```
void mergeSort(char *a, char *aux, int N, int lo, int hi) {
    if (hi <= lo)  return;
    int mi = lo + (hi - lo) / 2;
    mergeSort(a, aux, N, lo,     mi);
    mergeSort(a, aux, N, mi + 1, hi);
    if ( a[mi + 1] > a[mi] ) return;          // already sorted
    merge(a, aux, lo, mi, hi);
}
```