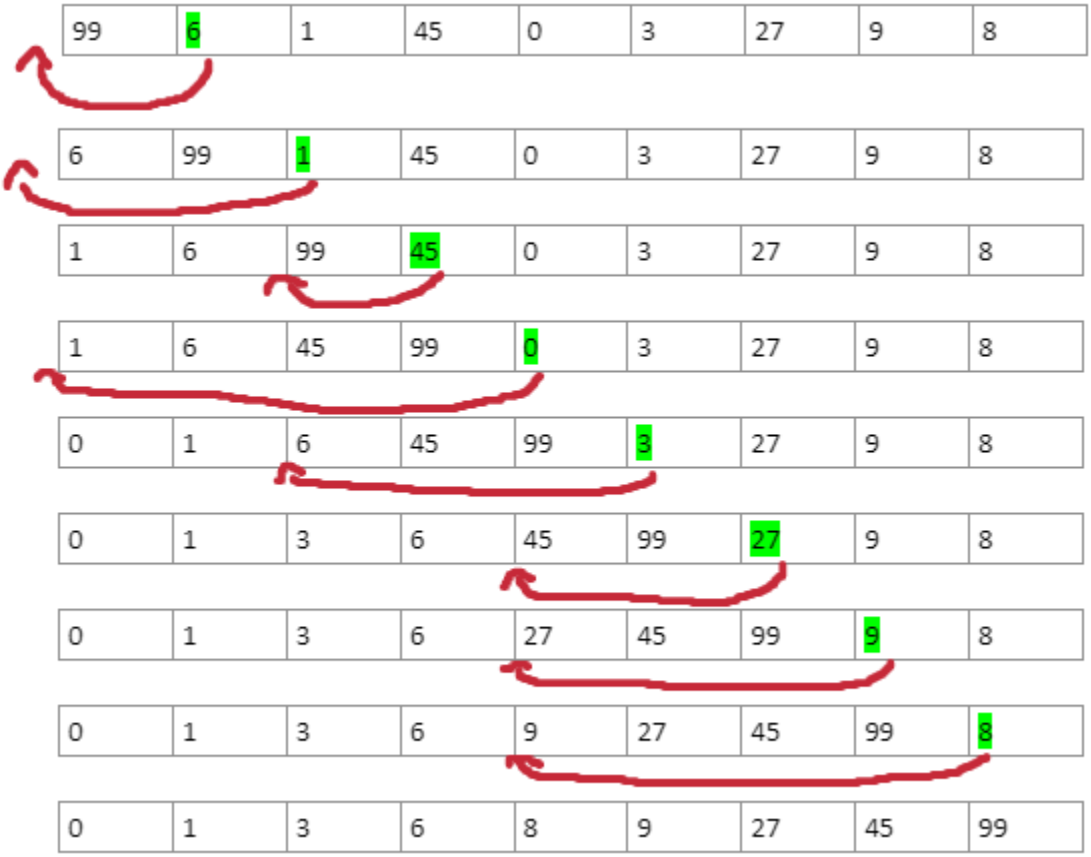
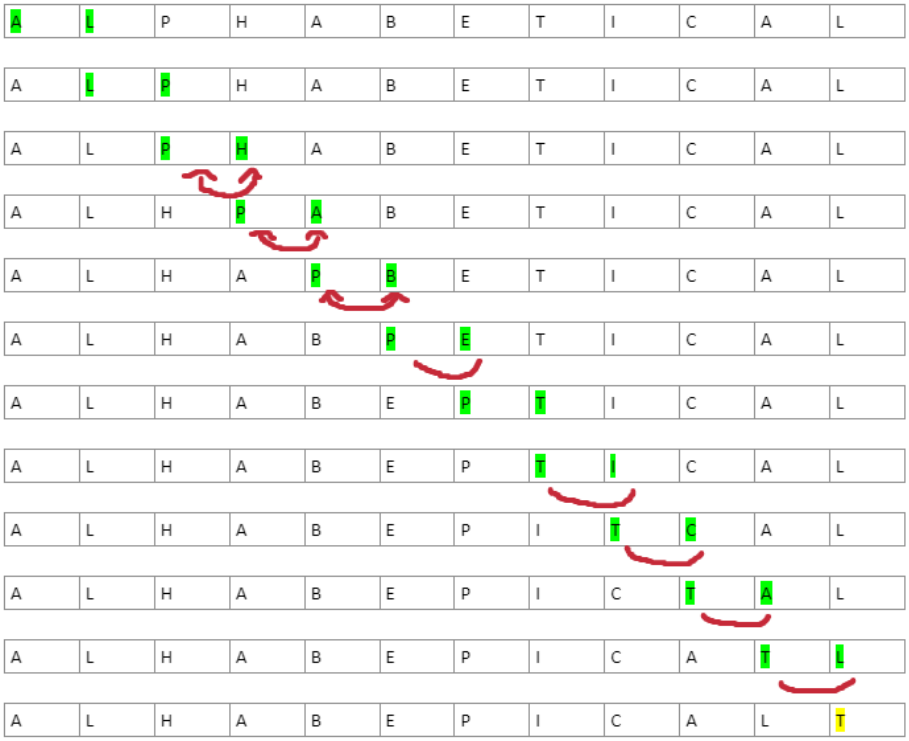
#1 insertion sort:

