**Project 3 part 3 complete recursive implementation**

[Project 3 part 3 complete recursive implementation](https://fcps.blackboard.com/webapps/blackboard/content/listContent.jsp?course_id=_1516674_1&content_id=_43601203_1&mode=reset)

**Part 3: create a method part3() that**

  Create and submit an application for the recursive, O(n\*log^2 n) algorithm presented in class (a variation of the algorithm in the paper:  [closestPairRecursive.pdf](https://fcps.blackboard.com/bbcswebdav/pid-44360208-dt-content-rid-41301464_2/xid-41301464_2) [closestPairRecursive.pdf - Alternative Formats](https://fcps.blackboard.com/webapps/blackboard/content/listContent.jsp?course_id=_1516674_1&content_id=_43601203_1&mode=reset)

 ).

Before you start you should read the points from the file points.txt into 2 vectors(this part should not be timed). One vector will be used by part2 and the other vector will be used by part3 (you may have as one parameter to this methods a reference to the vector.Complete recursive algorithm:  
 a) Sort the vector based on the x coordinate; O(nlogn). This step is outside the recursive method and should be done only once. Next steps describe the recursive method.  
 b) Divide the vector into 2 parts and recur for each part (logical division preferred since is faster); O(1) since is sorted  
 c) If there are 3 or 2 points just simply return the minimum distance since it can be done in constant time O(1)  
 d) When the 2 recursive calls return with 2 minimum distances (let's say d1 and d2), calculate d = min(d1,d2)  
 e) create a strip of distance d to the left of the middle and d to the right of middle (middle value). Save all the points in the strip in a vector (let's call it vectorStrip, but you may use any name you wish) f) sort the vectorStrip by y coordinate O(nlogn) f) for each pointin vectorStrip calculate the distance to max of 15 following points to see if you can  find a smaller distance than d, then update the d. This is O(15n)=O(n)

 g) return the minimum distance you obtained and the 2 points that have that distance

In the main you should:  
a) call part2(...)  
b) call part3(..)  
c) display on the screen and in the results.txt the 2 points and minimum distance obtained for both approaches also the time to complete each approach (you may do this either by creating global variables or by making both part2 and part3 to return some result)