George Lenz  
08/16/2018  
HW 8  
  
a)   
 array items  
 array bins  
 for item i …n  
 for item j to m (number of bins  
 if i fits in bin j, add it  
 if it does not j++  
 if j > m  
 open new bin and put in item  
 end loop for bins and go to next item  
 return length of bin array (i.e. amount of bins)  
  
 for first fit decreasing algorithm it will be the same, with the exception of a decreasing merge sort ran on the array of items first.  
  
The worst case scenario is that every item needs to go in a new bin, and since you have to loop through the bins each time you get a running time of O(n^2). Best case every item fits into one bin so you get omega(n).  
  
For the first fit decreasing the upper bound is the same but the lower bound is dominated by the sort which is omega(nlgn).  
  
c) The first algorithm generally performed better, but occasionally they were about the same. This makes sense because depending on the amount of bins that are needed, the algorithm will be dominated by the bin packing or the sort. If it’s dominated by the sort as in the lower bound, the decreasing algorithm will take slightly longer, however if the amount of bins is closer to the upper bound the algorithms will perform in a more similar time.