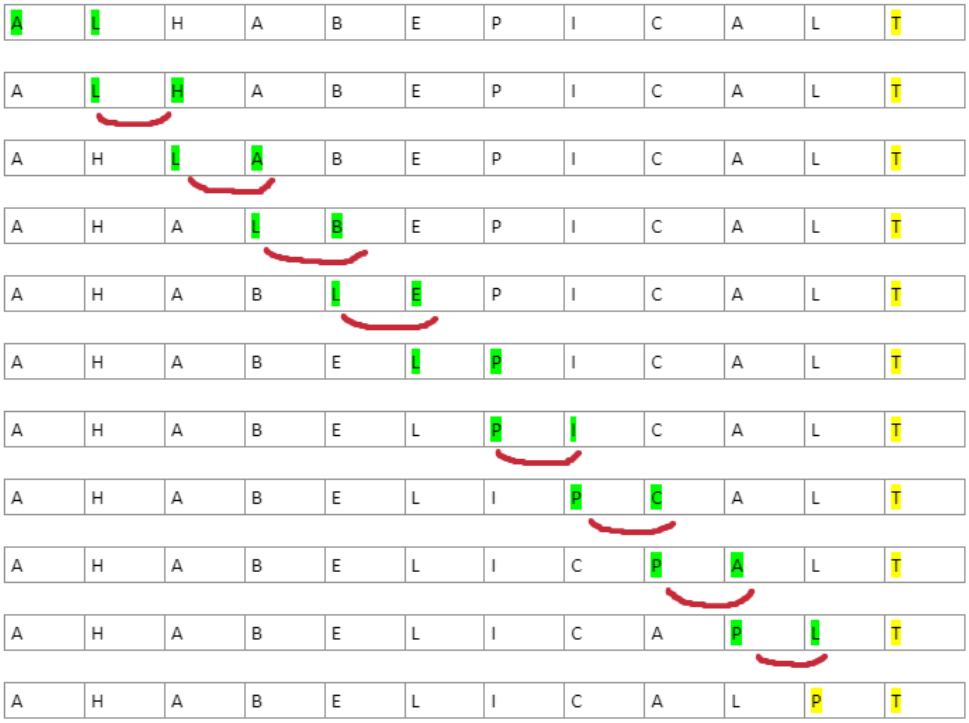


#2 bubble sort

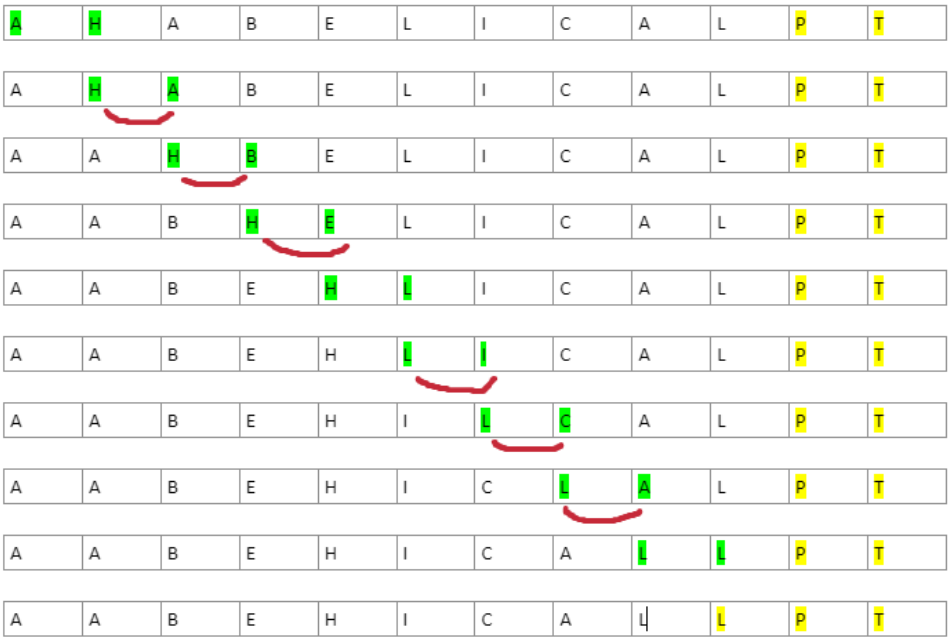
Pass 1:



