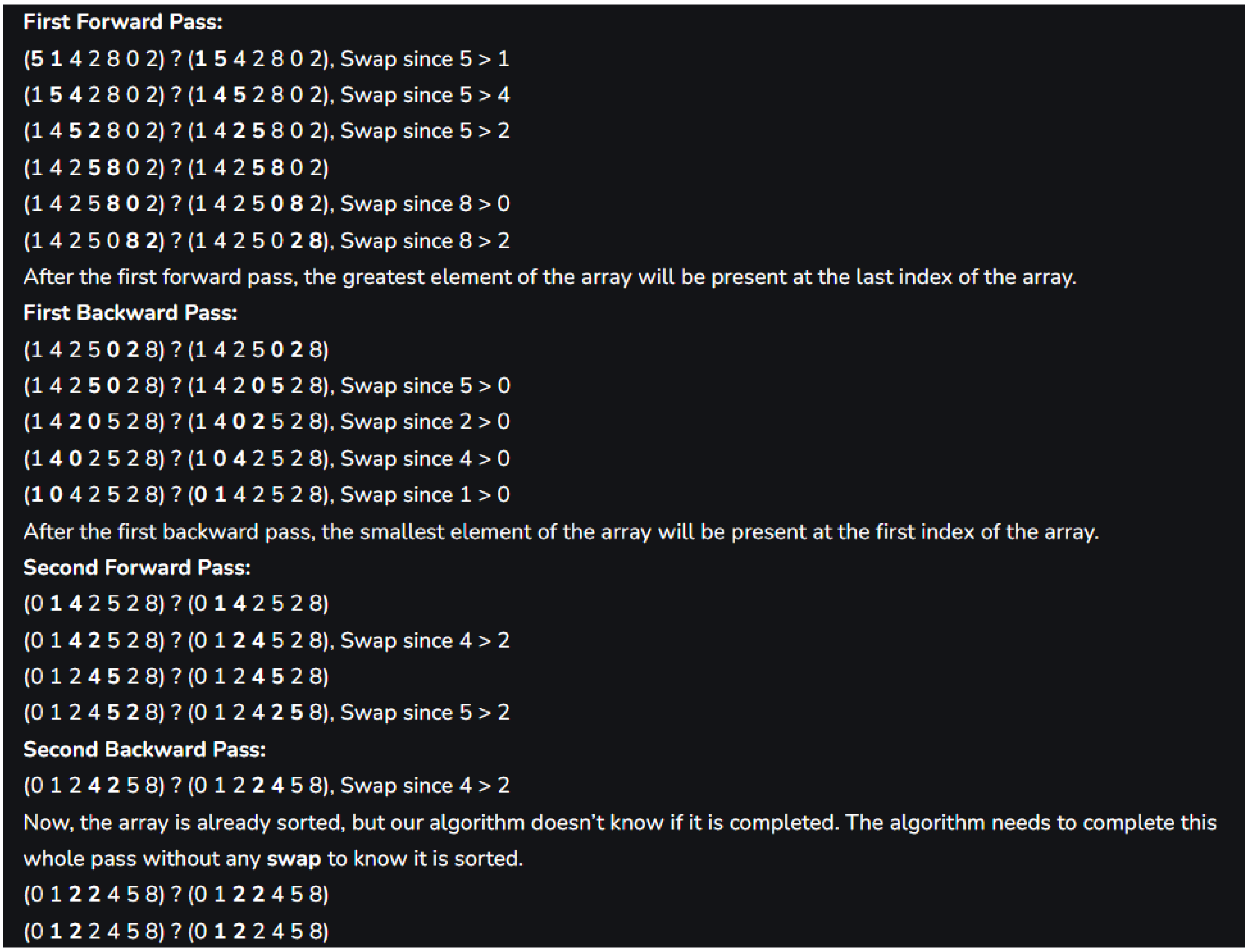
15. Popište princip ShakerSortu a HeapSortu

ShakerSort

- Bubblesort, který jde nejdříve zleva doprava (tedy úplně vpravo najdeme po této fázi největší prvek z pole) a poté jde opět zprava doleva (poslední, tedy největší a správně zařazený prvek, již přeskakuje)



First Forward Pass:

(**5 1** 4 2 8 0 2) ? (**1 5** 4 2 8 0 2), Swap since 5 > 1

(1 **5 4** 2 8 0 2) ? (1 **4 5** 2 8 0 2), Swap since 5 > 4

(1 4 **5 2** 8 0 2) ? (1 4 **2 5** 8 0 2), Swap since 5 > 2

(1 4 2 **5 8** 0 2) ? (1 4 2 **5 8** 0 2)

(1 4 2 5 **8 0** 2) ? (1 4 2 5 **0 8** 2), Swap since 8 > 0

(1 4 2 5 0 **8 2**) ? (1 4 2 5 0 **2 8**), Swap since 8 > 2

After the first forward pass, the greatest element of the array will be present at the last index of the array.

First Backward Pass:

(1 4 2 5 **0 2** 8) ? (1 4 2 5 **0 2** 8)

(1 4 2 **5 0** 2 8) ? (1 4 2 **0 5** 2 8), Swap since 5 > 0

(1 4 **2 0** 5 2 8) ? (1 4 **0 2** 5 2 8), Swap since 2 > 0

(1 **4 0** 2 5 2 8) ? (1 **0 4** 2 5 2 8), Swap since 4 > 0

(**1 0** 4 2 5 2 8) ? (**0 1** 4 2 5 2 8), Swap since 1 > 0

After the first backward pass, the smallest element of the array will be present at the first index of the array.

Second Forward Pass:

(0 **1 4** 2 5 2 8) ? (0 **1 4** 2 5 2 8)

(0 1 **4 2** 5 2 8) ? (0 1 **2 4** 5 2 8), Swap since 4 > 2

(0 1 2 **4 5** 2 8) ? (0 1 2 **4 5** 2 8)

(0 1 2 4 **5 2** 8) ? (0 1 2 4 **2 5** 8), Swap since 5 > 2

Second Backward Pass:

(0 1 2 **4 2** 5 8) ? (0 1 2 **2 4** 5 8), Swap since 4 > 2

Now, the array is already sorted, but our algorithm doesn't know if it is completed. The algorithm needs to complete this whole pass without any **swap** to know it is sorted.

(0 1 **2 2** 4 5 8) ? (0 1 **2 2** 4 5 8)

(0 **1 2** 2 4 5 8) ? (0 **1 2** 2 4 5 8)

HeapSort

Zde jsem to pochopil: https://www.geeksforgeeks.org/heap-sort/