



this is week 7

fall 2013

playlist50

Ya Hey (Vampire Weekend)

Mountain Sound (Of Monsters and Men)

Monster Mash (Bobby Pickett & The Crypt-Kickers)



agenda

resources

linked lists

hash tables

resources

lecture notes & source code

cs50.net/shorts

study.cs50.net

man

Google

cs50.net/discuss

ohs

me!

pset6

gdb

valgrind --leak-check=full

diff -y

[Sign Up](#)[Log in](#)[English ▾](#)

Tell Jason what you think about his teaching - both positive and constructive comments welcome!

Be honest and sincere, you'll stay anonymous:

Describe Jason Hirschhorn's good or bad qualities here -- this will help him/her to develop.

☐ **Recommended:** Allow Jason Hirschhorn to respond privately. You'll stay anonymous.

Say it.



Jason Hirschhorn has got **5** anonymous opinions

Get your feedback URL - 20 second sign-up

Your full name

Your password

Your feedback URL

Sign Up

<http://sayat.me/cs50>

linked lists

i.qkme.me/3to4j.jpg

node

```
typedef struct node
{
    int n;
    struct node* next;
}
node;
```

node

```
node new_node;
```

```
new_node.n = 1;
```

```
printf("%i\n", new_node.n);
```

```
node* ptr_node = &new_node;
```

```
printf("%i\n", (*ptr_node).n);
```