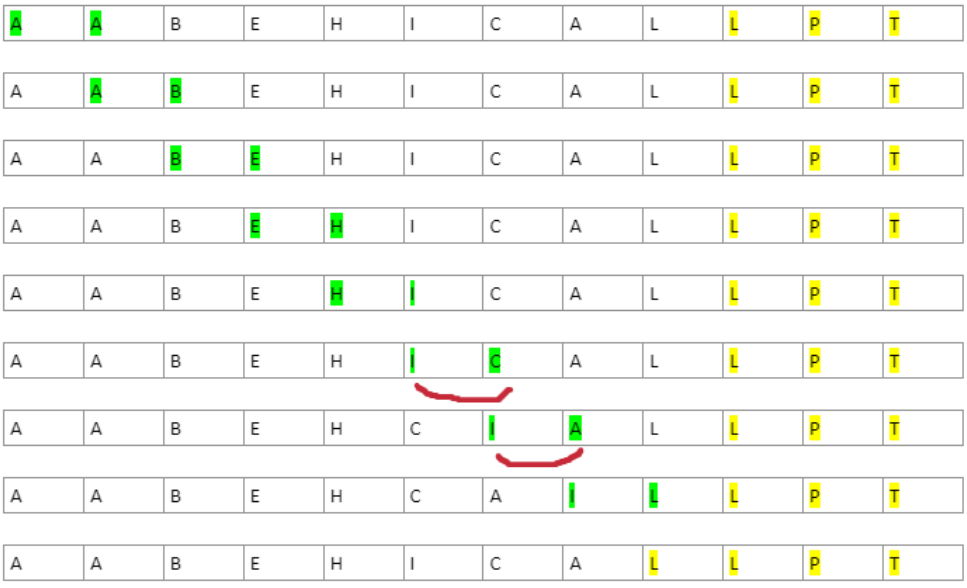
Pass 2:



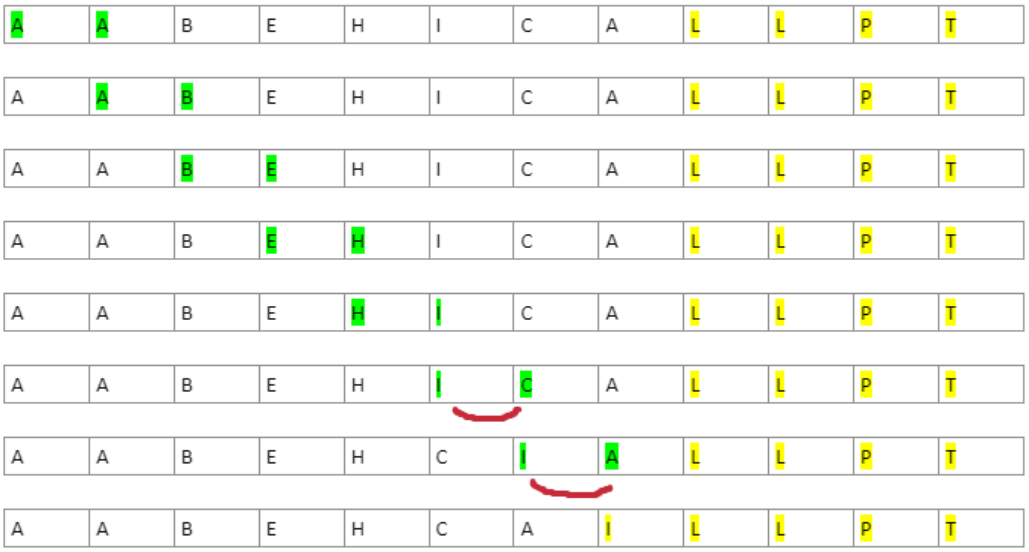
Pass 3:


A H A B E L I C A L P T
A H A B E L I C A L P T
A A H B E L I C A L P T
A A B H E L I C A L P T
A A B E H L I C A L P T
A A B E H L I C A L P T
A A B E H I L C A L P T
A A B E H I C L A L P T
A A B E H I C A L L P T
A A B E H I C A L L P T

