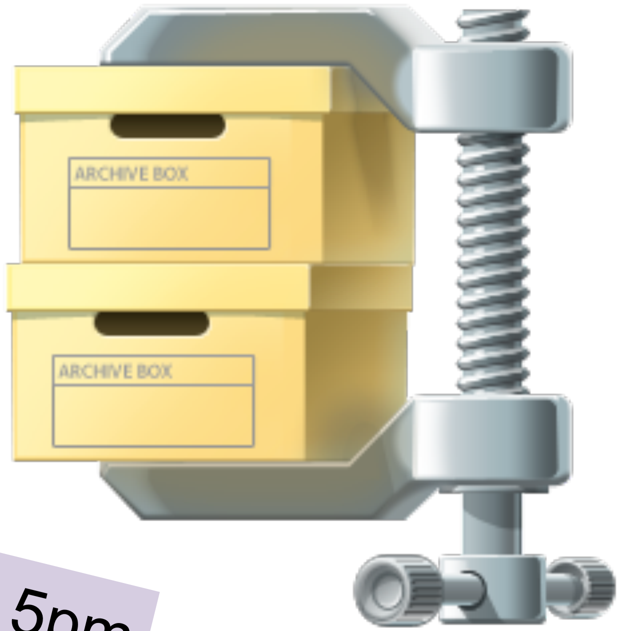
Binary Search Trees

Chris Piech

CS 106B
Lecture 20
Feb 24, 2016

Announcements



Due March 2nd at 5pm

YEAH tomorrow at 5pm in braunAud

Mid Quarter Evaluations

Socrative

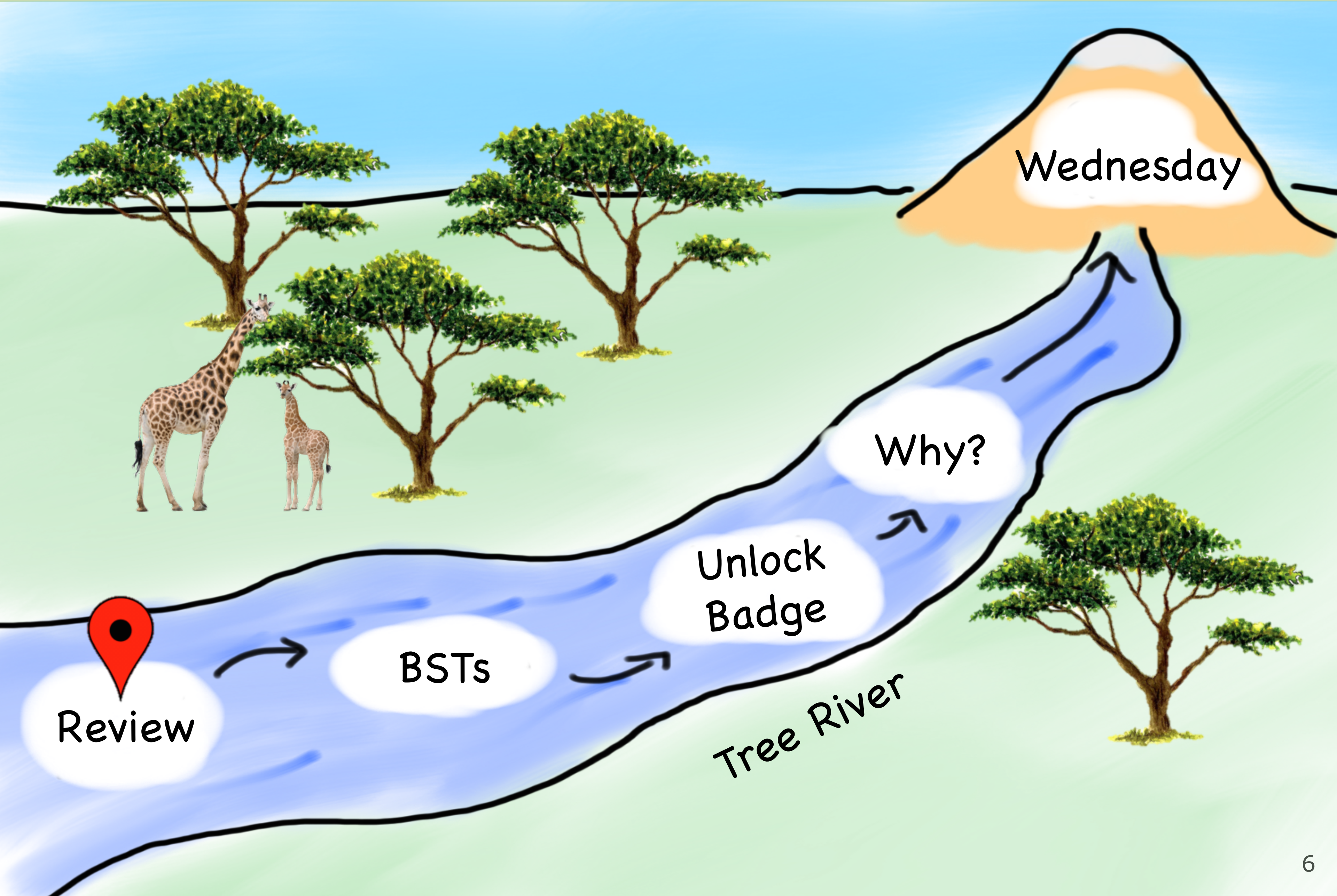
STUDENT

Room: **106BWIN16**

Today's Goal

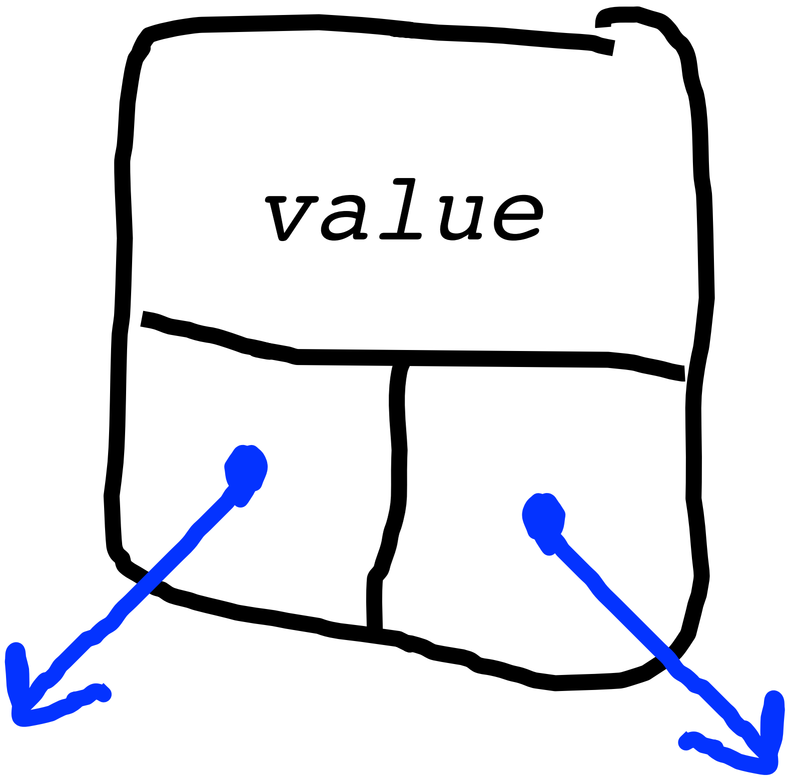
1. Binary Search Trees
2. Review Under the Hood

Today's Route



Binary Tree

```
struct Tree {
  string value;
  Tree * left;
  Tree * right;
};
```



This is how a Binary Tree is defined

Pointers by Reference

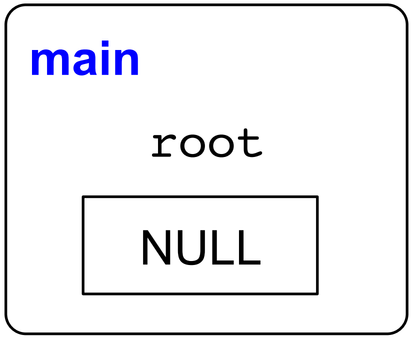
Pointers by Reference

```
int main() {
   Tree * root = NULL;
   startTree(root);
}

void startTree(Tree * tree) {
   tree = new Tree;
   tree->label = “S”;
}
```

This is not going to do anything

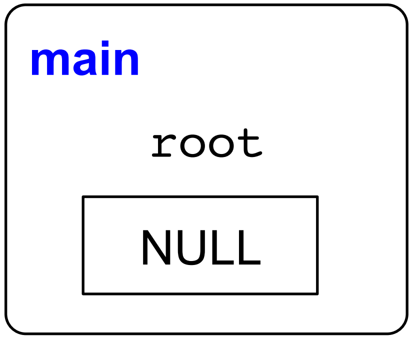
Pointers by Reference

```
int main() {
    Tree * root = NULL;
    startTree(root);
}

void startTree(Tree * tree) {
    tree = new Tree;
    tree->label = “S”;
}
```



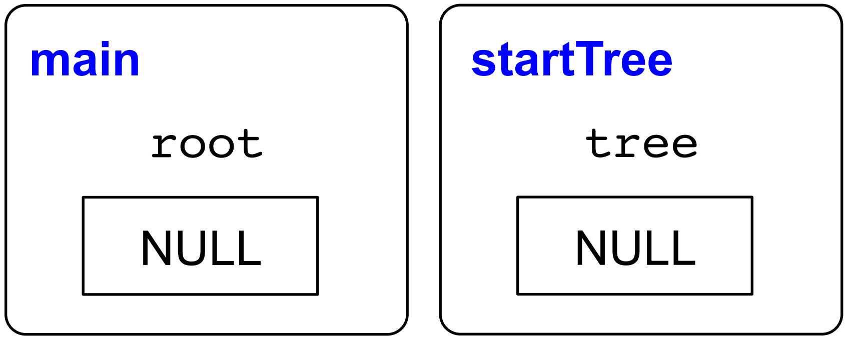
Pointers by Reference

```
int main() {
    Tree * root = NULL;
    startTree(root);
}

void startTree(Tree * tree) {
    tree = new Tree;
    tree->label = "S";
}
```



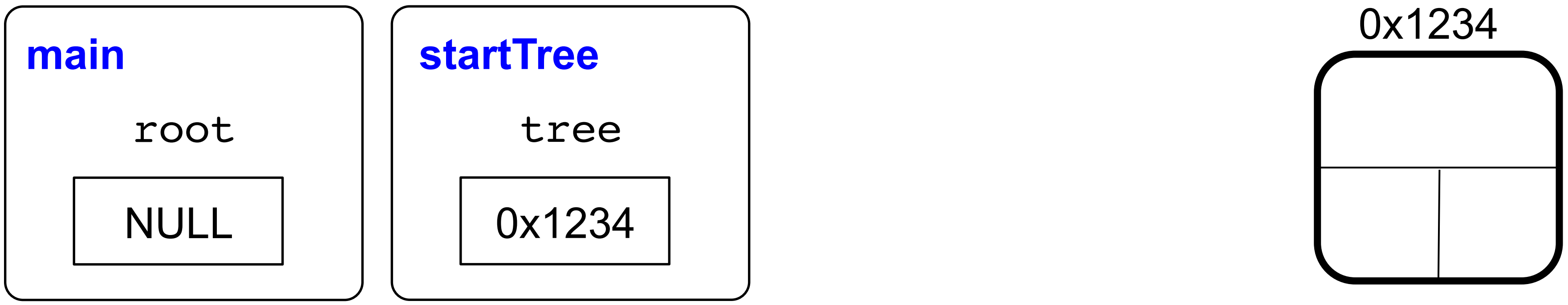
Pointers by Reference

```
int main() {
    Tree * root = NULL;
    startTree(root);
}

void startTree(Tree * tree) {
    tree = new Tree;
    tree->label = “S”;
}
```



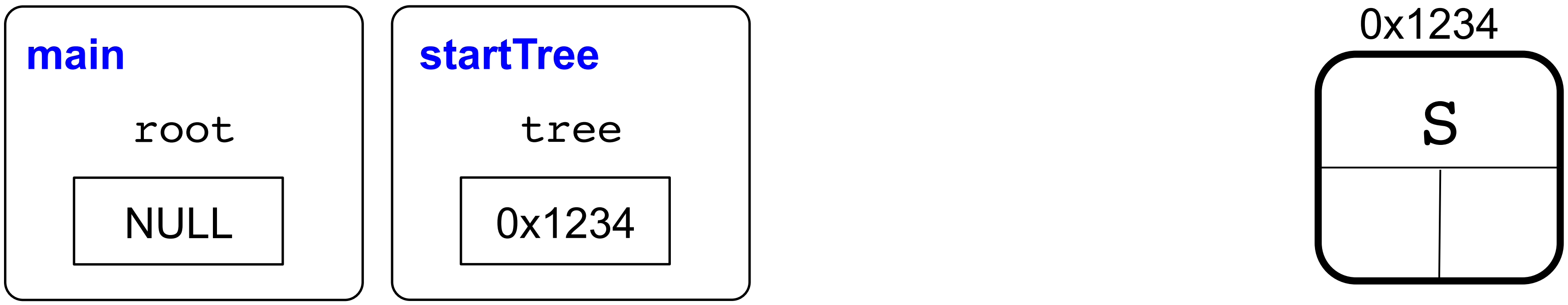
Pointers by Reference

```
int main() {
    Tree * root = NULL;
    startTree(root);
}

void startTree(Tree * tree) {
    tree = new Tree;
    tree->label = “S”;
}
```



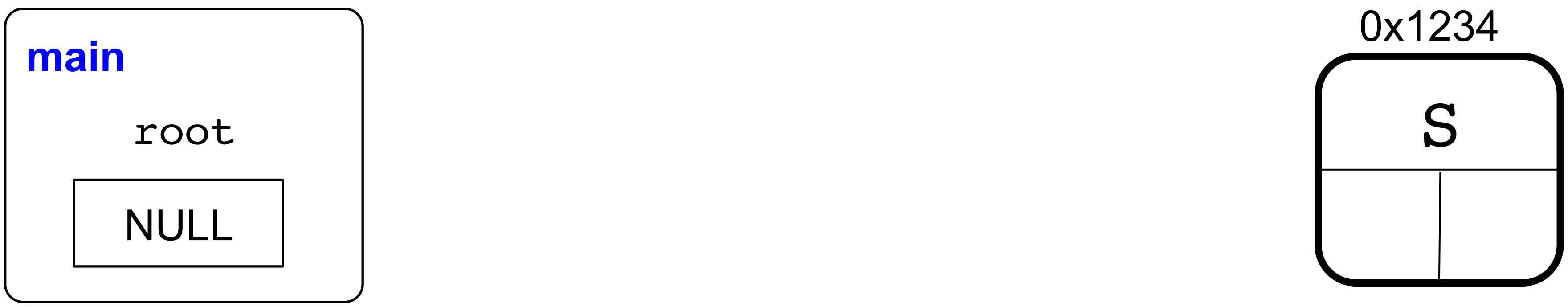
Pointers by Reference

```
int main() {
    Tree * root = NULL;
    startTree(root);
}

void startTree(Tree * tree) {
    tree = new Tree;
    tree->label = “S”;
}
```



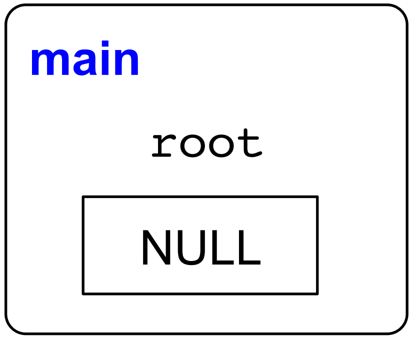
Pointers by Reference

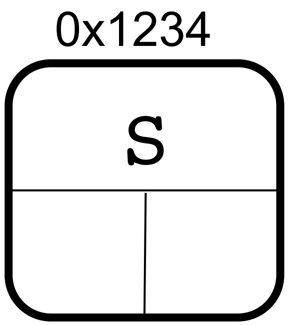
```
int main() {
    Tree * root = NULL;
    startTree(root);
}

void startTree(Tree * tree) {
    tree = new Tree;
    tree->label = “S”;
}
```



Pointers by Reference

```
int main() {
    Tree * root = NULL;
    startTree(root);
}

void startTree(Tree * tree) {
    tree = new Tree;
    tree->label = "S";
}
```





Fix

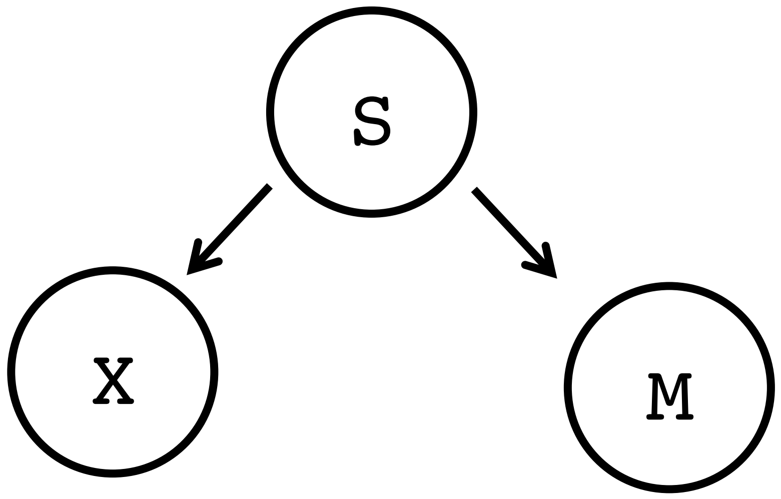
Pointers by Reference

```
int main() {
   Tree * root = NULL;
   startTree(root);
}

void startTree(Tree * & tree) {
   tree = new Tree;
   tree->label = “S”;
}
```

This is the fix

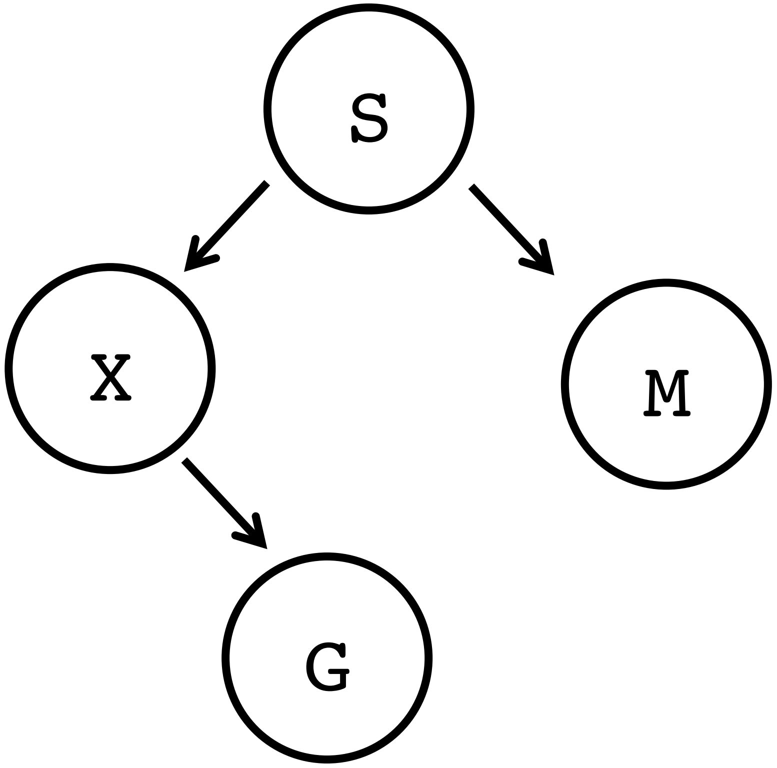
Add Random Leaf

Add Random Leaf



Add a random node to this tree…

Add Random Leaf



… and you might get something like this

Add Random Leaf

```
void addRandomLeaf(Tree * tree) {
  if(tree == NULL) {
    tree = new Tree;
    tree->value = randomChar();
    return;
  }
  if(randomBool()) {
    addRandomLeaf(tree->left);
  } else {
    addRandomLeaf(tree->right);
  }
}
```

Add Random Leaf

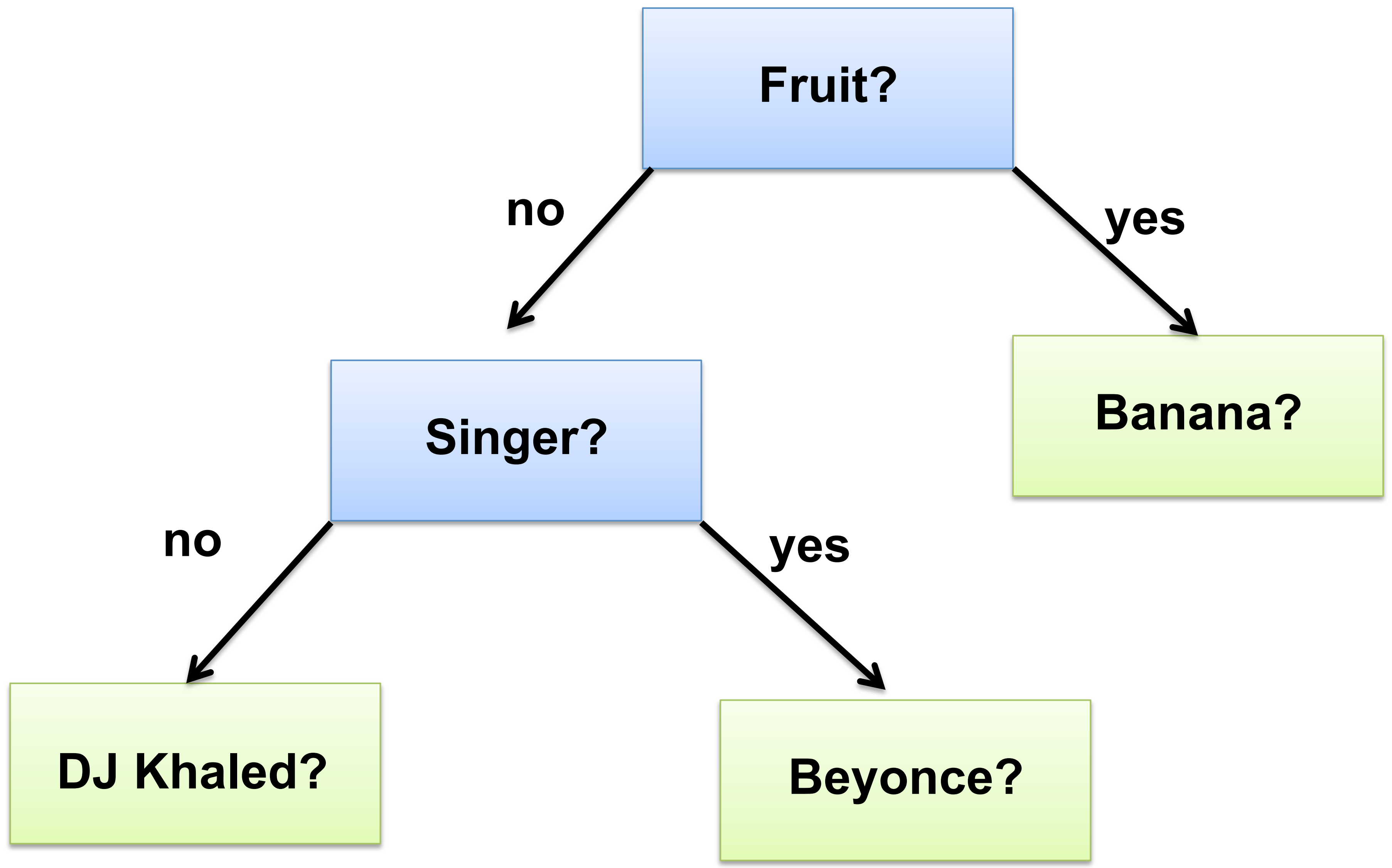
```
void addRandomLeaf(Tree * & tree) {
  if(tree == NULL) {
    tree = new Tree;
    tree->value = randomChar();
    return;
  }
  if(randomBool()) {
    addRandomLeaf(tree->left);
  } else {
    addRandomLeaf(tree->right);
  }
}
```

Full trace in Monday's code

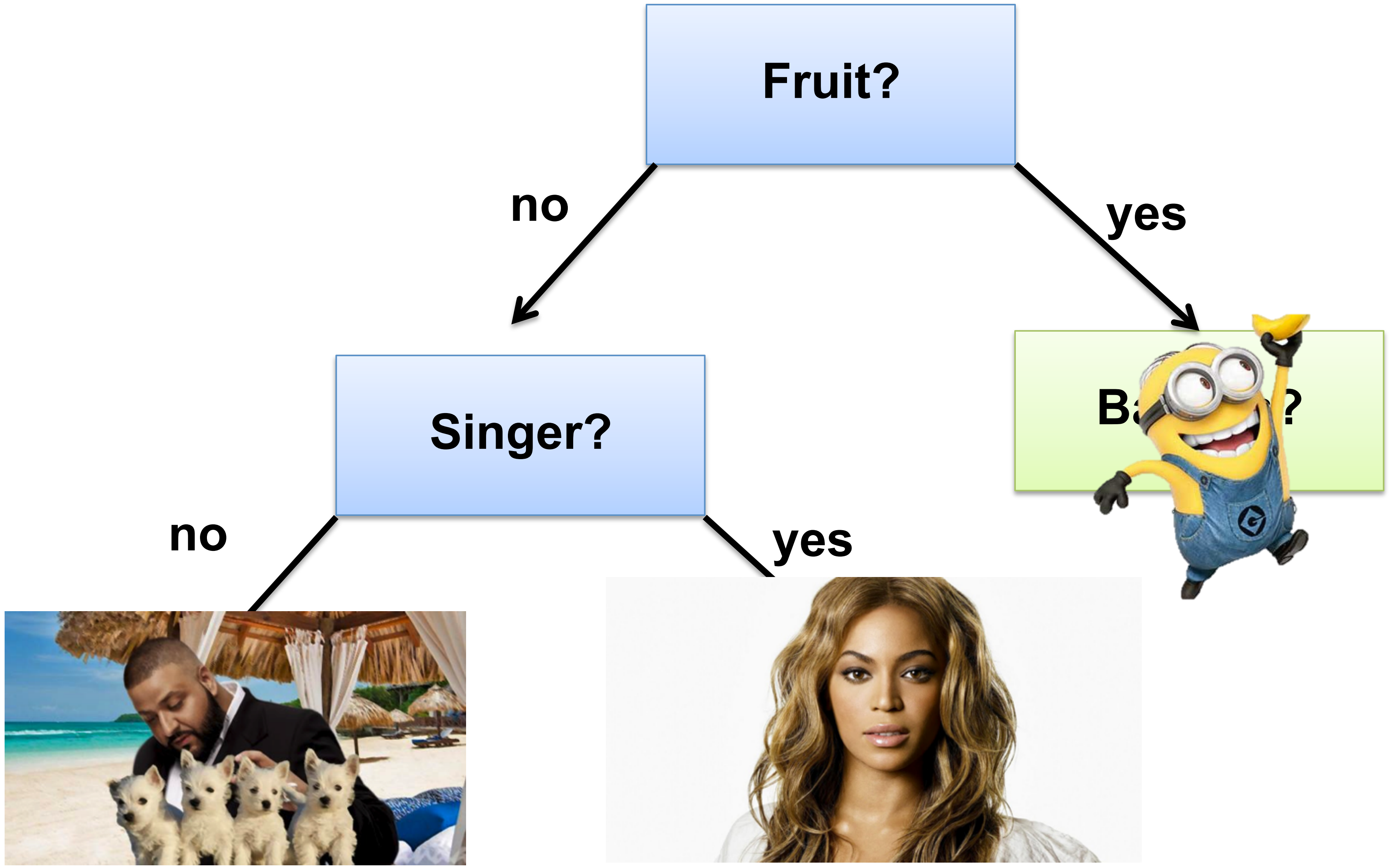
Add Random Leaf

```
void addRandomLeaf(Tree * & tree) {
  if(tree == NULL) {
    tree = new Tree;
    tree->value = randomChar();
    return;
  }
  if(randomBool()) {
    addRandomLeaf(tree->left);
  } else {
    addRandomLeaf(tree->right);
  }
}

int main() {
  Tree * root = NULL;
  addRandomLeaf(root);
}
```

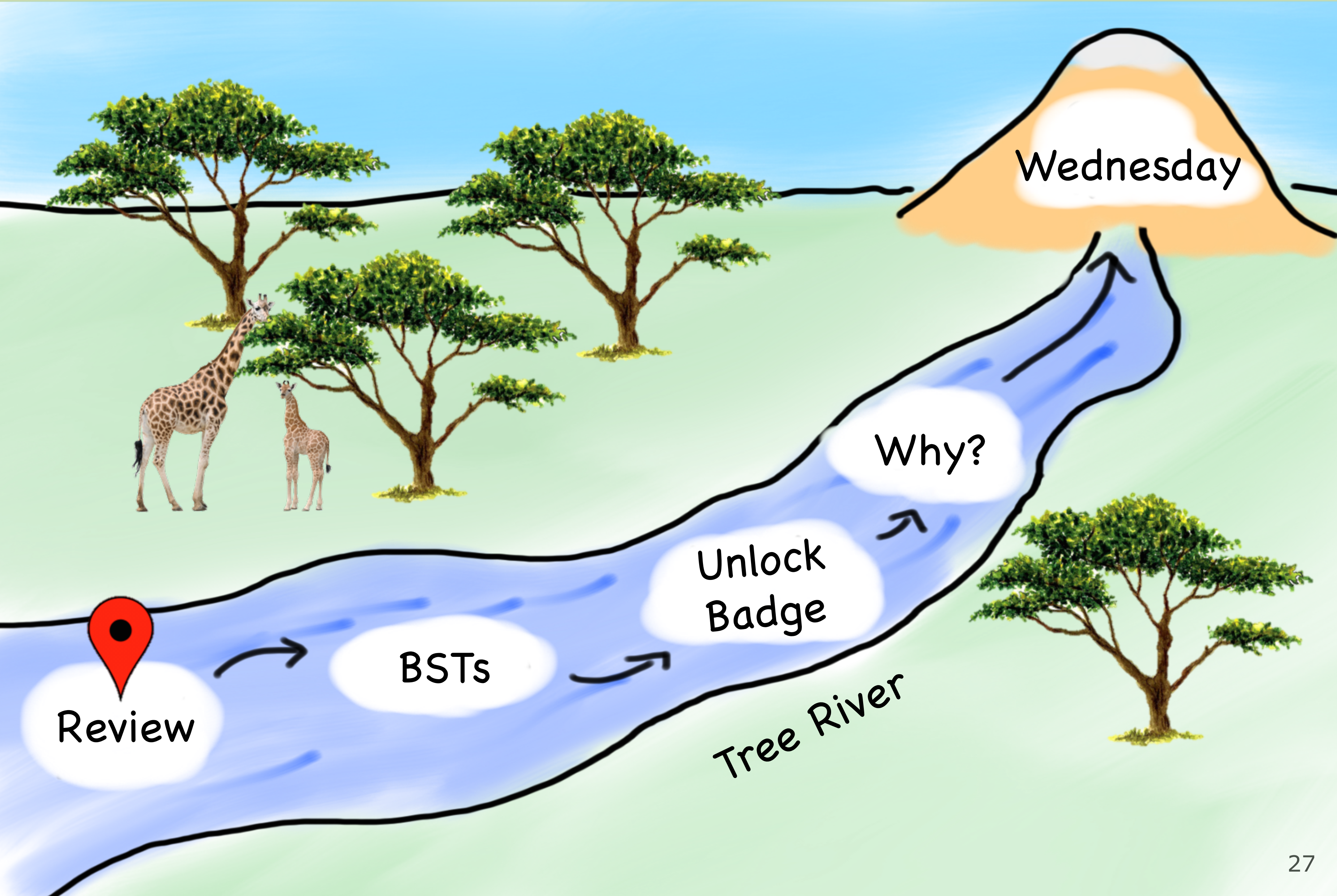
Pensive



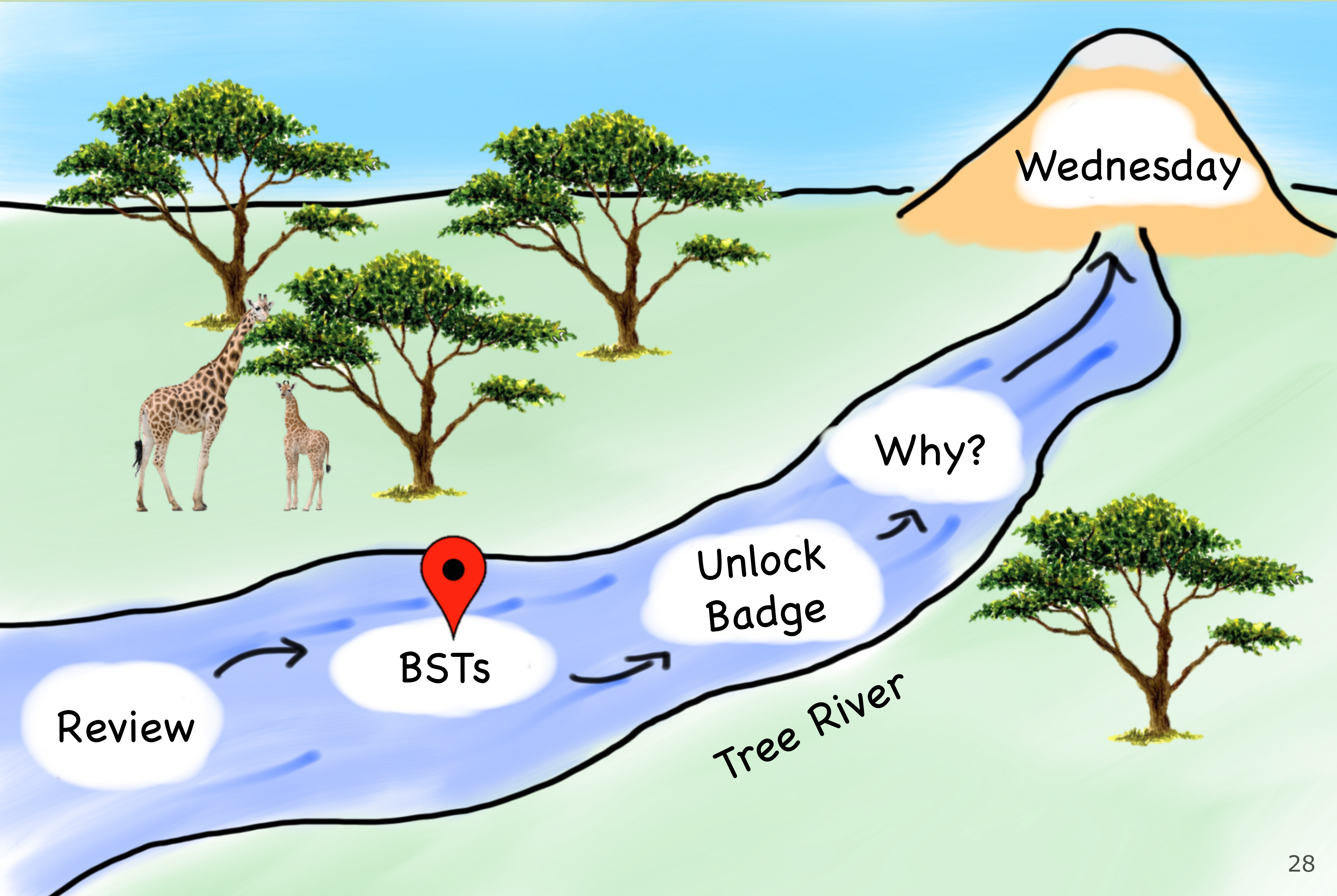
Pensive



Today's Route



Today's Route



How does the Map/Set work?

Talked About the Others

Vector

Grid

HashMap

Stack

Queue

PQueue

HashSet

Map

Set

Talked About the Others

Can I play too?

Grid

Vector

HashMap

Stack

Queue

PQueue

HashSet

Map

Set

Talked About the Others

Vector

Grid

HashMap

Stack

Queue

PQueue

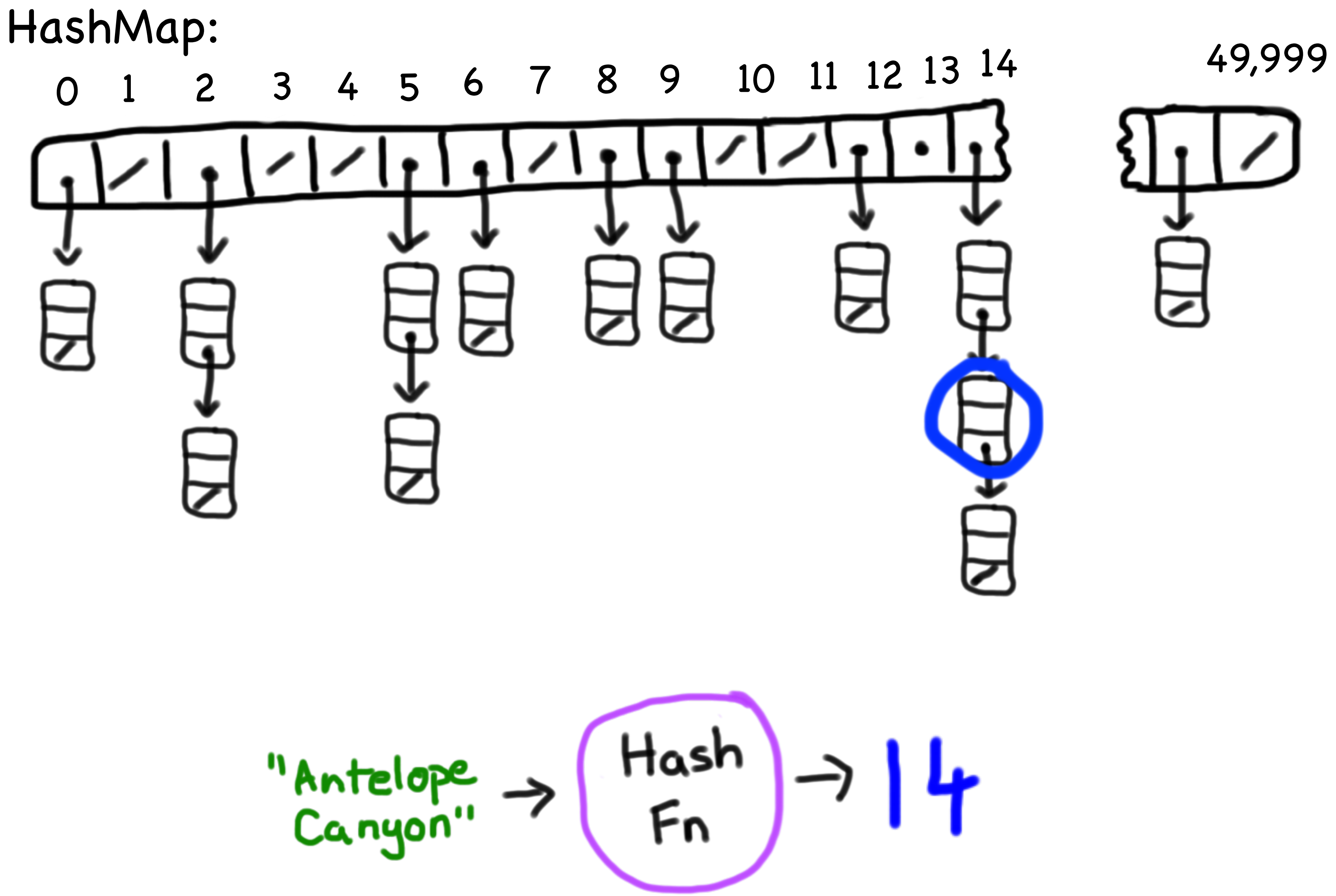
HashSet

Map

Set

How does the Map/Set work?

