For this assignment. The advance features are

1. Recursive Insertion Sort  
   -Meant to test understanding of Recursion.   
   Time complexity: O(n\*n)
2. Shell Sort  
   -Improves time complexity, by using “Insertion Sort” that results in less number of passes  
   Time complexity   
   best-case: O(n\* log n)   
   worst case: O(n\* log2n)
3. Comb Sort  
   -Time complexity can be compared to Quick Sort. Can be used for data containing Number or String, no worst case scenario, like Quick sort  
   best case: O(n log n)  
   average case: O(n^2/2^p) (p is a number of increment)   
   worst case: O(n^2)
4. Odd Even Sort

-Based on bubble sort, stable sort[maintains the order, doesn’t jump around]  
[Worst complexity](https://www.google.com/search?rlz=1C1ONGR_enSG983SG983&sxsrf=ALiCzsax7XMVeKq0TYA8gIkMXBM03fZ4rQ:1654366063597&q=odd%E2%80%93even+sort+worst+complexity&stick=H4sIAAAAAAAAAOPgE-LSz9U3MCoszC021DLOTrbSzy6IzynXL84vKsnMS49PzEnPL8osyci1Ks8vKi6JT87PLchJrcgsqYwvzkgsSk1ZxKqQn5LyqGFyallqngJImwJYpQJCJQDD9CKwZgAAAA&sa=X&ved=2ahUKEwisiLLssZT4AhW6TmwGHbm9CHwQ6BMoAHoECFgQAg): O( n^2)  
[Average complexity](https://www.google.com/search?rlz=1C1ONGR_enSG983SG983&sxsrf=ALiCzsax7XMVeKq0TYA8gIkMXBM03fZ4rQ:1654366063597&q=odd%E2%80%93even+sort+average+complexity&stick=H4sIAAAAAAAAAOPgE-LSz9U3MCoszC021DLNTrbSzy6IzynXL84vKsnMS49PzEnPL8osyci1SixLLUpMT41Pzs8tyEmtyCypjC_OSCxKTVnEqpSfkvKoYXJqWWqeAkijAlStAkItADkqH7hqAAAA&sa=X&ved=2ahUKEwisiLLssZT4AhW6TmwGHbm9CHwQ6BMoAHoECFoQAg): O( n^2)  
[Best complexity](https://www.google.com/search?rlz=1C1ONGR_enSG983SG983&sxsrf=ALiCzsax7XMVeKq0TYA8gIkMXBM03fZ4rQ:1654366063597&q=odd%E2%80%93even+sort+best+complexity&stick=H4sIAAAAAAAAAOPgE-LSz9U3MCoszC021DLKTrbSzy6IzynXL84vKsnMS49PzEnPL8osyci1SkotLolPzs8tyEmtyCypjC_OSCxKTVnEKp-fkvKoYXJqWWqeAkiXAkihAkIhAPX9gbxkAAAA&sa=X&ved=2ahUKEwisiLLssZT4AhW6TmwGHbm9CHwQ6BMoAHoECFEQAg): O(n)

1. Pancake Sort

-Flips array until the array is sorted. Tries to sort array with fewest reversals as possible, in-place)

Best: O(n)  
Worst: O(n\*n)

1. Cocktail Sort:

-Similar but better than Bubble Sort, fastest when elements are nearby their sorted position,

Best: O(n)  
Average&Worst: O(n\*n)

1. Gnome Sort

-In-place sort[does not require extra place to store data], combination of insertion and bubble sorts, once swapped, other unsorted pairs can be found easily.

Average: O(n^2)

1. Heap Sort

-A tree-based sort, consistent performance, very efficient, minimal memory usage, simple to understand

Best&Worst case: O(n log n)

The advanced sorts can be found in ‘advanced\_sorts\_functions.py’.

Additional Algorithms includes a Shift Cipher encryption, a programme shutdown timer, which can be found in ‘additional\_functions.py’. PASSWORD FOR LOG IN 🡪 ‘Windows10’[CASE\_SENSITIVE]