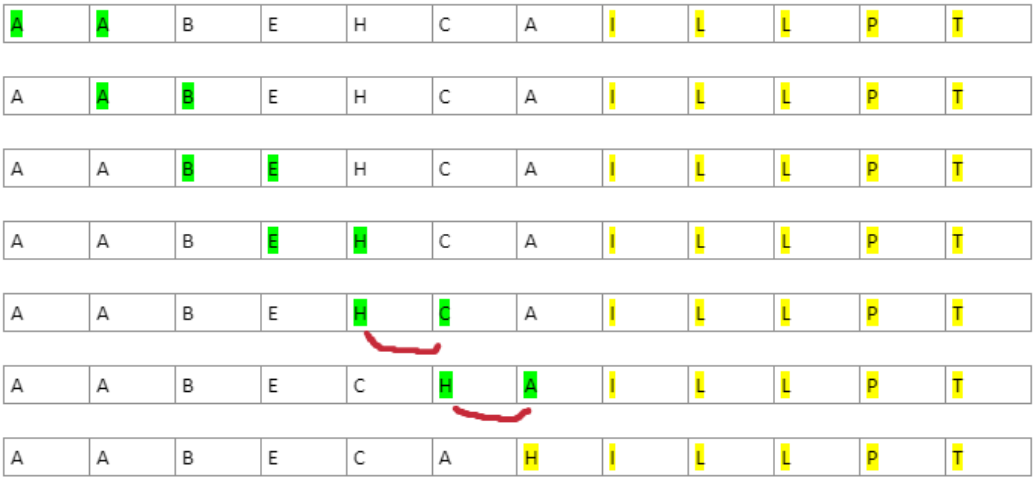
Pass 4:


A A B E H I C A L L P T
A A B E H I C A L L P T
A A B E H I C A L L P T
A A B E H I C A L L P T
A A B E H I C A L L P T
A A B E H I C A L L P T
A A B E H C I A L L P T
A A B E H C A I L L P T
A A B E H I C A L L P T

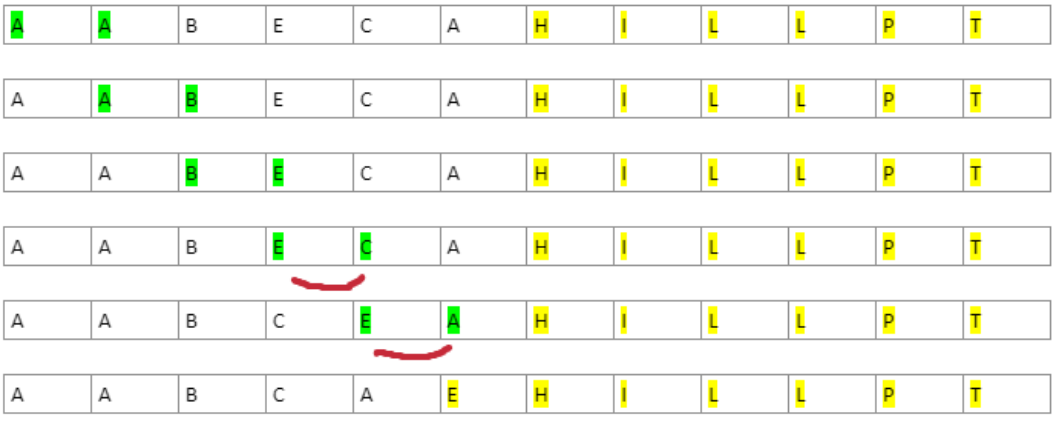
Pass 5:



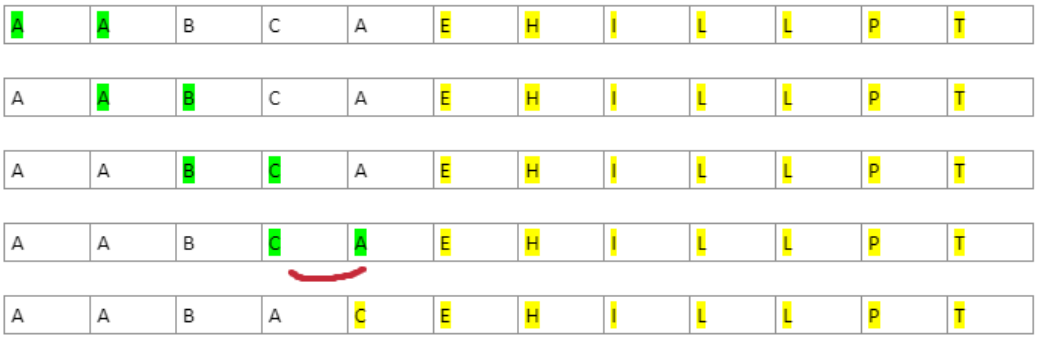
Pass 6:



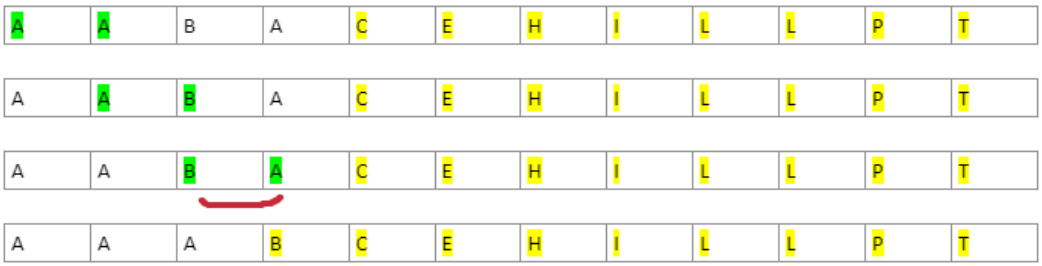
Pass 7:



