

1 Insertion

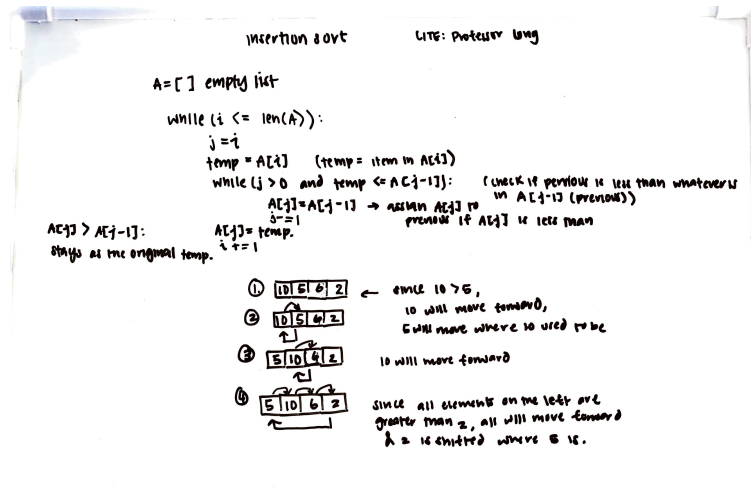


Figure 1: Insertion Sort

This is the most basic sorting, by going through each element in the list to check whether the current element is larger or smaller than the next element. Sudo code was provided by Professor Long.

2 Shell Sort

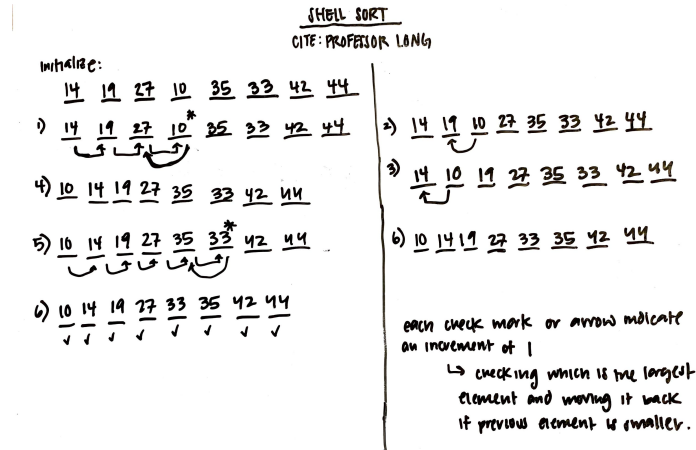


Figure 2: Shell sort

Shell sort relies on gaps to sort a certain number of elements. Then the gap decrease each time in a for loop. If the gap reaches to 1, it looks like it's doing insertion sort.

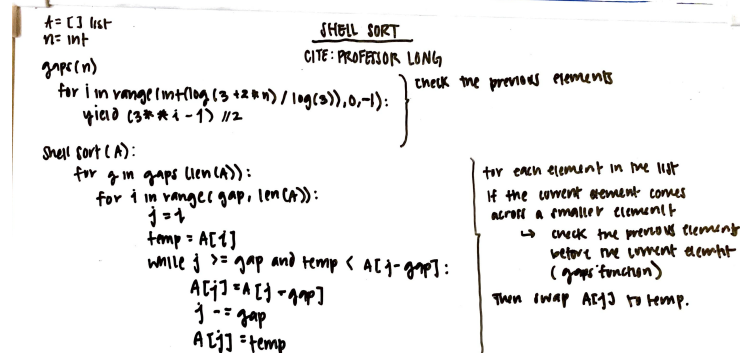


Figure 3: The sudo code is provided by Professor Long.

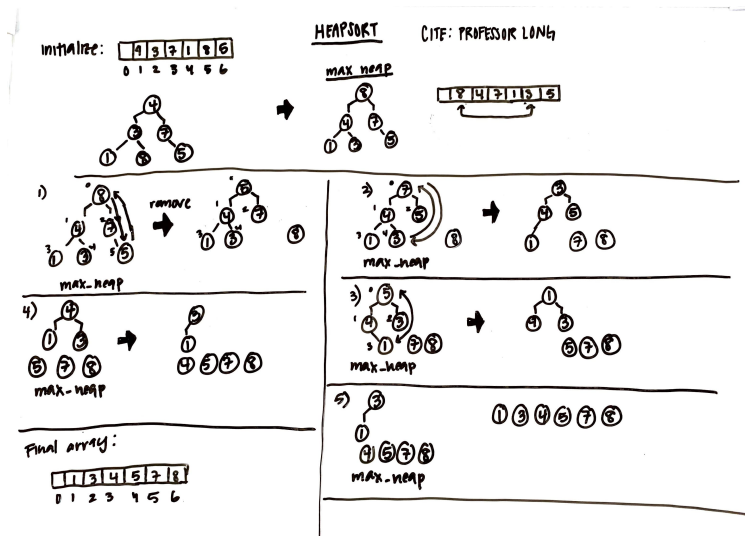


Figure 4: Heap Sort

Heap sort makes sense in a tree (kinda like a family tree). Max heap finds the largest parent element while the min heap finds the smallest parent element. The parent will always be the largest than the child elements.

HEAPSORT CITE: PROFESSOR LONG

Forming a heap:

```

A = list
first = int
last = int

max_heap(A, first, last):
    left = 2 * first
    right = left + 1
    if right <= last AND A[left-1] > A[right-1]:
        return right
    return left

```

getting the maximum number in the list as a max-heap

```

fix_heap(A, first, last):
    found = false
    init = first
    heap = max_heap(A, first, last):
    while init <= last // 2 AND NOT found:
        if A[init-1] < A[heap-1]:
            A[init-1], A[heap-1] = A[heap-1], A[init-1]
            init = heap
            heap = max_heap(A, init, last)
        else:
            found = true

```

go through the list and compare each one in the list to check if the max-heap is larger than the last in the list

Figure 5: But, the sorting creates a new list with the sorted elements. (Sudo code was provided by Professor Long)