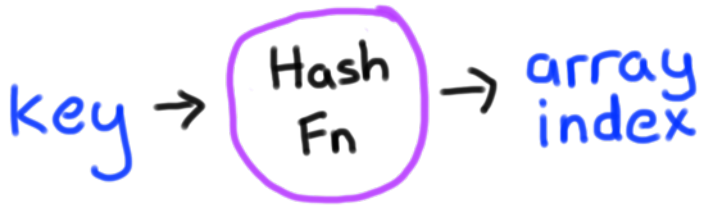
HashMap



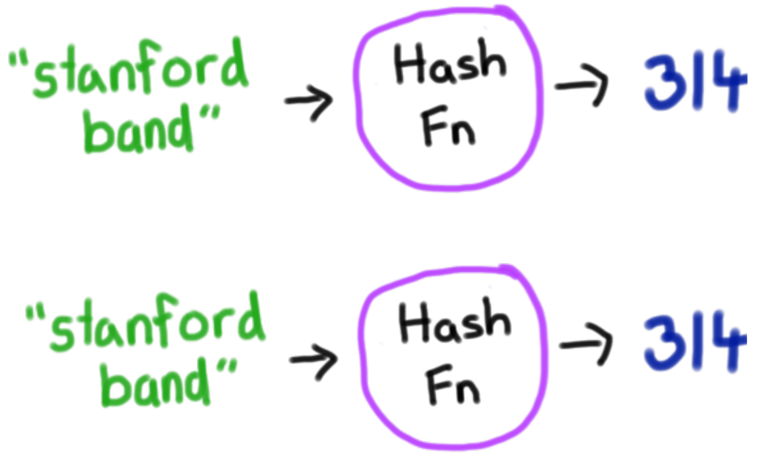
HashMap:
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
49,999
"Antelope Canyon"
Hash Fn
14

Hash Function

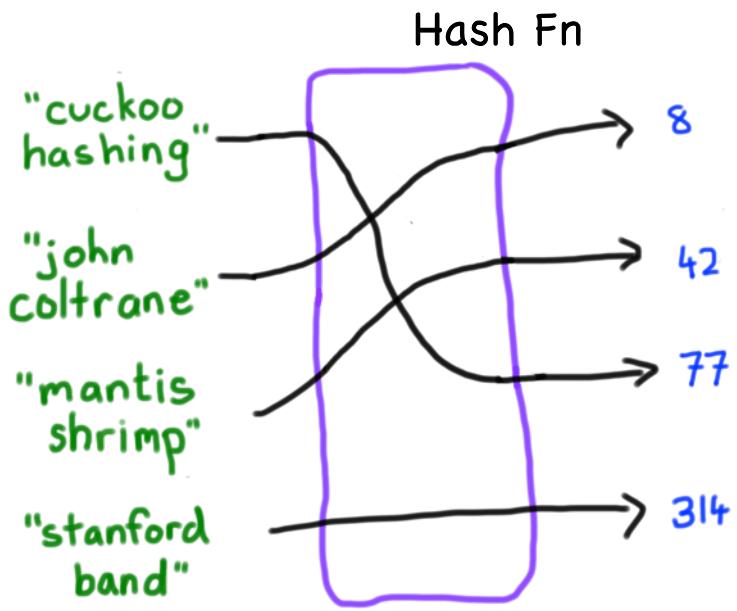
```
int hash(string key);
```



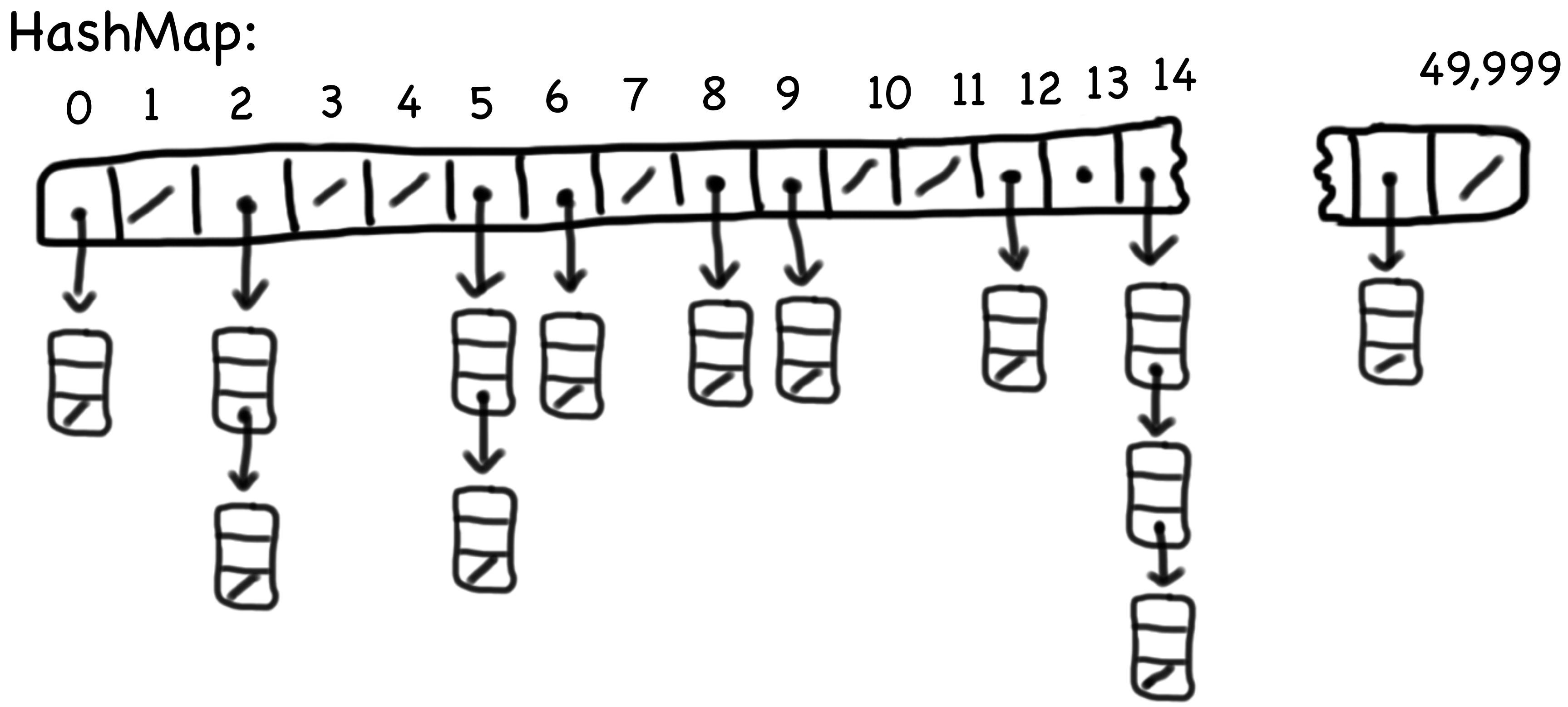
1. Consistent



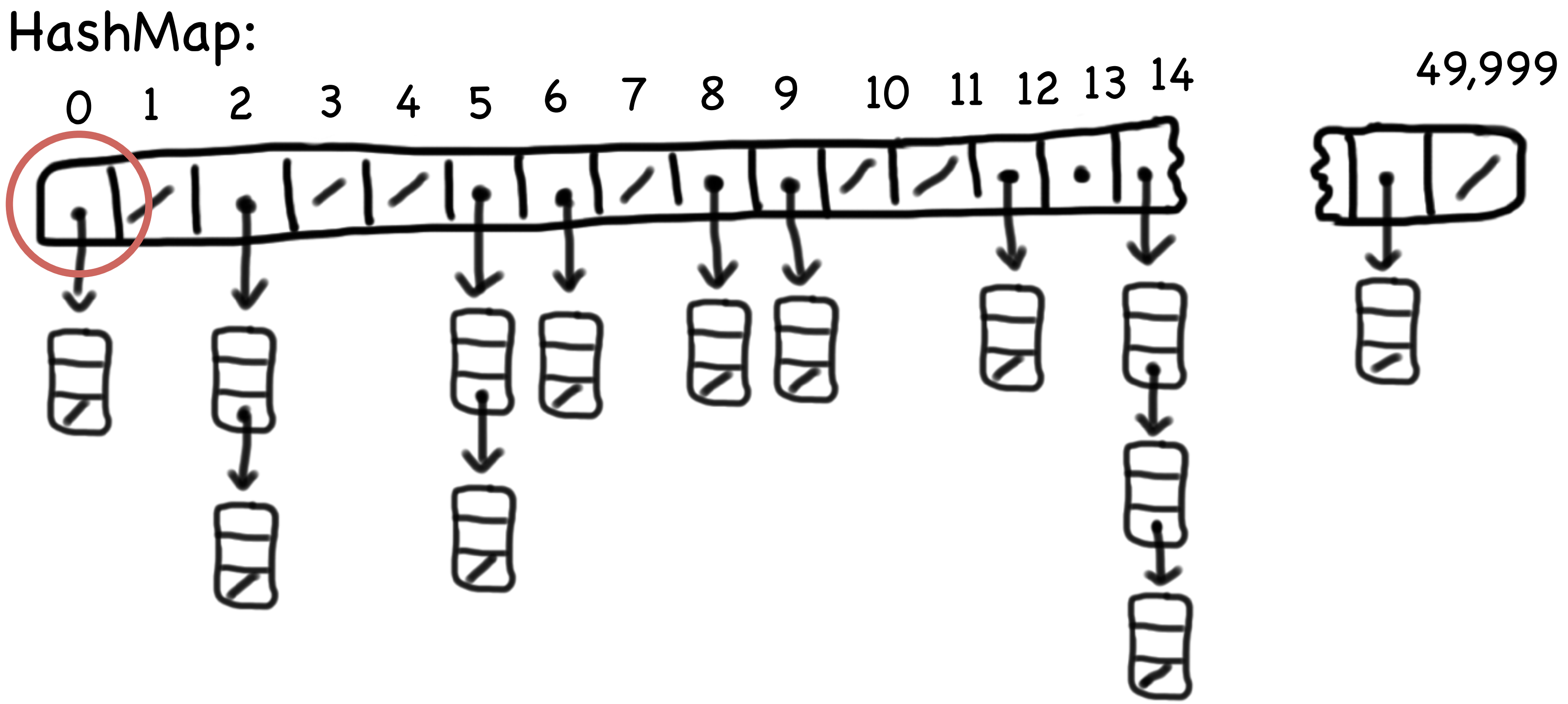
2. Well Distributed



Iteration

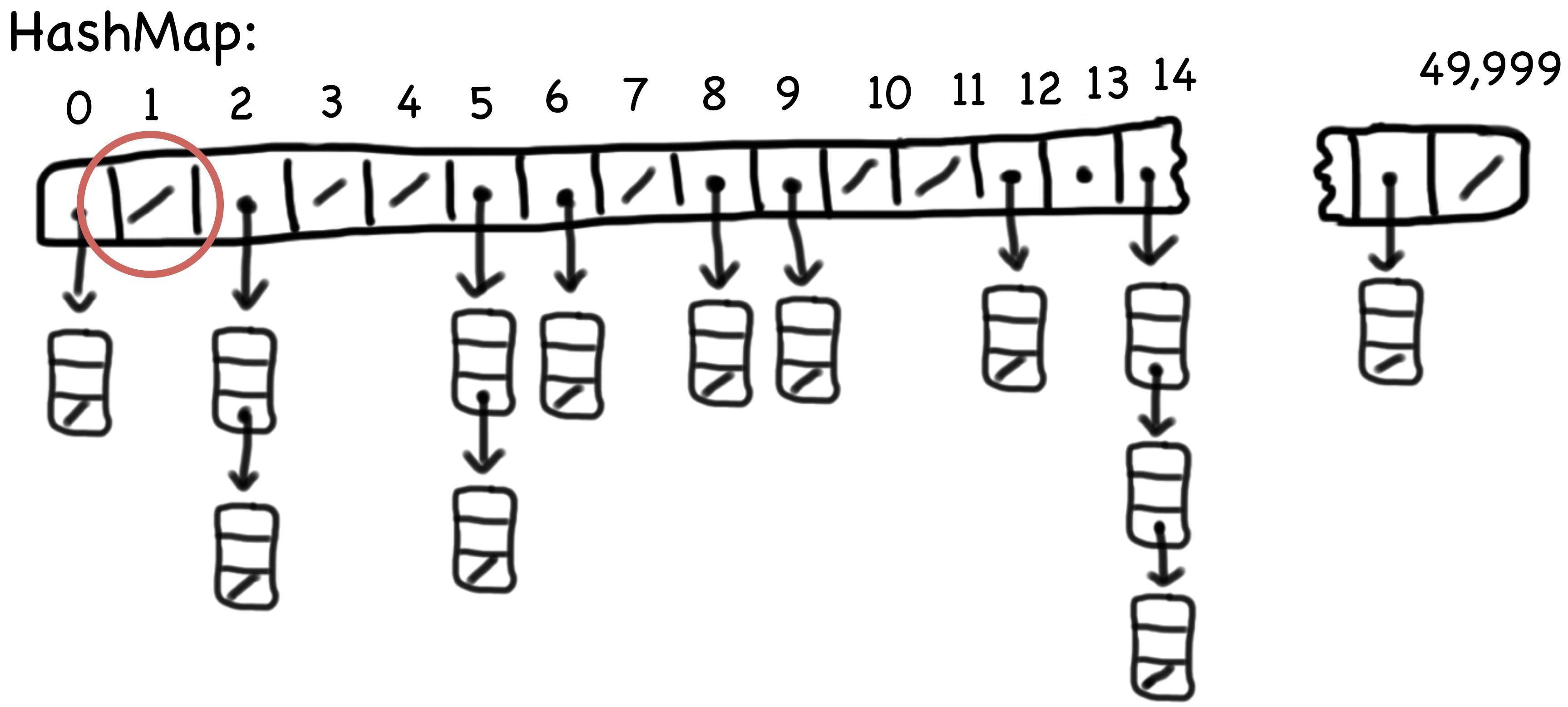


Iteration



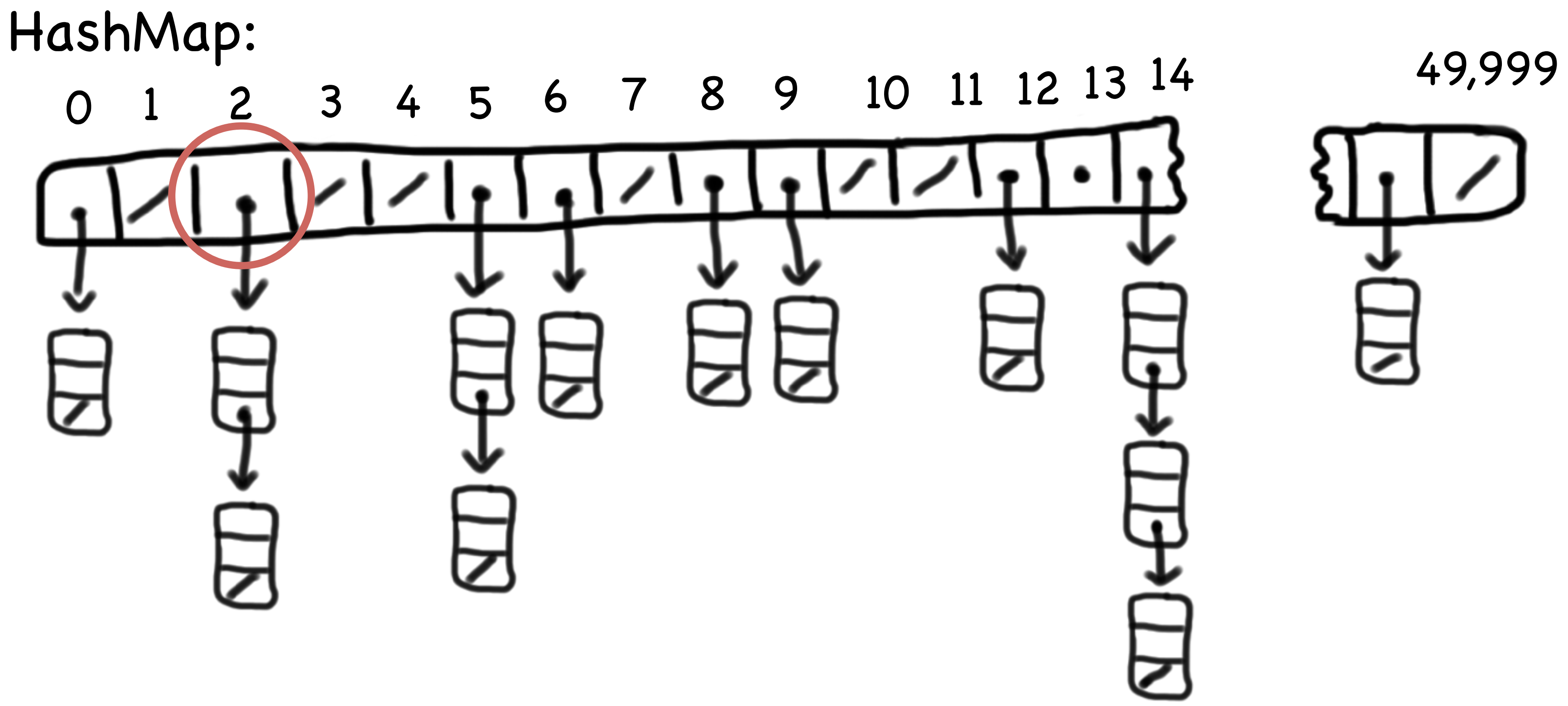
Iterating over a map doesn't return elements in sorted order

Iteration



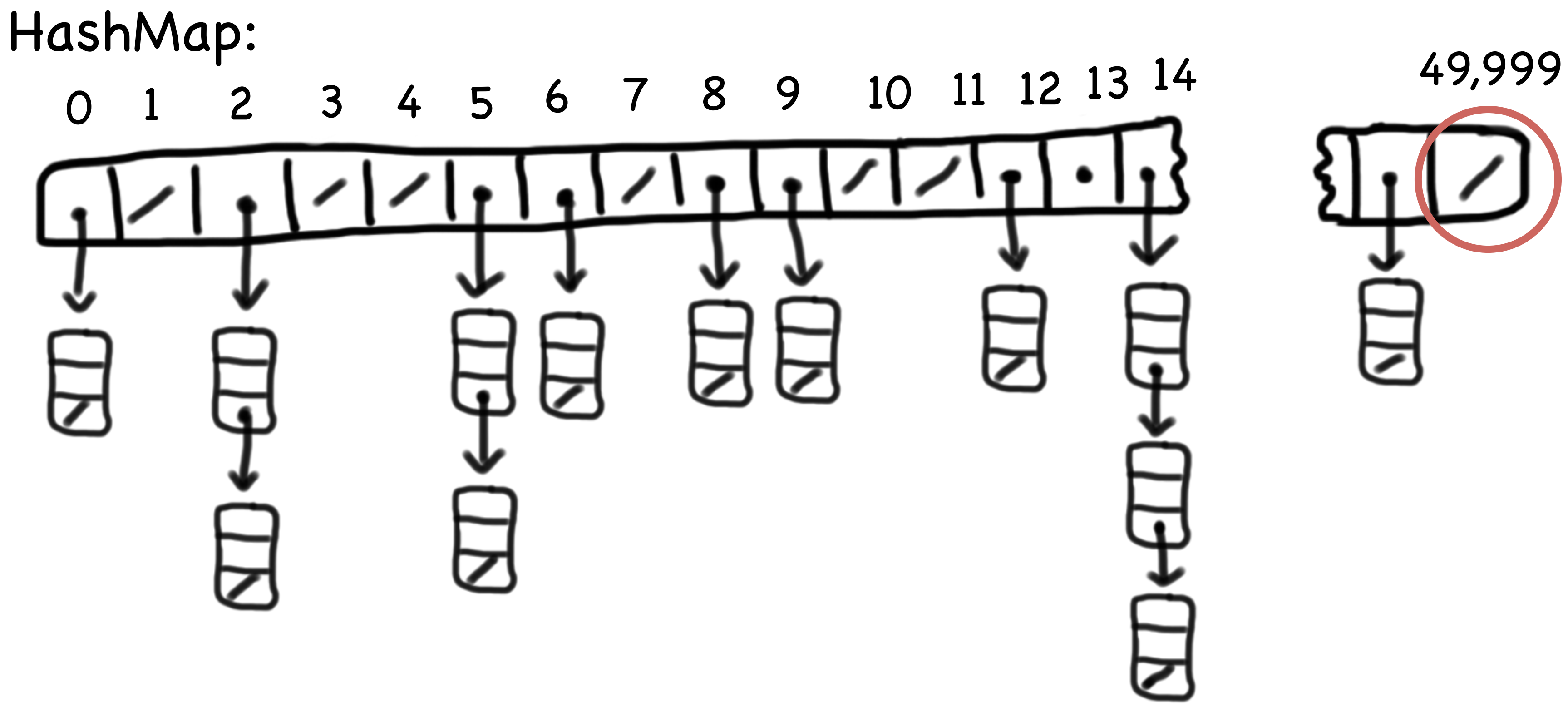
Iterating over a map doesn't return elements in sorted order

Iteration



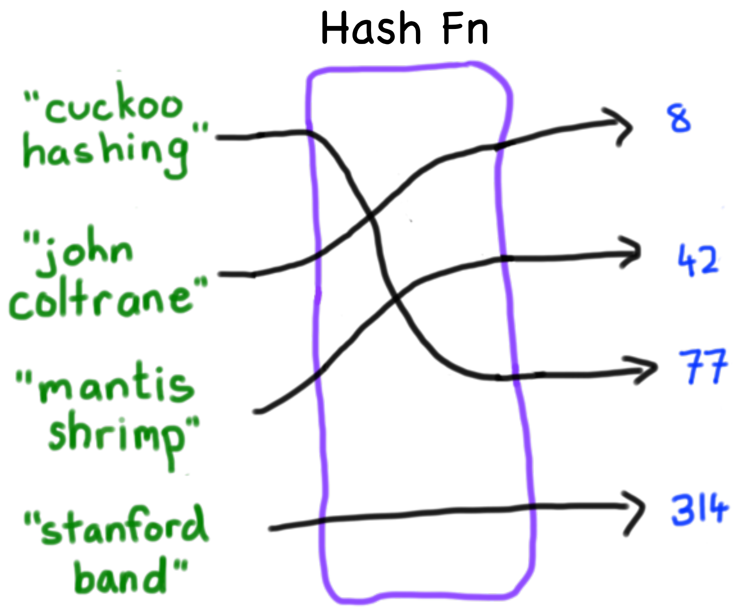
Iterating over a map doesn't return elements in sorted order

Iteration



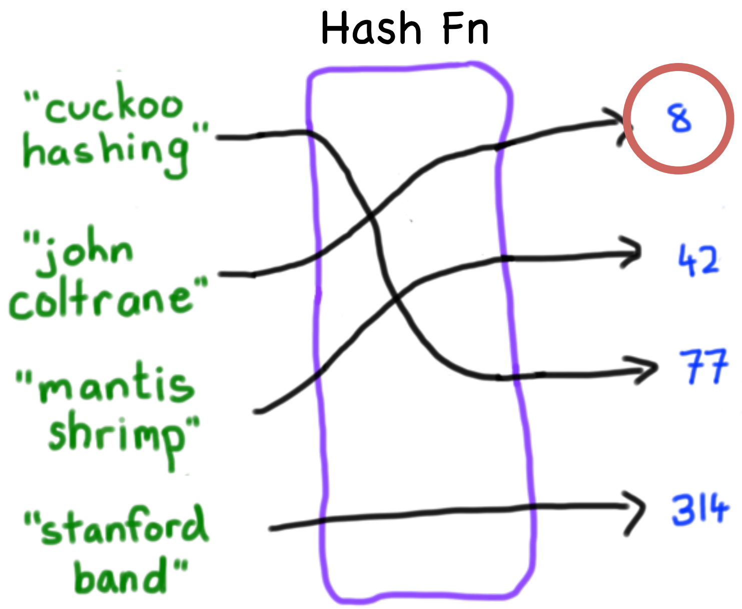
Iterating over a map doesn't return elements in sorted order

Iteration?



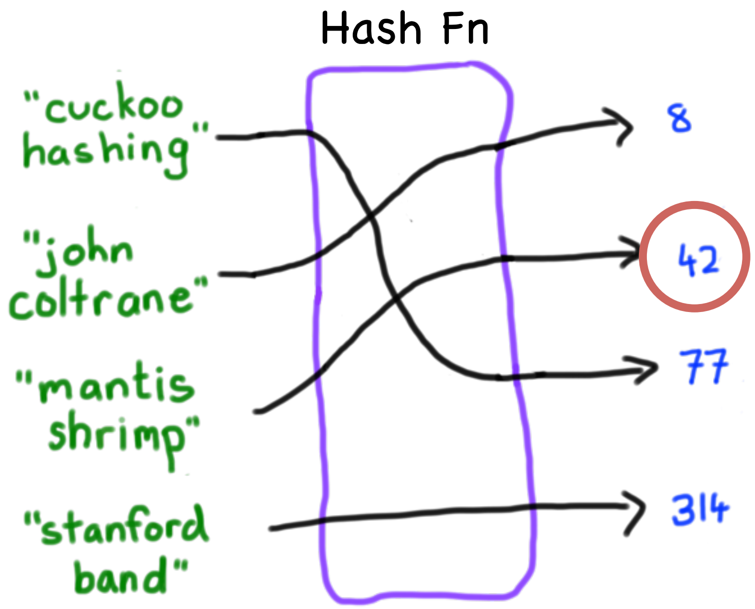
Iterating over a map doesn't return elements in sorted order

Iteration?



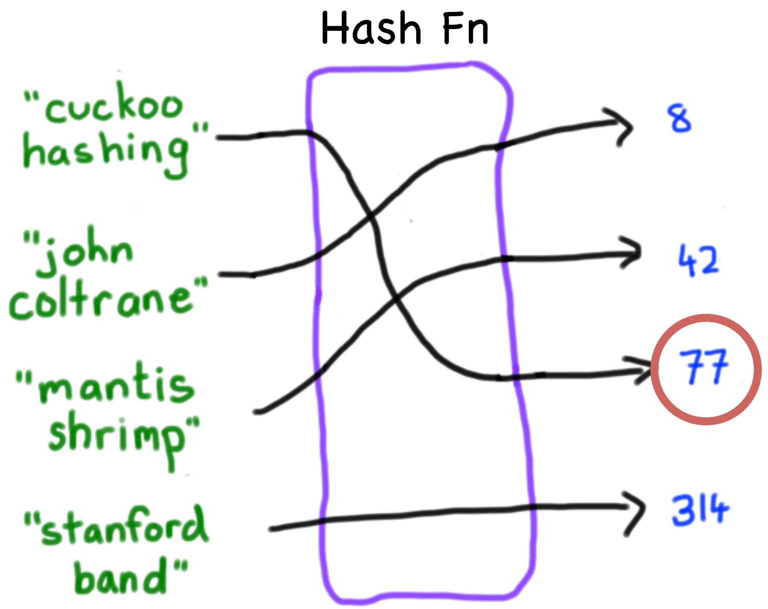
Iterating over a map doesn't return elements in sorted order

Iteration?



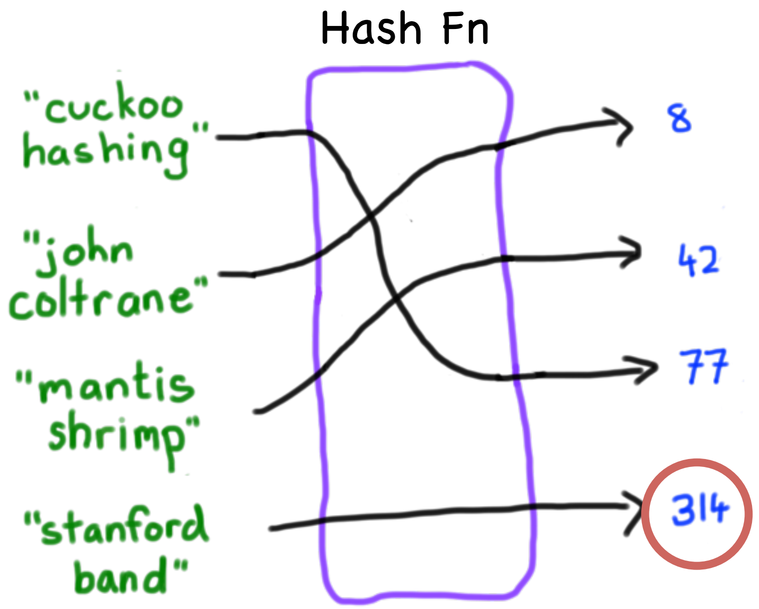
Iterating over a map doesn’t return elements in sorted order

Iteration?



Iterating over a map doesn't return elements in sorted order

Iteration?



Iterating over a map doesn't return elements in sorted order

Maps and Sets Print in Order

[Suspense]

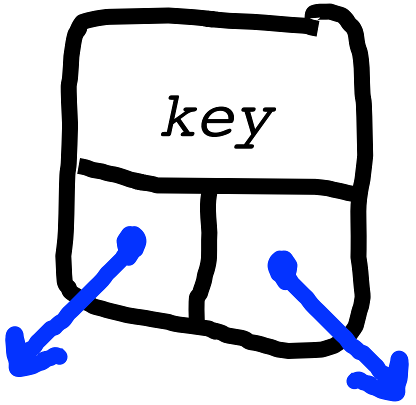
Binary $\vee$ Tree

Search

Binary $^{\vee}$ Tree

Search

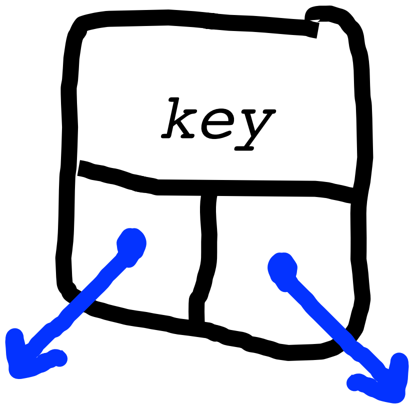
Binary Tree



Binary $\vee$ Tree

Search

Binary Tree

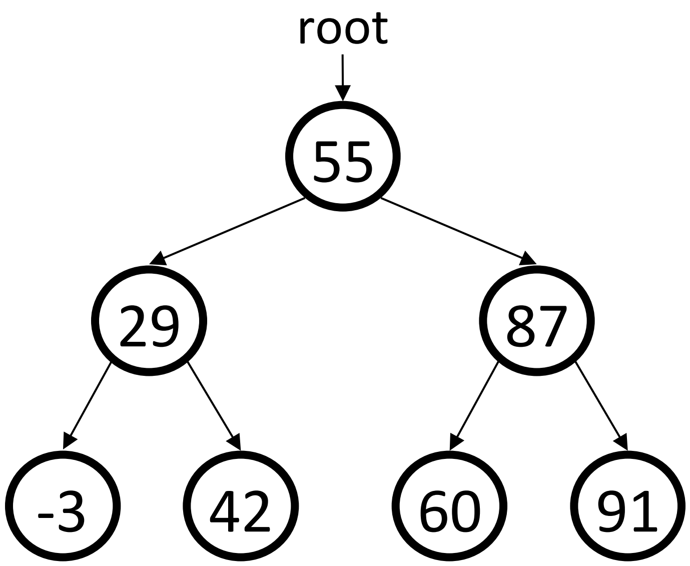


Search Tree

For a tree ***root***:

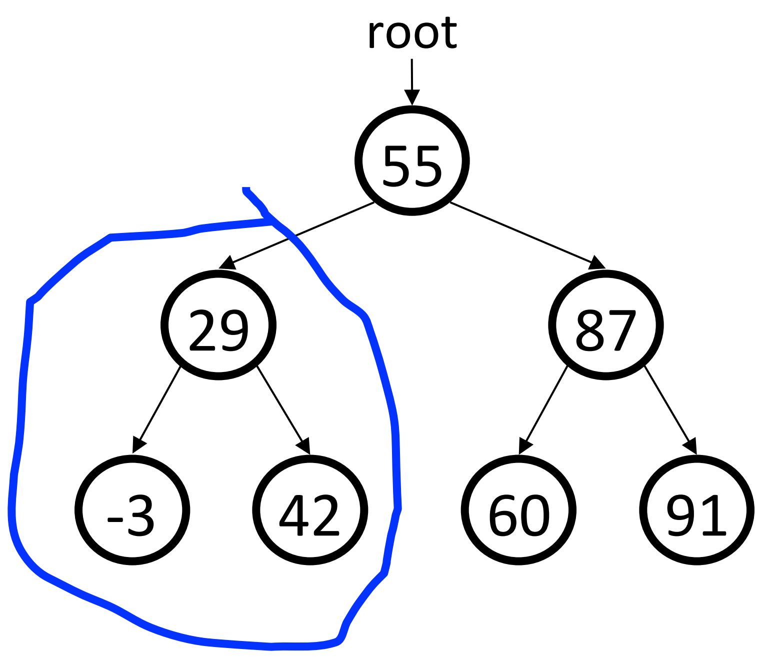
1. Every node in root's left subtree has a key < root's key
2. Every node of root's right subtree has a key > root's key
3. All children of root are also binary search trees

Binary Search Tree



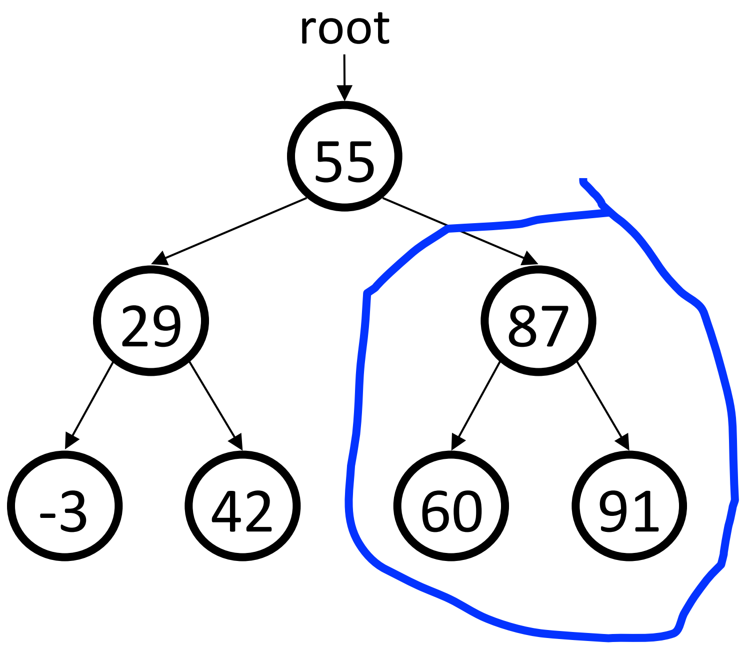
What makes a tree a Binary Search Tree?

Binary Search Tree



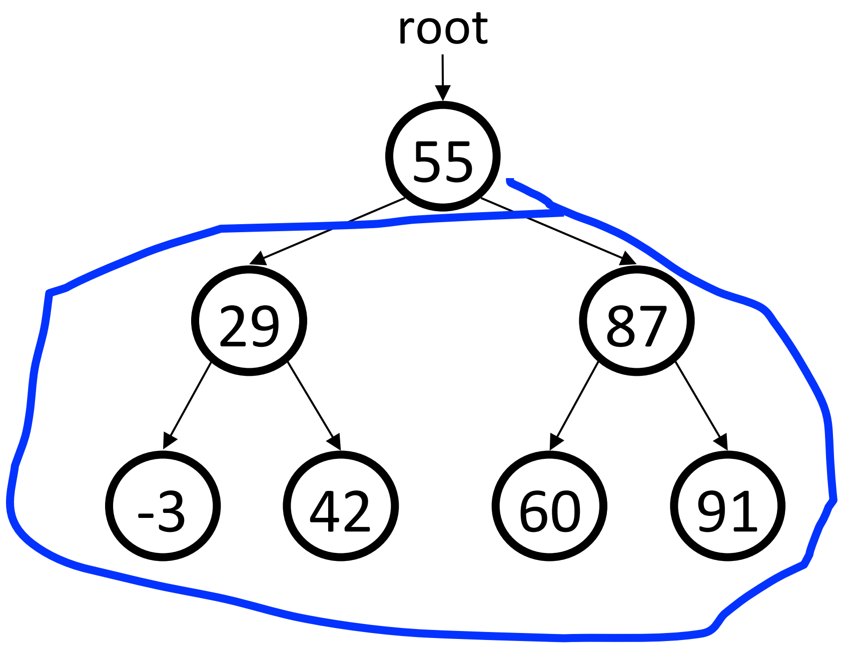
1. Every node in root's left subtree has a key < root's key

Binary Search Tree



2. Every node in root's right subtree has a key > root's key

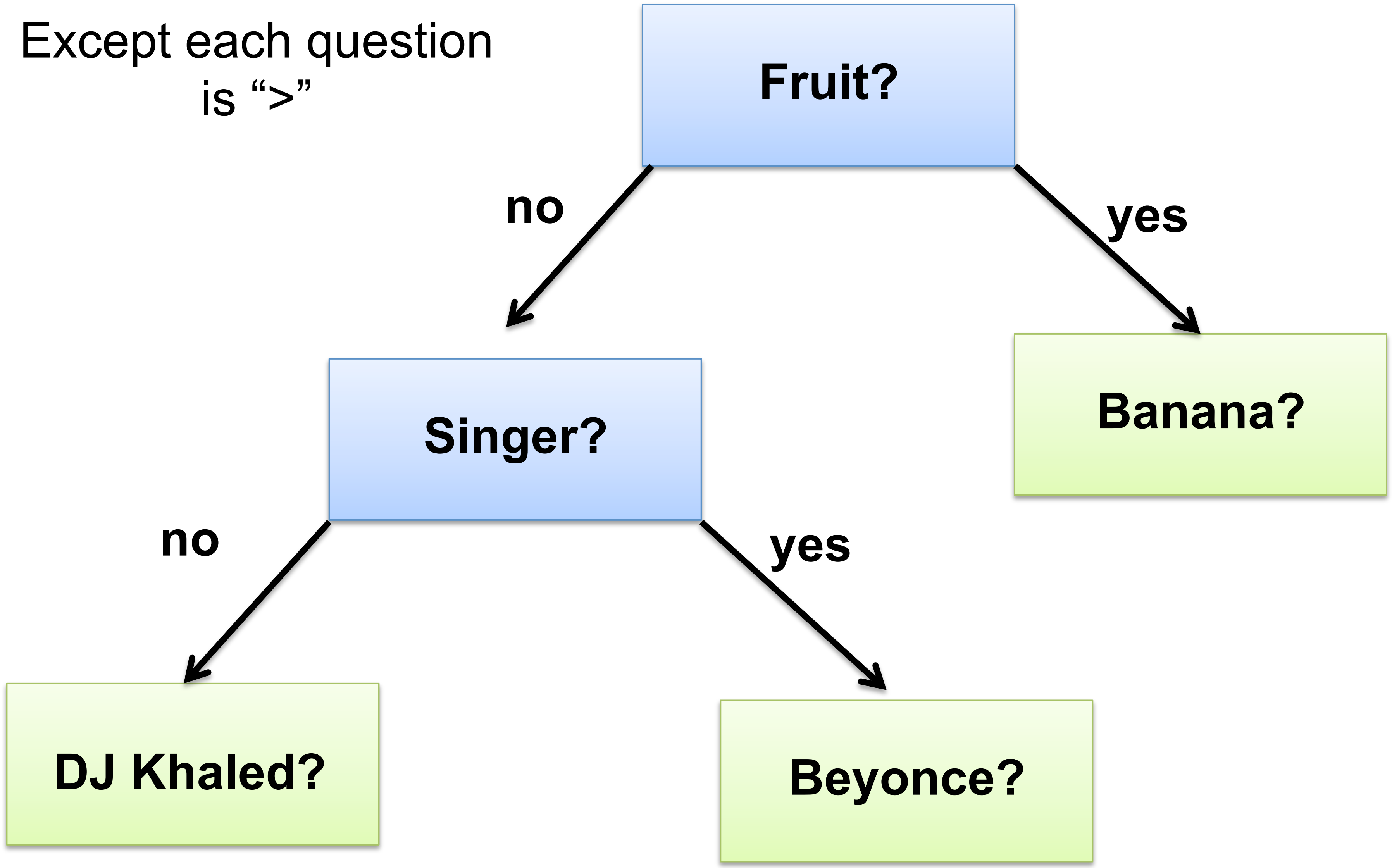
Binary Search Tree



3. All children of root are also binary search trees.

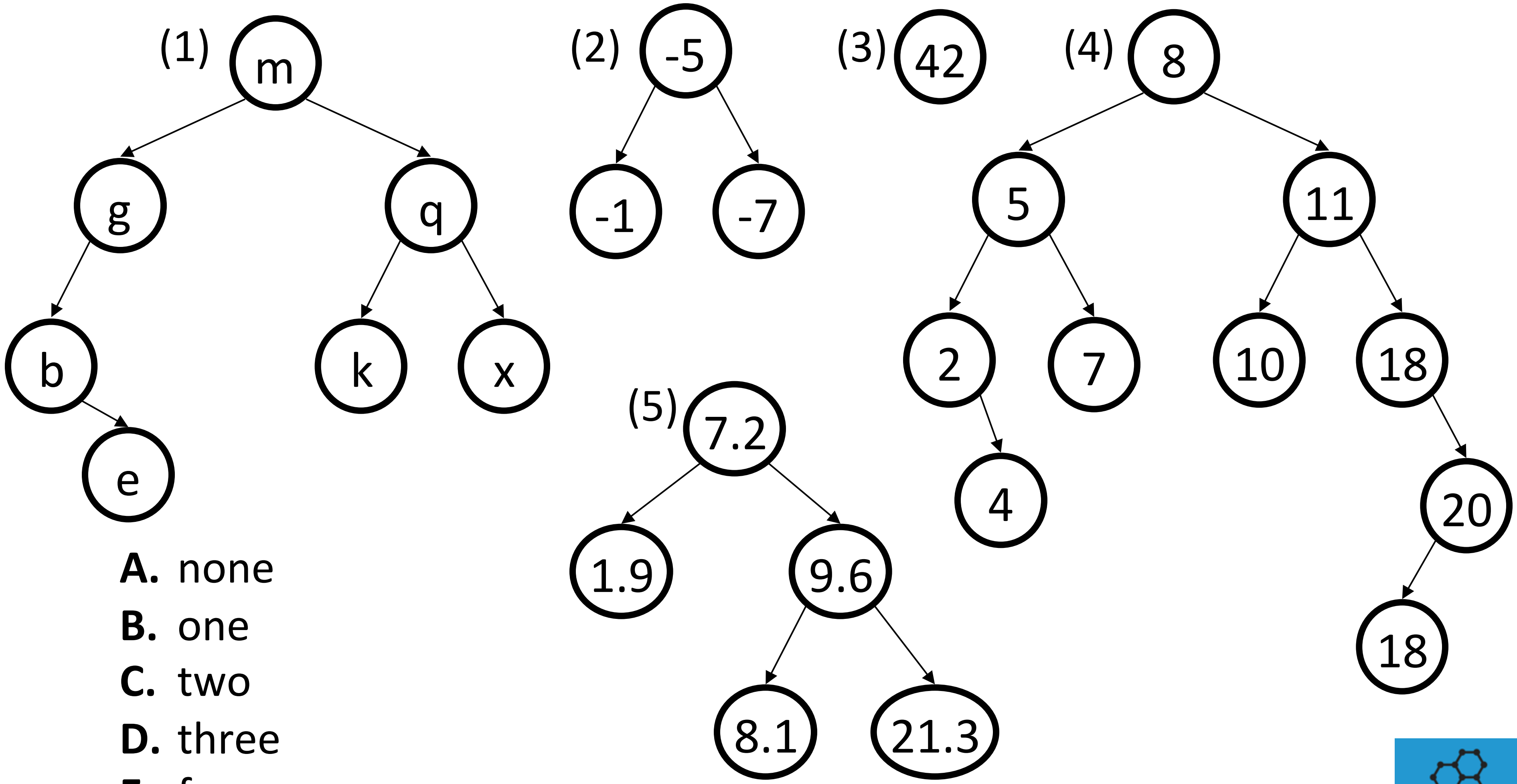
Like a Pensive

Except each question is “>”