Pass 8:

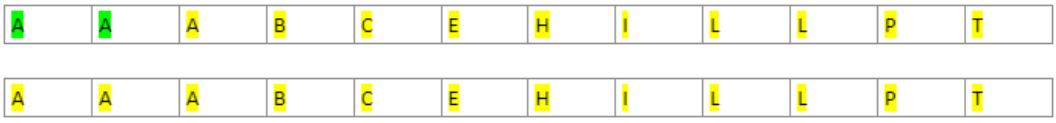


Pass 9:



Pass 10:

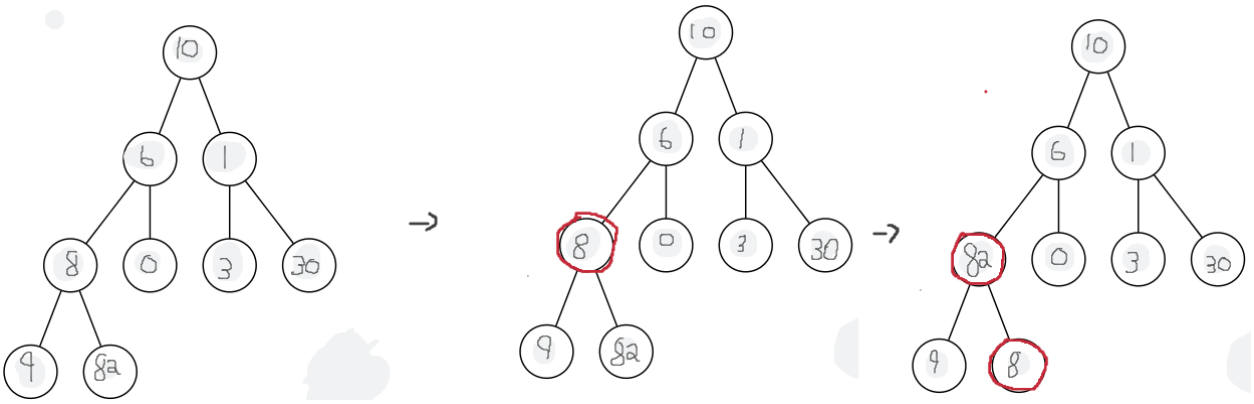
Pass 11:

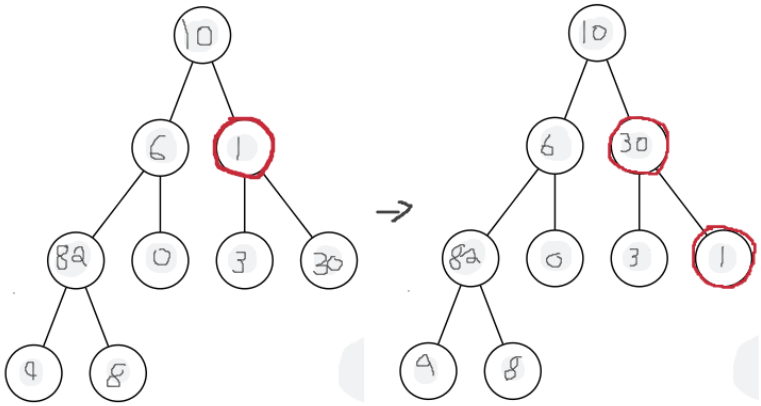


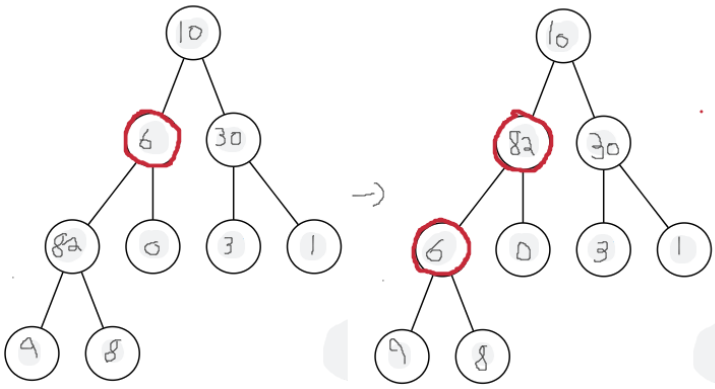
#3
O(n^2)

