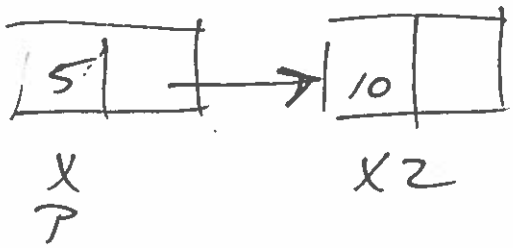



```
node *p = X;
while(p != NULL){
    cout << p->key << endl;
    p = p->next;
} // how we traverse a LL
```

Traverse array

```
for i = 0 to n
    array[i]
```

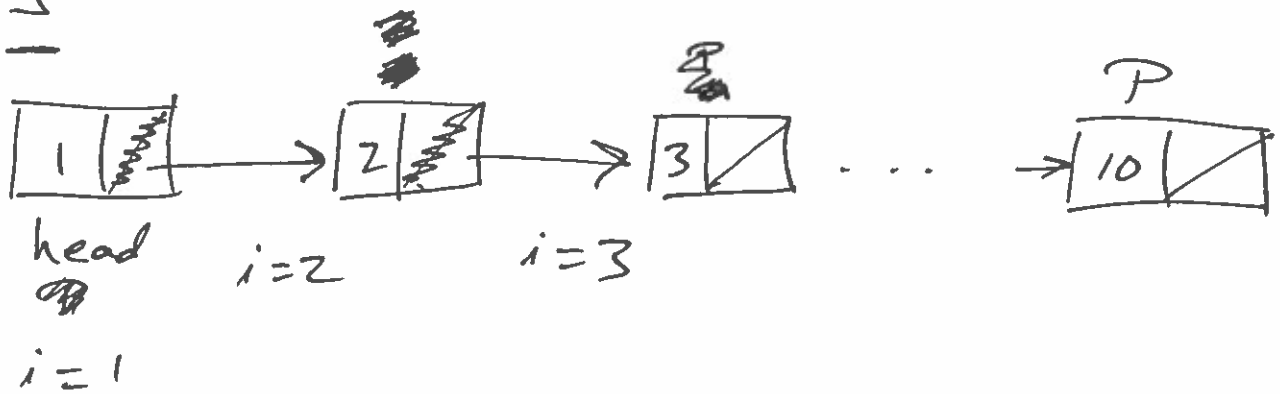
Traverse LL

```
while node != NULL
    node = node->next
```

Build a linked list for length n, whose key is 1-10

```
int i = 1
// create first node in list
// head of the list
node *head = new node(i, NULL)
```

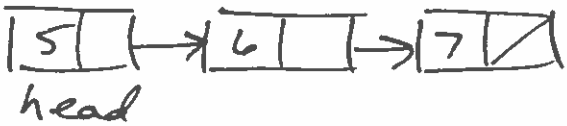
```
node *p = head
i++
while(i <= 10){
    node *n = new node(i, NULL)
    p->next = n        // set p->next to point to n
    p = n              // p = p->next
    i++
}
```



How do I traverse list?

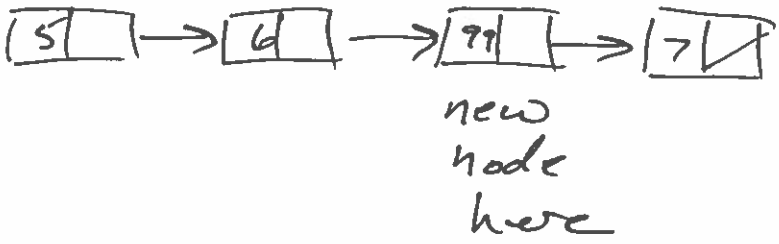
`p = head;` // puts me back at beginning of list

Insert a node -

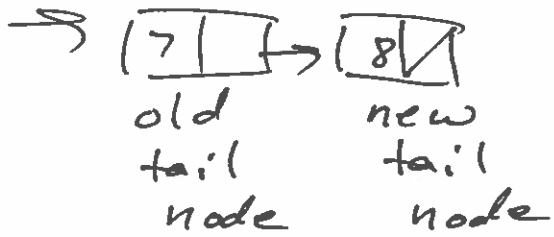


1. New head node
   - insert node before current head node

2. Middle node



3. New tail node - end of the list



Singly-linked list - only have next
We have to know previous node.

search

```
node *search(value, node* head){
    node *current = head;
    while(current != NULL)
        if(current->key == value)
            return current;
        current=current->next
    return NULL;
}

insertNode(value of node previous, value, new node head) beginning of list
    node *left = search(previous, head)
    //create new node
    node *n = new node(value, NULL)
```