Nicholas DeNobrega Assignment 2 Write Up

1. Divide and Conquer Algorithm Description:
   1. For divide and conquer implementation, the array of points (p) is first sorted in ascending order by their x coordinate
   2. We then recursively divide p until we reach one of the two cases:
      1. 2 points are left meaning the closest pair is those two points
      2. 1 point left meaning there is no “pair”, so we don’t return a pair. Also this means there is no “distance” so we just return infinity, since when this distance is compared to the next level ups distance it will obviously be larger and overwritten by the next level ups distance
   3. For each recursive call, we:
      1. Find the closet pairs of the left and right halfs of the current points array, and their corresponding distances
      2. We then compare the distances of the two closest pairs and determine the minimum of the two
      3. A “strip” of all points are created, and within the strip we have points that are within a distance of dMin(minimum distance found from step ii) from the dividing line (mid point of current points array).
      4. Creating the strip allows us to reduce the number of comparisons between two points, since only nearby points are considered. Since the strip already contains points that are close in x-direction, sort the strip by y coordinates so that we can compare the points that are close in the y-direction.
      5. We then loop through the points in strip, and check if each pair of points distance between eachother in the y direction is less than dMin. If it is, we have found a pair of points that is within dMin in the x direction, and also less then dMin in the y direction, meaning we update our closet pair.
      6. This is done for every point in the strip, and the final minimum distance between the two closet points in the strip is compared with the original dMin. We return the minimum of the two as our new minimum distance
2. Brute Force Description:
   1. We take the array of points p, and double for loop through the array. Within the inner loop of this nested for loop, we compare the point at p[i], where I is index of outer loop, with p[j], where j is index of inner loop. We calculate the distance between the two, and update the closest pair and minimum distance if the distance between the two is less then current minimum.
3. Running Results:
   1. Overall
   2. For this case of input data, my Divide and Conquer algorithm ran in 5346 microseconds, and the Brute Force algorithm ran in 496113 microseconds
      1. n = 10000;
      2. for(i=0; i<n; i++)
      3. {
      4. p[i].x= n- i;
      5. p[i].y= n- i;
      6. }
   3. For this case of input data, my Divide and Conquer algorithm and Brute Force Algorithm ran the same as first case
      1. n = 10000;
      2. for(i=0; i<n; i++)
      3. {
      4. p[i].x= i\*i;
      5. p[i].y= i\*i;
      6. }