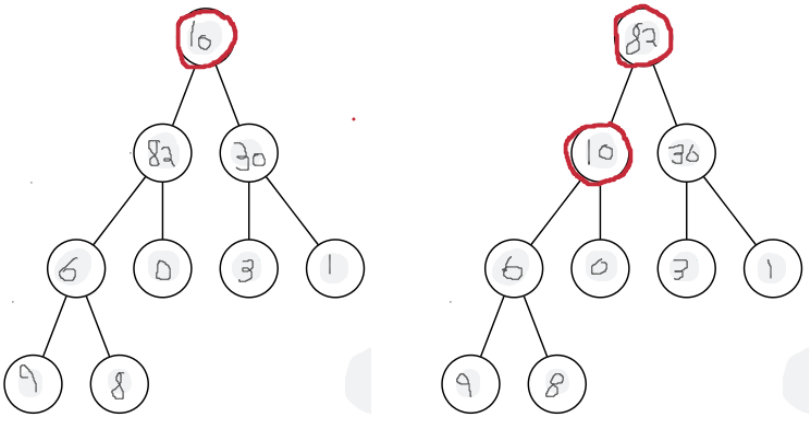
#4

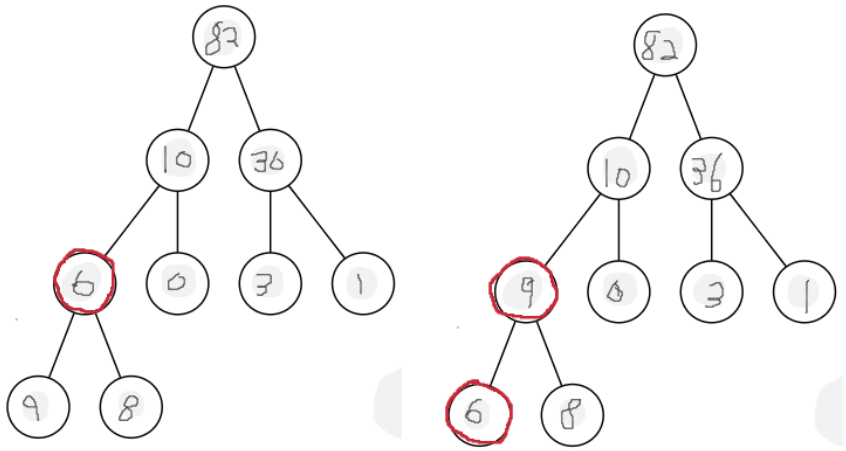
Building initial heap / running heapify():