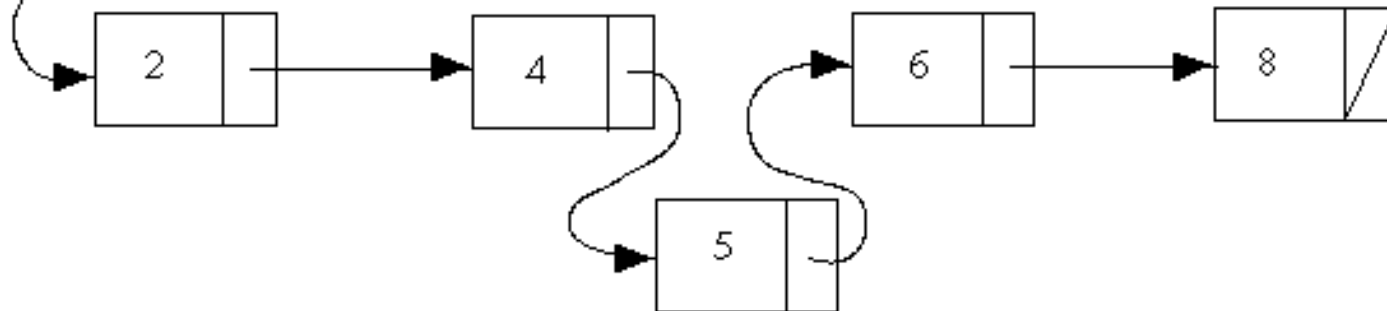
```
printf("%i\n", ptr_node->n);
```

insert

Original List
first

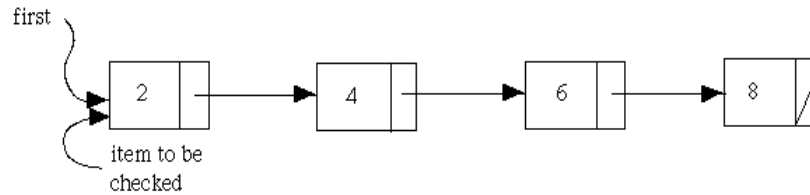


List with 5 added
first

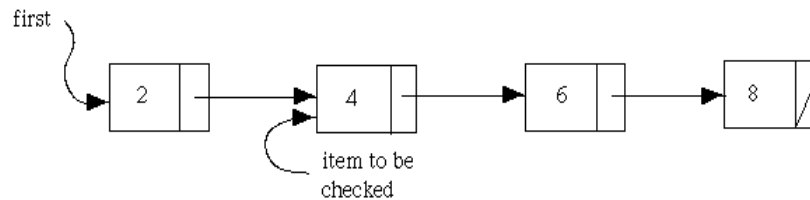


find

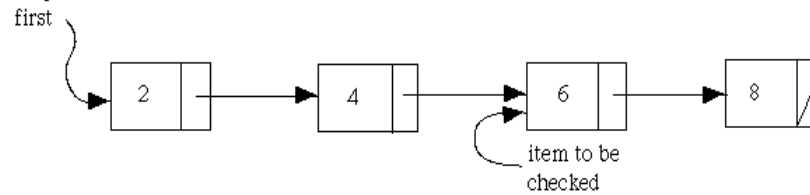
Step 1: Prepare to check first item



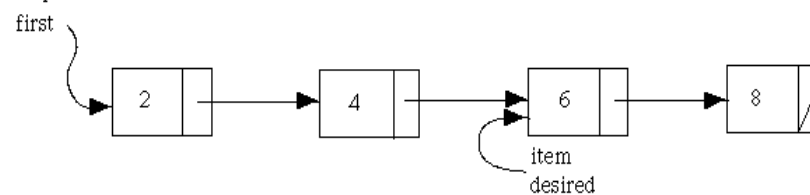
Step 2: Move to second item



Step 3: Move to next item

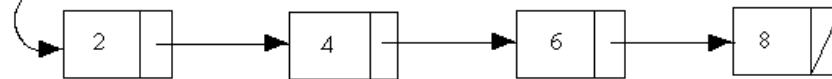


Step 4: Item found

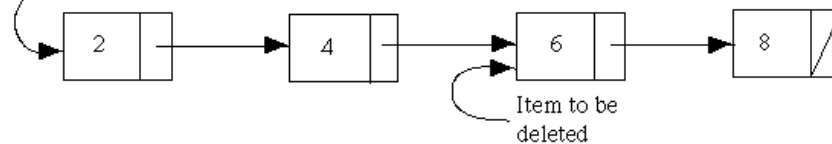


delete

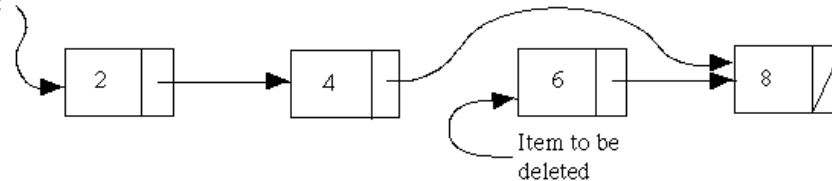
Original List
first



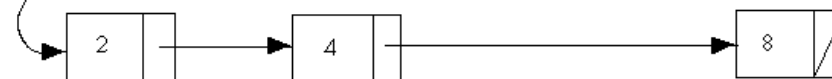
Step 1: Find item to be deleted
first



Step 2: Change previous pointer
first



Step 3: Throw away old item
first



your turn: linked.c

Write a function that inserts an int into a linked list. Keep the list sorted from smallest to largest. Do not insert duplicates. Let the user know whether or not the insert was successful. Don't worry about freeing nodes at the end of the program.

```
bool insert_node(int value);
```

Skeleton code has been provided for you.

strategy

// logic

draw a picture

write some pseudocode

// syntax

map it onto C

code the program

your turn: linked.c

Write a function that prints out all of the ints in a linked list. Print out the number of each node (0-indexed) as well.

```
void print_nodes(node* list);
```

Skeleton code has been provided for you.

your turn: linked.c

Write a function that frees all of the nodes in a linked list before the program exits.

```
void free_nodes(node* list);
```

Skeleton code has been provided for you.

