

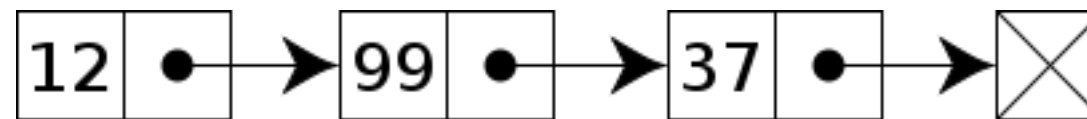
Week 4 Lecture I I

Theory

What's in this lecture?

- Linked Lists

List



- Look Familiar?
- A singly-linked list consists of list nodes with **value** and **next** pointers
- In this case, the **head** of the list is the node that has value 12

List in JavaScript

```
function make_listnode(value, next) {  
    var node = new Object();  
    node["value"] = value;  
    node["next"] = next;  
  
    return node;  
}
```

List in JavaScript

```
var alist = make_listnode(12, null);  
var blist = make_listnode(24, alist);  
var clist = make_listnode(17, blist);
```

What do alist, blist, and clist “look like”?

List Find()

```
function find(value, alist) {  
    if (alist == null) {  
        return null;  
    }  
    if (alist["value"] == value) {  
        return alist;  
    }  
    return find(value, alist["next"]);  
}
```

List Contains()

```
function contains(value, alist) {  
    return find(value, alist) != null;  
}
```

List Insert()

```
function insert(value, alist) {  
    return make_listnode(value, alist);  
}
```


List Remove()

```
function remove(value, alist) {  
  var last = null;  
  
  for (var cur = alist; cur != null; cur = cur["next"]) {  
    if (cur["value"] == value) {  
      if (last == null) {  
        alist = cur["next"];  
      } else {  
        last["next"] = cur["next"];  
      }  
      break;  
    }  
    last = cur;  
  }  
  
  return alist;  
}
```

List Length()

```
function length(alist) {  
    var len = 0;  
    for (var cur = alist; cur != null; cur = cur["next"]) {  
        len = len + 1;  
    }  
  
    return len;  
}
```

Exercises

- Read Intro to Algorithms, 3rd Edition, Chapter 10
- Implement `make_listnode`, `find`, `contains`, `insert`, and `delete`, and `length` for a **doubly**-linked list in JavaScript