BST examples

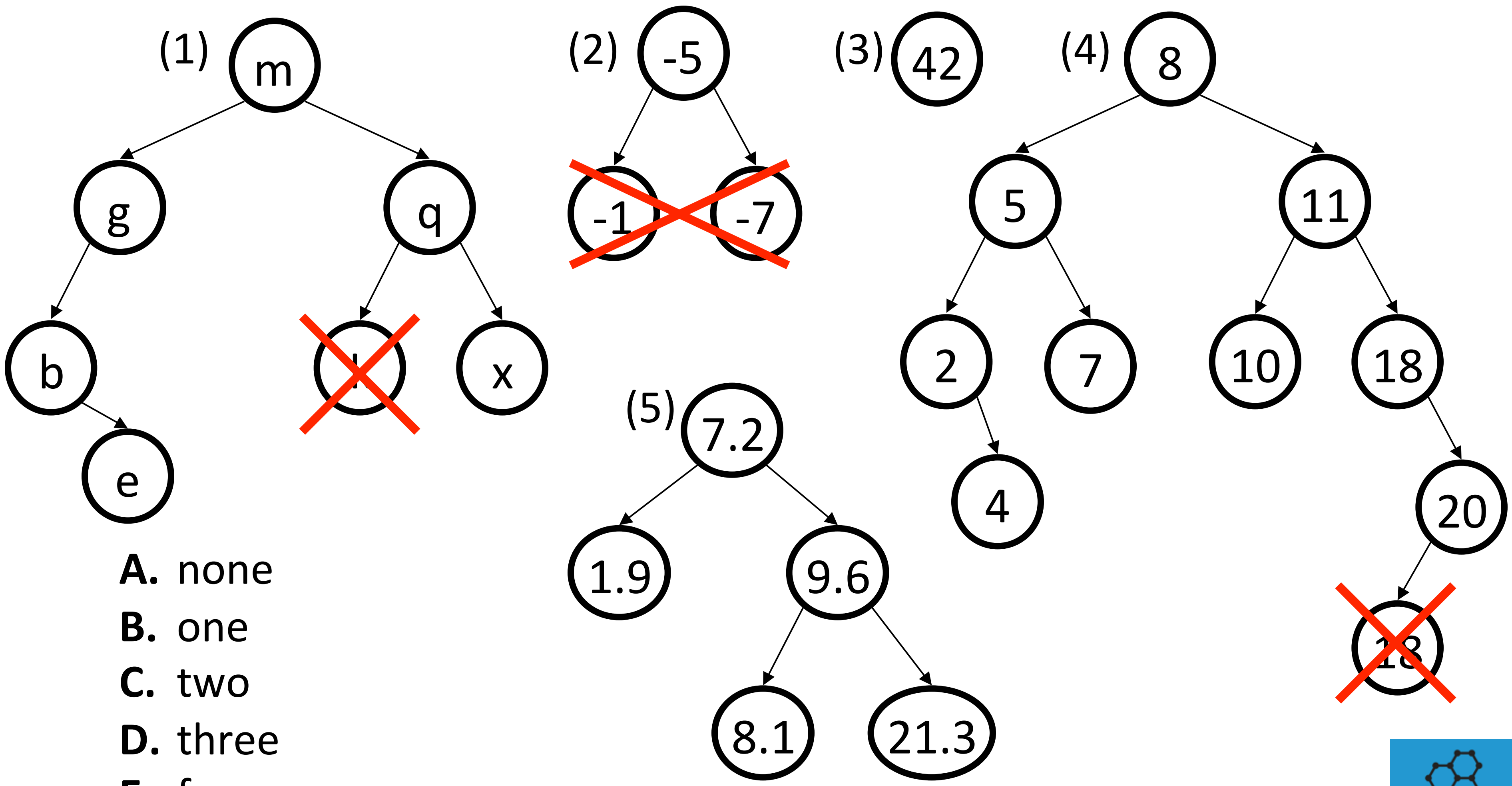
Q: How many of the trees shown are legal binary search trees?



A. none
B. one
C. two
D. three
E. four or more

BST examples

Q: How many of the trees shown are legal binary search trees?

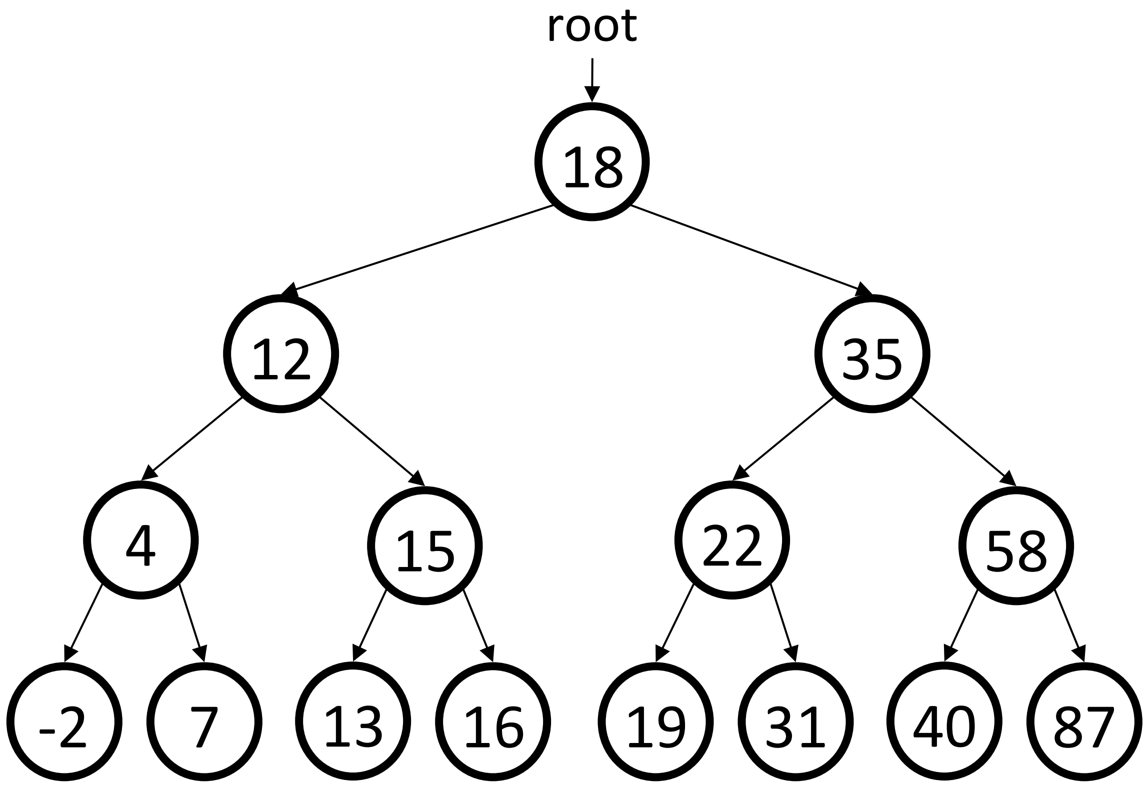


A. none
B. one
C. two
D. three
E. four or more

STUDENT

Searching a BST

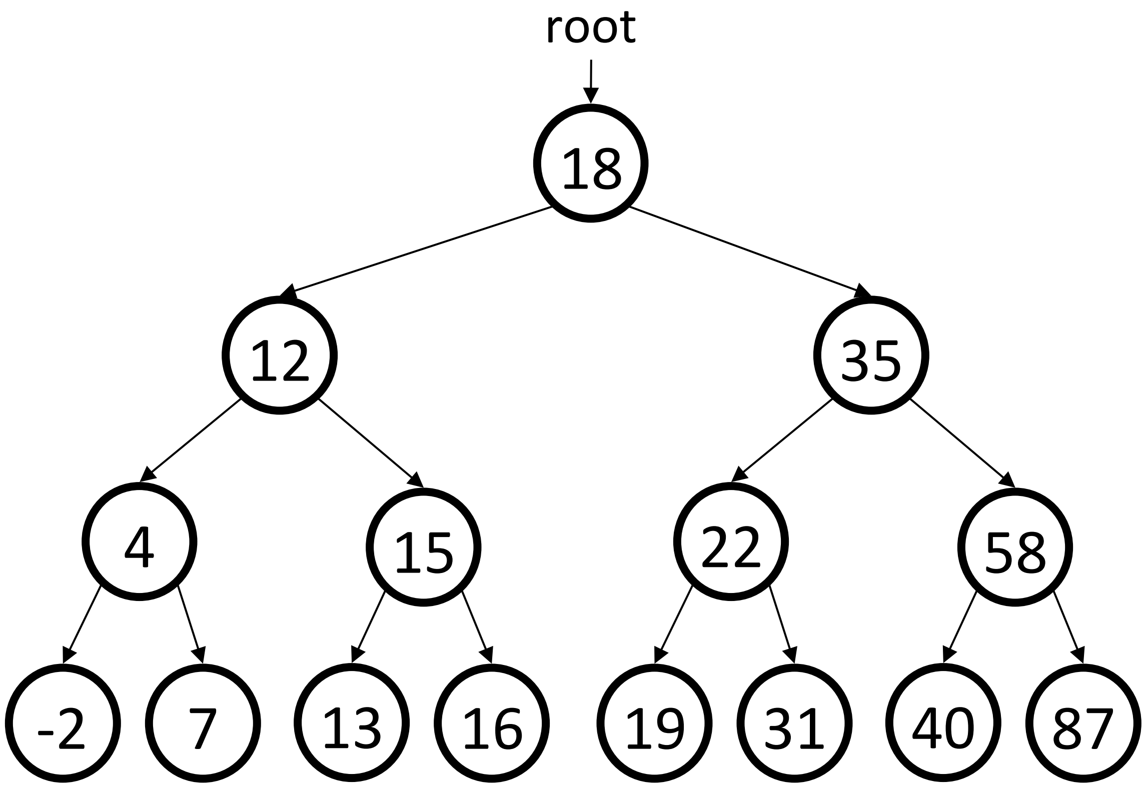
1. Try searching for the value 31, then 6 from the root



Describe your algorithm

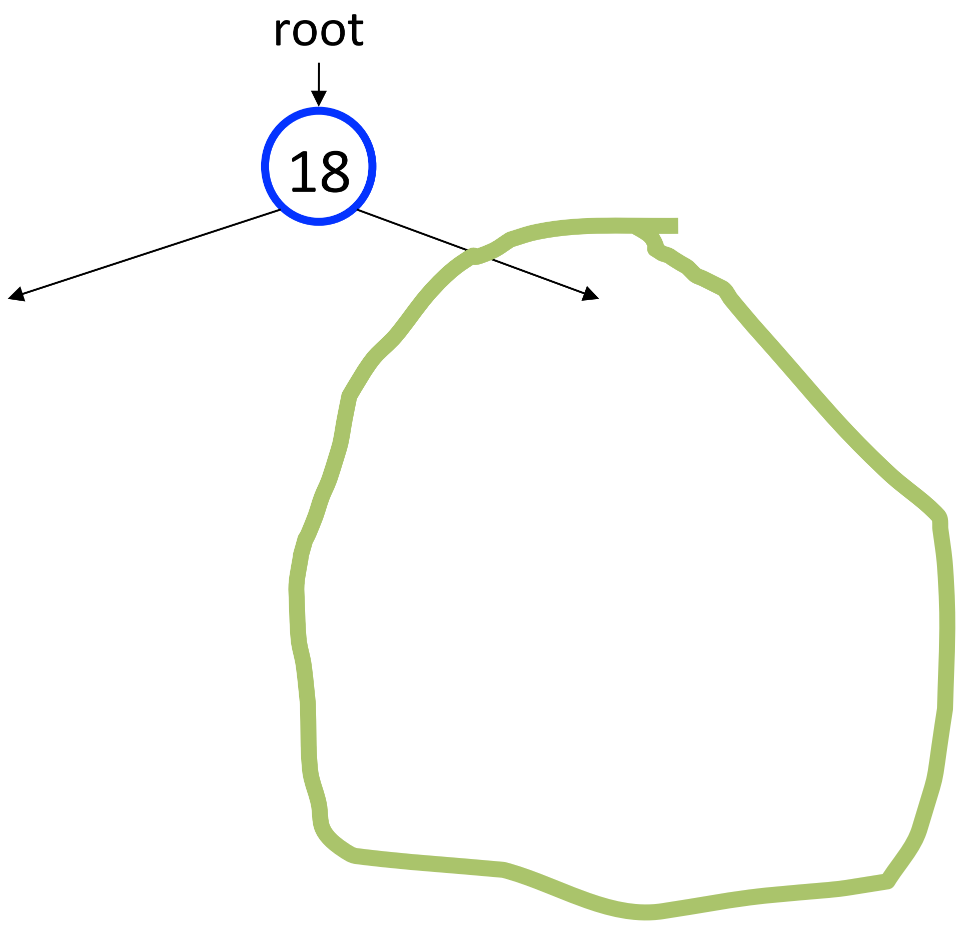
Searching a BST

2. Describe your algorithm



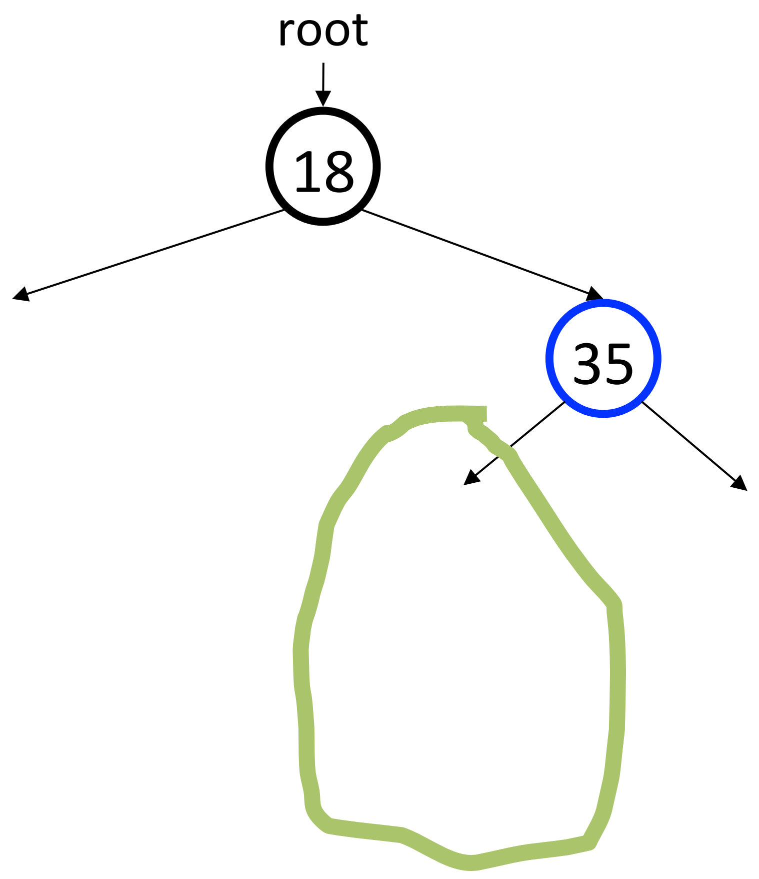
Searching a BST

Searching for the value **31** from the root



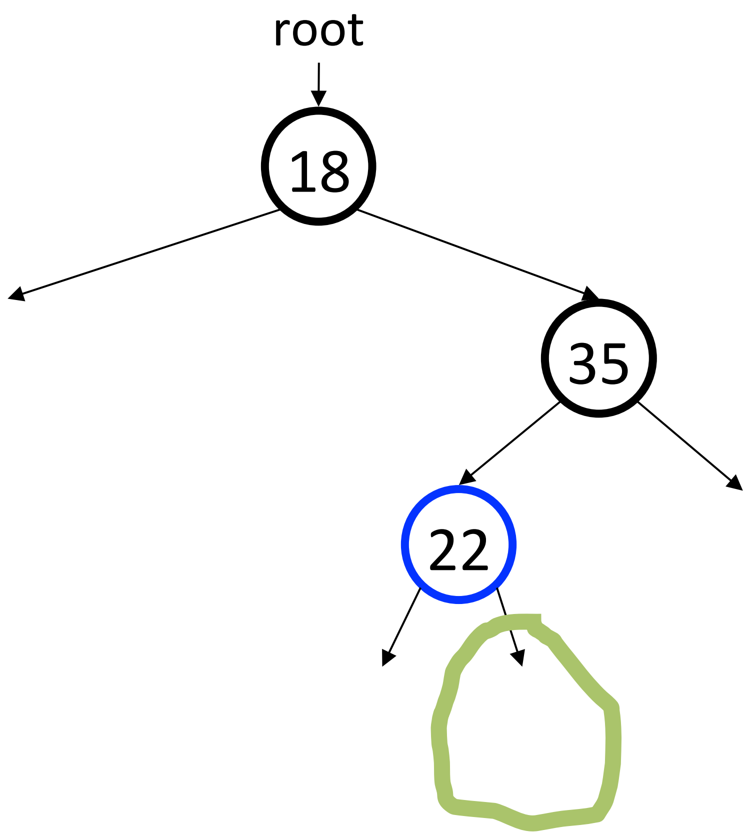
Searching a BST

Searching for the value **31** from the root



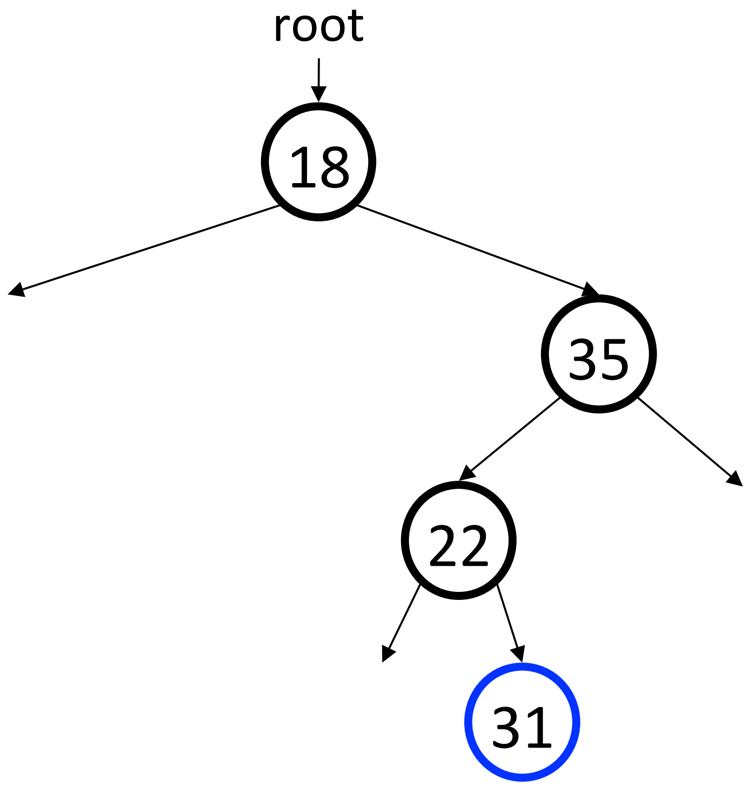
Searching a BST

Searching for the value **31** from the root



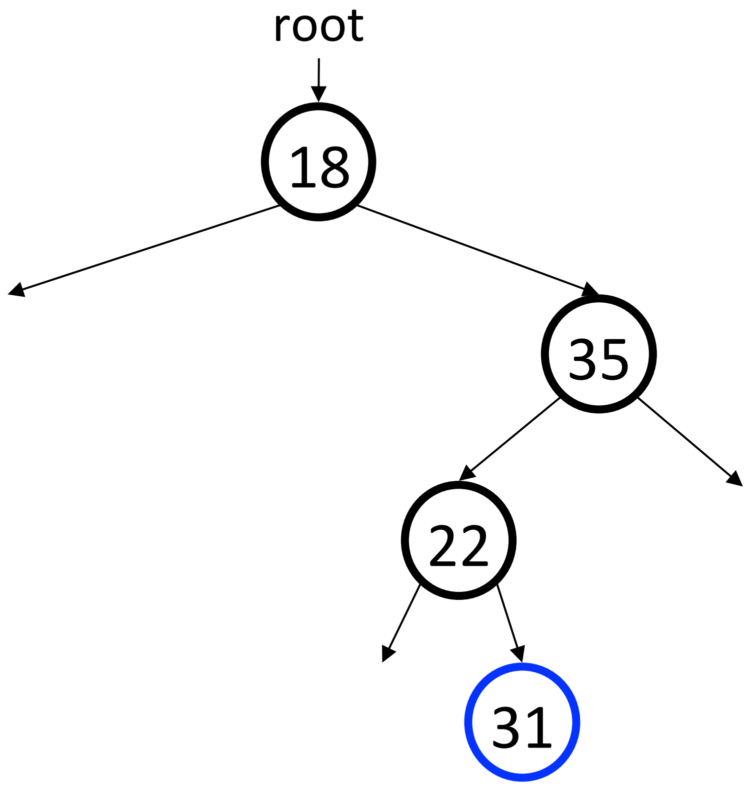
Searching a BST

Searching for the value **31** from the root



Searching a BST

Searching for the value **31** from the root



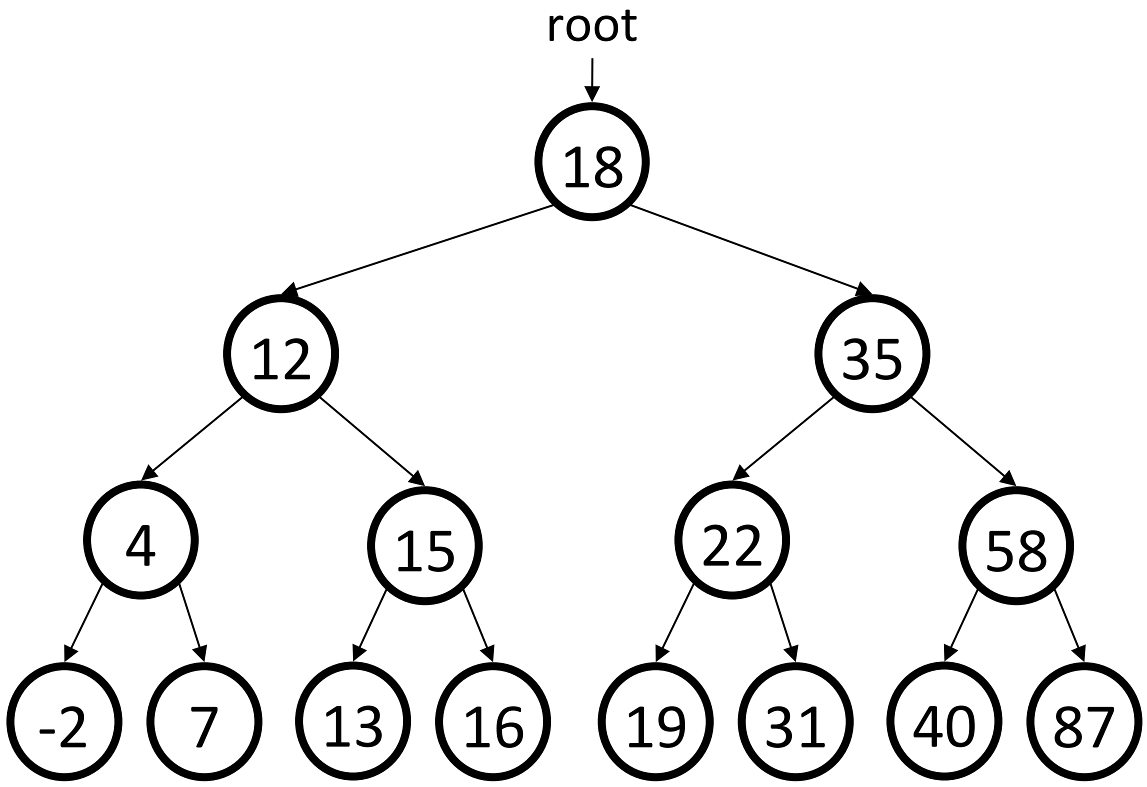
Contains Algorithm

```
bool contains(Tree * tree, int value) {
  // you hit a leaf. Only happens if no value
  if(tree == NULL) return false;
  // you found it! Yes tree contains value.
  if(tree->value == value) return true;

  if(value < tree->value) {
    // if value is less, recurse on left.
    return contains(tree->left);
  } else {
    // if value is greater, recurse on right.
    return contains(tree->right);
  }
}
```

Big O?

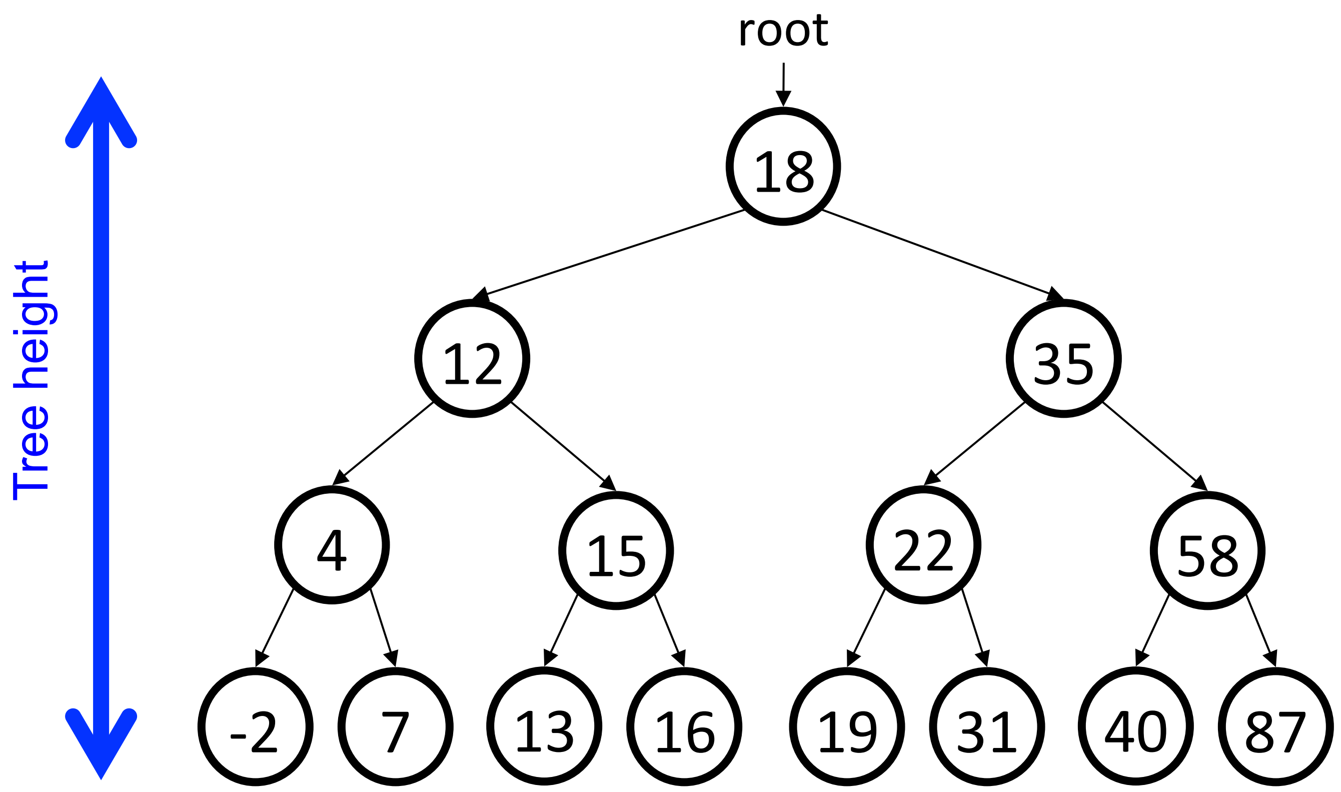
Searching a BST

3. What's the maximum number of nodes to check?



Searching a BST

3. What's the maximum number of nodes to check?

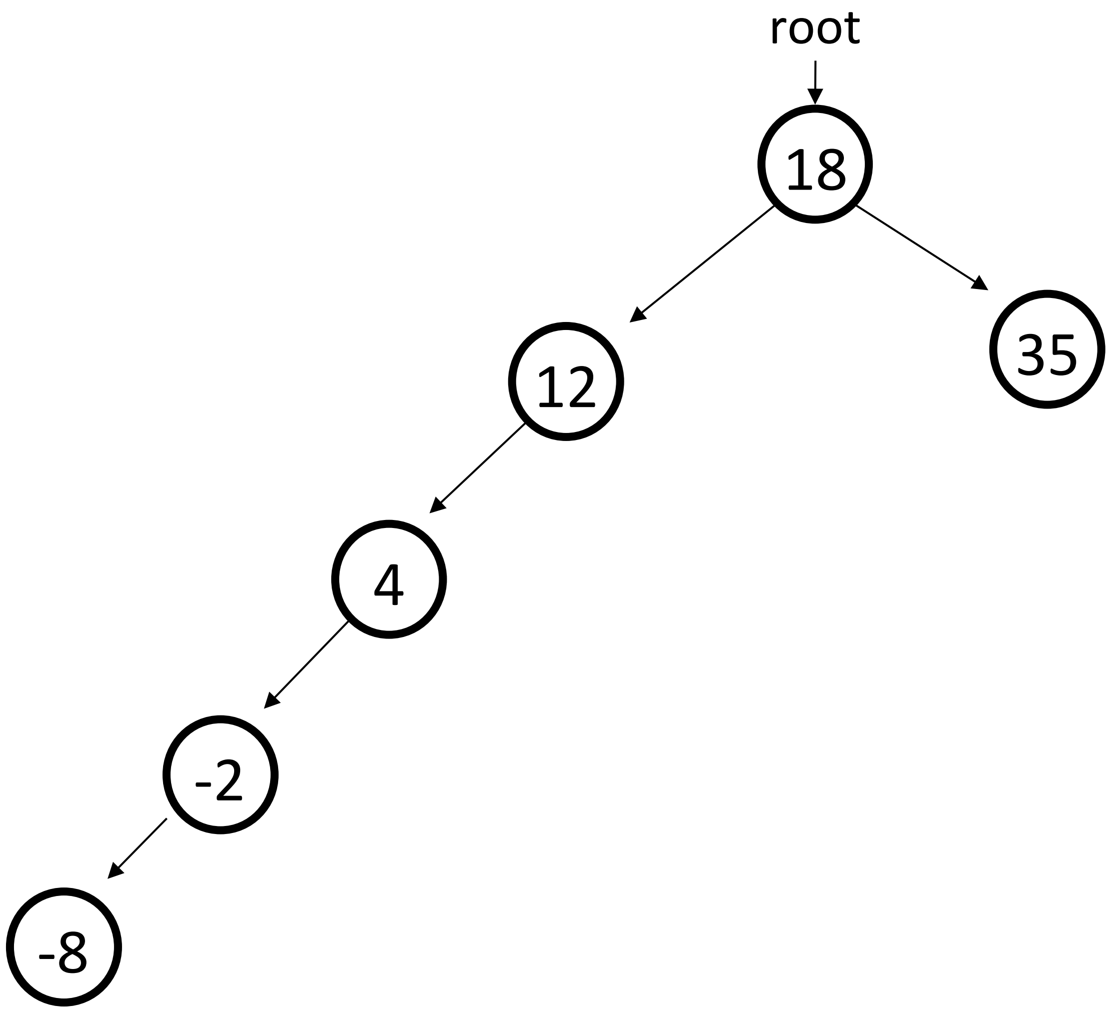


Contains Big O

$$\mathcal{O}(\log n)$$

* Assuming the tree is balanced

Unbalance



This is what an unbalanced tree looks like.

Add

Choice 1

```
void add(Tree * tree, int value) {
  if(tree == NULL) {
    tree = new Tree;
    tree->value = value;
    tree->left = NULL;
    tree->right = NULL;
 } else {
    if(value < tree->value) {
      add(tree->left, value);
    } else if(value > tree->value){
      add(tree->right, value);
    }
  }
}
```

Choice 2

```
void add(Tree * & tree, int value) {
  if(tree == NULL) {
    tree = new Tree;
    tree->value = value;
    tree->left = NULL;
    tree->right = NULL;
  } else {
    if(value < tree->value) {
      add(tree->left, value);
    } else if(value > tree->value){
      add(tree->right, value);
    }
  }
}
```

Which one is better?

Add

Choice 1

```
void add(Tree * tree, int value) {
   if(tree == NULL) {
     tree = new Tree;
     tree->value = value;
     tree->left = NULL;
     tree->right = NULL;
  } else {
     if(value < tree->value) {
       add(tree->left, value);
     } else if(value > tree->value){
       add(tree->right, value);
     }
   }
}
```

Choice 2

```
void add(Tree * & tree, int value) {
   if(tree == NULL) {
     tree = new Tree;
     tree->value = value;
     tree->left = NULL;
     tree->right = NULL;
   } else {
     if(value < tree->value) {
       add(tree->left, value);
     } else if(value > tree->value){
       add(tree->right, value);
     }
   }
}
```

Which one is better?

