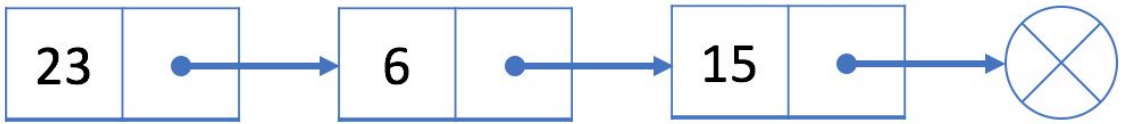
Atom #3

Linked Lists

Linked List

A linked list is a linear data structure, in which the elements are not stored at contiguous memory locations. The elements in a linked list are linked using pointers as shown in the below image:



Representation

A linked list is represented by a pointer to the first node of the linked list. The first node is called head. If the linked list is empty, then value of head is NULL.

Each node in a list consists of at least two parts:

1) data

2) Pointer (Or Reference) to the next node

```
#include <bits/stdc++.h>

using namespace std;

struct SinglyListNode {

    int val;

    SinglyListNode *next;

    SinglyListNode(int x) : val(x), next(NULL) {}

};
```

Linked Lists vs Arrays

Linked lists are preferable over arrays when:

1. you need constant-time insertions/deletions from the list (such as in real-time computing where time predictability is absolutely critical)
2. you don't know how many items will be in the list. With arrays, you may need to re-declare and copy memory if the array grows too big
3. you don't need random access to any elements
4. you want to be able to insert items in the middle of the list (such as a priority queue)

Arrays are preferable when:

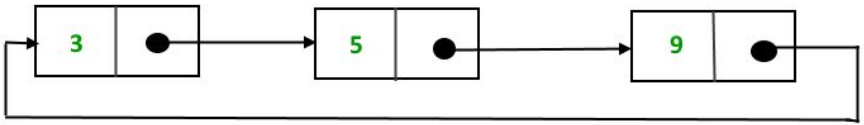
1. you need indexed/random access to elements
2. you know the number of elements in the array ahead of time so that you can allocate the correct amount of memory for the array
3. you need speed when iterating through all the elements in sequence. You can use pointer math on the array to access each element, whereas you need to lookup the node based on the pointer for each element in linked list, which may result in page faults which may result in performance hits.
4. memory is a concern. Filled arrays take up less memory than linked lists. Each element in the array is just the data. Each linked list node requires the data as well as one (or more) pointers to the other elements in the linked list.

