Consider the following sorting algorithm on a list of size $n = k(k + 1)/2$.


 Problem 1. Write the pseudo code. You may assume that you have available a bubble sort routine

\[
\text{BubbleSort}(A,p,r)
\]

which sorts the elements in $A[p, \ldots , r]$ (using bubble sort), and a merge routine

\[
\text{Merge}(A,p,q,r)
\]

which merges the sorted list $A[p, \ldots , q]$ with the sorted list $A[q + 1, \ldots , r]$ putting it into $A[p, \ldots , r]$.

 Problem 2.

(a) What is the number of comparisons for Phase 1 (the bubble sort phase) as a function of $k$? Show your work.

(b) What is the exact high order term for Phase 1 as a function of $n$?

(c) What is the (worst case) number of comparisons for Phase 2 (the merge phase) as a function of $k$? Show your work.

(d) What is the exact high order term for Phase 2 as a function of $n$?

(e) What is the exact high order term for the total number of comparisons as a function of $n$?

(f) How does this compare to the high order term of bubble sort (as a ratio)?

(g) How does this compare to the high order term of mergesort (as a ratio)?

 Problem 3.

(a) What is the average case number of comparisons for $n = 1$ ($k = 1$). Justify.

(b) What is the average case number of comparisons for $n = 3$ ($k = 2$). Justify.

(c) What is the average case number of comparisons for $n = 6$ ($k = 3$). Justify. (This part may not be fun, but it is very doable.)