Balanced Add and Remove

Leonidas Guibas

Note: Beyond Scope

Illustrating AVL

H

He

Li

Be

B

C

Illustrating AVL

H

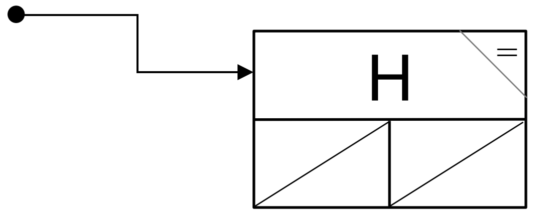
He

Li

Be

B

C



Illustrating AVL

H

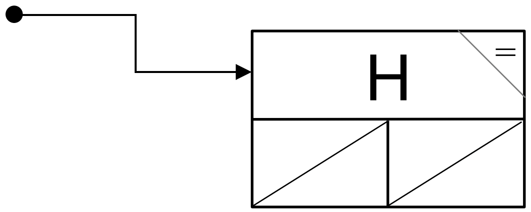
He

Li

Be

B

C



Illustrating AVL

H

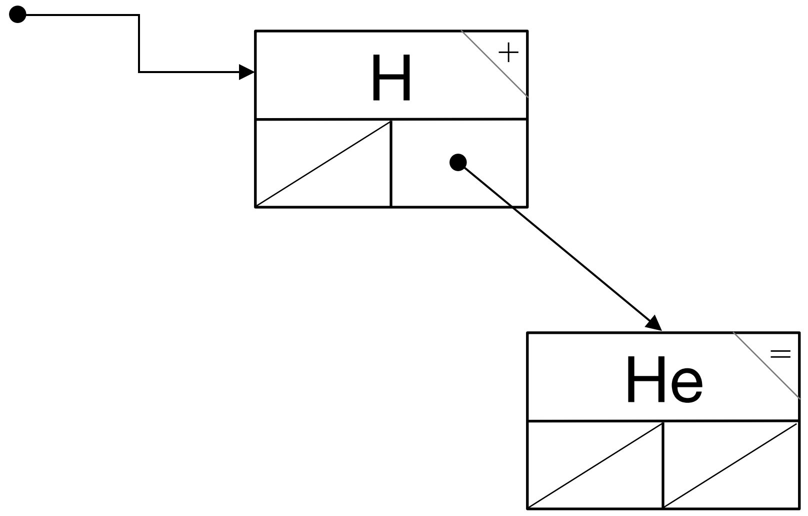
He

Li

Be

B

C



Illustrating AVL

H

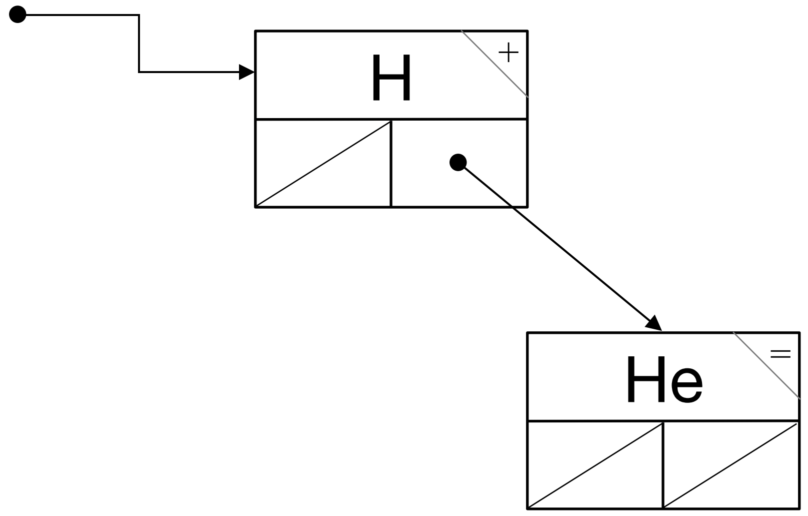
He

Li

Be

B

C



Illustrating AVL

H

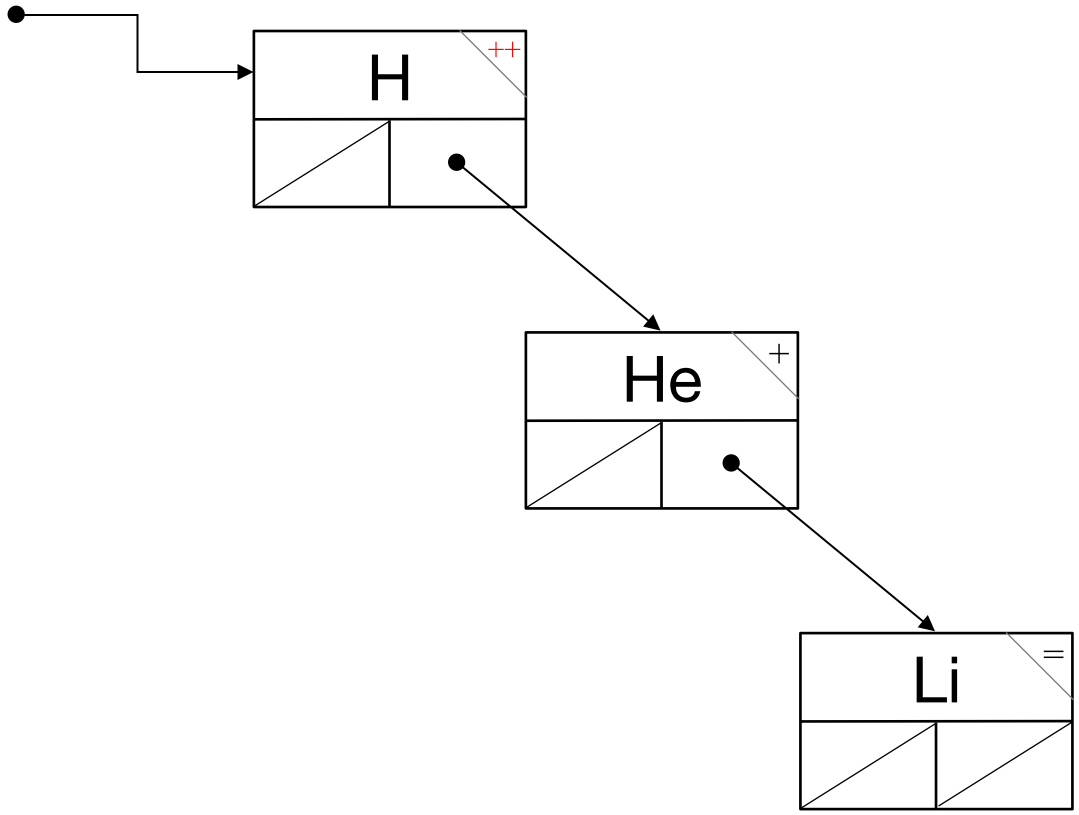
He

Li

Be

B

C



Illustrating AVL

H

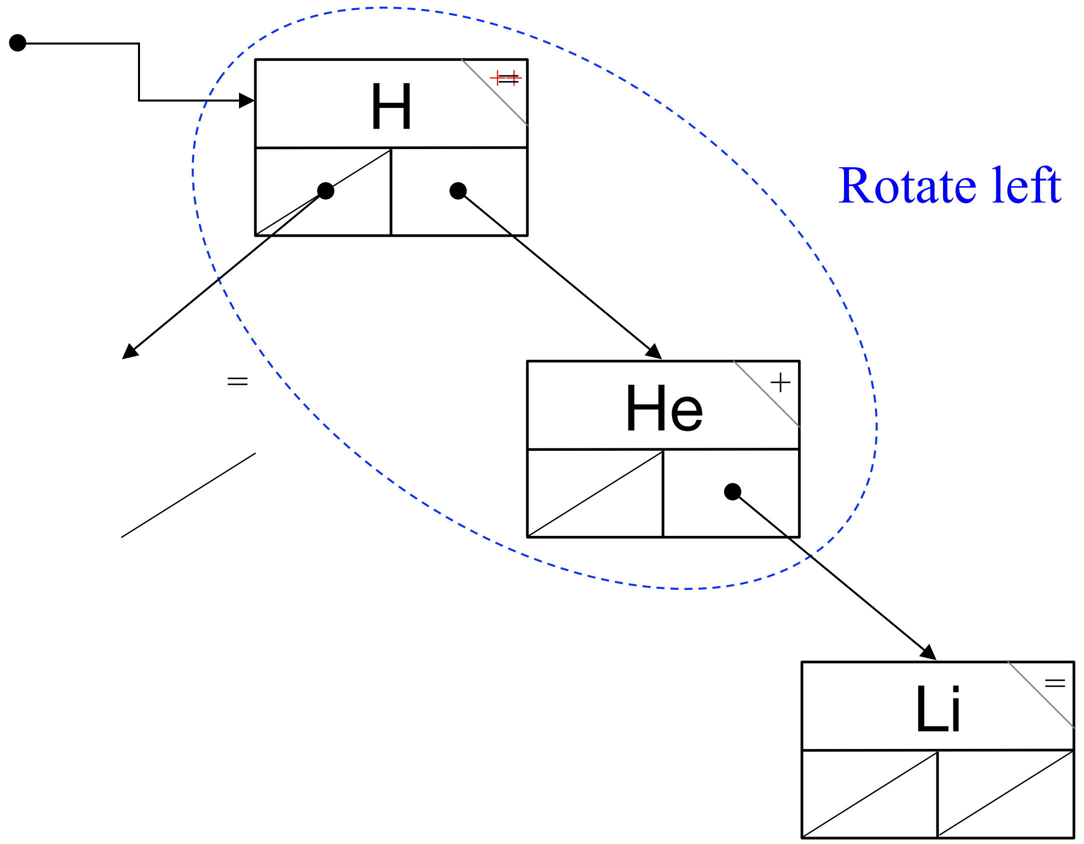
He

Li

Be

B

C



Illustrating AVL

H

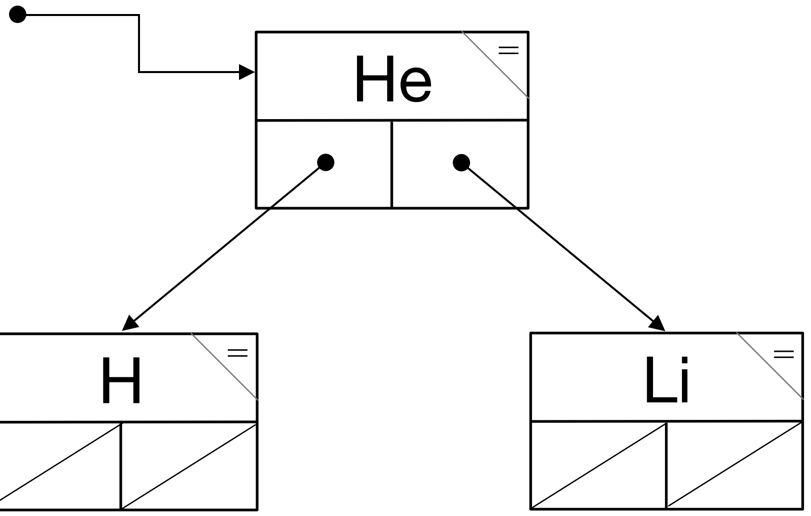
He

Li

Be

B

C



Illustrating AVL

H

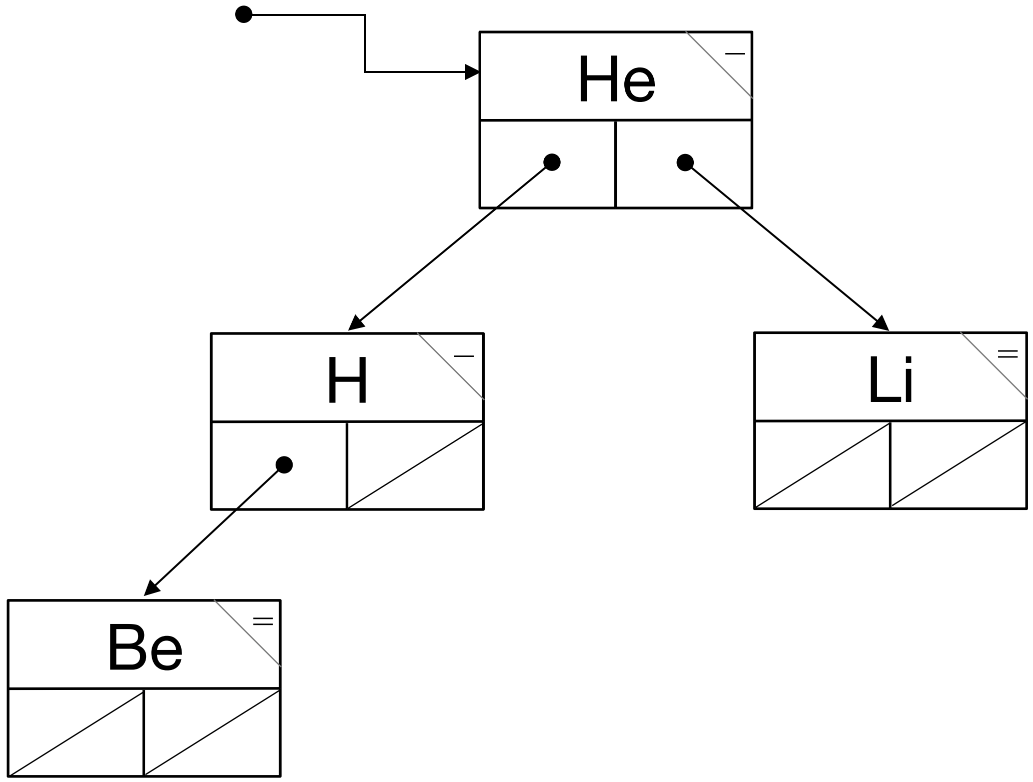
He

Li

Be

B

C



Illustrating AVL

H

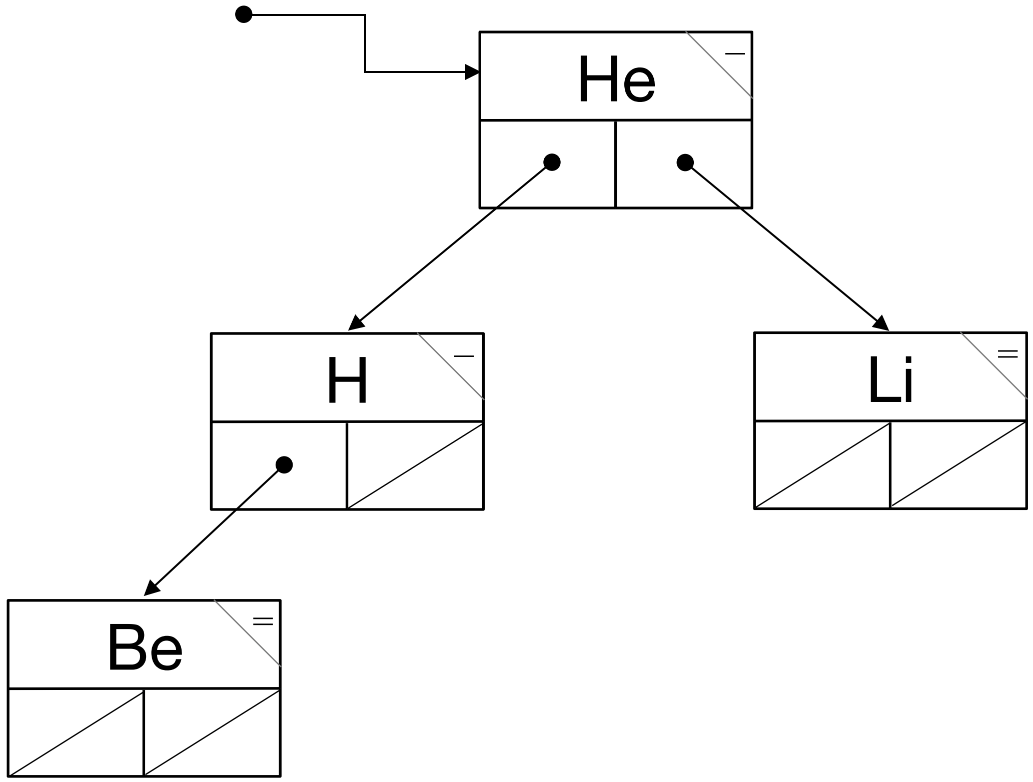
He

Li

Be

B

C



Illustrating AVL

H

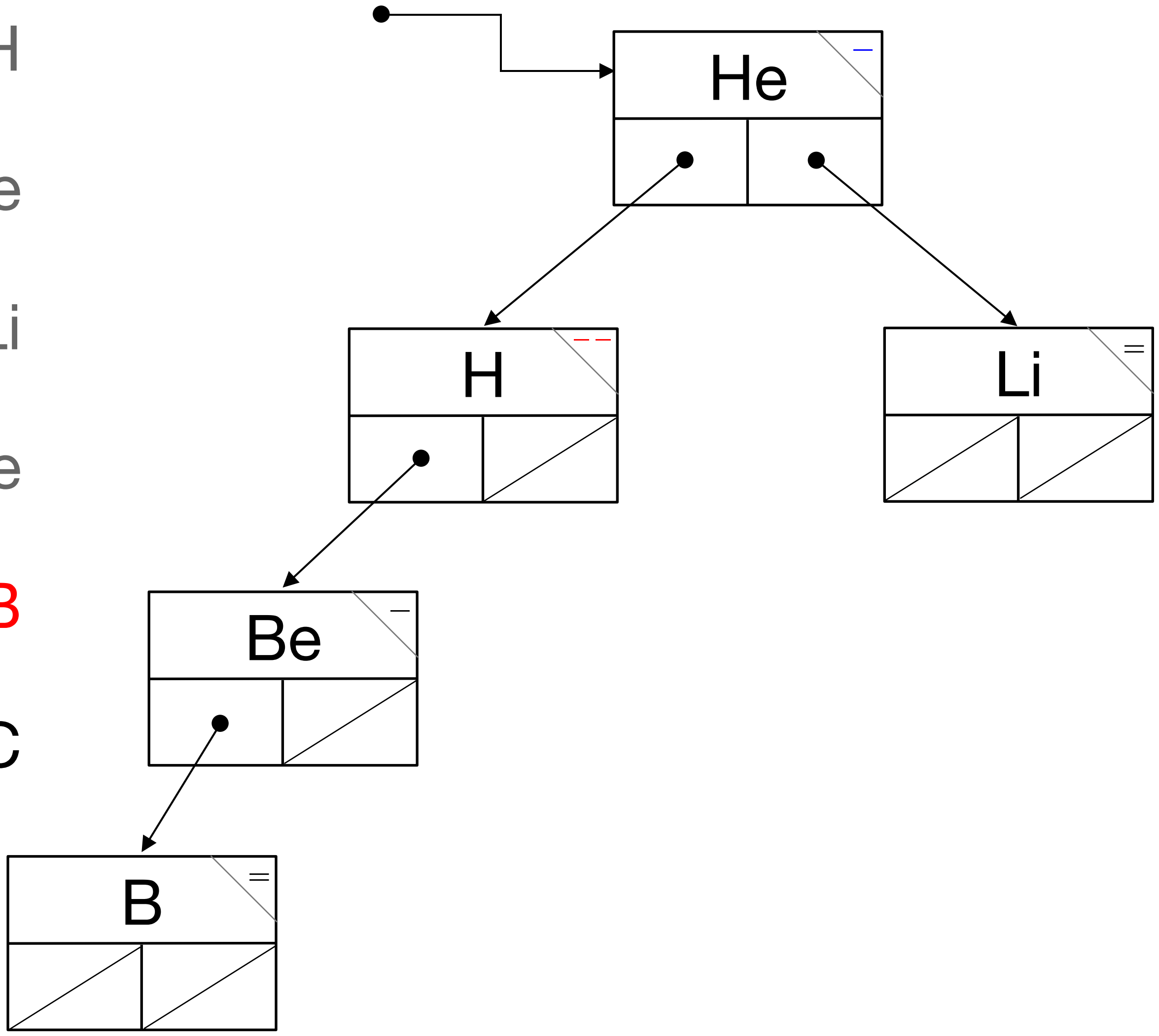
He

Li

Be

B

C



Illustrating AVL

H

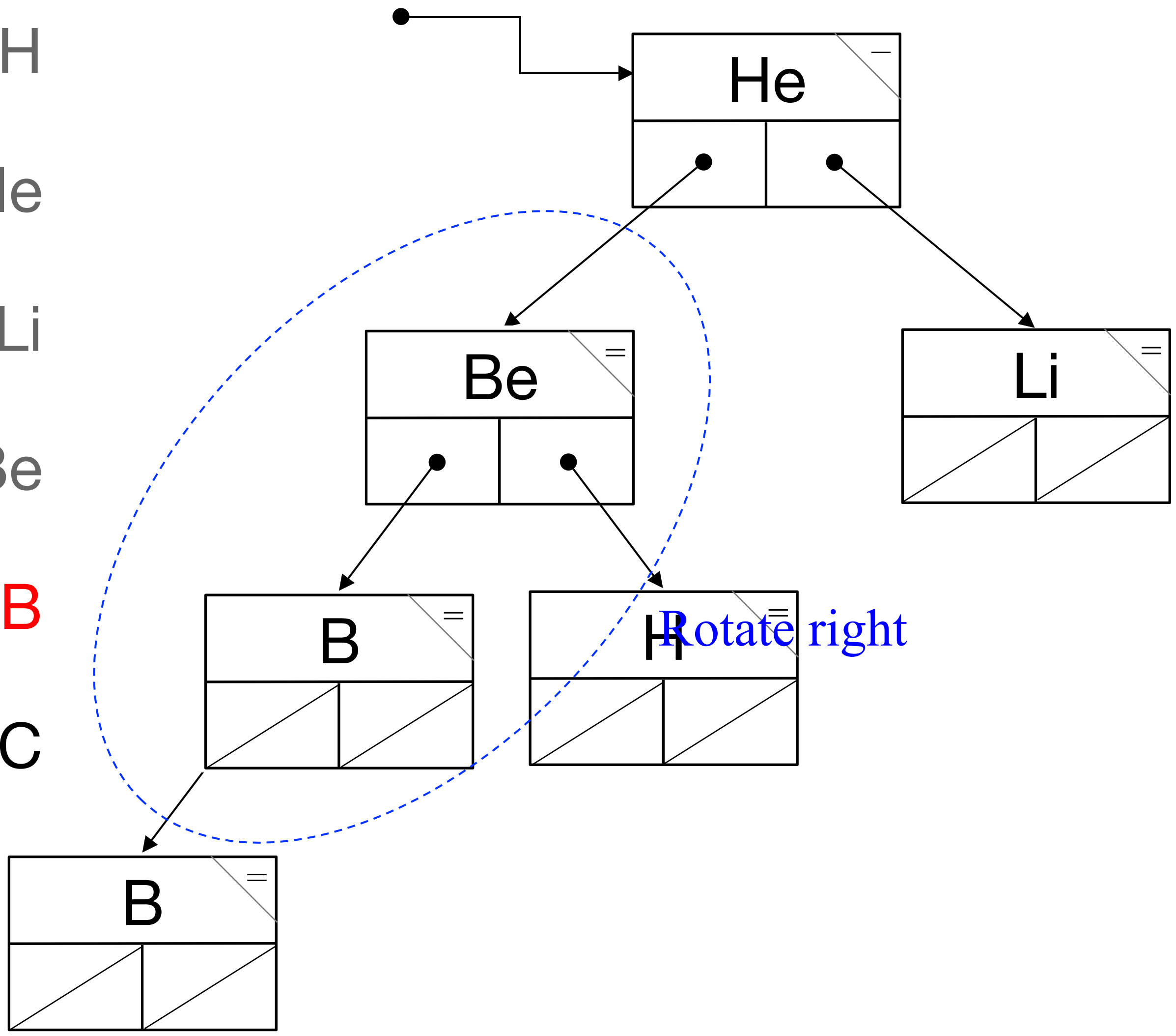
He

Li

Be

B

C



Illustrating AVL

H

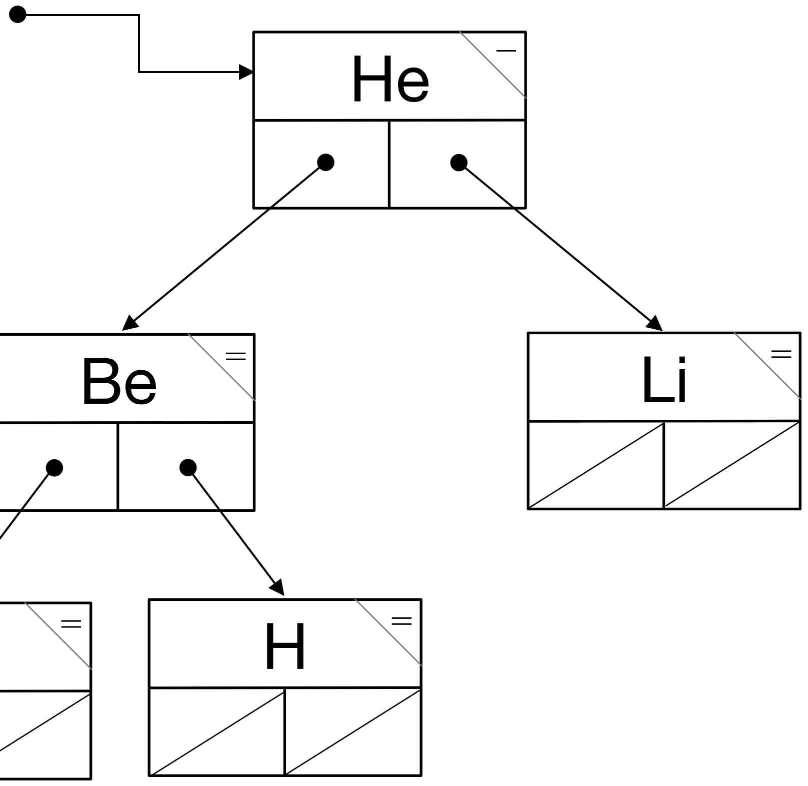
He

Li

Be

B

C



Illustrating AVL

H

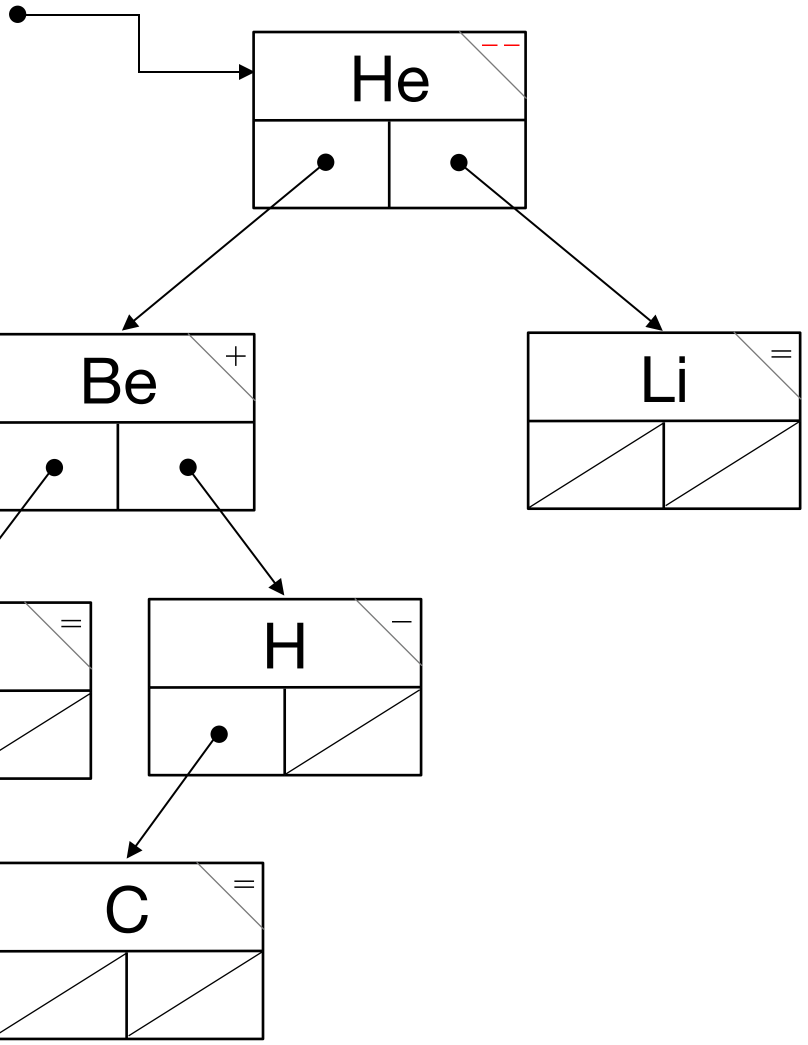
He

Li

Be

B

C



Illustrating AVL

H

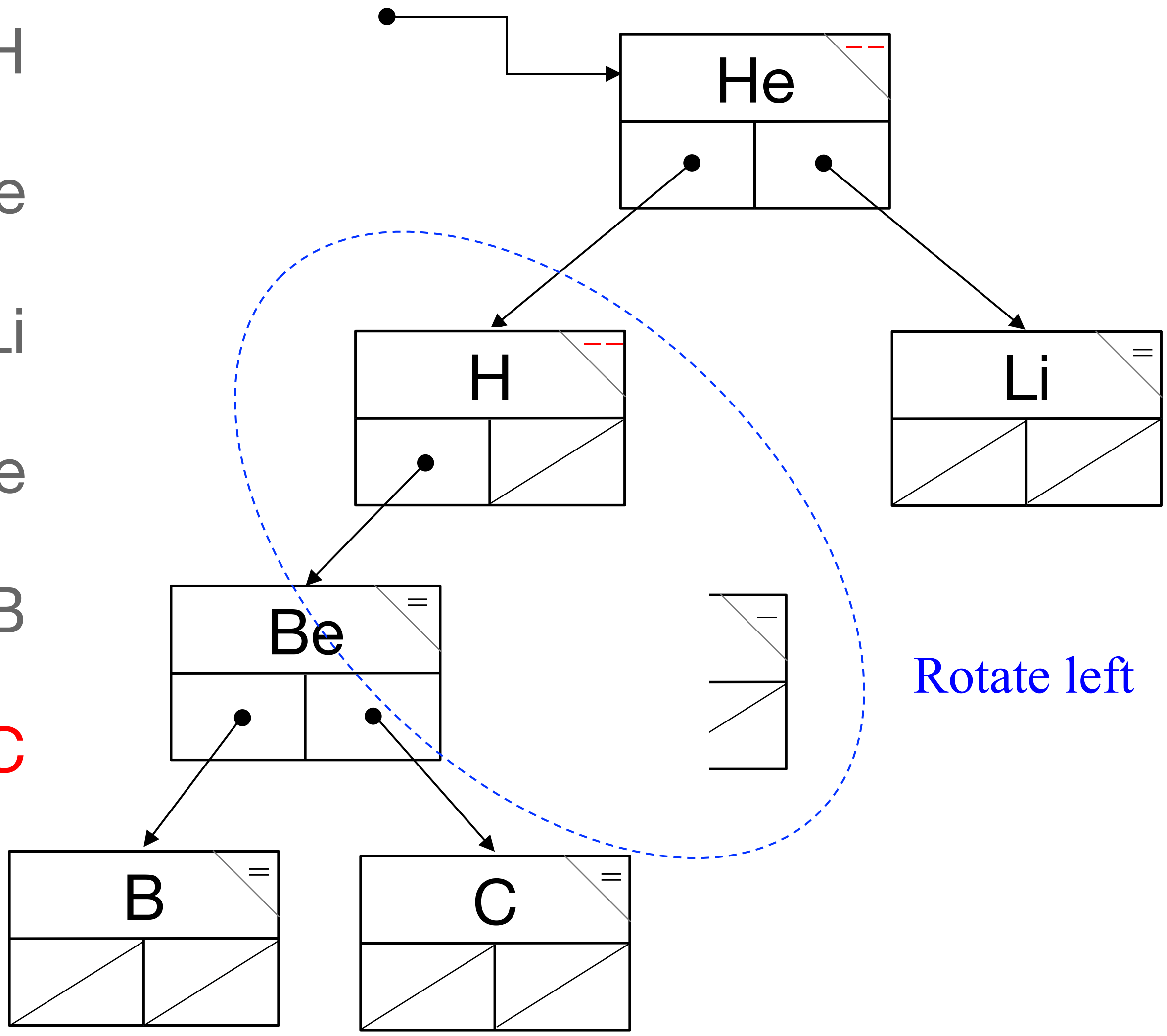
He

Li

Be

B

C



Illustrating AVL

H

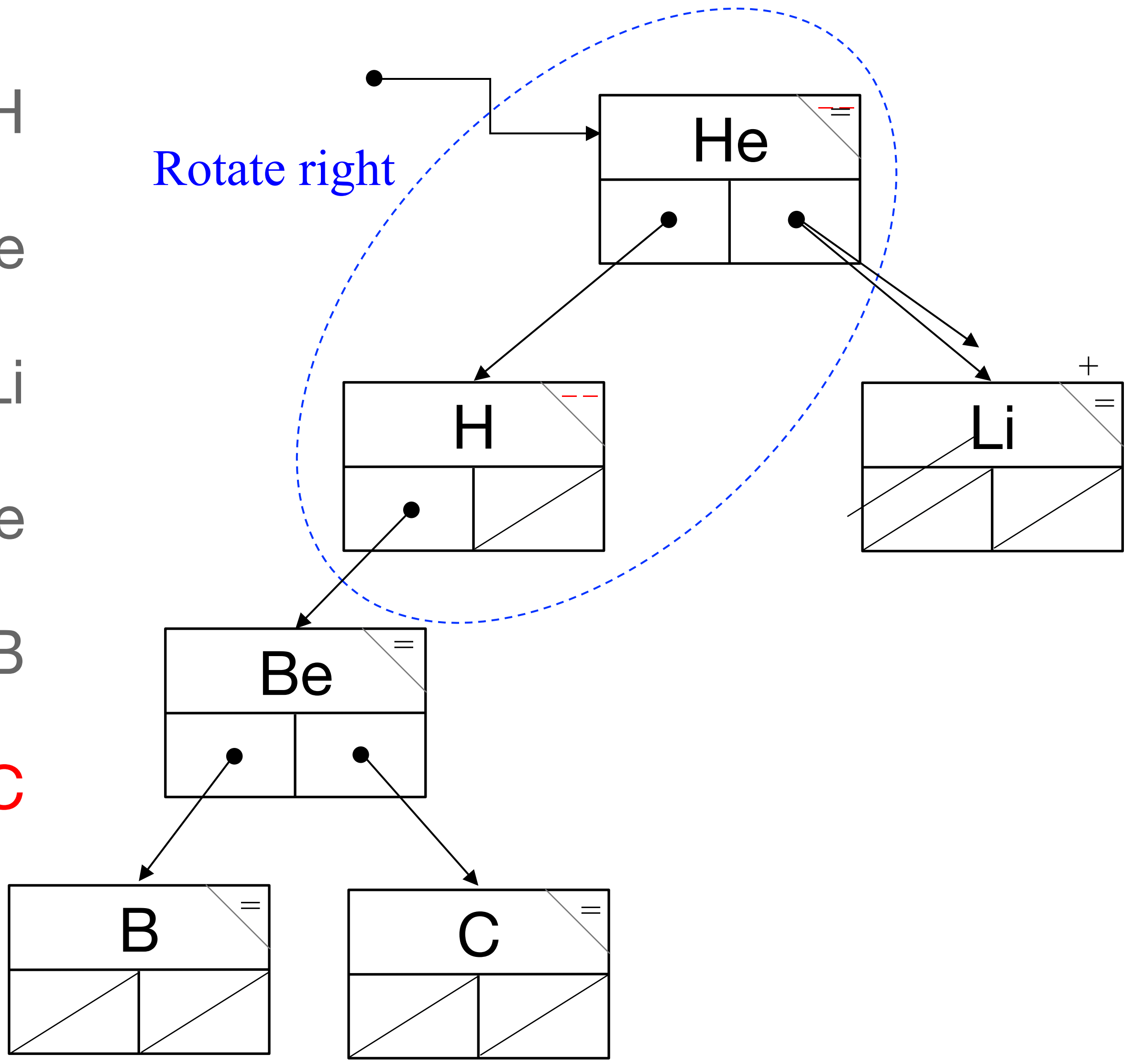
He

Li

Be

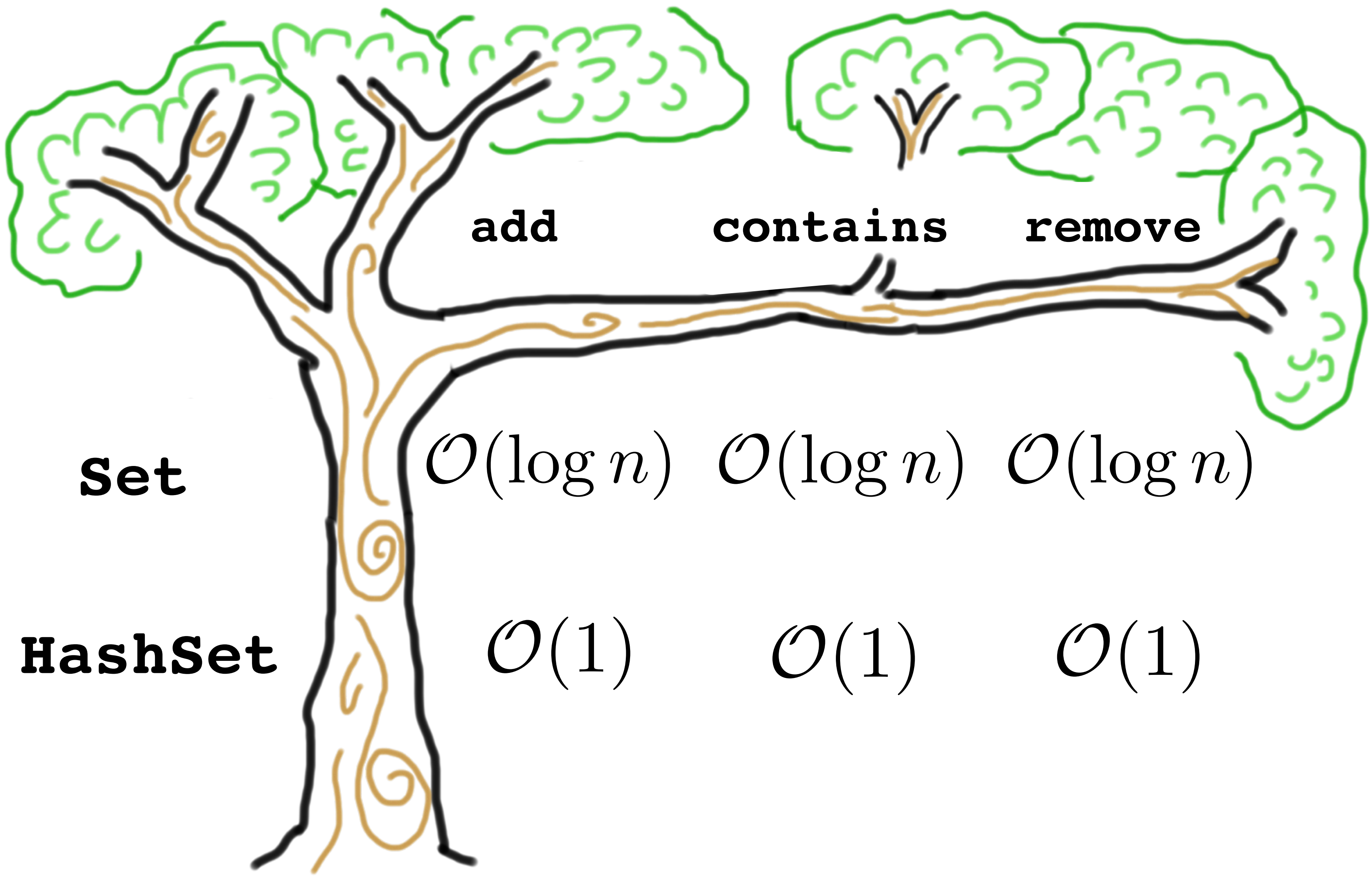
B

C



Back to Scope

Binary Search Tree Big O



	add	contains	remove
Set	$\mathcal{O}(\log n)$	$\mathcal{O}(\log n)$	$\mathcal{O}(\log n)$
HashSet	$\mathcal{O}(1)$	$\mathcal{O}(1)$	$\mathcal{O}(1)$

Hash Map is faster…

Log *n*

Lets say *n* is 1 **billion**

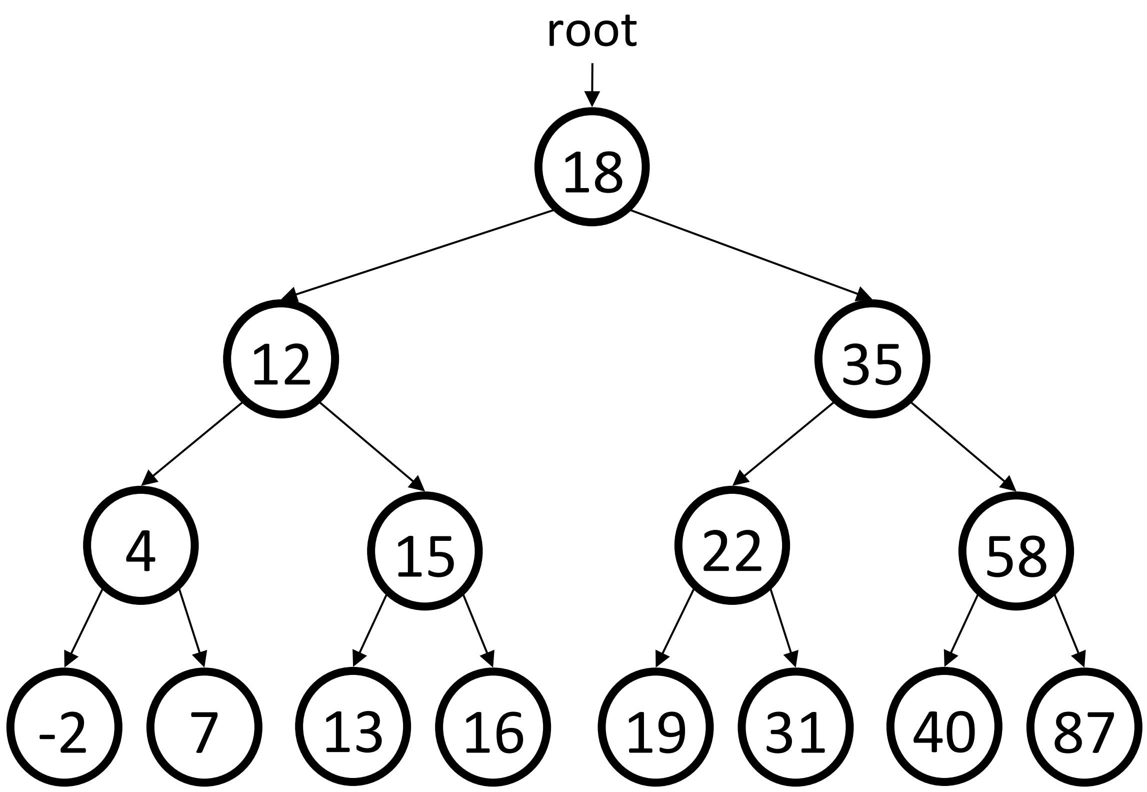
$$\log_2(billion) = 30$$

..But not by much

How does the Map/Set work?

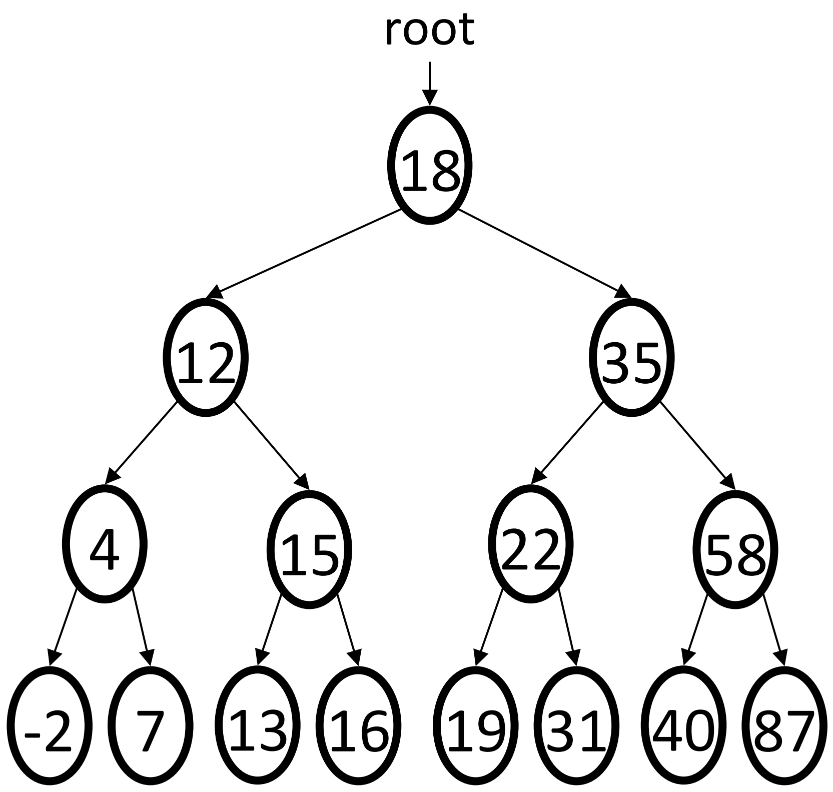
Maps and Sets Print in Order

Iteration?