Type of Linked Lists

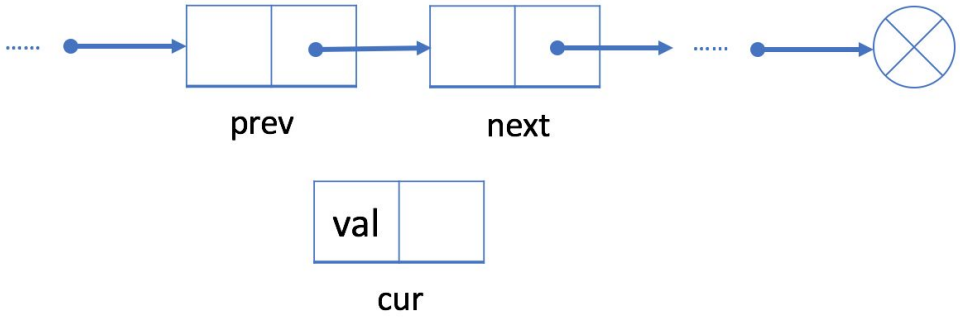
- SINGLY LINKED LIST

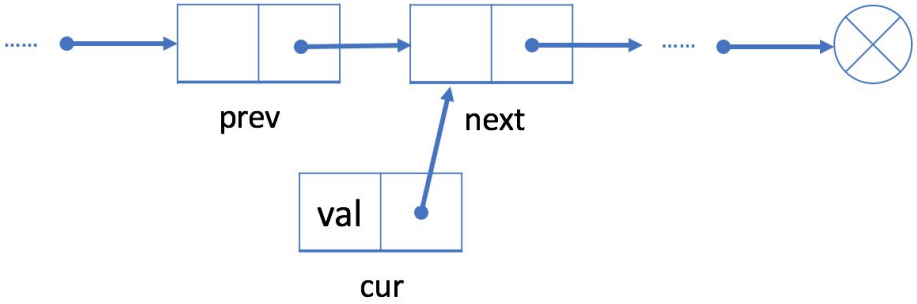
- DOUBLY LINKED LIST

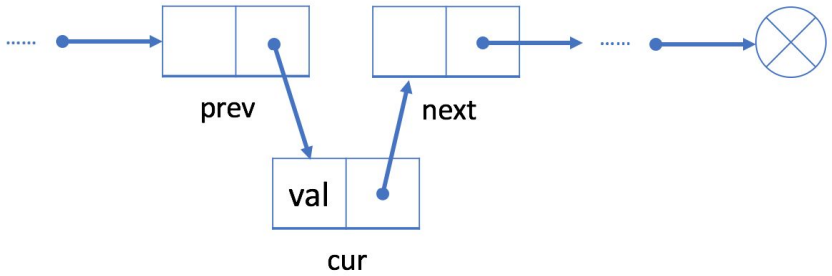
- CIRCULAR LINKED LIST



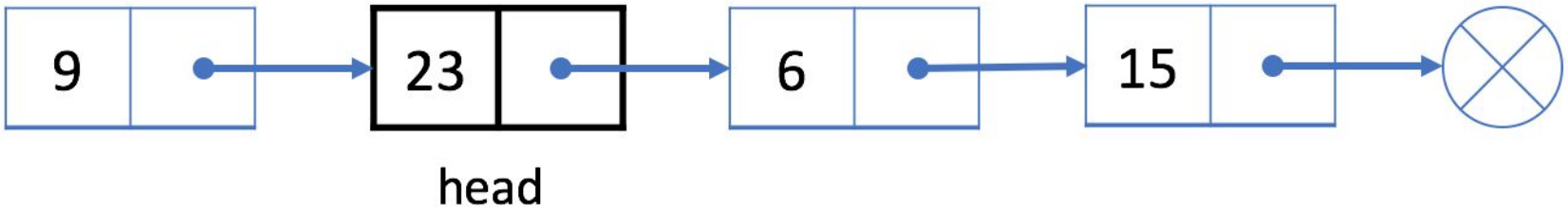
INSERTION OF A NODE BETWEEN TWO NODES

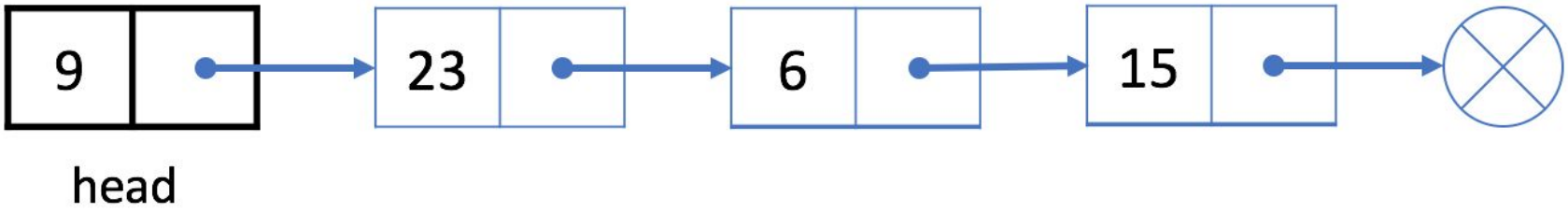




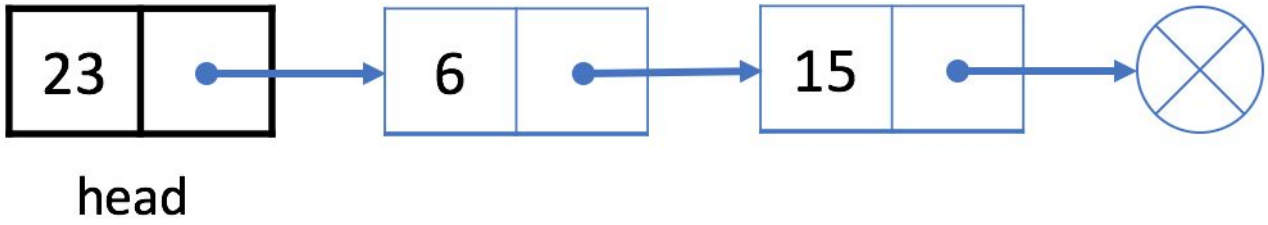


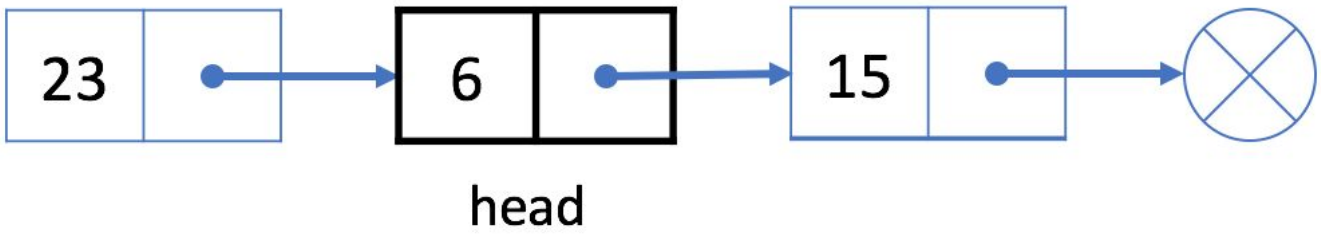