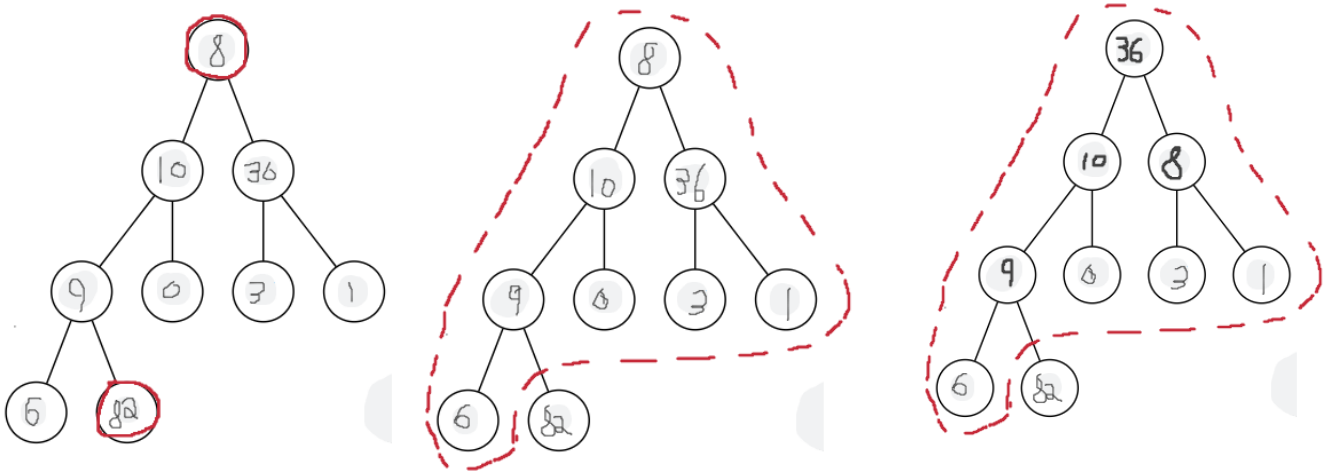




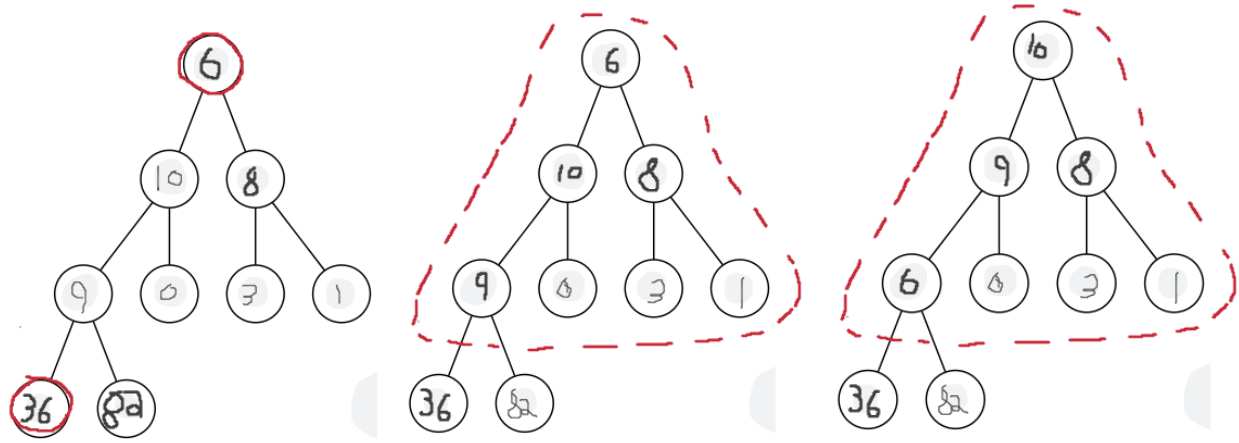




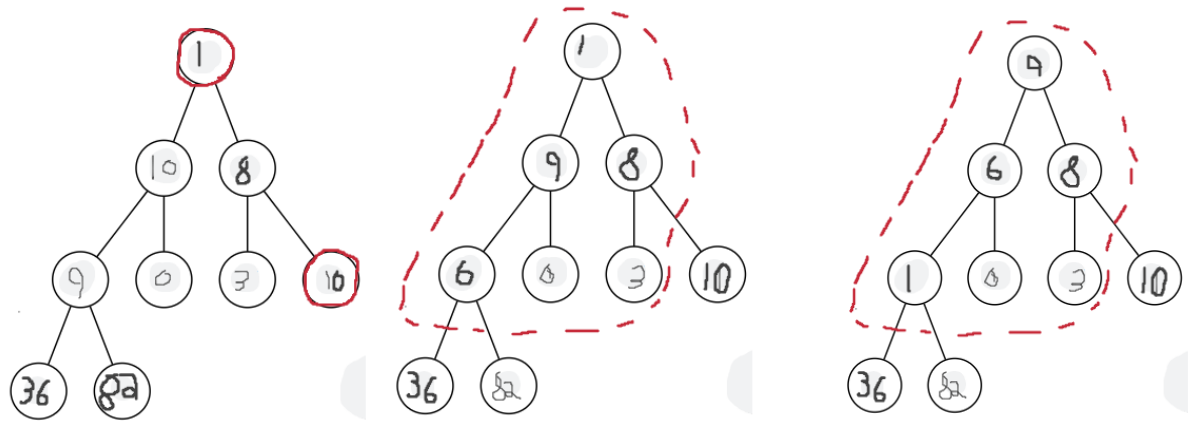
Heapsort:



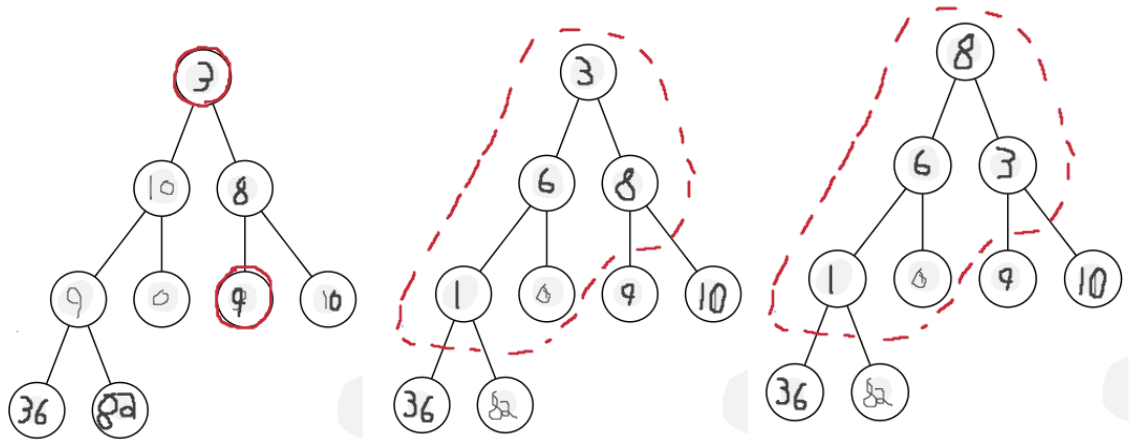
Swap first and last value, last value now sorted, then run heapify() on unsorted values