Tree Traversal

```
void preOrder(Tree * tree) {
    if(tree == NULL) return;
    cout<<tree->value<<" ";
    preOrder(tree->left);
    preOrder(tree->right);
}

void inOrder(Tree * tree) {
    if(tree == NULL) return;
    inOrder(tree->left);
    cout<<tree->value<<" ";
    inOrder(tree->right);
}

Void postOrder(Tree * tree) {
    if(tree == NULL) return;
    postOrder(tree->left);
    postOrder(tree->right);
    cout<<tree->value<<" ";
}
```

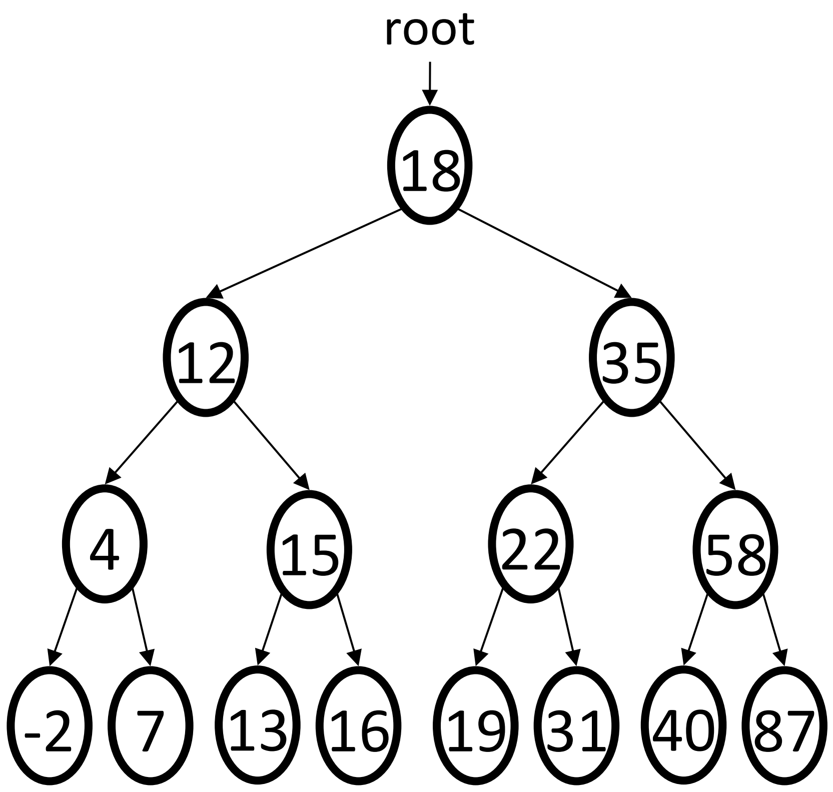


Tree Traversal

```
void preOrder(Tree * tree) {
    if(tree == NULL) return;
    cout<<tree->value<<" ";
    preOrder(tree->left);
    preOrder(tree->right);
}
```

```
void inOrder(Tree * tree) {
    if(tree == NULL) return;
    inOrder(tree->left);
    cout<<tree->value<<" ";
    inOrder(tree->right);
}
```

```
void postOrder(Tree * tree) {
    if(tree == NULL) return;
    postOrder(tree->left);
    postOrder(tree->right);
    cout<<tree->value<<" ";
}
```

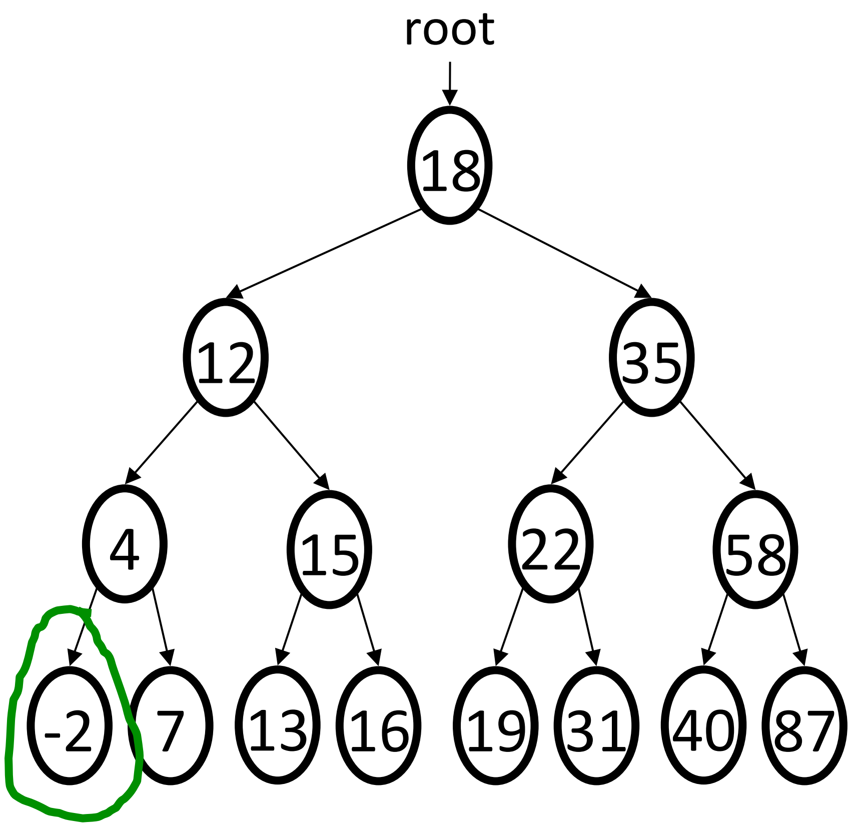


Tree Traversal

```
void preOrder(Tree * tree) {
    if(tree == NULL) return;
    cout<<tree->value<<" ";
    preOrder(tree->left);
    preOrder(tree->right);
}
```

```
void inOrder(Tree * tree) {
    if(tree == NULL) return;
    inOrder(tree->left);
    cout<<tree->value<<" ";
    inOrder(tree->right);
}
```

```
void postOrder(Tree * tree) {
    if(tree == NULL) return;
    postOrder(tree->left);
    postOrder(tree->right);
    cout<<tree->value<<" ";
}
```

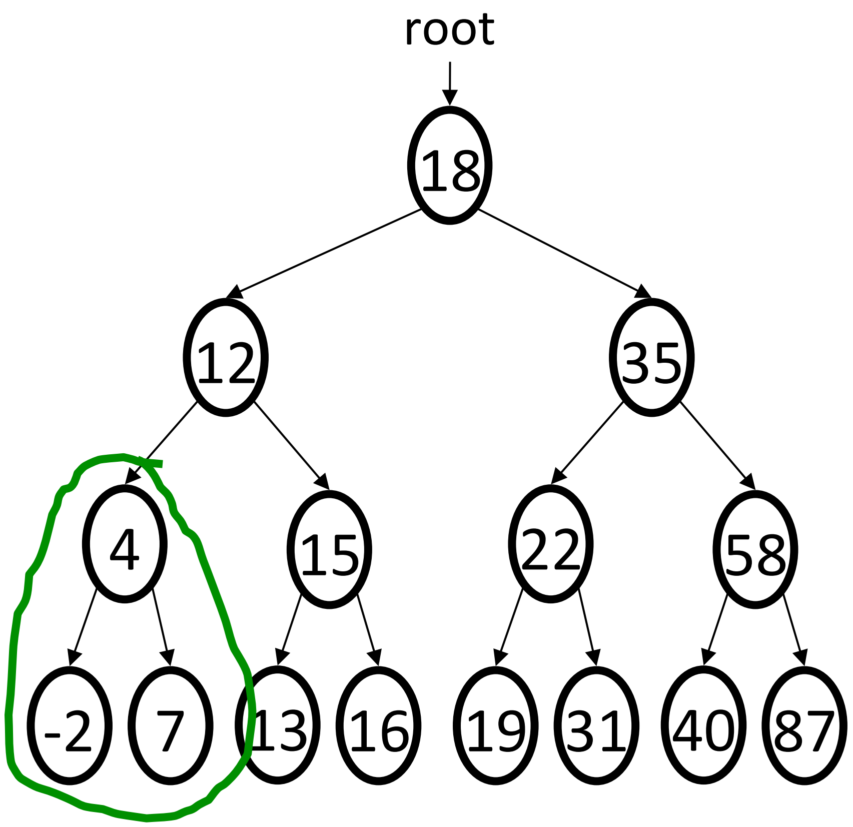


Tree Traversal

```
void preOrder(Tree * tree) {
    if(tree == NULL) return;
    cout<<tree->value<<" ";
    preOrder(tree->left);
    preOrder(tree->right);
}
```

```
void inOrder(Tree * tree) {
    if(tree == NULL) return;
    inOrder(tree->left);
    cout<<tree->value<<" ";
    inOrder(tree->right);
}
```

```
void postOrder(Tree * tree) {
    if(tree == NULL) return;
    postOrder(tree->left);
    postOrder(tree->right);
    cout<<tree->value<<" ";
}
```

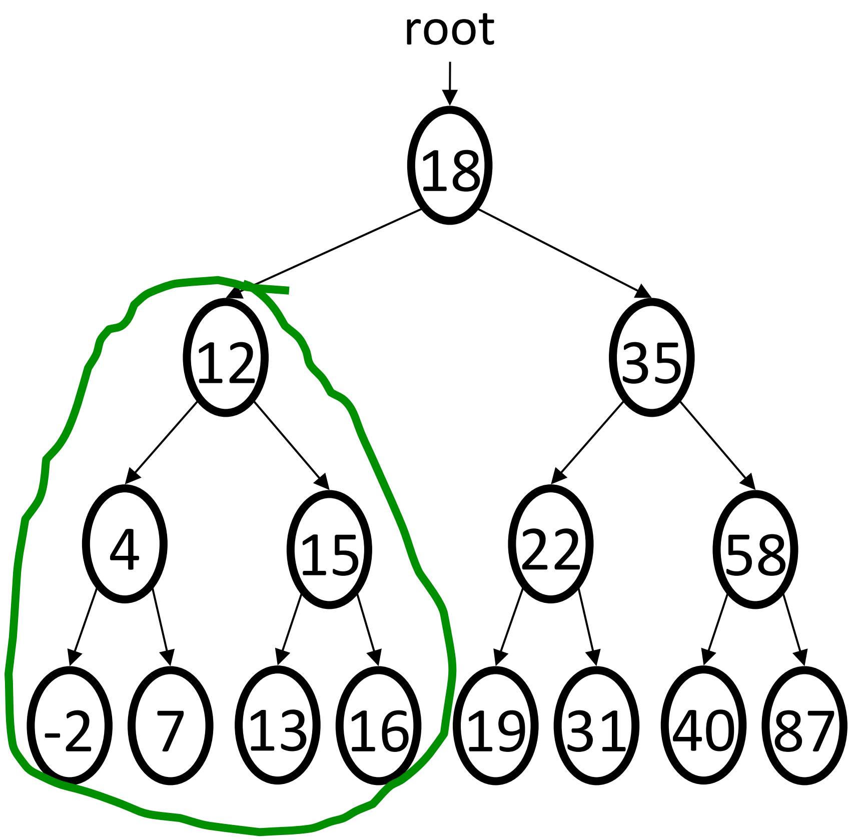


Tree Traversal

```
void preOrder(Tree * tree) {
    if(tree == NULL) return;
    cout<<tree->value<<" ";
    preOrder(tree->left);
    preOrder(tree->right);
}
```

```
void inOrder(Tree * tree) {
    if(tree == NULL) return;
    inOrder(tree->left);
    cout<<tree->value<<" ";
    inOrder(tree->right);
}
```

```
void postOrder(Tree * tree) {
    if(tree == NULL) return;
    postOrder(tree->left);
    postOrder(tree->right);
    cout<<tree->value<<" ";
}
```

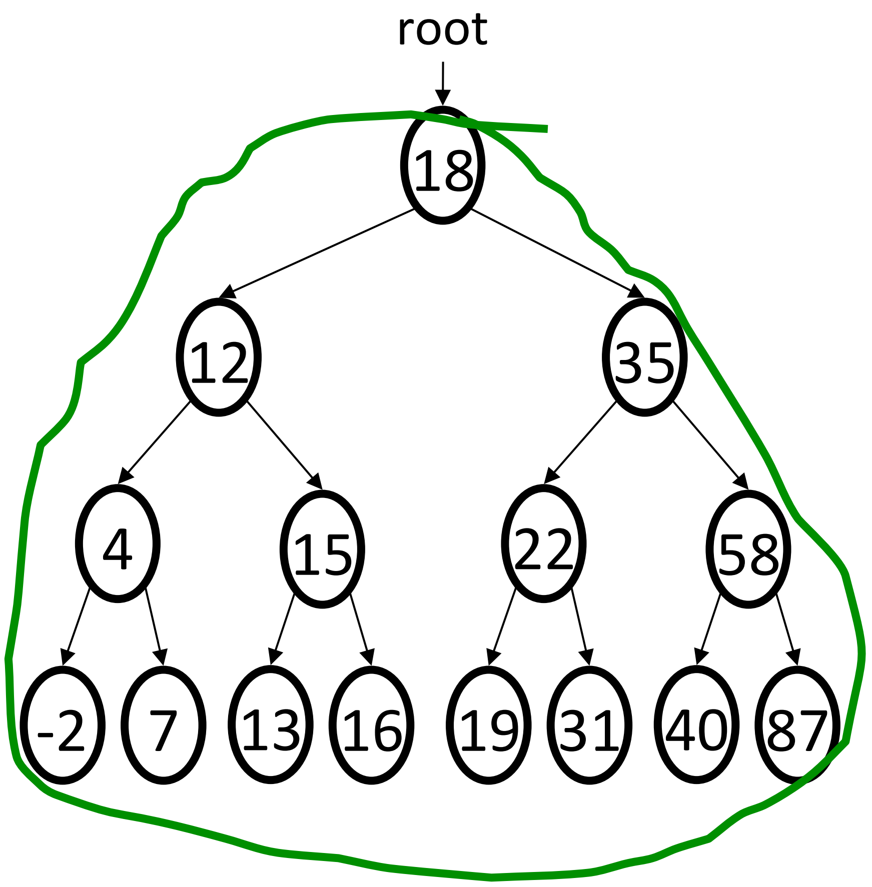


Tree Traversal

```
void preOrder(Tree * tree) {
    if(tree == NULL) return;
    cout<<tree->value<<" ";
    preOrder(tree->left);
    preOrder(tree->right);
}
```

```
void inOrder(Tree * tree) {
    if(tree == NULL) return;
    inOrder(tree->left);
    cout<<tree->value<<" ";
    inOrder(tree->right);
}
```

```
void postOrder(Tree * tree) {
    if(tree == NULL) return;
    postOrder(tree->left);
    postOrder(tree->right);
    cout<<tree->value<<" ";
}
```

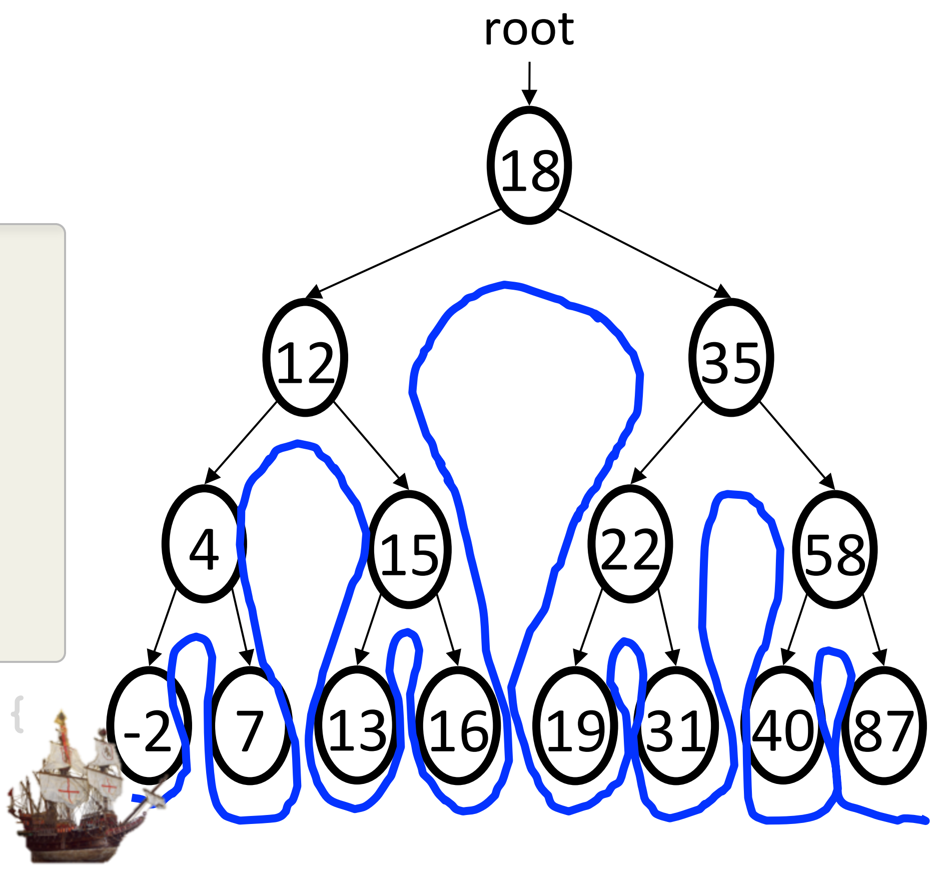


Tree Traversal

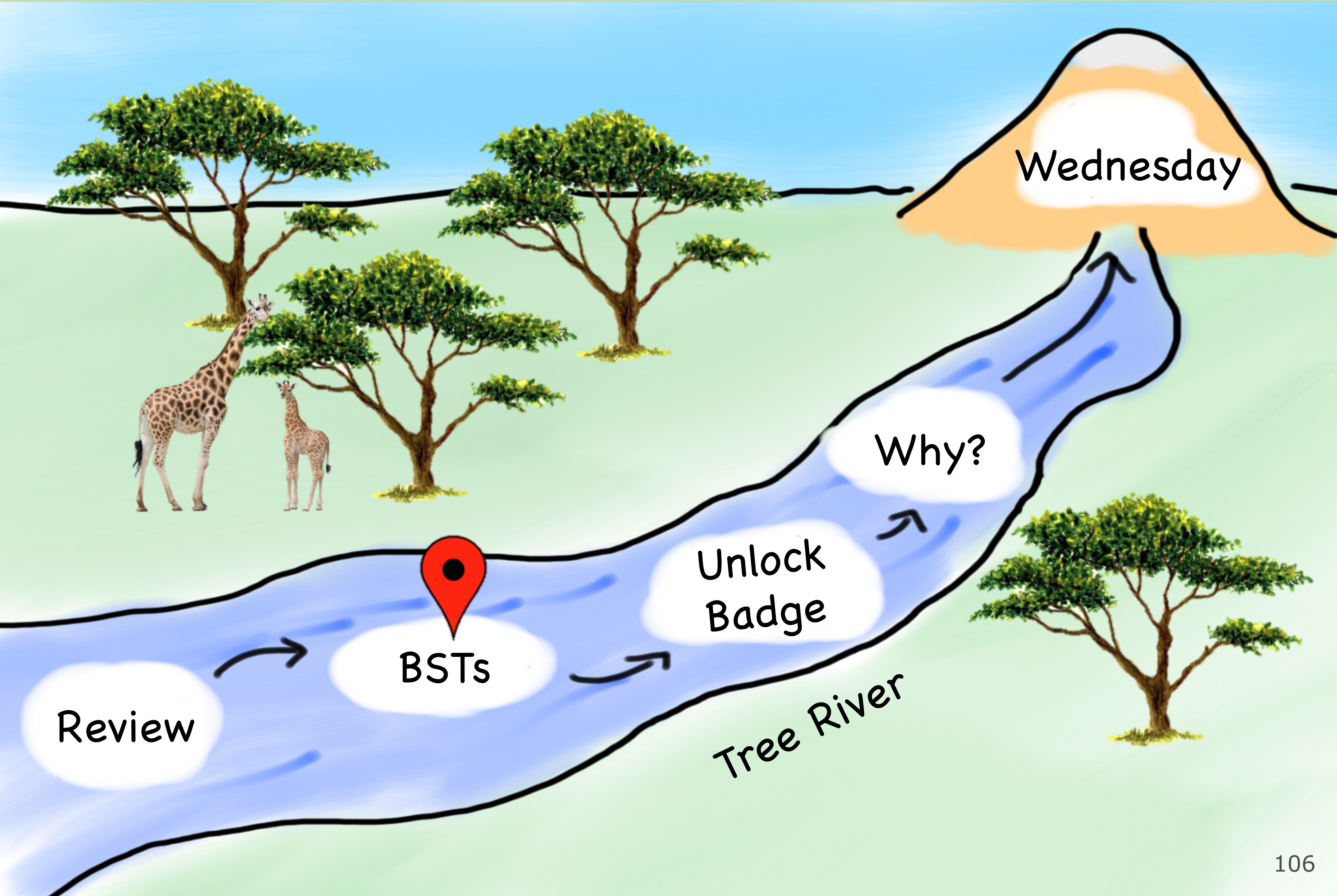
```
void preOrder(Tree * tree) {
    if(tree == NULL) return;
    cout<<tree->value<<" ";
    preOrder(tree->left);
    preOrder(tree->right);
}
```

```
void inOrder(Tree * tree) {
    if(tree == NULL) return;
    inOrder(tree->left);
    cout<<tree->value<<" ";
    inOrder(tree->right);
}
```

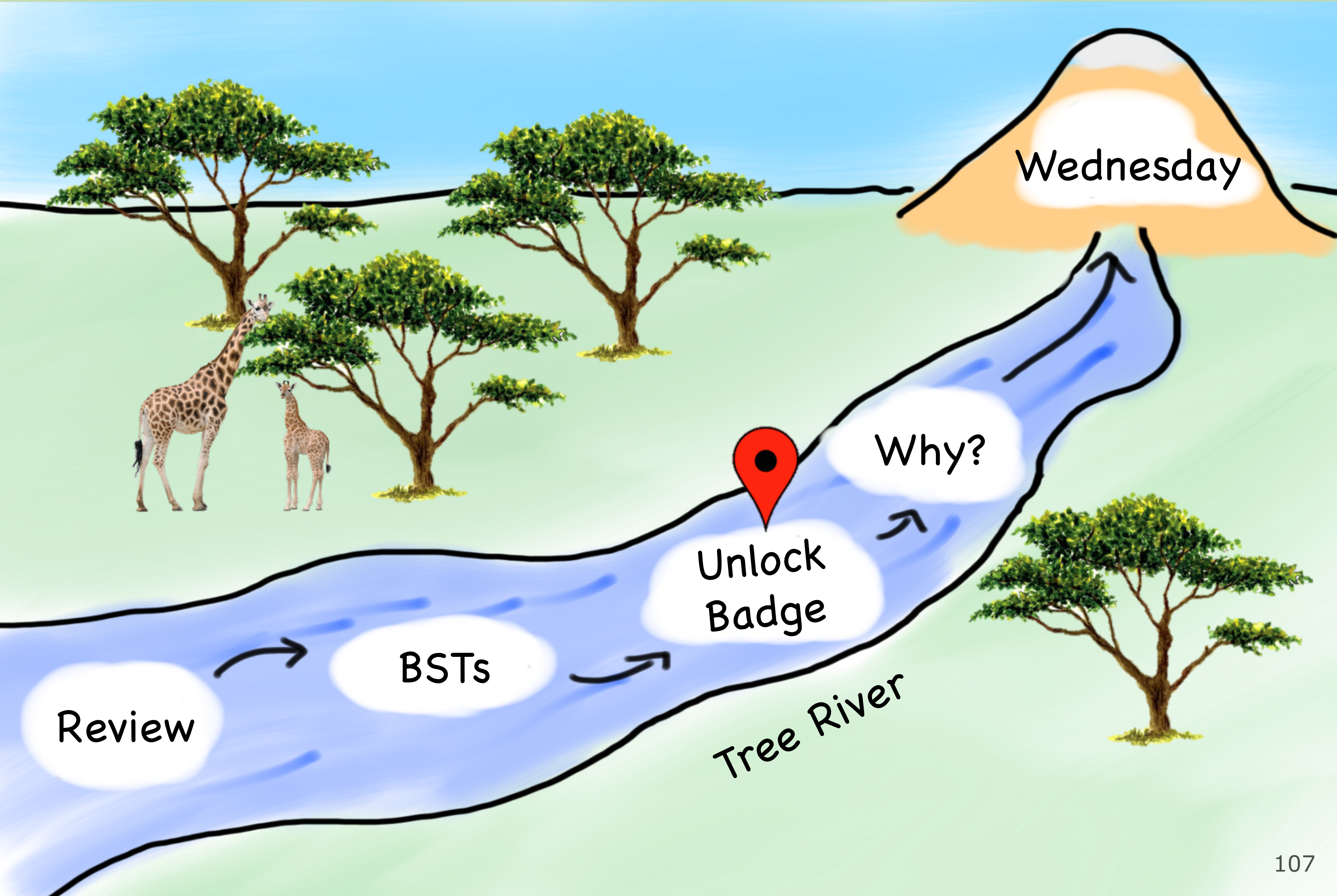
```
void postOrder(Tree * tree) {
    if(tree == NULL) return;
    postOrder(tree->left);
    postOrder(tree->right);
    cout<<tree->value<<" ";
}
```



Today's Route

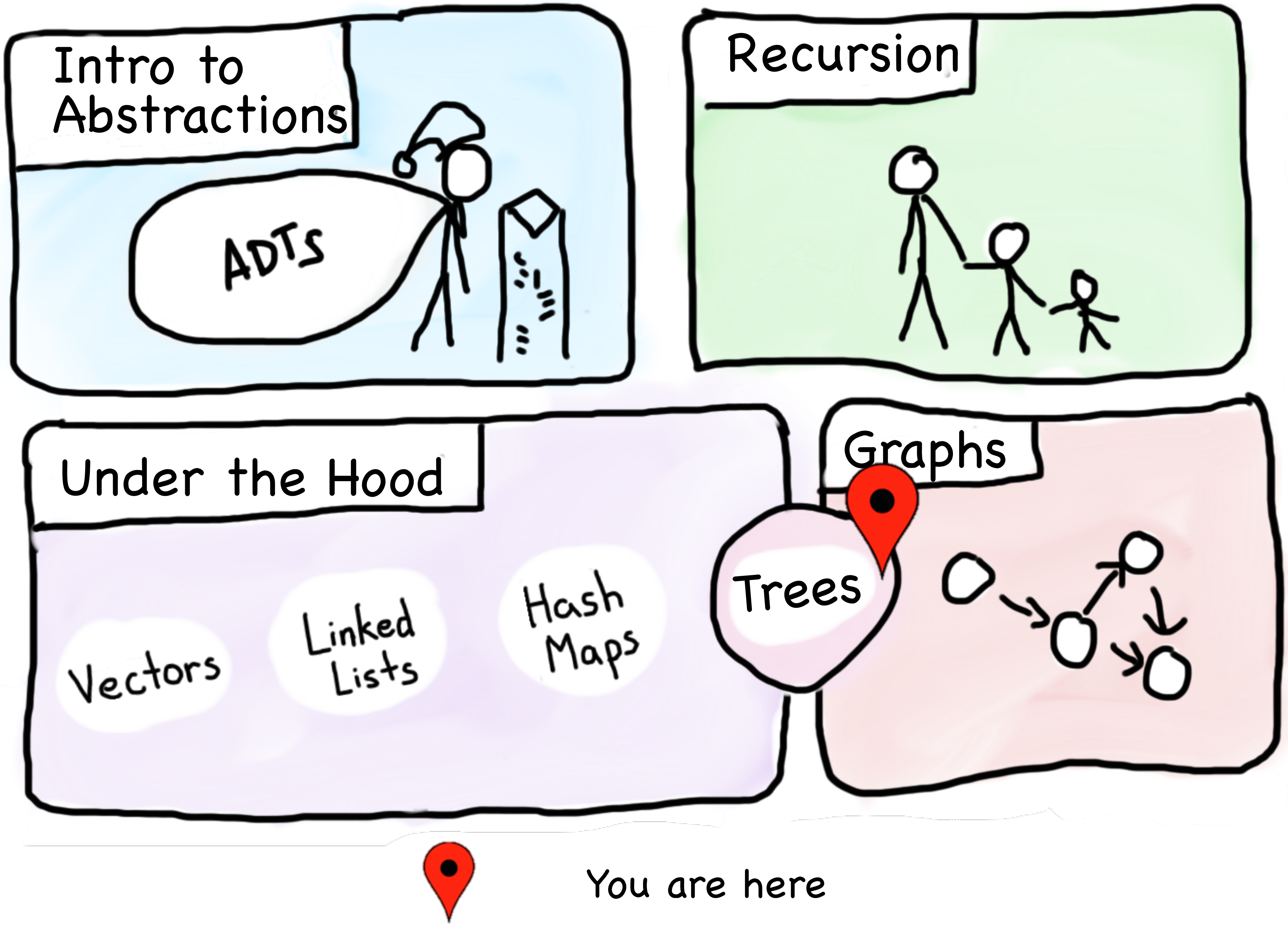


Today's Route



Where Am I?

