Vanessa Joyce Tan, 30556864

3.

I used the same code from Exercise 2 for creating a list in this task which means its time complexity is still in O(n) which is why it produces a linear graph. However, for the shaker sort function, I’ve used nested loops and all the loops depend on n, the length of the list. Therefore, it has a time complexity of O(n^2). That’s why the line graph produced for shaker sort rapidly increases as n becomes larger.

5.

The shape of both graphs is expected. Creating 100 random lists has the same time complexity (O(n)) as creating one random list, hence it still produces a linear line graph. The average time taken to sort those 100 random lists with shaker sort should be similar to the time taken for shaker sort in Exercise 3 part 3. Fortunately, in this case, they both have similar shape, only difference being that the average has a slightly lower time complexity than the shaker sort in part 3.

8.

Shape of graph produced for creating a list is linear as it is in O(n) time. It has a much lower time complexity than sorting the three different lists, as they have a complexity of O(n^2). The reversed list is the worst case for sorting so it should take the longest time to sort. The sorted list is the best case for sorting so it should take the shortest time sort which is why it has the lowest time complexity of the three sorting graphs. And sorting the random list is in between both the worst case and best case time complexity so it is where it should be.