hash tables

```
#####  
#      #  
#      #
```

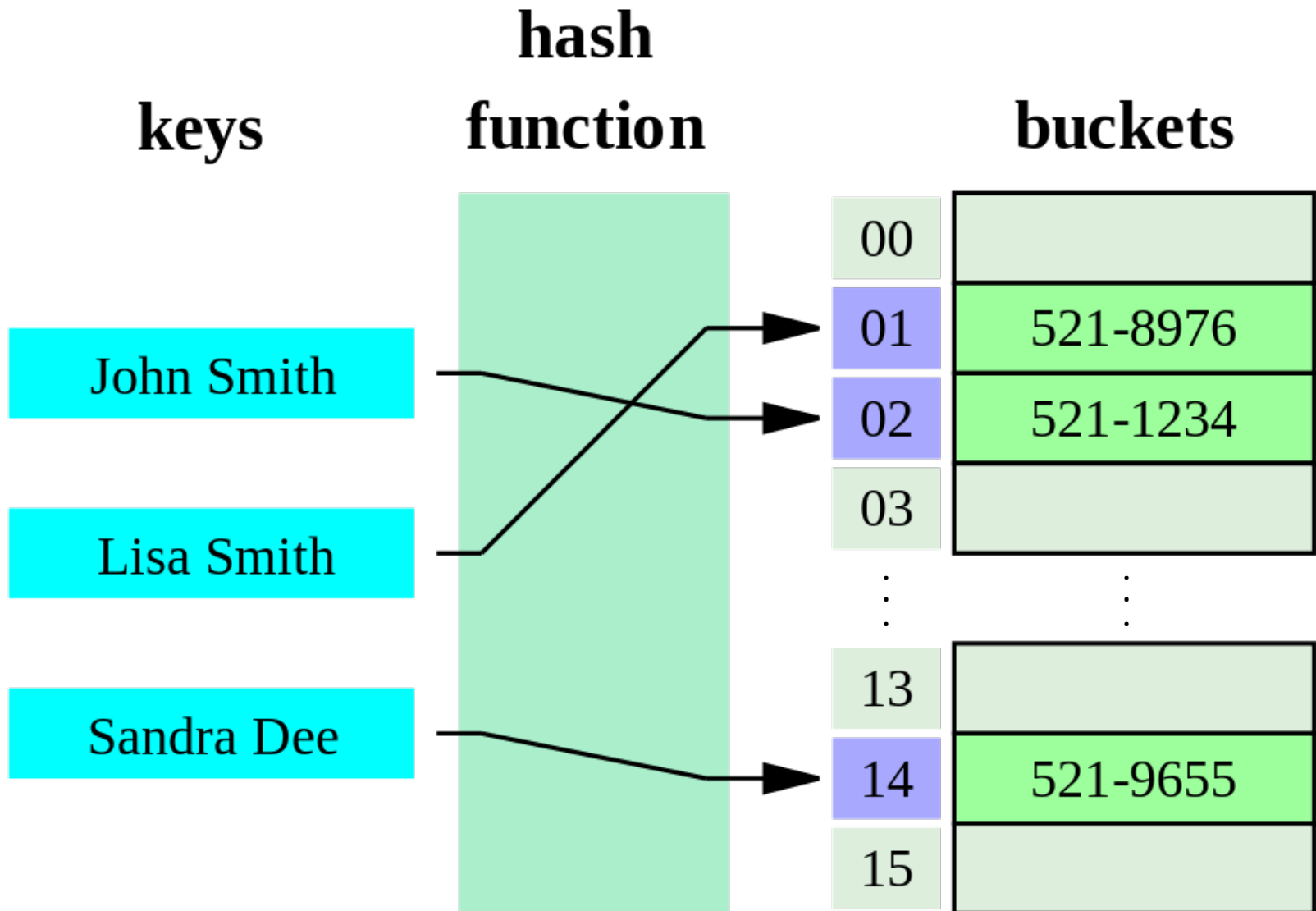
hash tables

array + hash function

0) key

1) `value = hash_function(key)`

2) `array[value] = key`



potential pitfalls

What make a good
hash function?

0) deterministic

1) returns valid indices

potential pitfalls

What if two keys map to the same value?

0) linear probing

1) separate chaining

