**Array Query**

**Kadane’ s Algorithm**

1. Subarray operation
   1. Sum
   2. Minimum or max
2. Efficiency O(n)

**Segment Tree**

* N numbers there are
  + 2n -1 nodes
  + O(n) efficiency (not sure)
  + Tree height is log n
* Query and Update
  + Query time is log n
  + Query time number of nodes traversed is 2\*log n (here log n is the height of the tree and at any level 2 nodes are traversed)
  + Deletion or addition of nodes are not allowed

**Lookup Table**

* Size depends nature of problem
* zBut the query time has to be O (1)

**Sparse Table**

* Query time O (1)
* Space Efficiency is n log n
* Build Time is n log n.

**Square root decomposition (MO’s Algorithm)**

* Space complexity sqroot(n)
* Processing Time is sqroot(n) (3 sqroot(n) don’t know how)
* Not going to work with dynamic queries

**Fenwick Tree**

* Space efficiency is O(n)
* Query is log n

**Notes:** Merge Sort Tree, Persistent Tree, 2-d Fenwick Tree