Let x and L be LinkedList Nodes.

Analyzing append

```
append(x, L) {
   Node curr = L;
   while (curr != null && curr.next != null) {
      curr = curr.next;
   }
   curr.next = x;
}
```

$O(n)$

LinkedList Reversal

```
reverse(L) {
   if (L == null) {
      return null;
   }
   else if (L.next == null) {
      return L;
   } else {
      Node front = L;
      Node rest = L.next;
      L.next = null;

      Node restReversed = reverse(rest);
      append(front, restReversed);
   }
}
```

$$T(n) = \begin{cases} d_0 & \text{if } n = 0 \text{ or } 1 \\ c_0 + c_1 n + T(n-1) & \text{otherwise} \end{cases}$$

$$\begin{aligned} T(n) &= (c_0 + c_1 n) + T(n-1) \\ &= (c_0 + c_1 n) + (c_0 + c_1(n-1)) + T(n-2) \\ &= (c_0 + c_1 n) + (c_0 + c_1(n-1)) + (c_0 + c_1(n-2)) + \ldots + (c_0 + c_1(1)) + d_0 \end{aligned}$$

$$= \sum_{i=1}^{n} (c_0 + c_1 i) + d_0$$

$$= n \cdot c_0 + c_1 \sum_{i=1}^{n} i + d_0$$

$$= n \cdot c_0 + c_1 \left(\frac{n(n+1)}{2} \right) + d_0 = O(n^2)$$

```
int sum(int[] arr){
    return help(arr,0,arr.length);
}
int help(int[] arr, int lo, int hi) {
    if(lo==hi)      return 0;
    if(lo==hi-1)    return arr[lo];
    int mid = (hi+lo)/2;
    return help(arr,lo,mid) + help(arr,mid,hi);
}
```

C_3

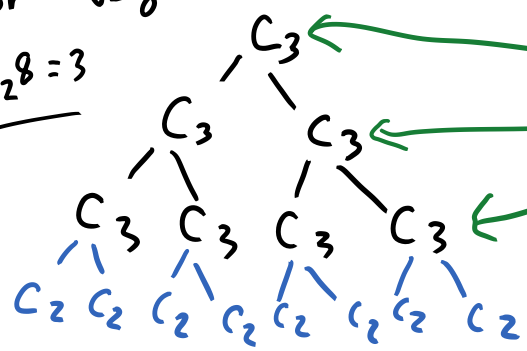
$$\boxed{\begin{aligned} T(0) &= C_1 \\ T(1) &= C_2 \\ T(N) &= C_3 + 2 \cdot T\left(\tfrac{N}{2}\right) \end{aligned}}$$

C_3

$T\left(\frac{N}{2}\right)$ $T\left(\frac{N}{2}\right)$

For $N=8$

$\log_2 8 = 3$



"level"	# nodes at this level	Total Work at this level
0	1	$1 \cdot C_3$
1	2^1	$2\ C_3$
2	2^2	$4 \cdot C_3$
i	2^i	$2^i \cdot C_3$

$$T(N) = \sum_{i=0}^{(\log_2 N)-1} 2^i C_3 + C_2 \cdot N$$

$$= C_3 \sum_{i=0}^{(\log_2 N)-1} 2^i + C_2 \cdot N$$

$$= C_3 \cdot \left(2^{\log_2 N} - 1\right) + C_2 \cdot N$$

$$= C_3 \cdot (N-1) + C_2 \cdot N = C_3 \cdot N - C_3 + C_2 \cdot N$$

$$= O(N)$$

$$\sum_{i=0}^{k} 2^i = 2^{k+1} - 1$$