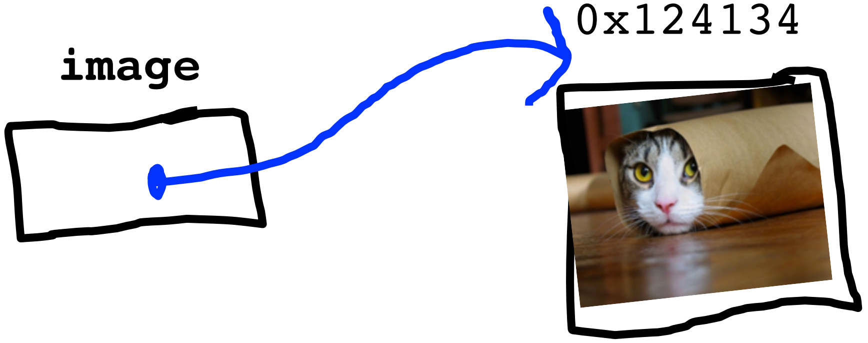
Course Syllabus



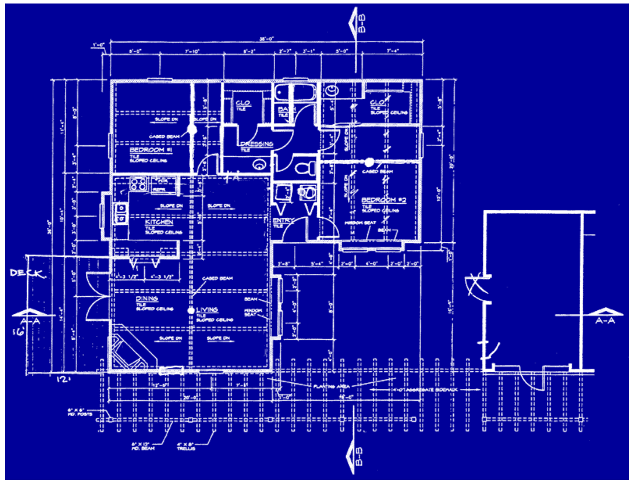
Building Blocks

Pointers

```
GImage * image =
 new GImage("cat.png");
```



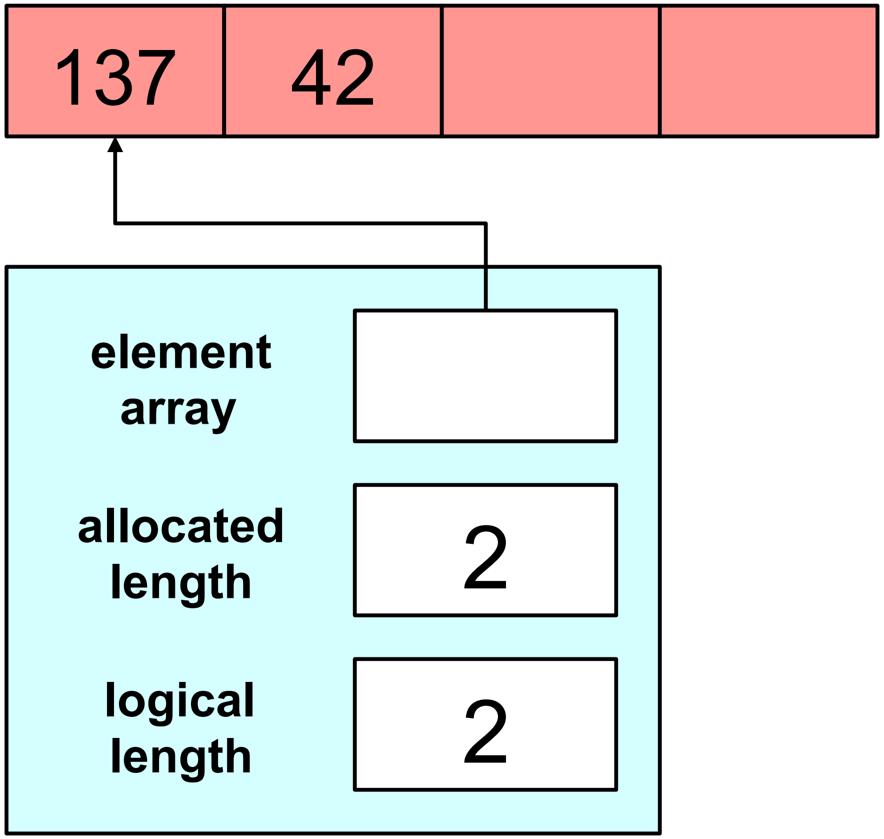
Classes / Structs



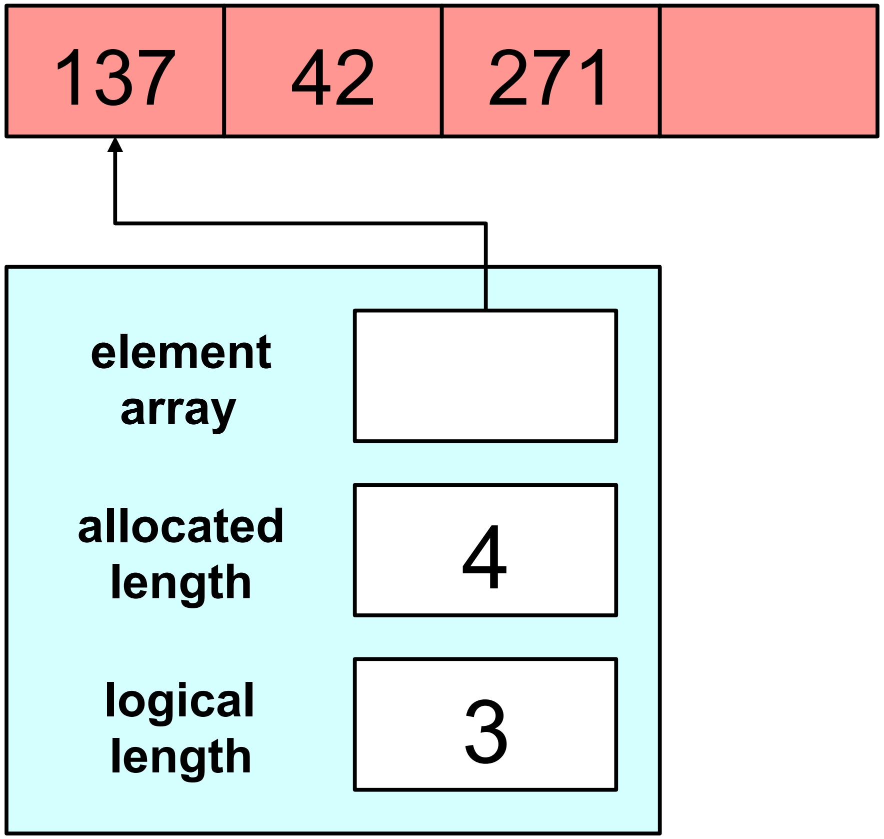
Blueprint for a new variable *type*

Vector

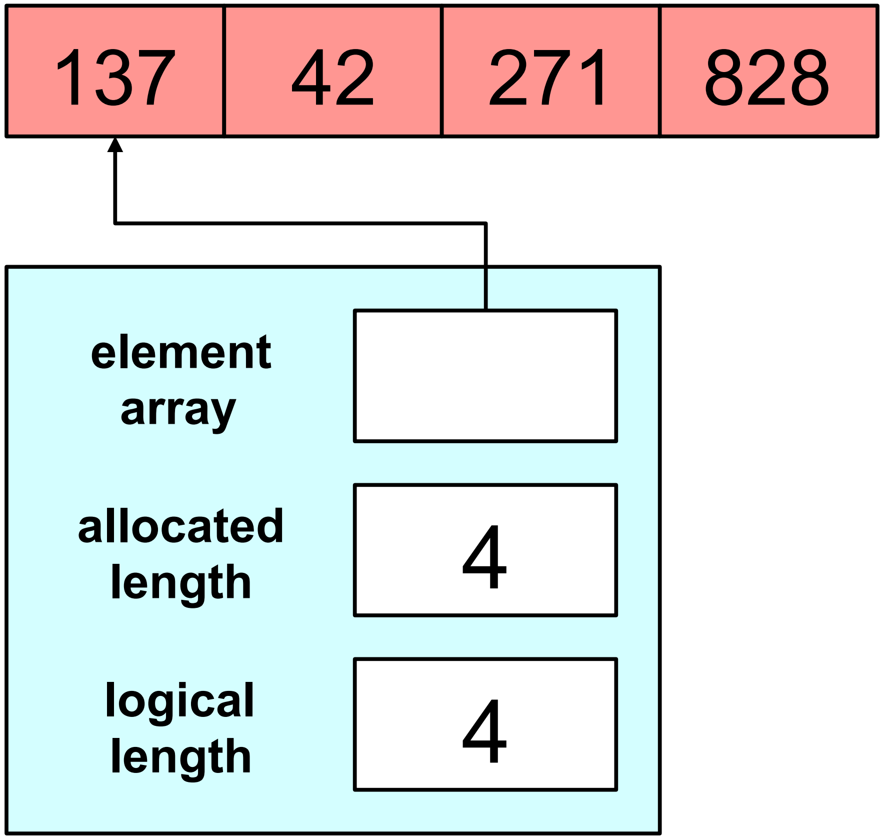
Vector



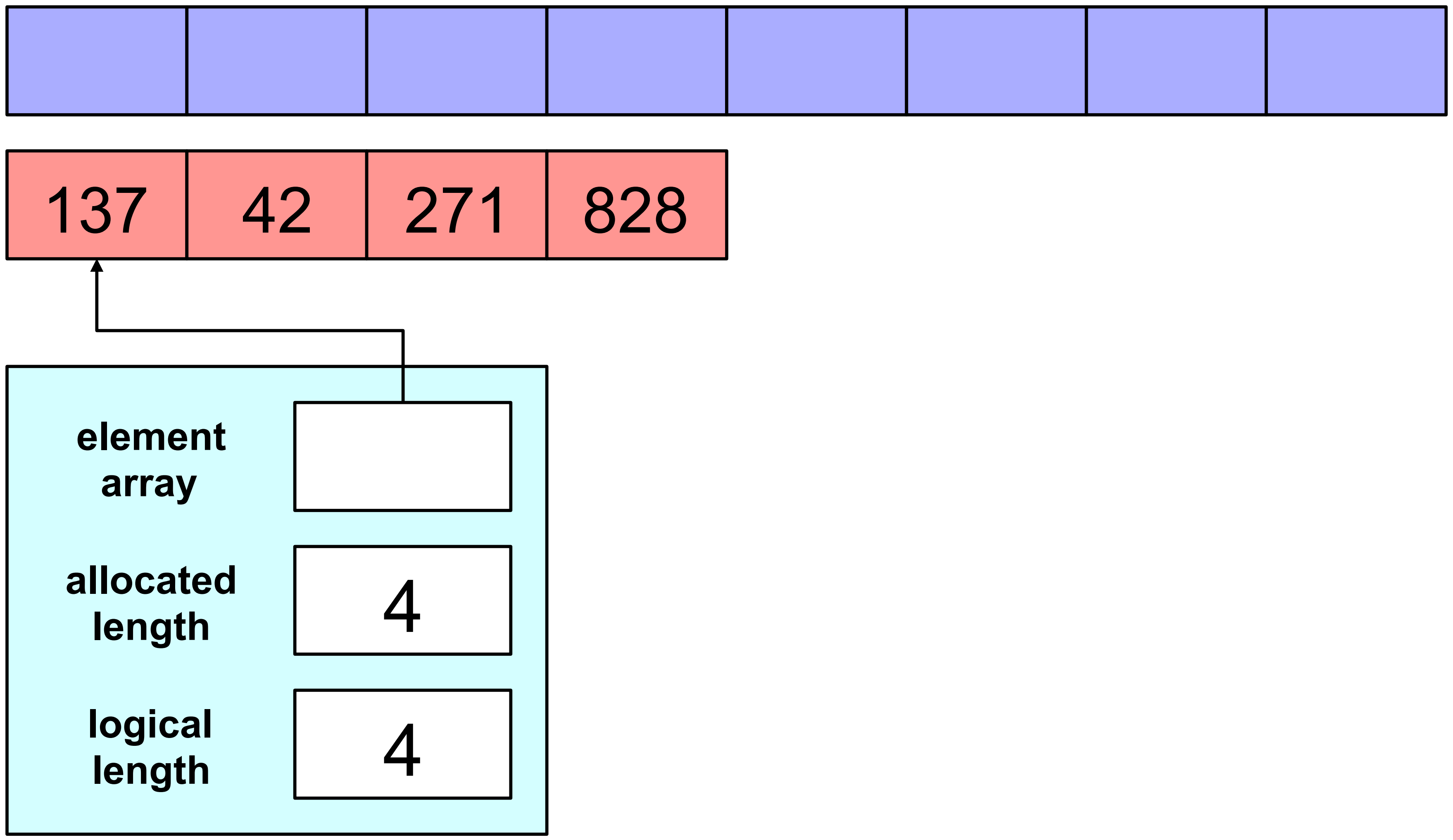
Actual Vector



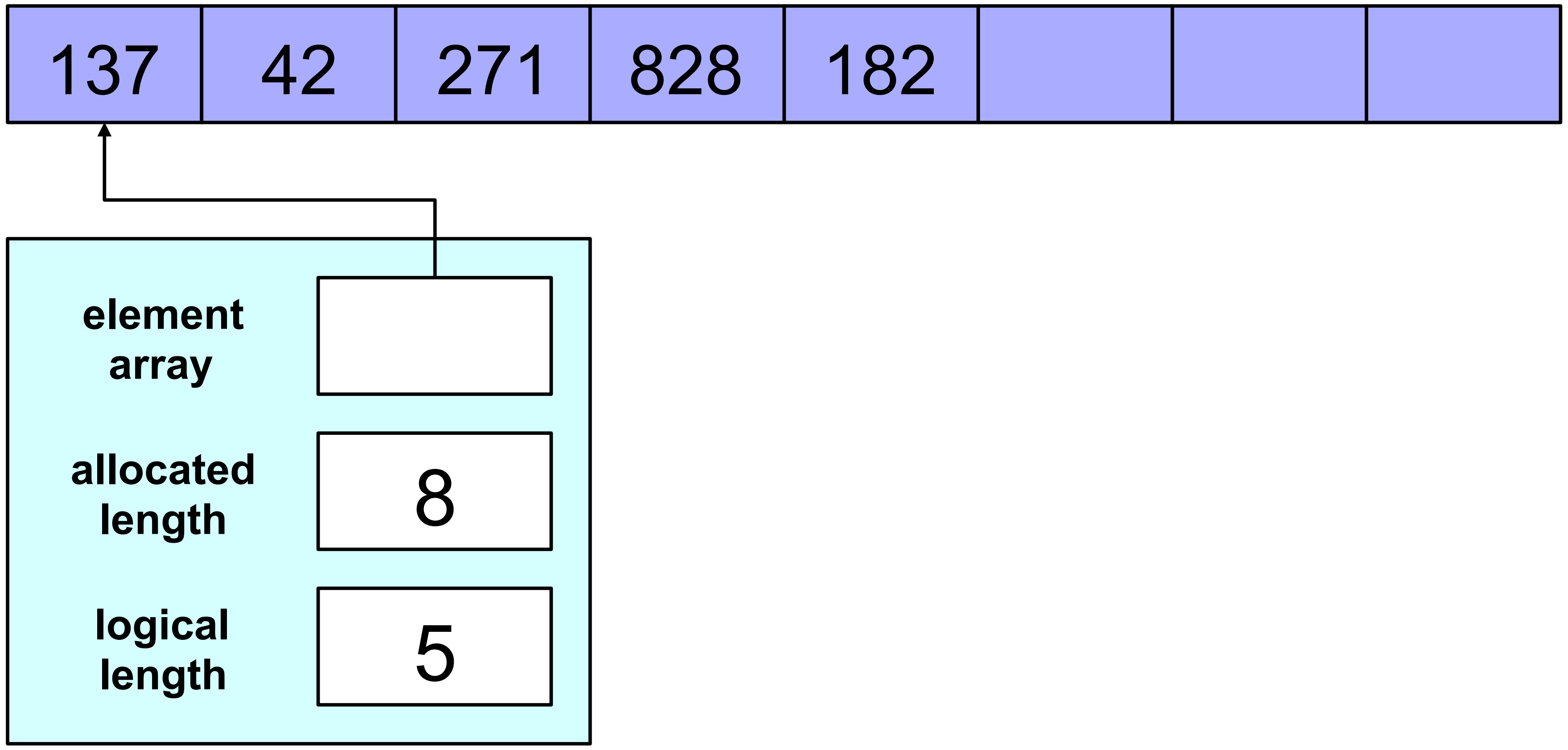
Actual Vector



Vector

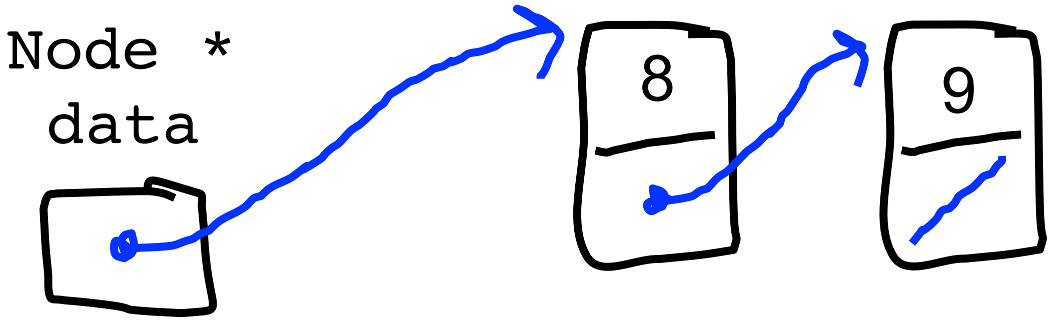

137
42
271
828
element array
allocated length
4
logical length
4

Vector


137
42
271
828
182
element array
allocated length
8
logical length
5

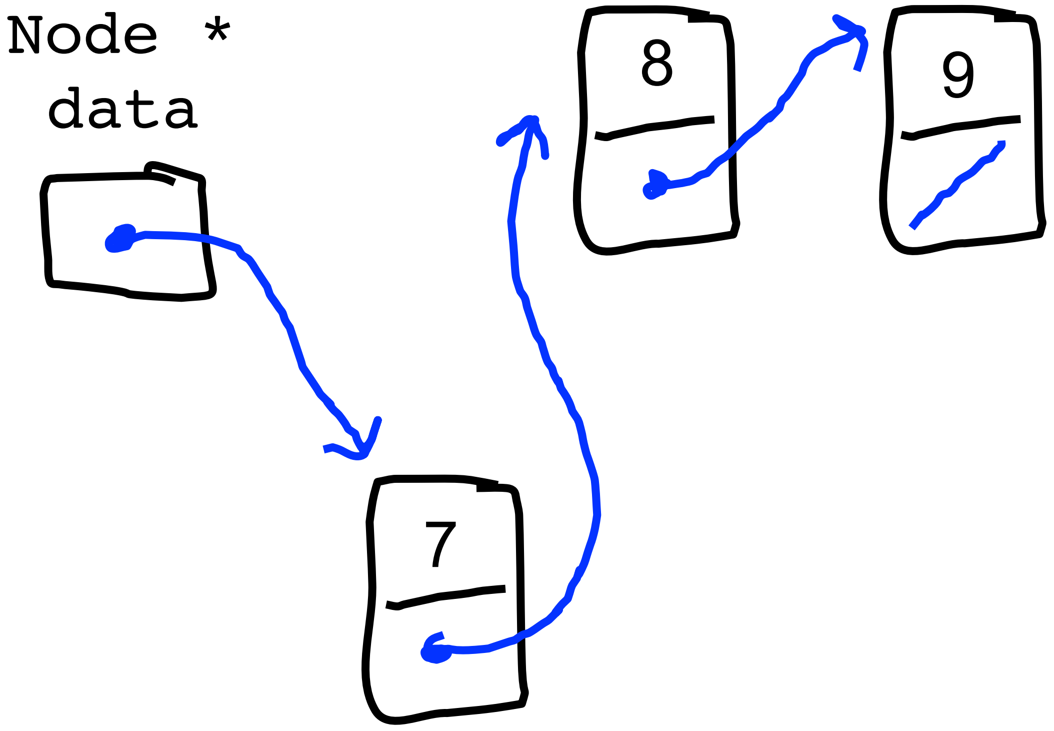
Stack

Stack



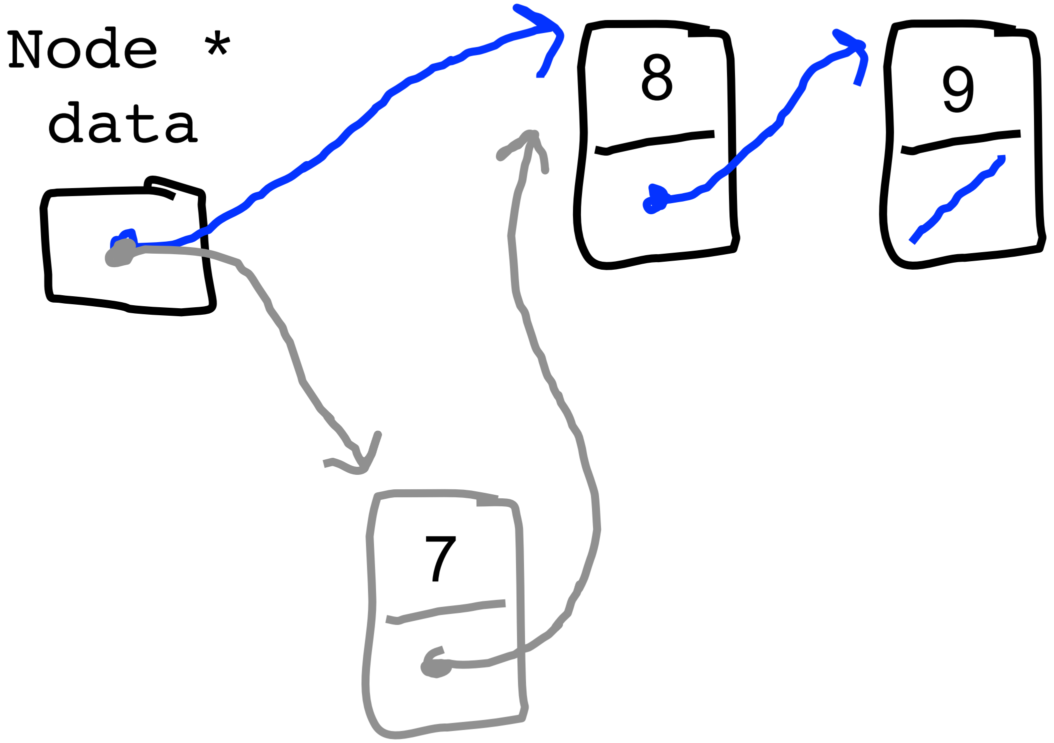
Stack Push

```
push(7);
```



Stack Pop

```
pop();
```

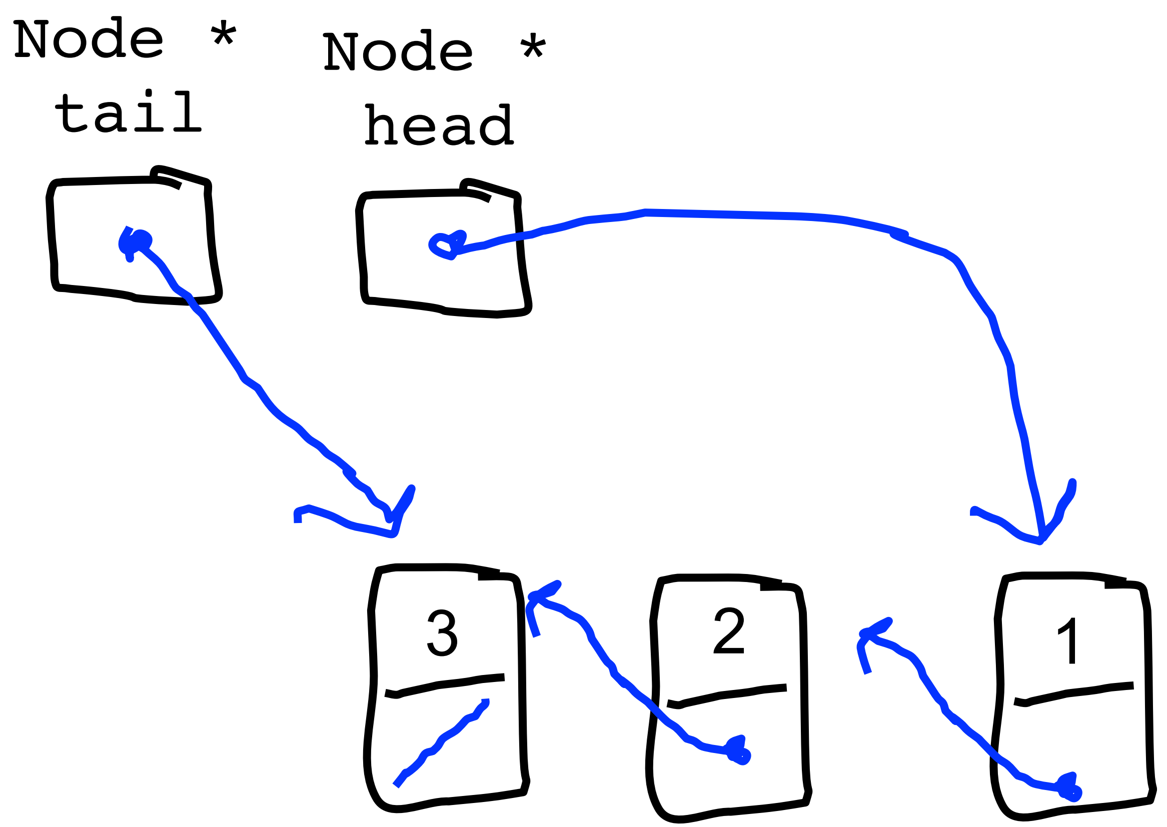


int return

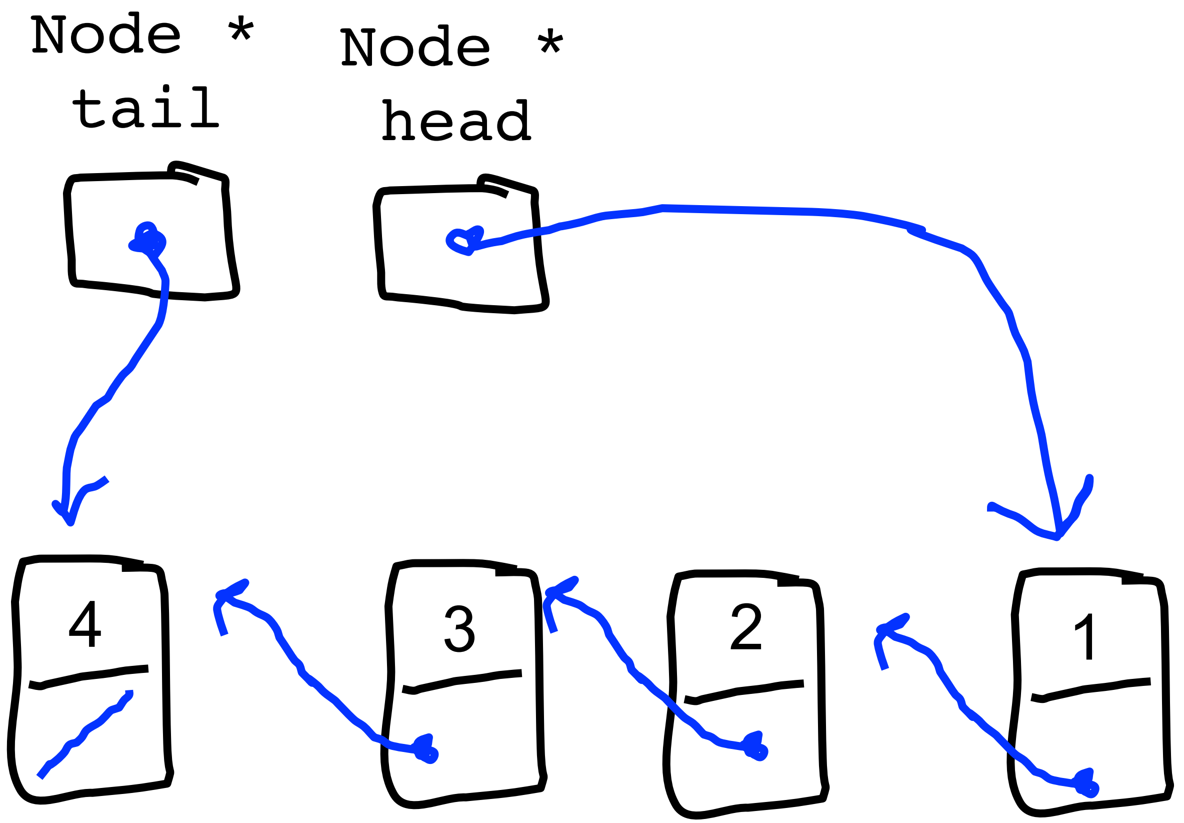
7

Queue

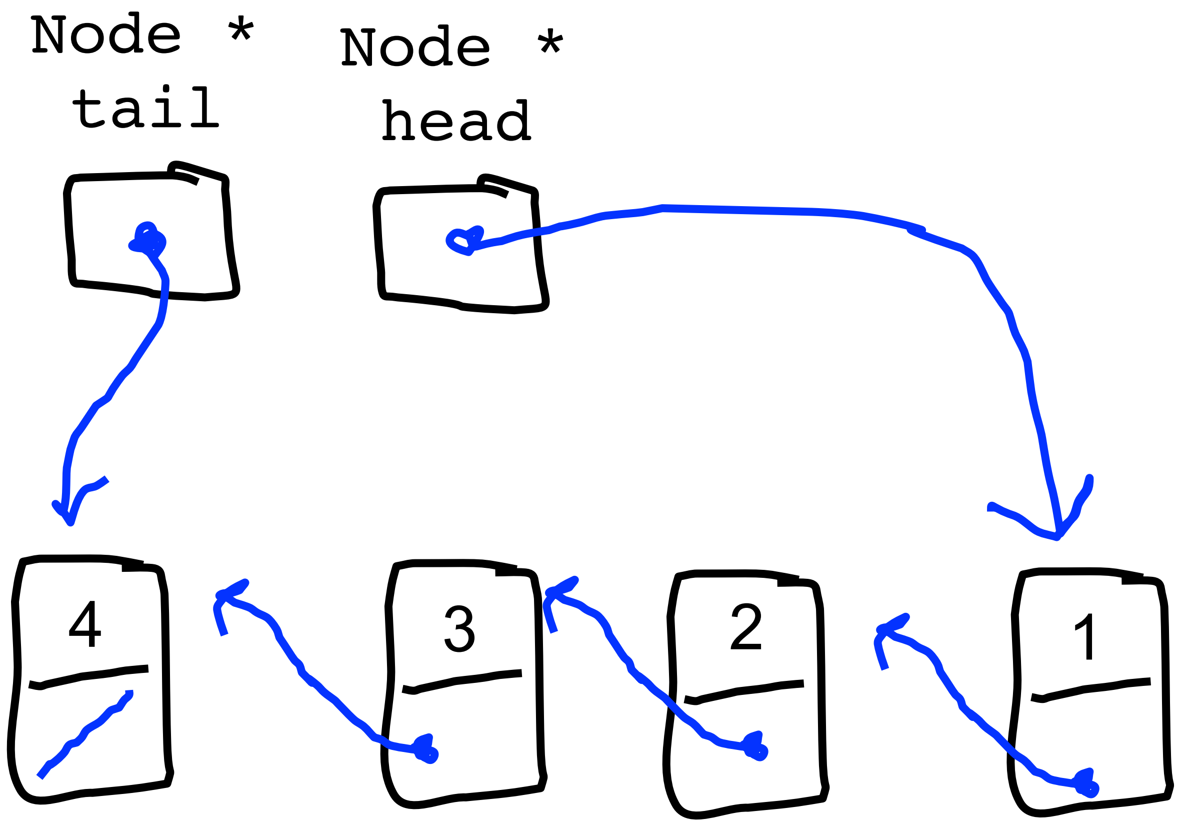
Queue



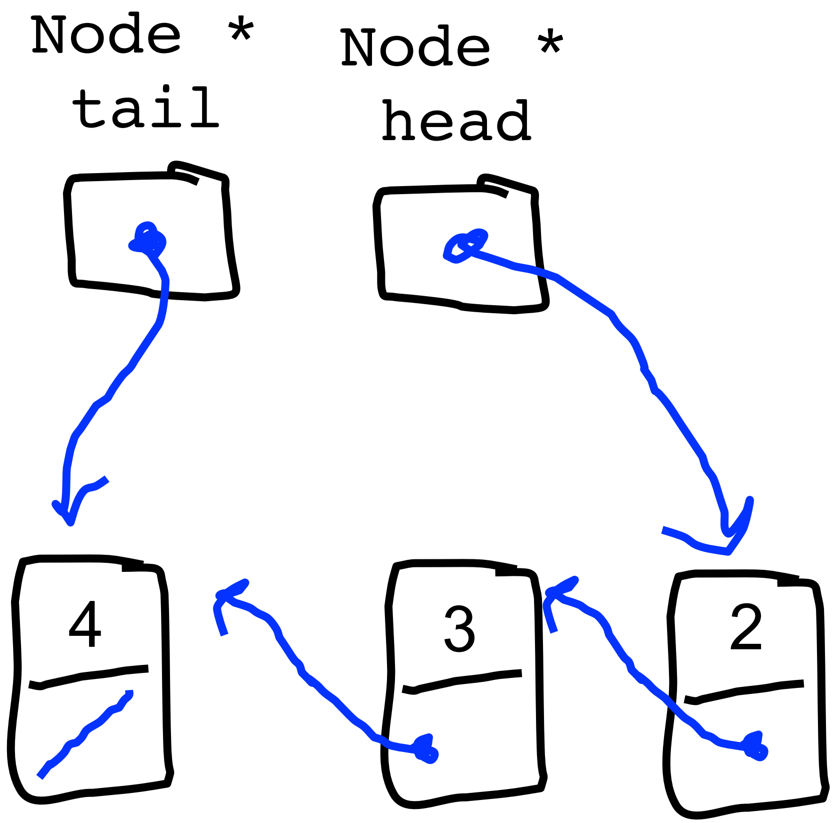
Queue Enqueue



Queue Dequeue

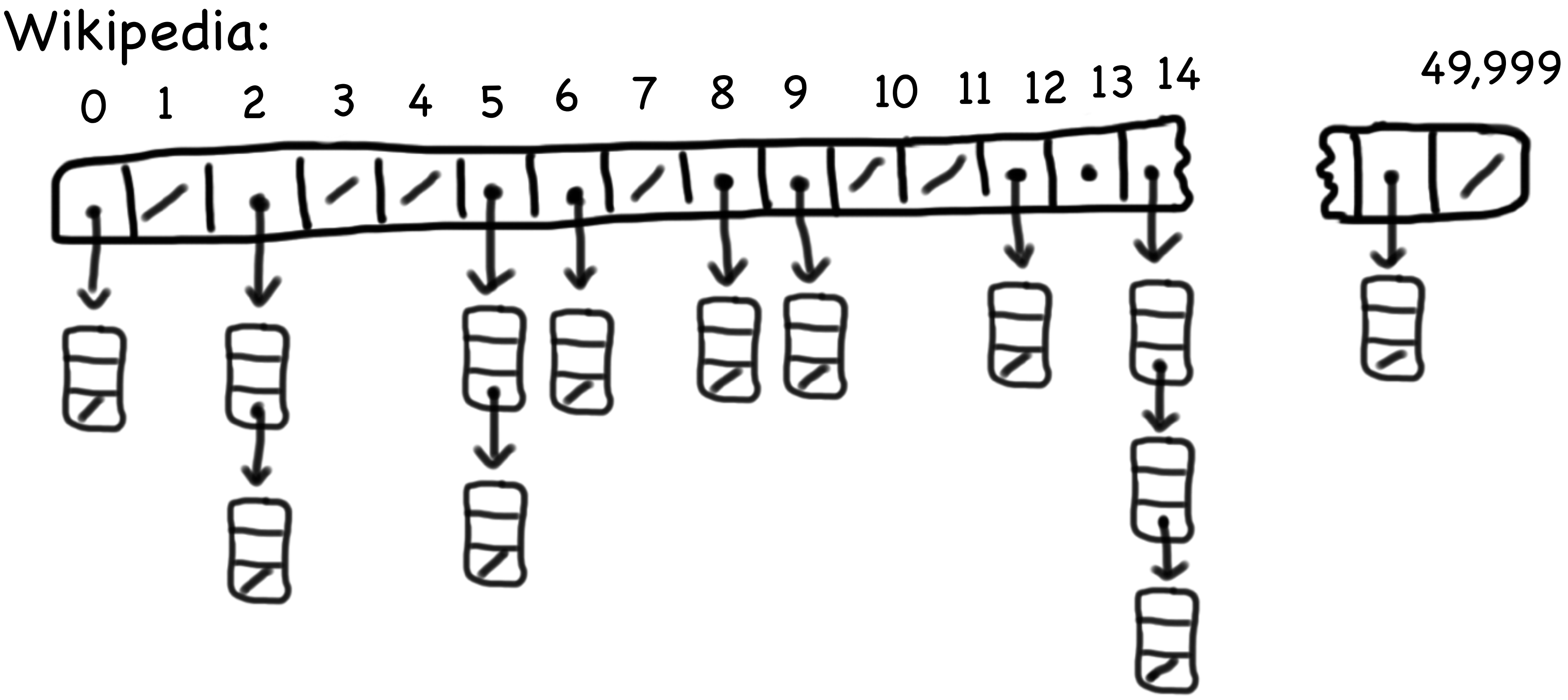


Queue Dequeue

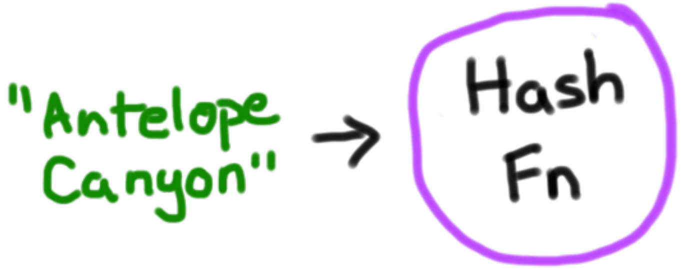


HashMap / HashSet

HashMap



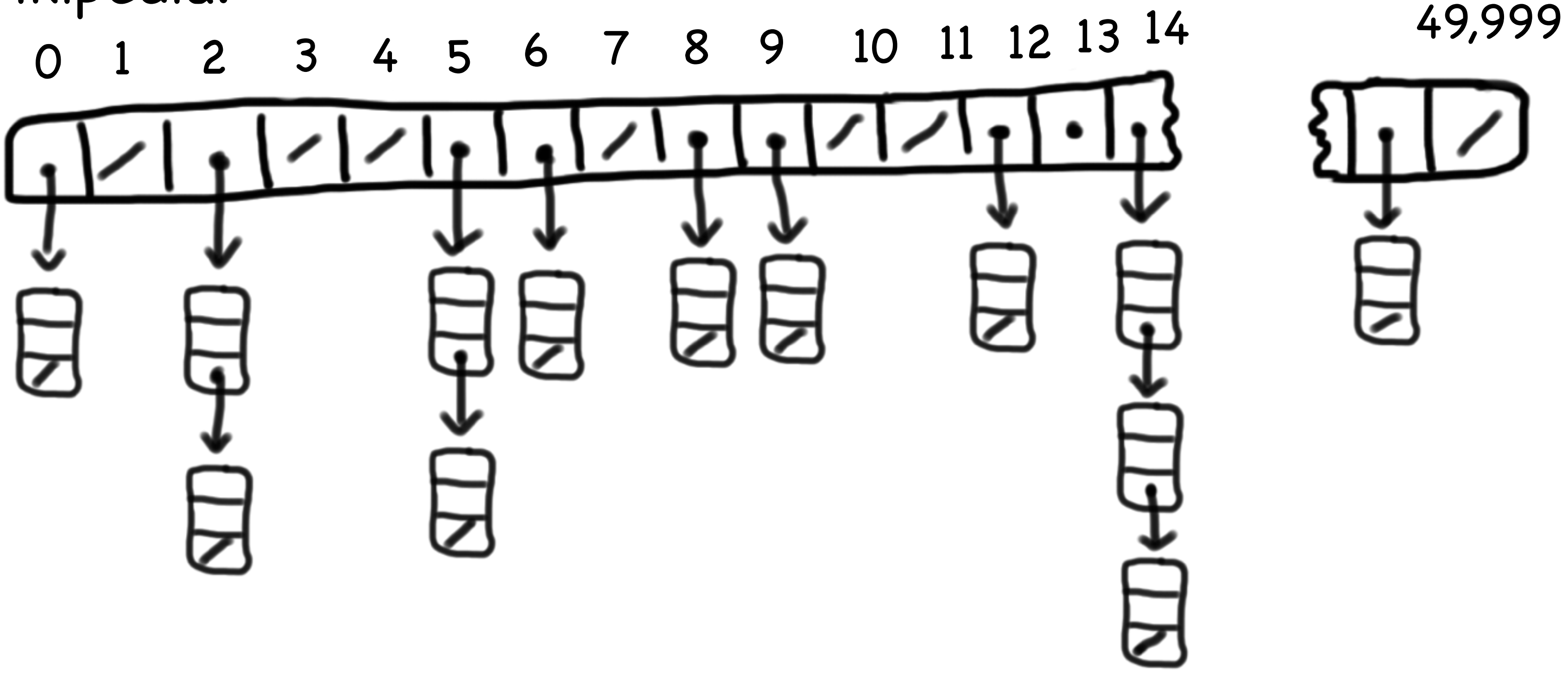
Wikipedia:
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
49,999

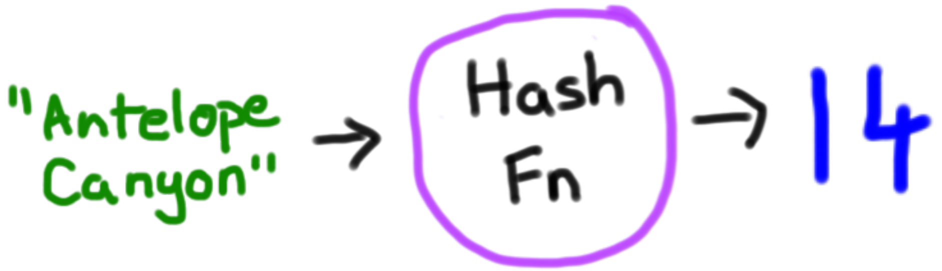


"Antelope Canyon"
Hash Fn

HashMap

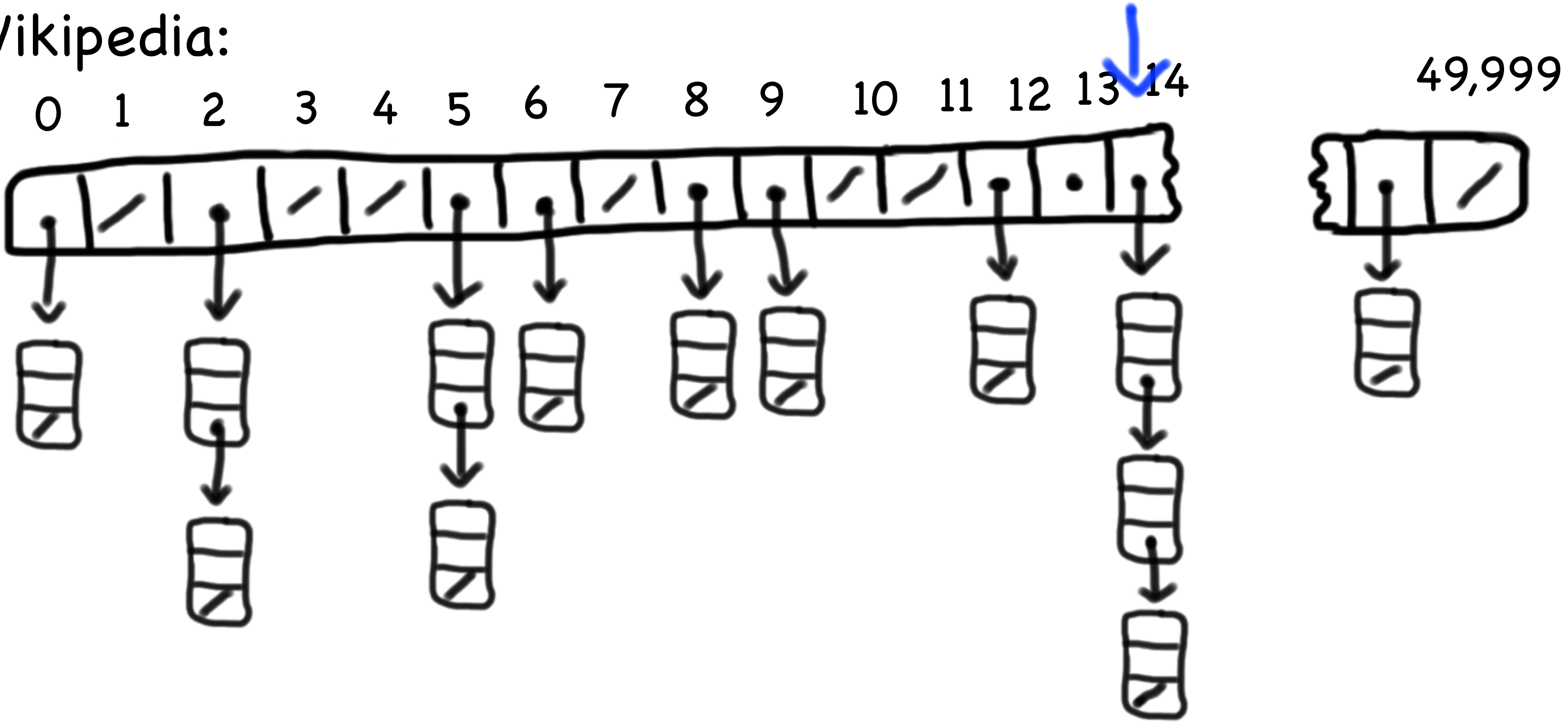
Wikipedia:

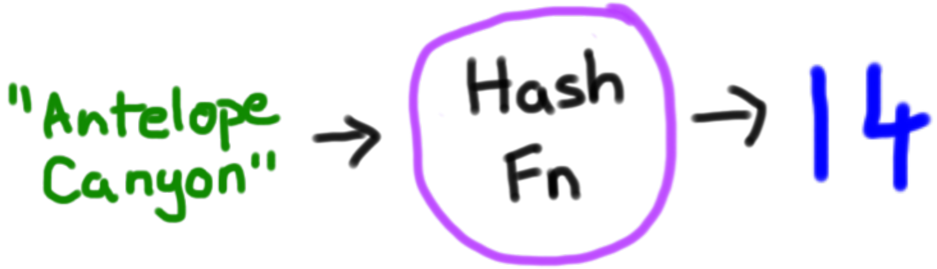




HashMap

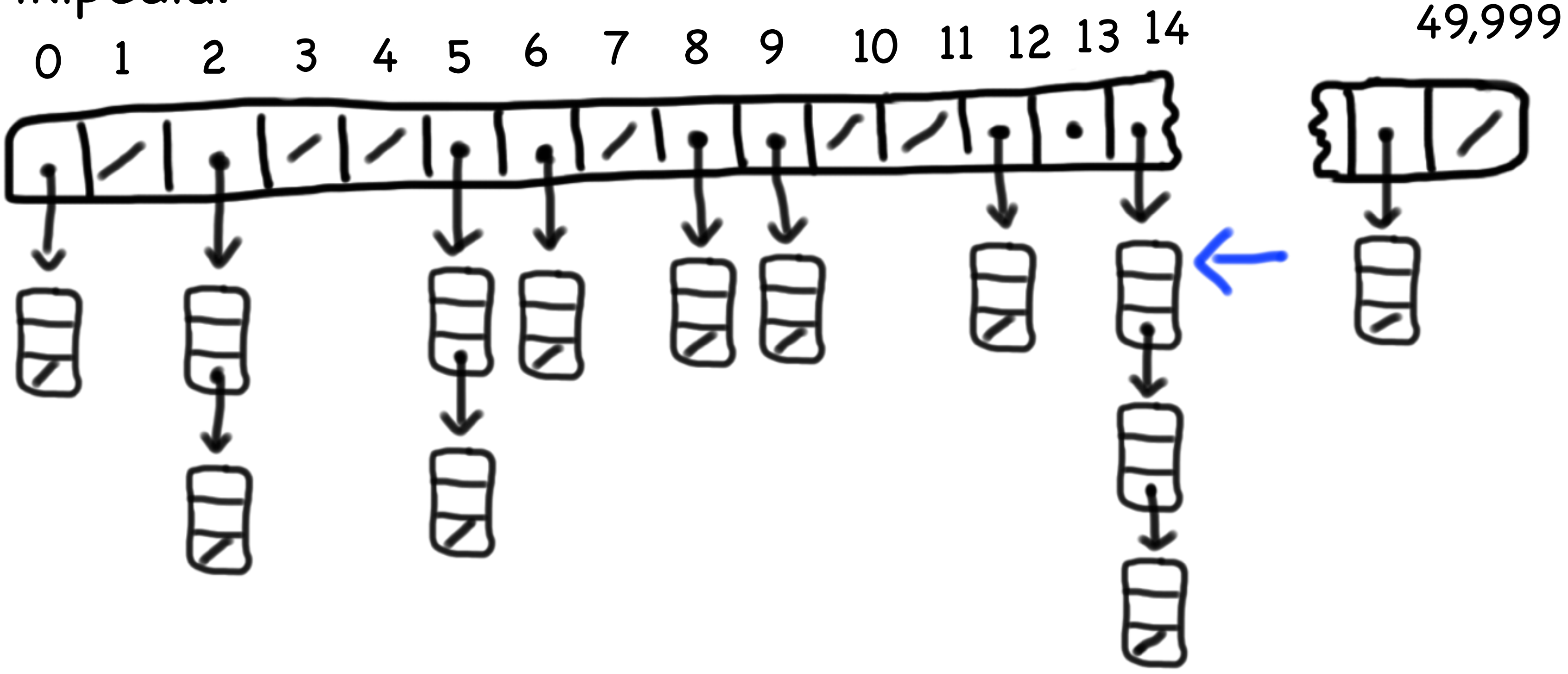
Wikipedia:

