Midterm Study List

1. Algorithmic analysis (loops, recursion)
2. Strings
   1. Boyer Moore
   2. LCS
   3. Finite state machines
3. Number theoretic algorithms
   1. gcd
   2. lcm(a,b) = (a\* b)/ gcd(a,b)
   3. power,powermod
   4. brute force trial division (O(n))
   5. brute force trial division O(√n)
   6. Eratosthenes sieve
   7. Fermat
      1. Carmichael number
      2. Probabilistic
4. Sorting
   1. insertion sort
   2. quick sort
   3. heap sort
   4. merge sort
   5. fischer-Yates
5. Searching
   1. binary
   2. bisection
   3. golden mean
6. Lists
   1. GrowArray
      1. insertion
      2. deletion
      3. linear traversal
      4. random access traversal
   2. NO LINKED LIST! (Final) LinkedList
      1. insertion
      2. deletion
      3. linear traversal
      4. random access traversal

O(n ) (why???)

for (int i = 1; i < n; i \*= 2) //O(log n) = 1, 2, 4, 8, 16, …, n/4, n/2, n

for (int j = 1; j <= i; j++)

;

the inner loop goes up to the outer value, so it is:

1 + 2 + 4 + … + n/4 + n/2 + n

factor out an n

(1/n + 2/n + 4/ + … + 1/4 + 1/2 + 1) n

½ + ¼ + ⅛ + …. 1/n = 1

(1/n + 2/n + 4/ + … + 1/4 + 1/2 + 1) n = 2n = O(n)

//O(n log n)

for (int i = 1; i < n; i \*= 2) //O(log n) = 1, 2, 4, 8, 16, …, n/4, n/2, n

for (int j = 1; j <= n; j++) //O(n)

;

for (int i = 1; i < n; i++) //O(n) = 1, 2, 4, 8, 16, …, n/4, n/2, n

for (int j = 1; j <= i; j++) //O(n) first one is 1..1 last one is 1..n avg is 1..n/2

;

//O(n log n)

for (int j = 1; j <= n; j++) //O(n)

for (int i = 1; i < j; i \*= 2) // log (1) + log(2) + log (3) + … log(n-1) + log(n) = O(logn)

;

Where does the doubling algorithm come up? In lists where the grow method is called to double the list size every time

ArrayList a = new ArrayList(n);

for (int i = 0; i < n; i++)

a.add(i);

ArrayList a = new ArrayList(); // defaults to size 10

for (int i = 0; i < n; i++)

a.add(i);

// twice as slow

class GrowArray {  
 private int[] data;

public void add(int v) {  
 int[] old = data;

data = new int[data.length+ 1];

System.arraycopy(old, 0, data, 0, old.length);

data[old.length] = v;

}