**Linked List**

**Singly Linked List**

**prepend** O(1) – add item to front of list

**append** O(1) – add item to end of list

**lookup**  O(n) – we need to traverse the list from head to tail

**insert** O(n) – we need to traverse the list from head to tail

**delete**  O(n) – we need to traverse the list from head to tail

**Doubly Linked List**

Same as a singly linked list except it has a pointer to the previous node as well.

**prepend** O(1) – add item to front of list

**append** O(1) – add item to end of list

**lookup**  O(n) – it is really O(n/2) because we can start from head or tail and if we know which half of the list to search for, then we can choose which half is optimal for searching.

**insert** O(n) – we need to traverse the list from head to tail

**delete**  O(n) – we need to traverse the list from head to tail

**Singly linked list is good for:**

* insert
* delete
* uses less memory because there is no previous property

**Doubly linked list**

* can be traversed from head or tail
* uses more memory because there is a previous property