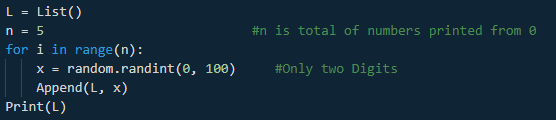
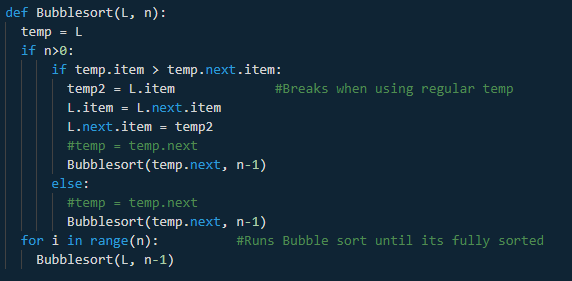
Lab 2

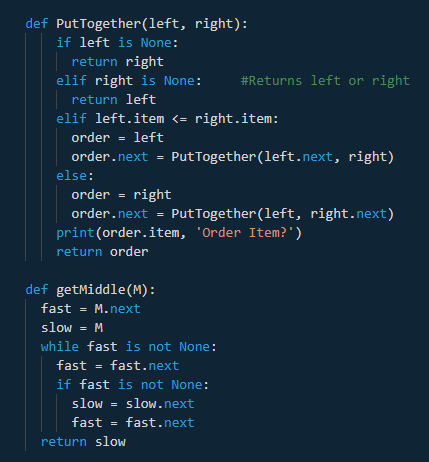
For this lab we need to create three different sorts, using Linked Lists (LL), being Bubble, Merge, and Quick sort, as well as an edited Quick sort. This edited quick sort will be set to use only one recursive call, rather than two. For our LL data, we will create a list of n length of random integers.

To create the list of random integers, I imported random, and created a loop that appended to the list n times. This list was then used to test/use for our sorting algorithms. I was able to get bubble sort to work by having an if statement that compares two adjacent values, and moves the larger one towards the end, and moves forward in the list. I use recursion until the end of the list is reached. I then use a loop to recursively call on the method, so that it calls it enough times until it is completely sorted. Bubble sort running time is O(n^2)





For the Merge Sort I separated the list in half until each piece was separated from one another. I then compared each piece and put them back together into a list until it was sorted. However, I ran into a problem I was unable to fix. Whenever I run merge sort, nothing less than the head of the initial List will make it into the sorted list. As such if the head was the smallest number, the list will be completely sorted. In the case of the head being the largest, the new list will only be that of the head. I was not sure if this was an error due to my initial separation, or when I was putting it back together. Running time for Merge sort should be O(n log(n))

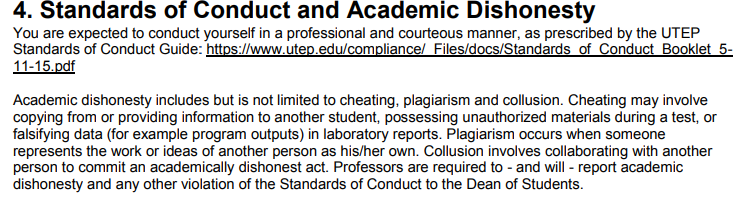
 

For quick sort, I was unable to get to work, and as such I was unable to start on the edited quick sort. My proposed solution for the initial quick sort was to create a pivot based off the value of the tail, and from there create two separate lists of those less than, and greater than the pivot. Rinse and repeat until all elements are isolated. From there, you put the elements back together in order. The average running time for Quick sort should be O(nlog(n))

This lab itself is late, and unfinished, but I would like to learn and understand where I failed to understand quick and merge sort. As for what I need to improve on, I need to practice tracing, so that I can better understand where I can find errors in code. I also need to better understand quick sort, and be able to devise different ways to come up with the same product. Lastly, I need to work on asking for help. I wasted too much time on this assignment trying to figure out how to make it work by myself, when asking for some guidance could have helped me better overall.

Github Link: <https://github.com/Penguinhedgehog/CS2302-PatrickBrannan/tree/master/CS2302/Lab%202>

Academic Honesty:



*-Patrick Brannan*