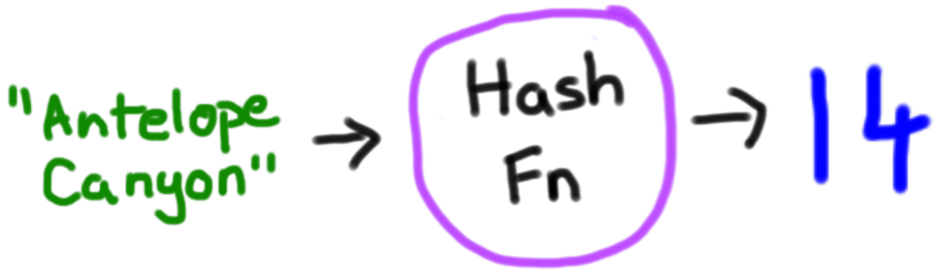




HashMap

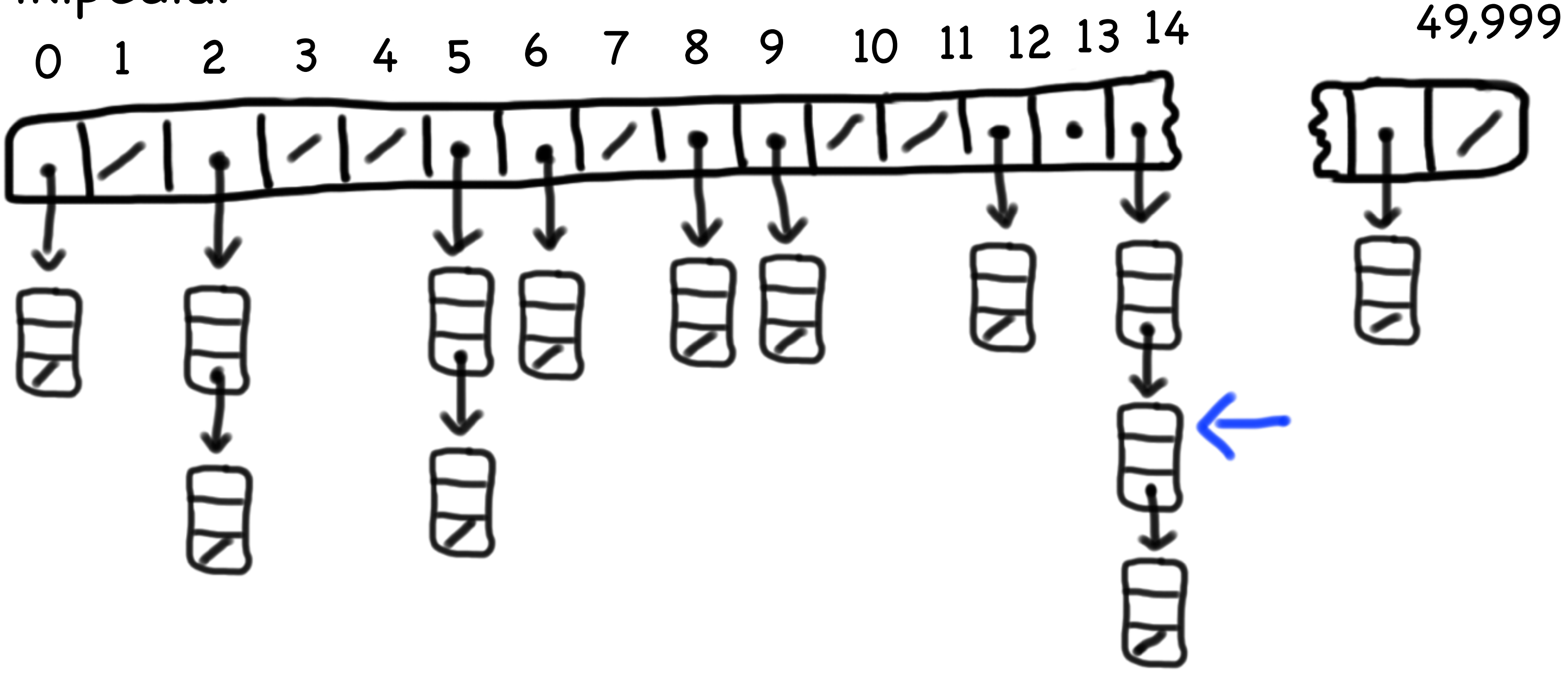
Wikipedia:

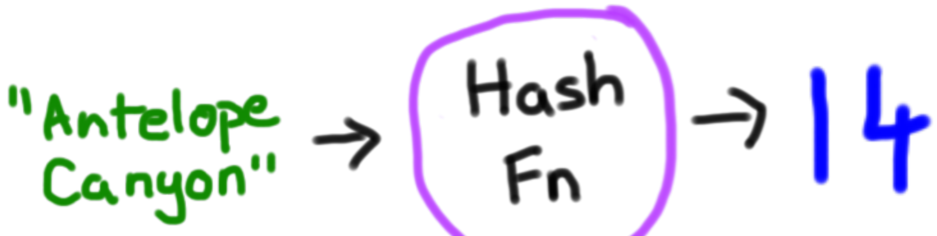




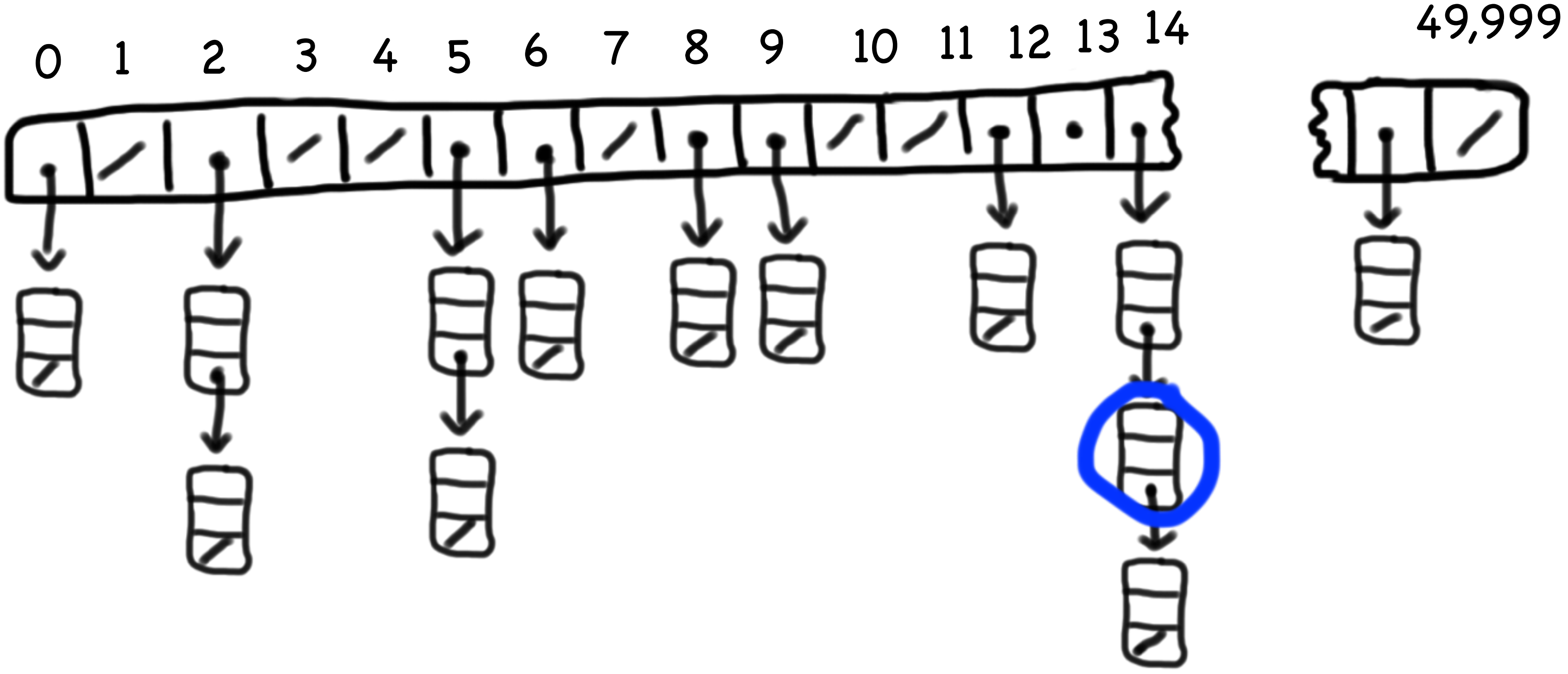
HashMap

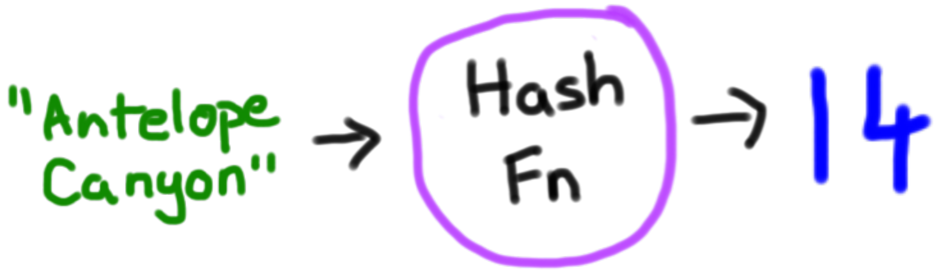
Wikipedia:





HashMap



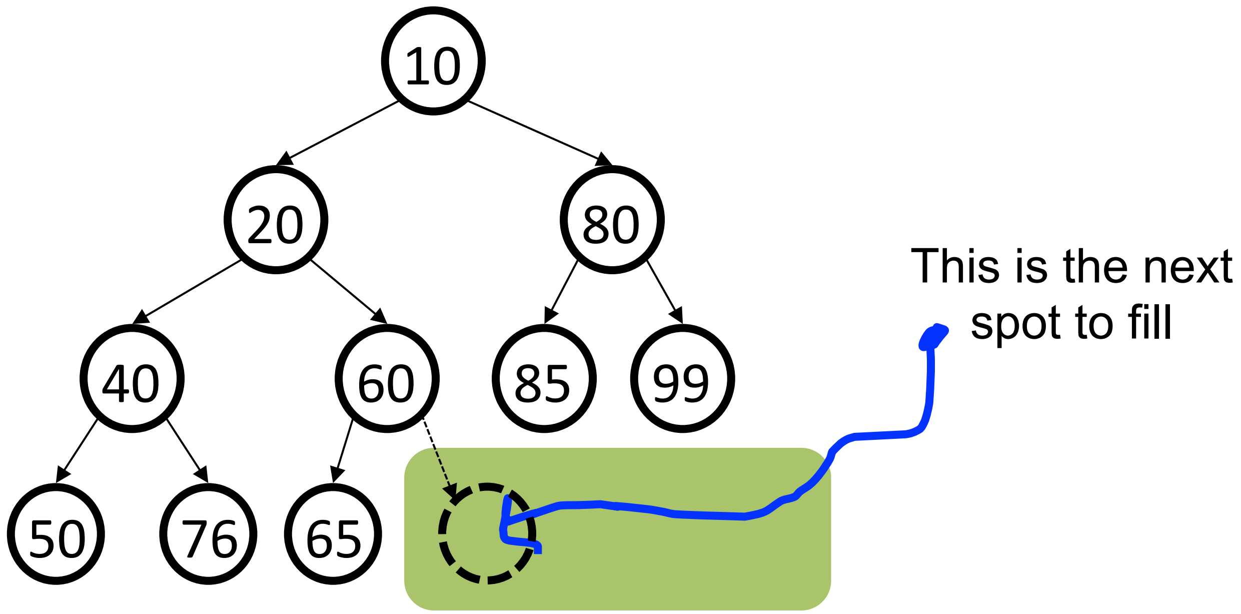


PQueue

Priority Queue

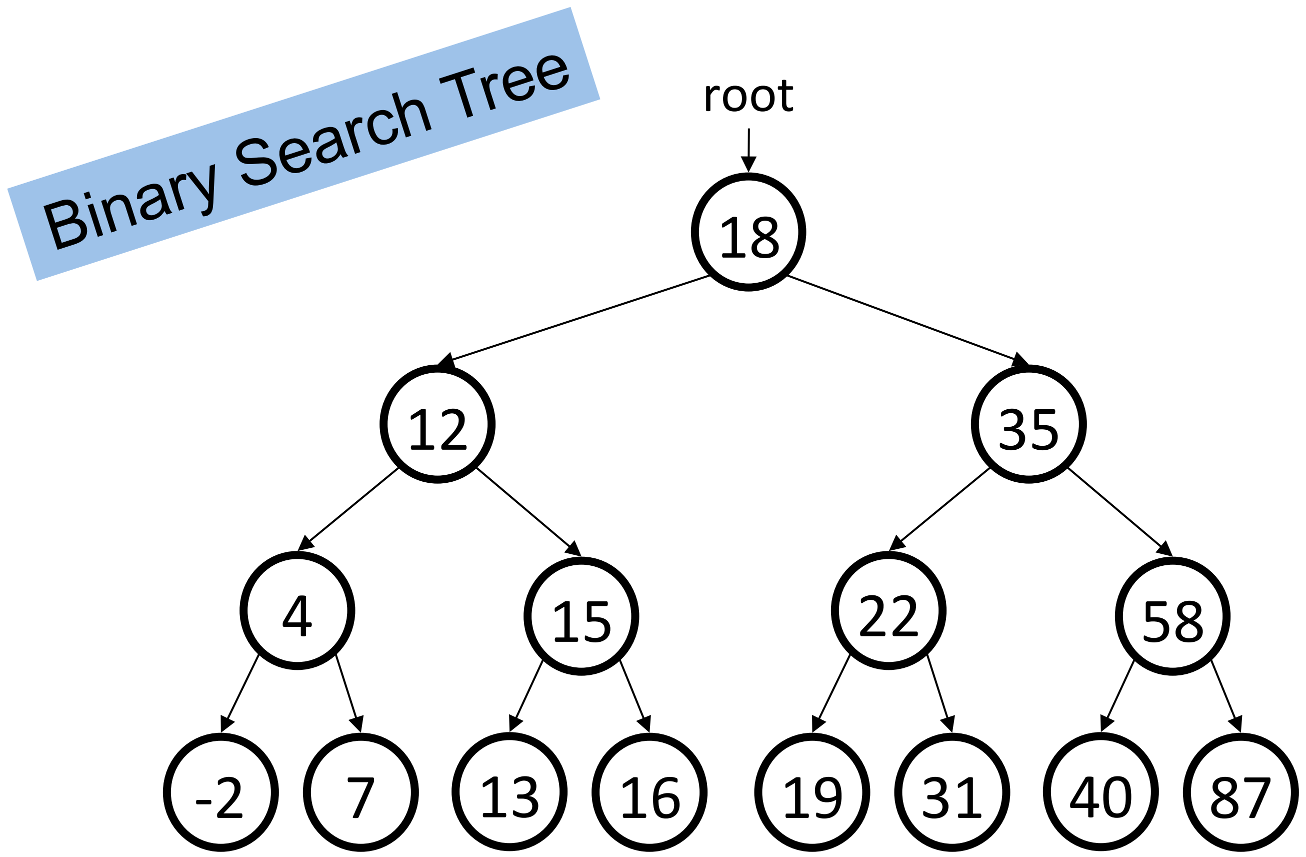
Pqueue Tree

Heap Ordered: Every branch takes you to a "greater" node.
Binary: Every node has at most two children.
Complete: There are no "gaps" in the tree.



Sets + Maps

Sets and Maps



That’s all of them

Main CS Collections

Vector

Grid

HashMap

Stack

Queue

PQueue

HashSet

Map

Set

Main CS Collections

Vector

Grid

HashMap

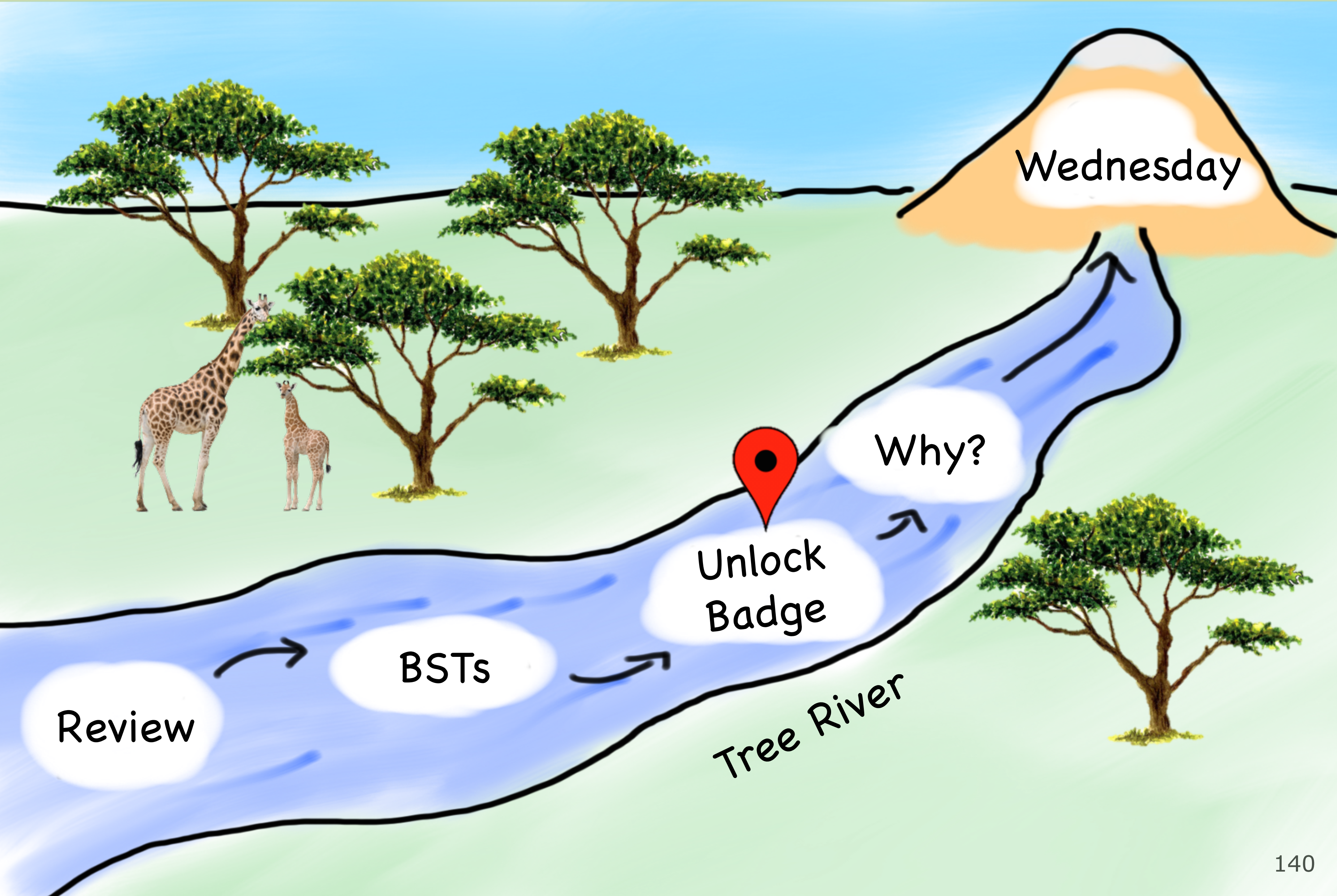
Achievement unlocked

Saw under the hood of the main CS collections

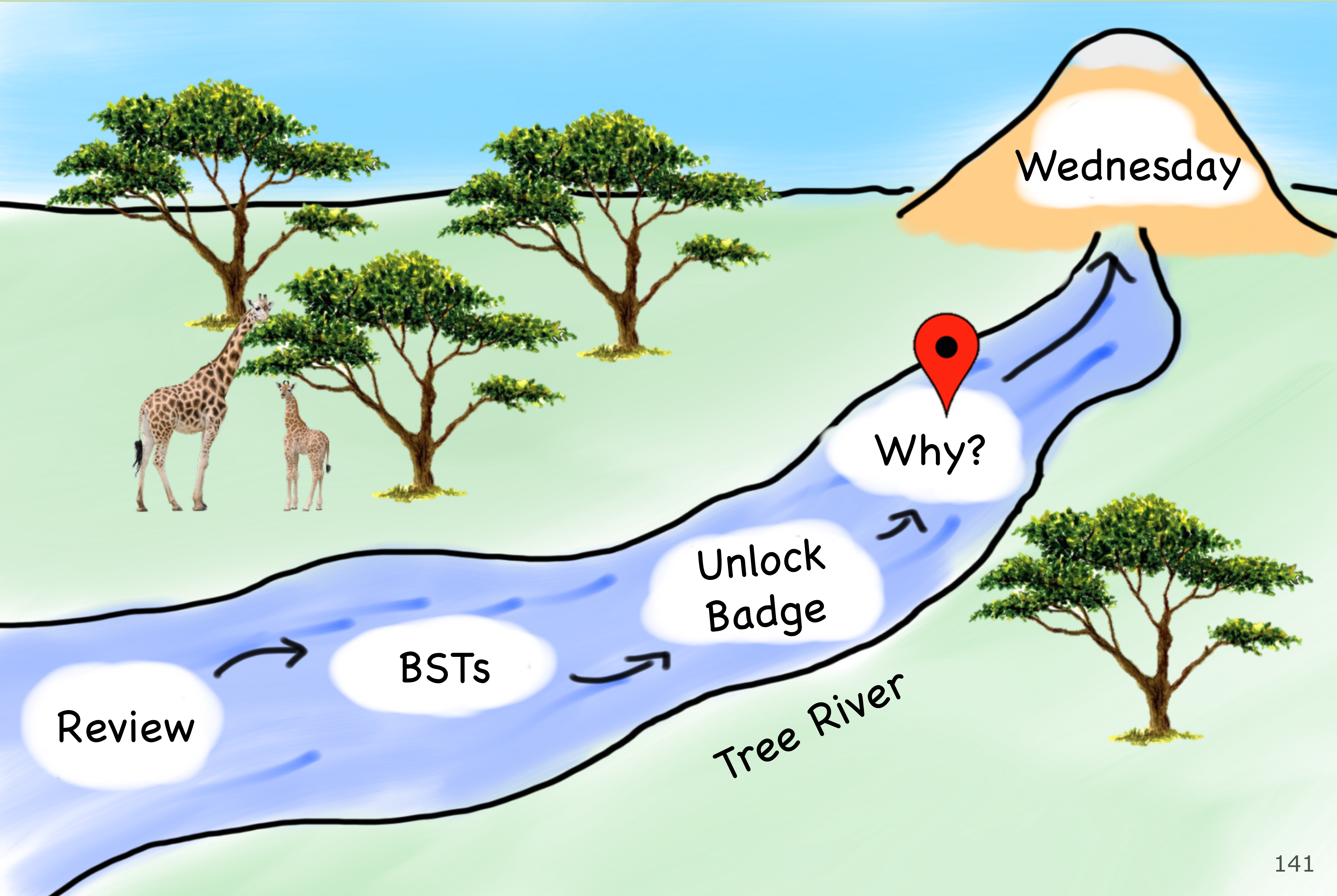
Map

Set

Today's Route



Today's Route



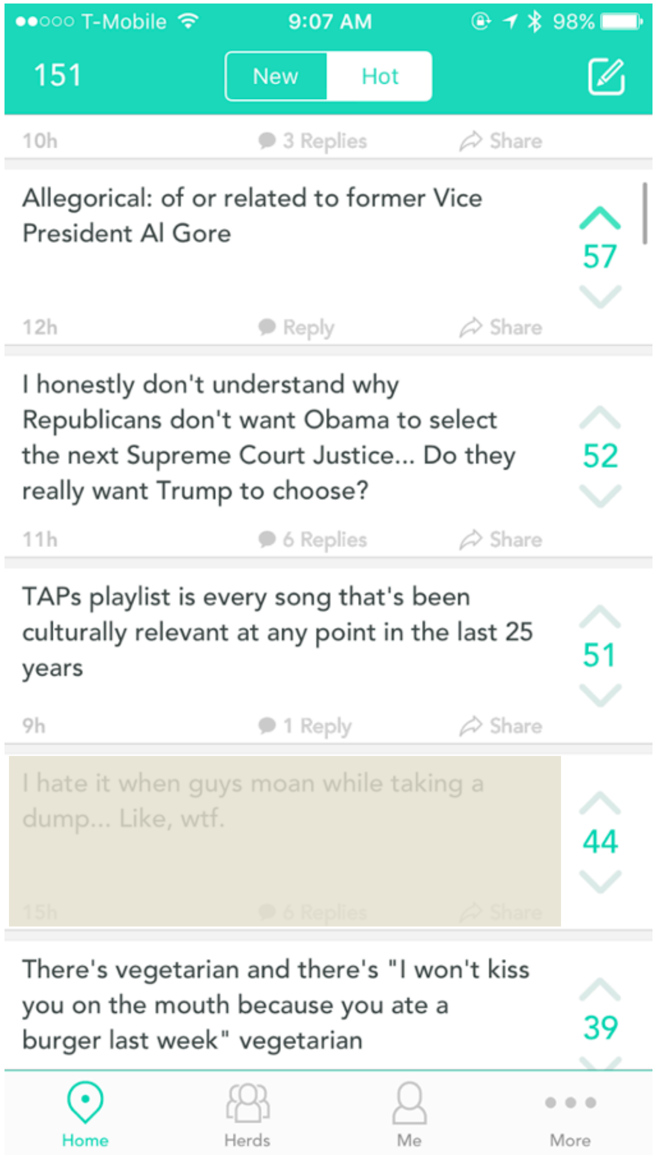
Great Ideas

Ok folks…

Let’s Talk about YikYak

How do you think its implemented?

YikYak



●●○○○ T-Mobile 9:07 AM 98%

151 New Hot

10h 3 Replies Share

Allegorical: of or related to former Vice President Al Gore

57

12h Reply Share

I honestly don't understand why Republicans don't want Obama to select the next Supreme Court Justice... Do they really want Trump to choose?

52

11h 6 Replies Share

TAPs playlist is every song that's been culturally relevant at any point in the last 25 years

51

9h 1 Reply Share

I hate it when guys moan while taking a dump... Like, wtf.

44

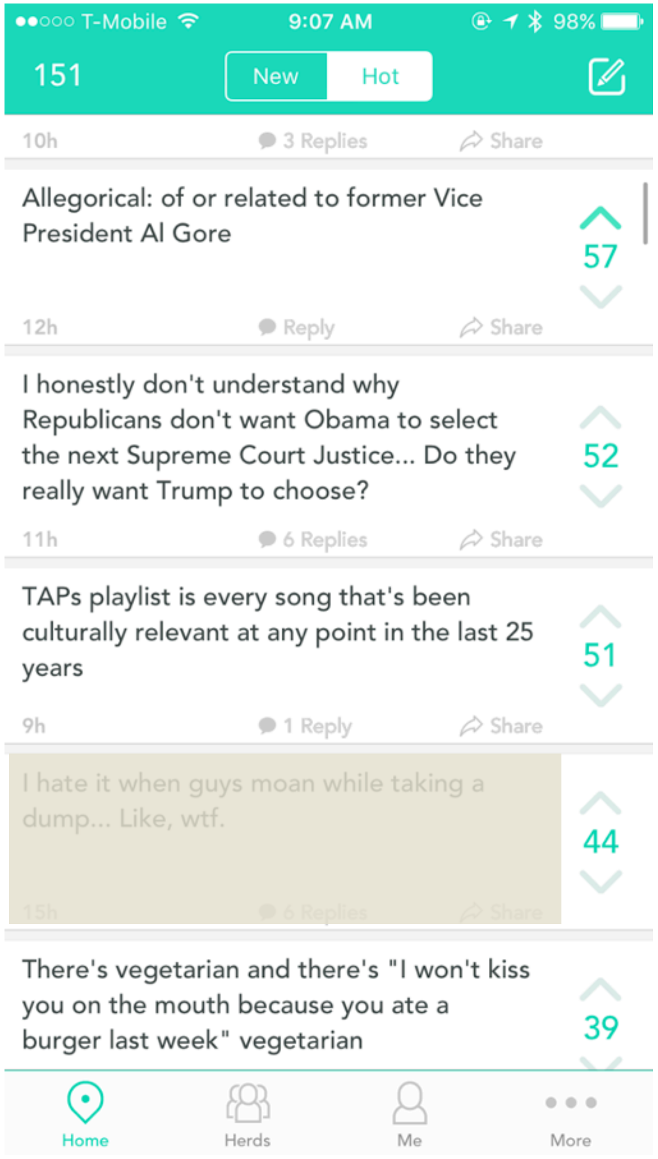
15h 6 Replies Share

There's vegetarian and there's "I won't kiss you on the mouth because you ate a burger last week" vegetarian

39

Lol

Newsfeed?



●●○○○ T-Mobile 9:07 AM 98%

151 New Hot

10h 3 Replies Share

Allegorical: of or related to former Vice President Al Gore

57

12h Reply Share

I honestly don't understand why Republicans don't want Obama to select the next Supreme Court Justice... Do they really want Trump to choose?

52

11h 6 Replies Share

TAPs playlist is every song that's been culturally relevant at any point in the last 25 years

51

9h 1 Reply Share

I hate it when guys moan while taking a dump... Like, wtf.

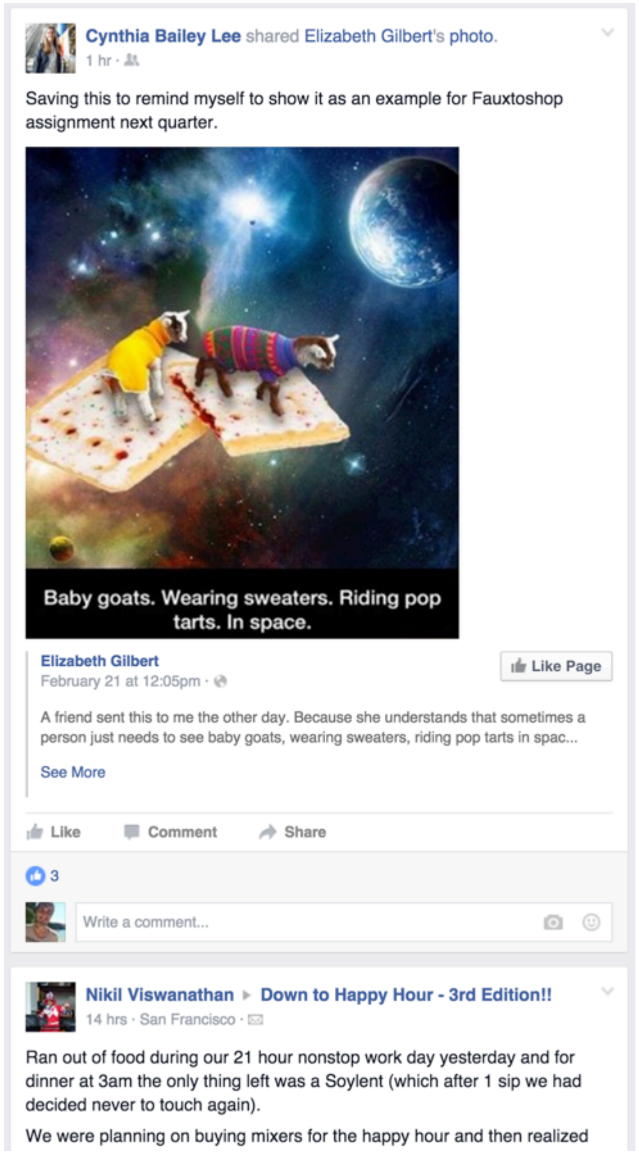
44

15h 6 Replies Share

There's vegetarian and there's "I won't kiss you on the mouth because you ate a burger last week" vegetarian

39

Home Herds Me More



Cynthia Bailey Lee shared Elizabeth Gilbert's photo.
1 hr ·

Saving this to remind myself to show it as an example for Fauxtoshop assignment next quarter.

Baby goats. Wearing sweaters. Riding pop tarts. In space.

Elizabeth Gilbert
February 21 at 12:05pm ·

Like Page

A friend sent this to me the other day. Because she understands that sometimes a person just needs to see baby goats, wearing sweaters, riding pop tarts in spac...

See More

Like Comment Share

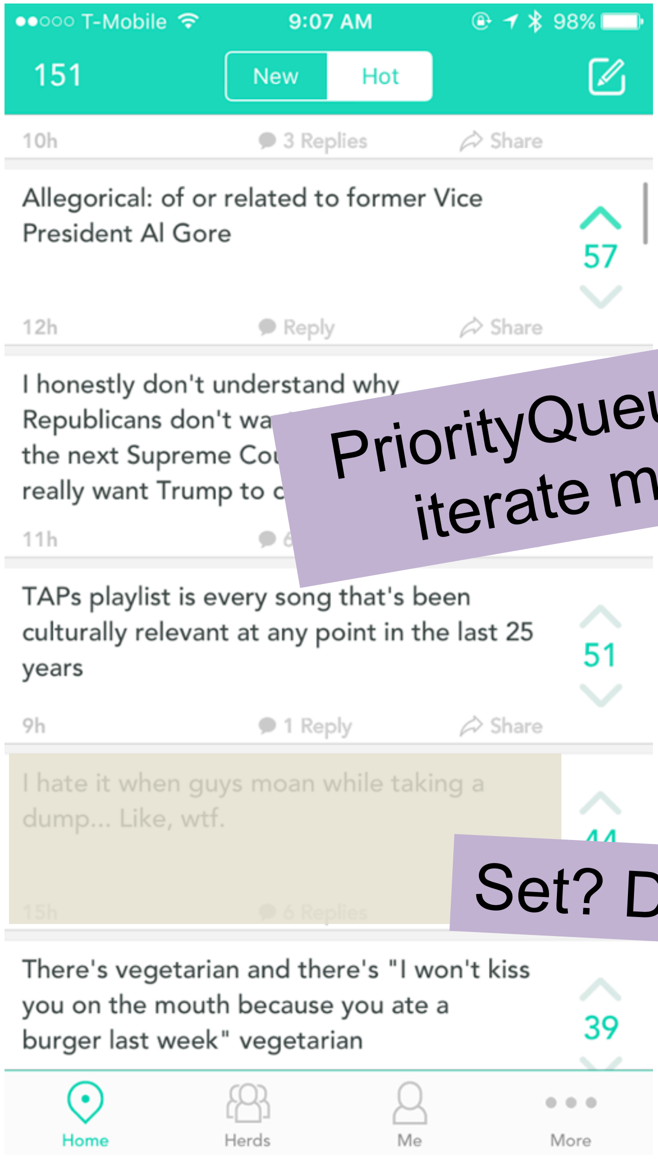
3

Write a comment...

Nikil Viswanathan ▸ Down to Happy Hour - 3rd Edition!!
14 hrs · San Francisco ·

Ran out of food during our 21 hour nonstop work day yesterday and for dinner at 3am the only thing left was a Soylent (which after 1 sip we had decided never to touch again).
We were planning on buying mixers for the happy hour and then realized

Newsfeed?



●●○○○ T-Mobile 9:07 AM 98%

151 New Hot

10h 3 Replies Share

Allegorical: of or related to former Vice President Al Gore 57

12h Reply Share

I honestly don't understand why Republicans don't wa the next Supreme Cou really want Trump to c

11h

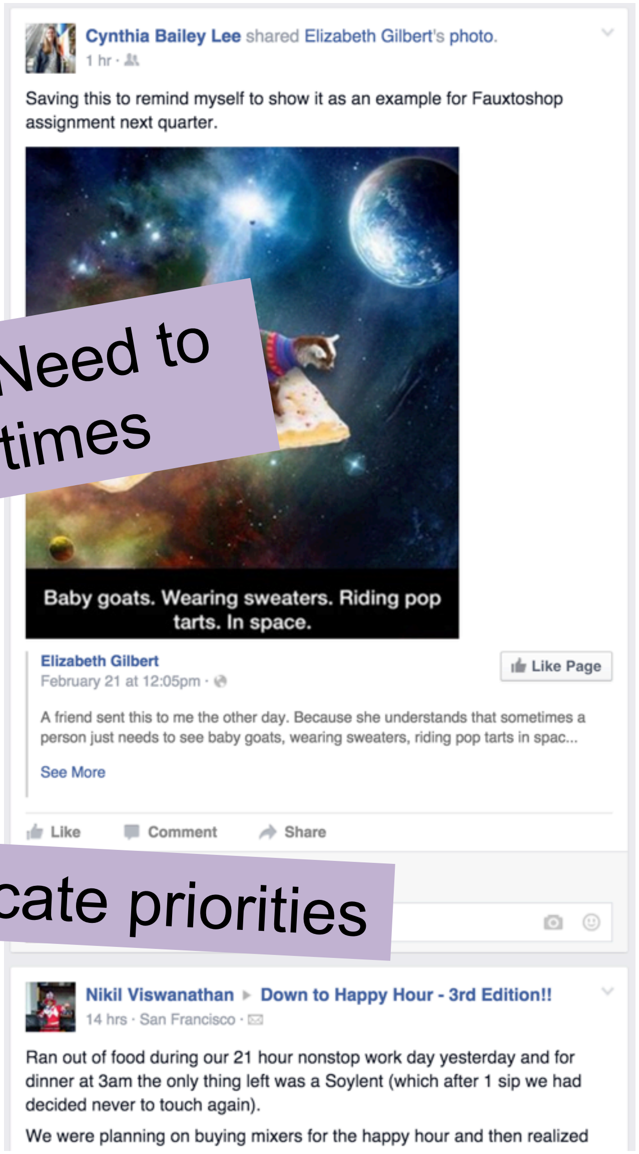
TAPs playlist is every song that's been culturally relevant at any point in the last 25 years 51

9h 1 Reply Share

I hate it when guys moan while taking a dump... Like, wtf. 44

15h 6 Replies

There's vegetarian and there's "I won't kiss you on the mouth because you ate a burger last week" vegetarian 39



Cynthia Bailey Lee shared Elizabeth Gilbert's photo.
1 hr

Saving this to remind myself to show it as an example for Fauxtoshop assignment next quarter.

Baby goats. Wearing sweaters. Riding pop tarts. In space.

Elizabeth Gilbert
February 21 at 12:05pm

Like Page

A friend sent this to me the other day. Because she understands that sometimes a person just needs to see baby goats, wearing sweaters, riding pop tarts in spac...

See More

Like Comment Share

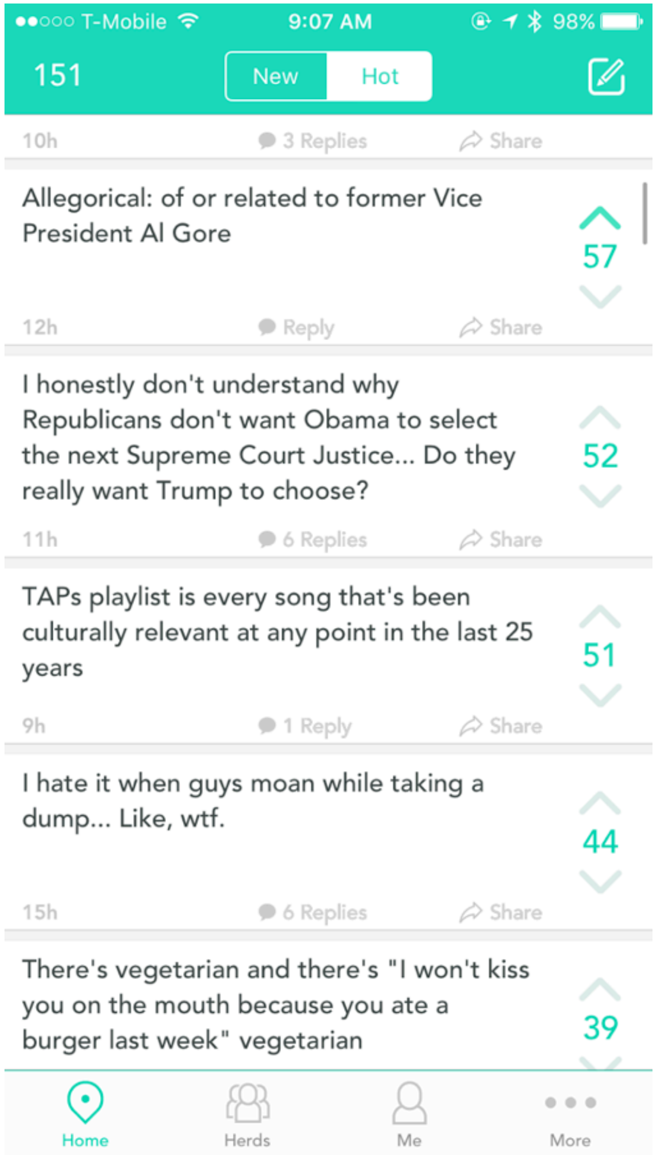
Nikil Viswanathan ▸ Down to Happy Hour - 3rd Edition!!
14 hrs · San Francisco

Ran out of food during our 21 hour nonstop work day yesterday and for dinner at 3am the only thing left was a Soylent (which after 1 sip we had decided never to touch again).
We were planning on buying mixers for the happy hour and then realized

PriorityQueue? Need to iterate many times

Set? Duplicate priorities

Priority Based with Iteration



●●○○○ T-Mobile 9:07 AM 98%

151 New Hot

10h 3 Replies Share

Allegorical: of or related to former Vice President Al Gore

57

12h Reply Share

I honestly don't understand why Republicans don't want Obama to select the next Supreme Court Justice... Do they really want Trump to choose?

