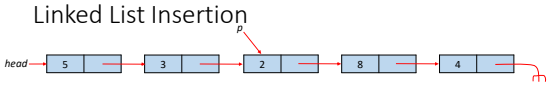
Lecture 4-2

Linked List
Insert, Find, & Delete

Teera Siriteerakul

1

Linked List Insertion

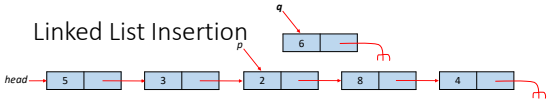


- To insert, we must have a pointer point to the node before the position to insert
  - For example, to insert between 2 and 8 we need a pointer point at 2.
- Steps are:
  - Create a new node q with the new data
  - Point next of q to next of p
  - Point next of p to q
- Big-O is O(1)
- The implementation is simple:

```
public void insert(int d, Node p) {
    Node q = new Node(d);
    q.next = p.next;
    p.next = q;
}
```


2

Linked List Insertion

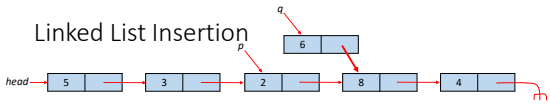


- To insert, we must have a pointer point to the node before the position to insert
  - For example, to insert between 2 and 8 we need a pointer point at 2.
- Steps are:
  - **Create a new node q with the new data**
  - Point next of q to next of p
  - Point next of p to q
- Big-O is O(1)
- The implementation is simple:

```
public void insert(int d, Node p) {
    Node q = new Node(d);
    q.next = p.next;
    p.next = q;
}
```


3

Linked List Insertion

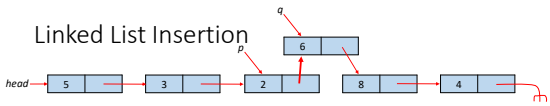


- To insert, we must have a pointer point to the node before the position to insert
  - For example, to insert between 2 and 8 we need a pointer point at 2.
- Steps are:
  - Create a new node q with the new data
  - **Point next of q to next of p**
  - Point next of p to q
- Big-O is O(1)
- The implementation is simple:

```
public void insert(int d, Node p) {
    Node q = new Node(d);
    q.next = p.next;
    p.next = q;
}
```


4

Linked List Insertion



- To insert, we must have a pointer point to the node before the position to insert
  - For example, to insert between 2 and 8 we need a pointer point at 2.
- Steps are:
  - Create a new node q with the new data
  - Point next of q to next of p
  - **Point next of p to q**
- Big-O is O(1)
- The implementation is simple:

```
public void insert(int d, Node p) {
    Node q = new Node(d);
    q.next = p.next;
    p.next = q;
}
```


5

Delete from Linked List

- To delete, we must have a pointer point to the node before the position to delete
  - For example, to delete 8 we need a pointer point at 2.
- Steps is:
  - Point next of p to next of next of p
- The left-over data node will be handled by the garbage collector.
- Big-O is O(1)
- The implementation is so simple:

```
public void delete(Node p) {
    p.next = p.next.next;
}
```


6

Search for Data in Linked List

- Binary search is NOT possible due to random access is O(n) in average.
- Best we can do is sequential search which is O(n) in average.
- The implementation is as follow:
- Note that, in main(), we must call find() like this:

```
MyLinkedList.Node p = mList.find(4);
```

```
public Node find(int d) {
    Node p = head;
    while(p!=null) {
        if(p.data==d) return p;
        p = p.next;
    }
    return null;
}
```

The result of `find()` cannot be use as an input of `insert()` or `delete()`!

7

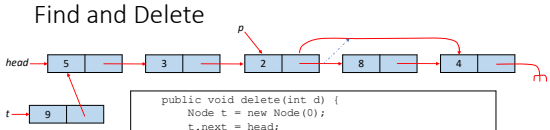
Find and Delete

- Since method `find()` and `delete()` previously implemented cannot work with each other, let implement another `delete()` where we need to find (take data `d` as input).

```
public void delete(int d) {
    Node t = new Node(0);
    t.next = head;
    Node p = t;
    while( (p.next!=null) && (p.next.data!=d) ) {
        p = p.next;
    }
    if(p.next!=null) {
        p.next = p.next.next;
    }
    head = t.next;
}
```


8

Find and Delete




```
public void delete(int d) {
    Node t = new Node(0);
    t.next = head;
    Node p = t;
    while( (p.next!=null) && (p.next.data!=d) ) {
        p = p.next;
    }
    if(p.next!=null) {
        p.next = p.next.next;
    }
    head = t.next;
}
```


9

Summary

- Let summarize all methods of linked list

Methods	Best case	Worst case	Average case
Add into a linked list	$O(1)$	$O(1)$	$O(1)$
Insert into a linked list	$O(1)$	$O(1)$	$O(1)$
Find in a linked list	$O(1)$	$O(n)$	$O(n)$
Delete from a linked list	$O(1)$	$O(1)$	$O(1)$

- Order or unordered linked list make no difference in its operations.
- We can use temporary head to help with delete
  - Can be done similarly in insertion