

210 ($n \times \log n$ sort) Write a program to sort a list. Execution time should be at most $n \times \log n$ where n is the length of the list.

After trying the question, scroll down to the solution.

§ There are many ways to sort in time $n \times \log n$. I'll do merge sort. Let the list variable be L . Let i , j , and k be natural variables. Define specifications S (for Sort) and T as follows.

$$S = (\forall a, b: 0..#L. a \leq b \Rightarrow L'a \leq L'b) \wedge \text{perm } L' L$$

$$T = (\forall a, b: i..k. a \leq b \Rightarrow L'a \leq L'b) \wedge \text{perm } (L'[i..k]) (L[i..k]) \\ \wedge L'[0..i] = L[0..i] \wedge L'[k..#L] = L[k..#L]$$

$$\text{perm } A B = \forall x. \phi(\exists i: 0..#A. A i = x) = \phi(\exists i: 0..#B. B i = x)$$

I have just realized that top-down mergesort (mergesort both halves of the list, then merge the two sorted halves) will require a stack of values, either as parameters (Chapter 5) or as an explicit stack (Chapter 7). So I'll try bottom-up mergesort (merge pairs of singles, then pairs of pairs, then pairs of 4s, and so on). UNFINISHED