Inserting a Node in the beginning



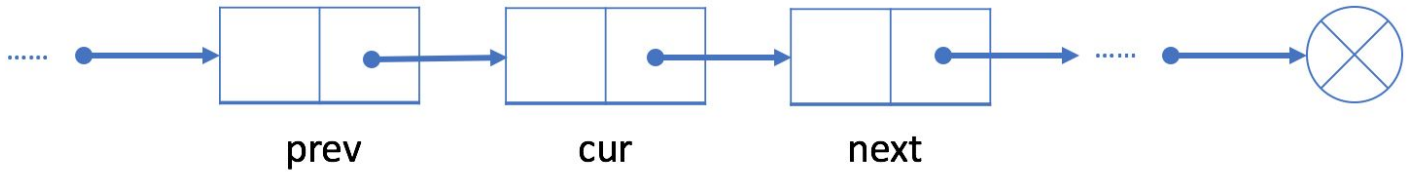


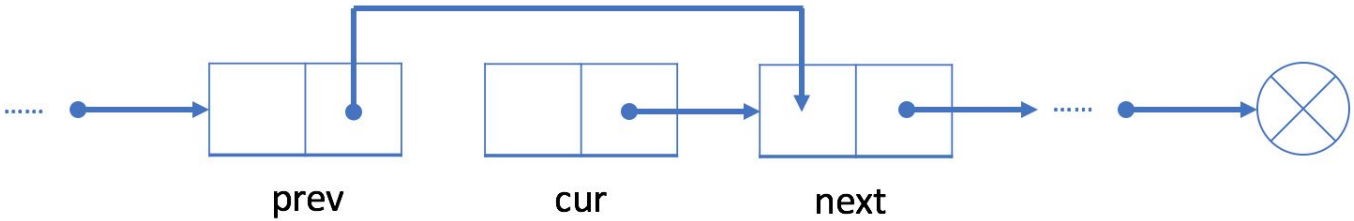
Deleting head node





Deleting a node from between





Comparison

		Array	Singly Linked List
Access	by index	O(1)	O(N)
Add	before first node	O(N)	O(1)
	after given node	O(N)	O(1)
	after last node	O(1)	O(1)
Delete	the first node	O(N)	O(1)
	a given node	O(N)	O(N)
	the last node	O(1)	O(N)
Search	a given value	O(N)	O(N)

LETS CODE