Braeden Pope
Algorithm Design
Sort Design

```
PROMPT for file name
READ file at file name into list

i_pivot = len(list) – 1

WHILE i_pivot >= 0
    i_largest = 0
    FOR i_check in range(len(list))

          IF list[i_check] > i_largest
                i_largest = list[i_check]

          IF i_largest > list[i_pivot]
                temp = list[i_pivot]
                list[i_pivot] = i_largest
                list[i_check] = temp

    i_pivot -= 1

PUT list on the screen
```

The algorithmic efficiency for this algorithm is O(log n), as after each iteration, the part of the list that gets checked (the unsorted side) becomes progressively smaller, and therefore quicker to check. This lines up with the O(log n) algorithmic efficiency.

Sort Design Trace Table

Test Case: [52, 15, 39, 26]

Line	i_pivot	i_largest	i_check	ist[i_check	ist[i_pivot]	temp
1	/	/	/	/	/	/
2	/	/	/	/	/	/
3	3	/	/	/	/	/
4	3	/	/	/	/	/
5	3	0	/	/	/	/
6	3	0	0	/	/	/
7	3	0	0	/	/	/
8	3	0	0	52	/	/
9	3	52	0	52	/	/
10	3	52	0	52	/	/
11	3	52	0	52	26	/
12	3	52	0	52	26	26
13	3	52	0	52	26	26
14	3	52	0	52	52	26
15	3	52	0	26	/	/
16	2	52	0	/	/	/
17	/	/	/	/	/	/
18	/	/	/	/	/	/