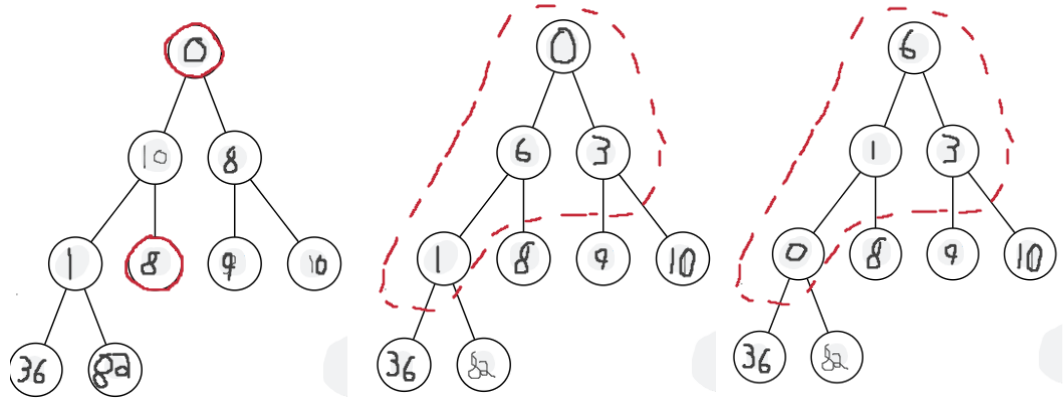
Again:



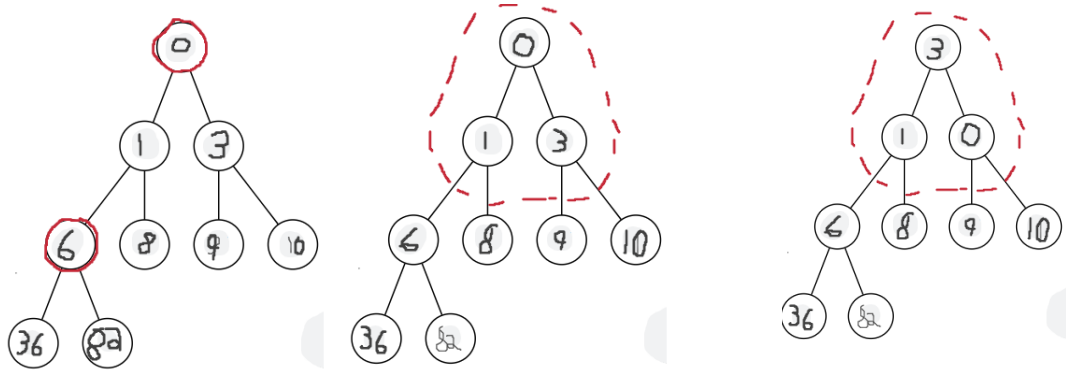
Again:



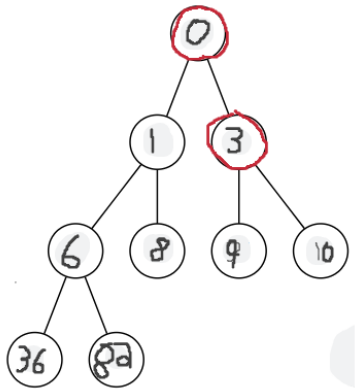
Once more:



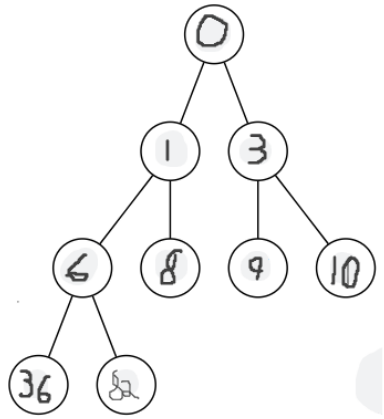
Again:



One final time:



And finally it is sorted:



#5

a.

- The root node is A[1]
–Node i is A[i]
–The parent of node i is A[i/3]
–The left child of node i is A[3i]
–The middle child of node i is A[3i + 1]
- The right child of node i is A[3i+2]

b. Log3(n)

c.

```
EXTRACT-MAX(A) {
        if (heapSize < 1){
                Error
        }
        Max = A[1]
        A[1] = A[heapSize[A]]
        heapSize[A]--
```

```
        heapify(A,1)
        Return Max

}
```

Running time: O(log3(n))

d.

```
INSERT(A, item) {
        heapSize[A]++
        i = heapSize[A]
        A[i] = -∞
        HeapChangeKey(A, i, item)
}
```

Running time: O(log3(n))

e.

```
INCREASE-KEY(A, i, k) {
        if( k < A[i]) {
                Error
        }
        Else {
                A[i] = k
                While (i > 1 && A[parent(i)] < A[i]) {
                        swap(A[i], A[parent(i)]
                }
        }
}
```

Running Time: O(log3(n))