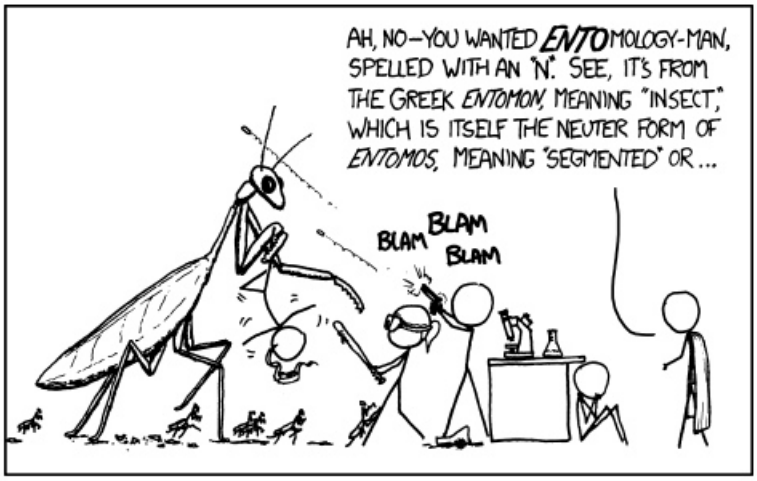
CS302 - Assignment 5

Due: Tuesday, Mar. 1 at the beginning of class
Hand-in method: paper



http://www.xkcd.com/1012/

For this assignment you must use latex to generate your work.

1. **[8 points]** Even $O(n \log n)$ may not be fast enough for certain cases where the data set is very large and/or when response time is very important. One way to solve this problem is to use multiple machines to sort the data.

   For the following sorting methods (Insertion-Sort, Merge-Sort, Quicksort and Counting-Sort), which algorithm is the most amenable to this situation? Why? What are the challenges for the others? Be clear, but concise.

2. **[12 points]** Consider the following sorting algorithm: sort the first two-thirds of the elements in the array, then sort the last two thirds of the array and finally sort the first two thirds again. Specifically:

TRIPLE-SORT(A, i, j)

```
1  if A[i] > A[j]
2          swap A[i] and A[j]
3  if i + 1 ≥ j
4          return
5  k ← ⌊(j − i + 1)/3⌋
6  TRIPLE-SORT(A, i, j − k) # sort the first two-thirds
7  TRIPLE-SORT(A, i + k, j)# sort the second two-thirds
8  TRIPLE-SORT(A, i, j − k) # sort the first two-thirds again
```

which you would call to sort A with TRIPLE-SORT$(A, 1, A.length)$.

(a) (4 points) Give an informal but convincing explanation of why the algorithm above is correct. The explanation does not need to be more than a few sentences, but be precise.

(b) (3 points) Describe the recurrence relation for the run-time of TRIPLE-SORT.

(c) (3 points) What is the run-time of TRIPLE-SORT (i.e. solve the recurrence)?

(d) (2 points) How does this algorithm compare to INSERTION-SORT, MERGE-SORT, QUICKSORT?