52

11h 6 Replies Share

TAPs playlist is every song that's been culturally relevant at any point in the last 25 years

51

9h 1 Reply Share

I hate it when guys moan while taking a dump... Like, wtf.

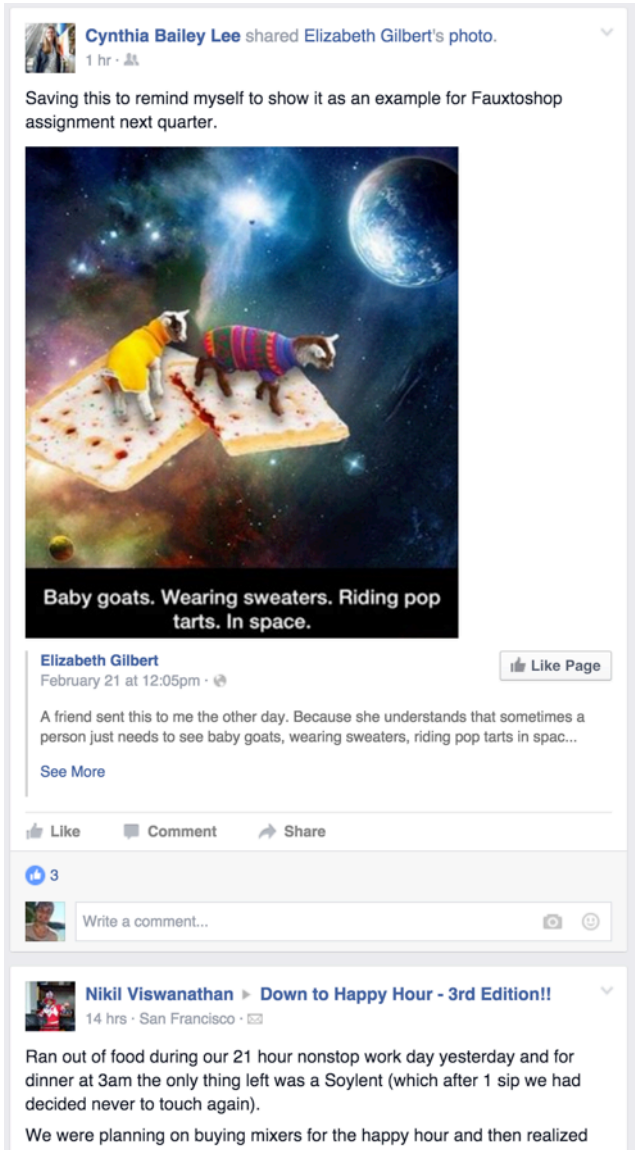
44

15h 6 Replies Share

There's vegetarian and there's "I won't kiss you on the mouth because you ate a burger last week" vegetarian

39

Home Herds Me More



Cynthia Bailey Lee shared Elizabeth Gilbert's photo.
1 hr ·

Saving this to remind myself to show it as an example for Fauxtoshop assignment next quarter.

Baby goats. Wearing sweaters. Riding pop tarts. In space.

Elizabeth Gilbert
February 21 at 12:05pm ·

Like Page

A friend sent this to me the other day. Because she understands that sometimes a person just needs to see baby goats, wearing sweaters, riding pop tarts in spac...

See More

Like Comment Share

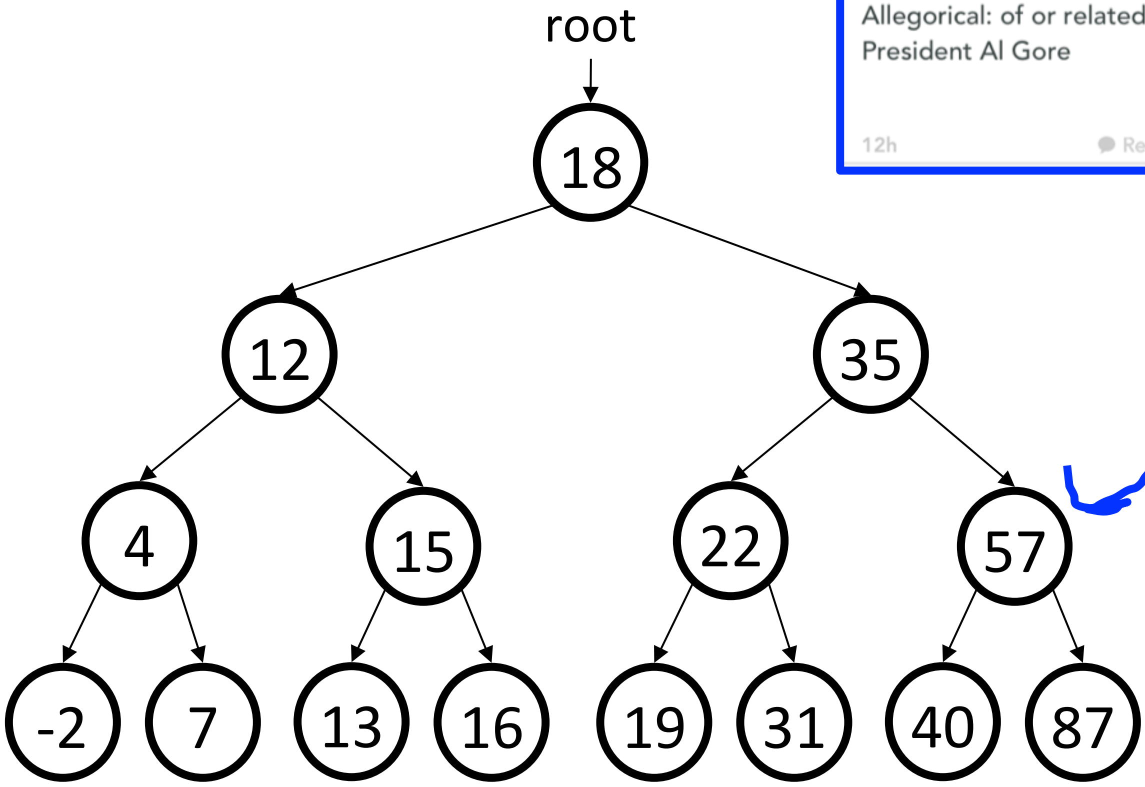
3

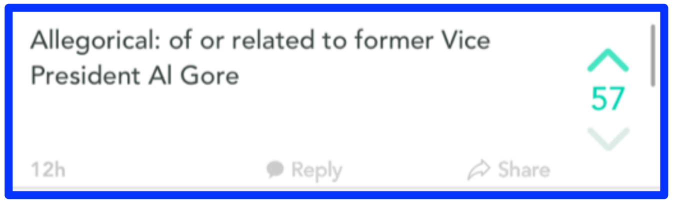
Write a comment...

Nikil Viswanathan ▸ Down to Happy Hour - 3rd Edition!!
14 hrs · San Francisco ·

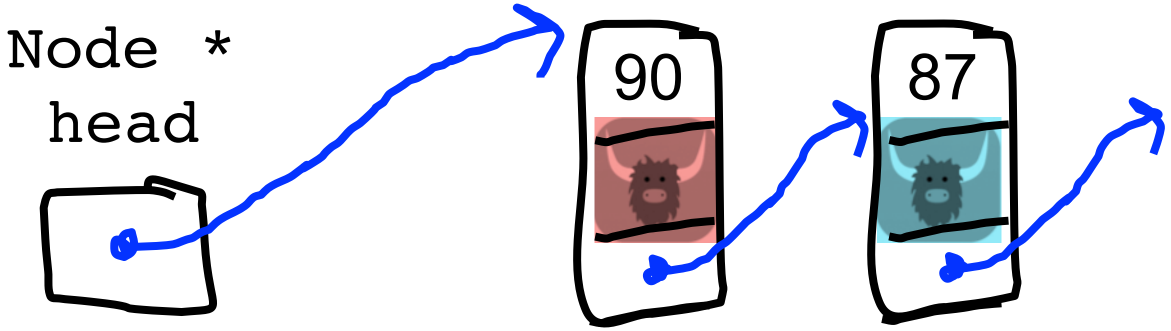
Ran out of food during our 21 hour nonstop work day yesterday and for dinner at 3am the only thing left was a Soylent (which after 1 sip we had decided never to touch again).
We were planning on buying mixers for the happy hour and then realized

BST that Allows Duplicates?

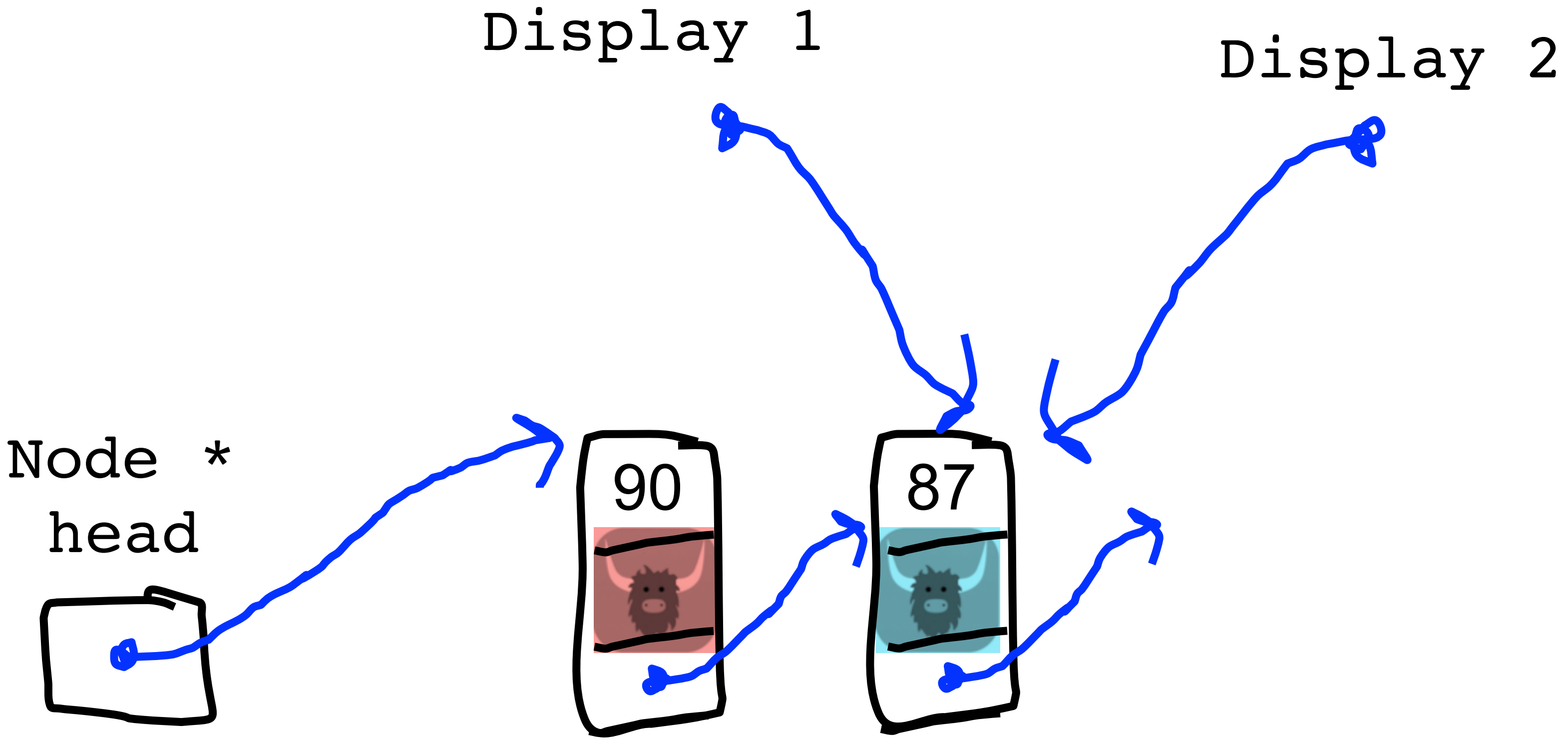



Allegorical: of or related to former Vice President Al Gore

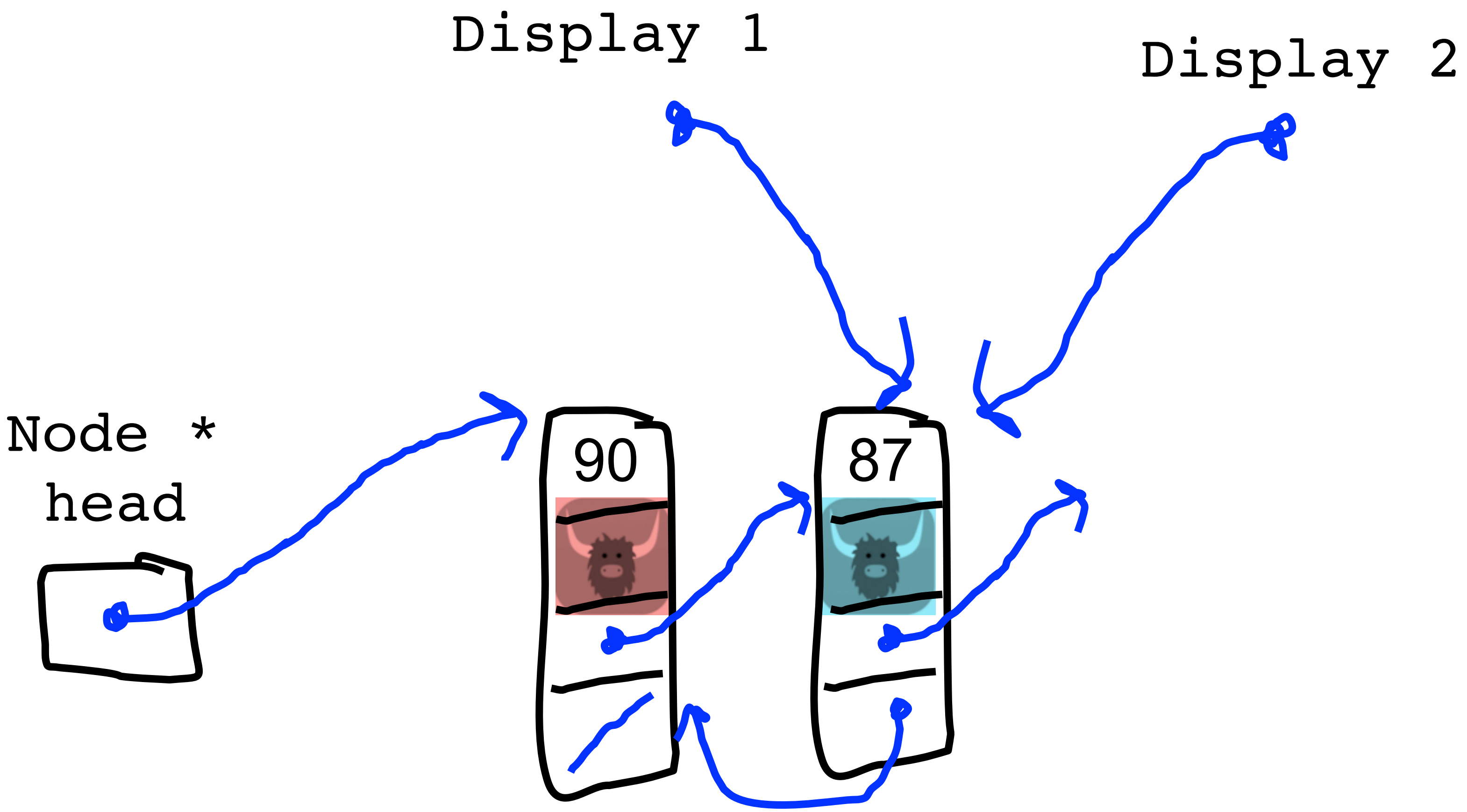
Linked List?



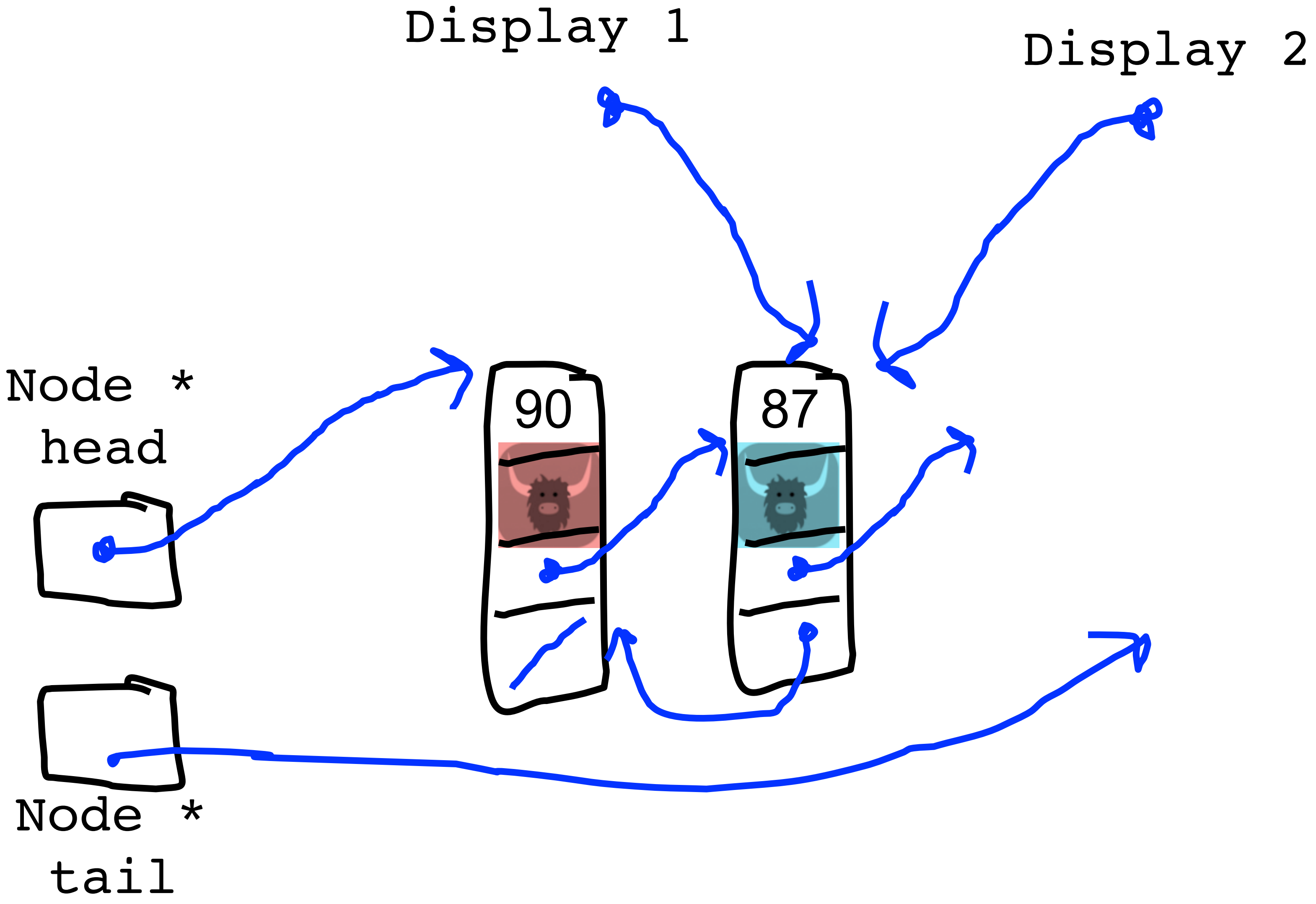
Linked List?



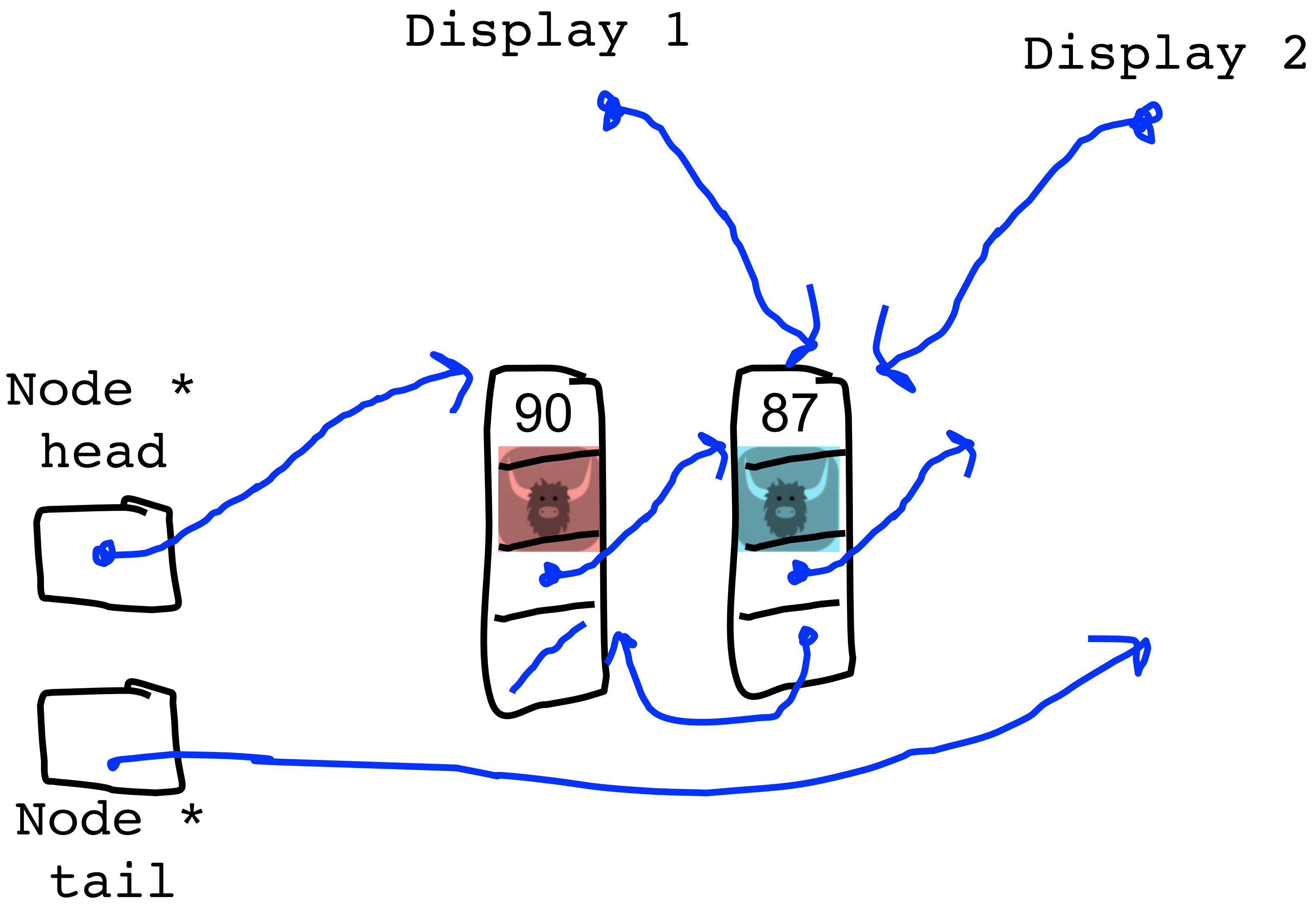
Linked List?



Linked List?

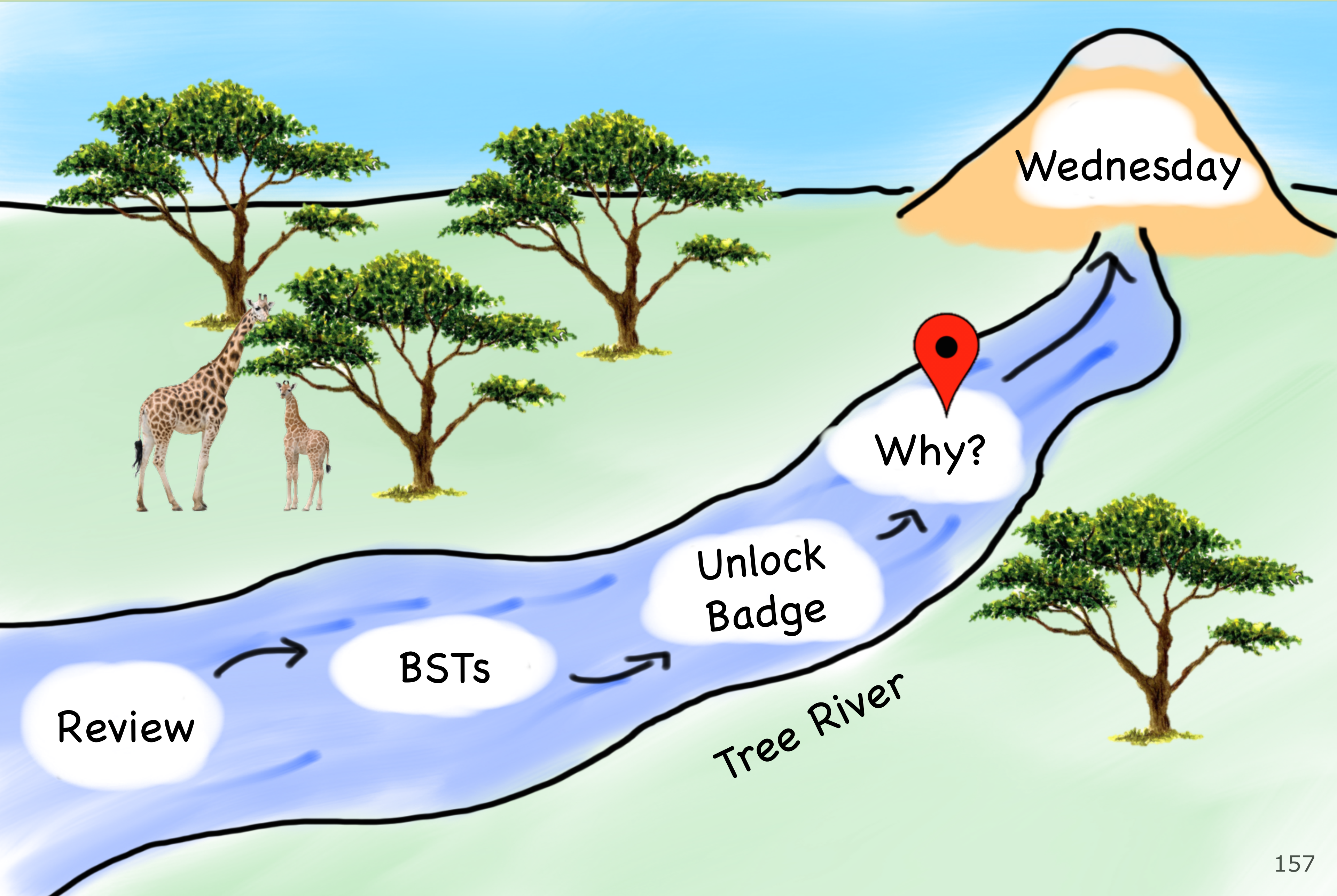


Yakray

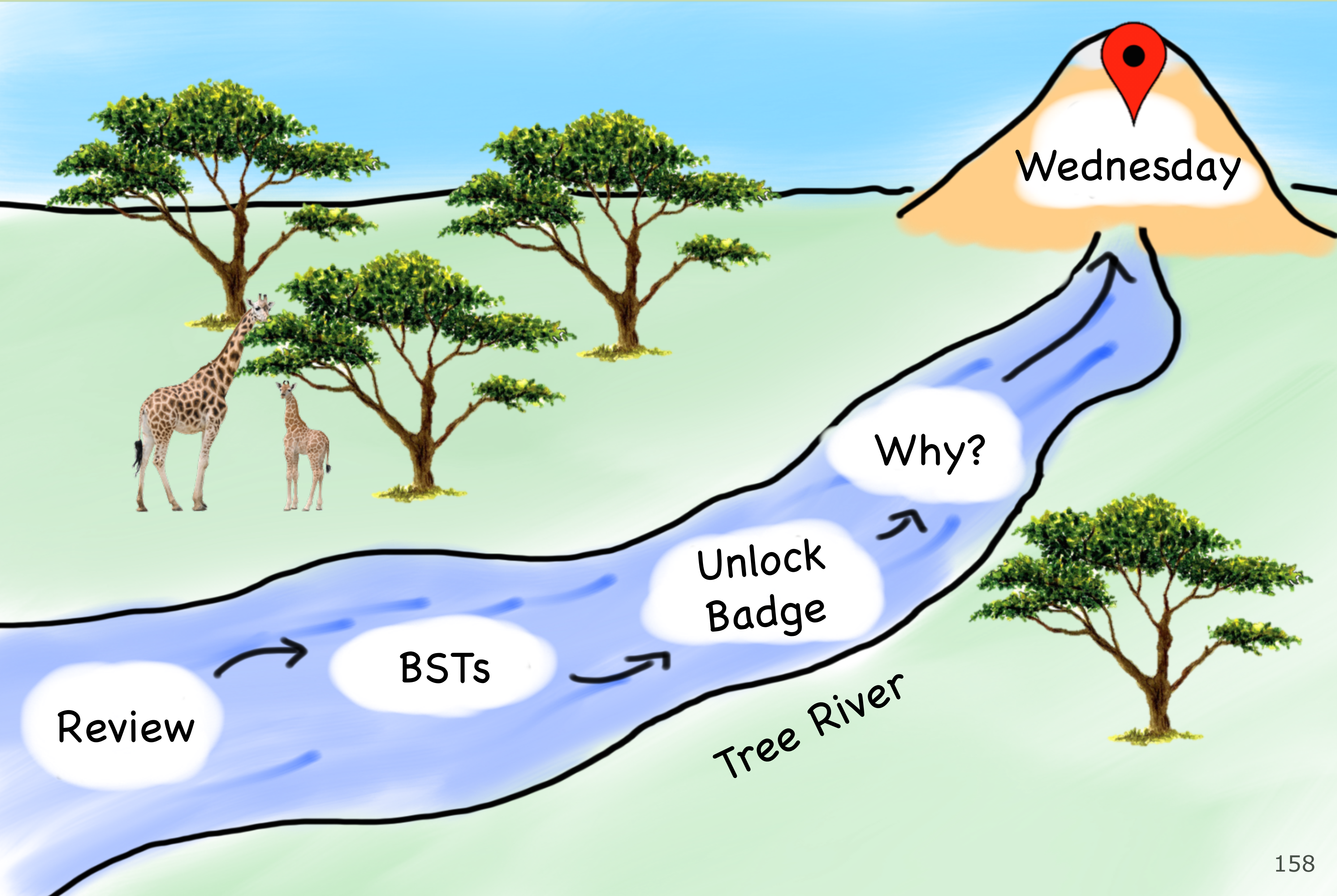


Happy with that

Today's Route



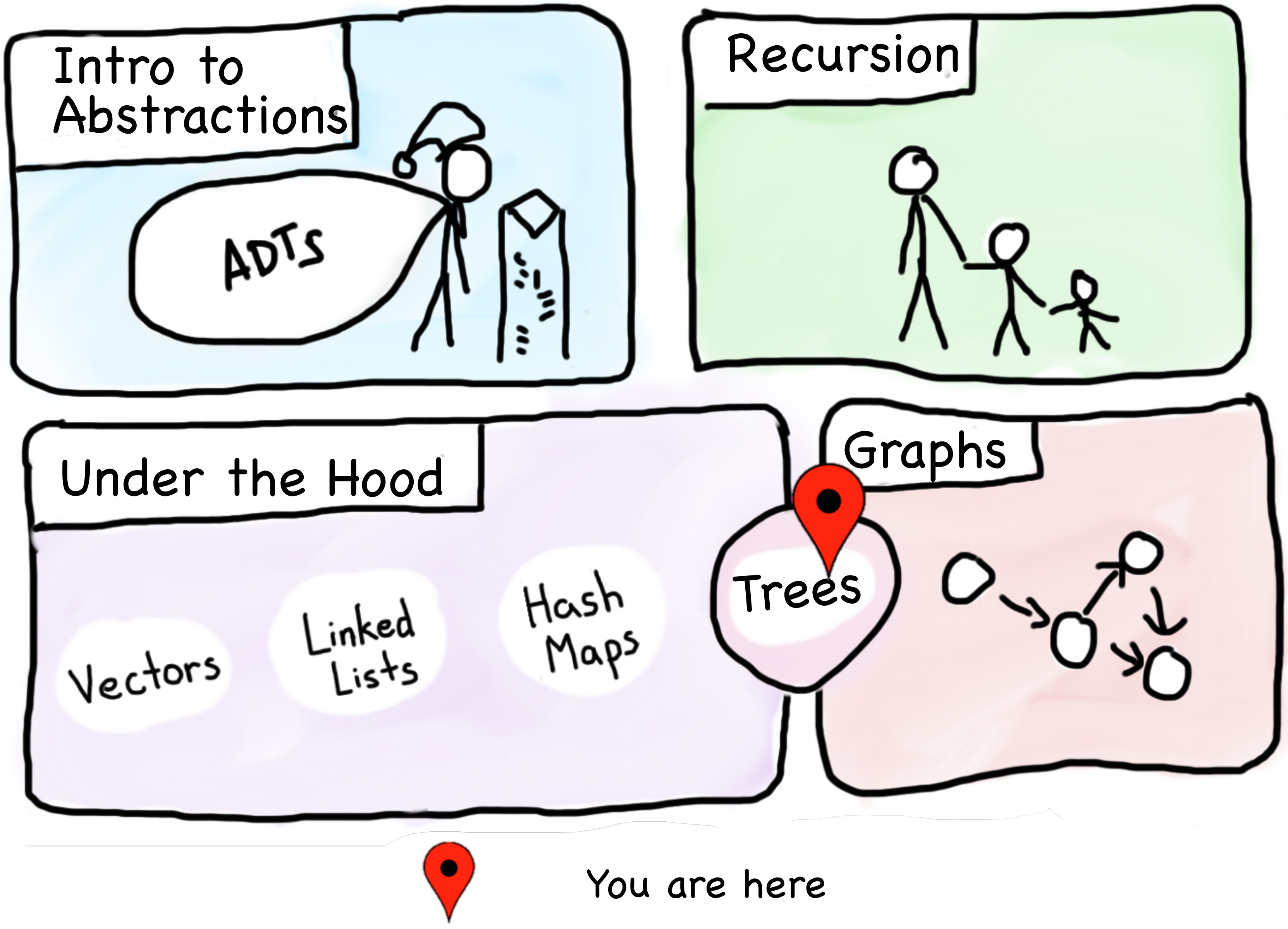
Today's Route



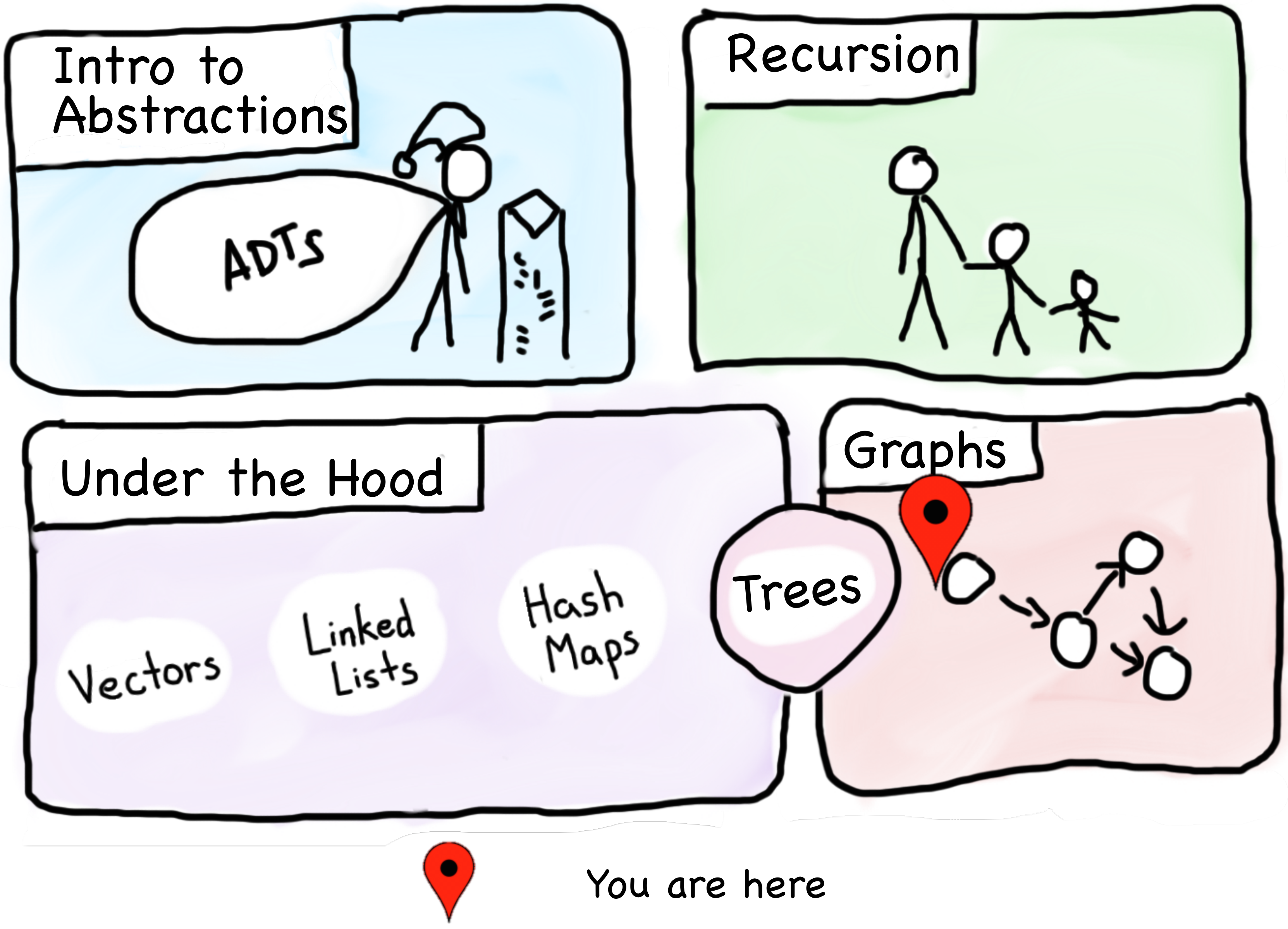
Today's Goal

1. Binary Search Trees
2. Review Under the Hood

Course Syllabus



Course Syllabus



[Drops mic, walk away]

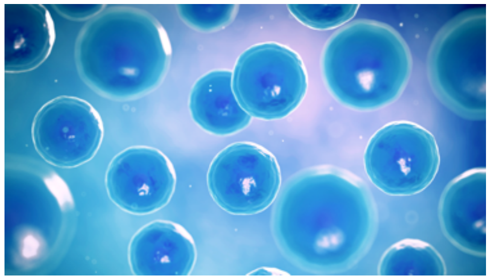
Incase of extra time, pick up mic

Supervised Machine Learning

$$\begin{tikzcd}\textit{High dimensional vector} \arrow[r] & \textit{Label}\end{tikzcd}$$

Not on Final

Supervised Machine Learning



High dimensional vector $\longrightarrow$ *Label*

Not on Final

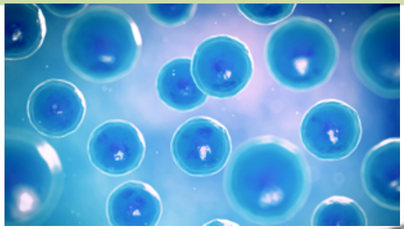
Supervised Machine Learning

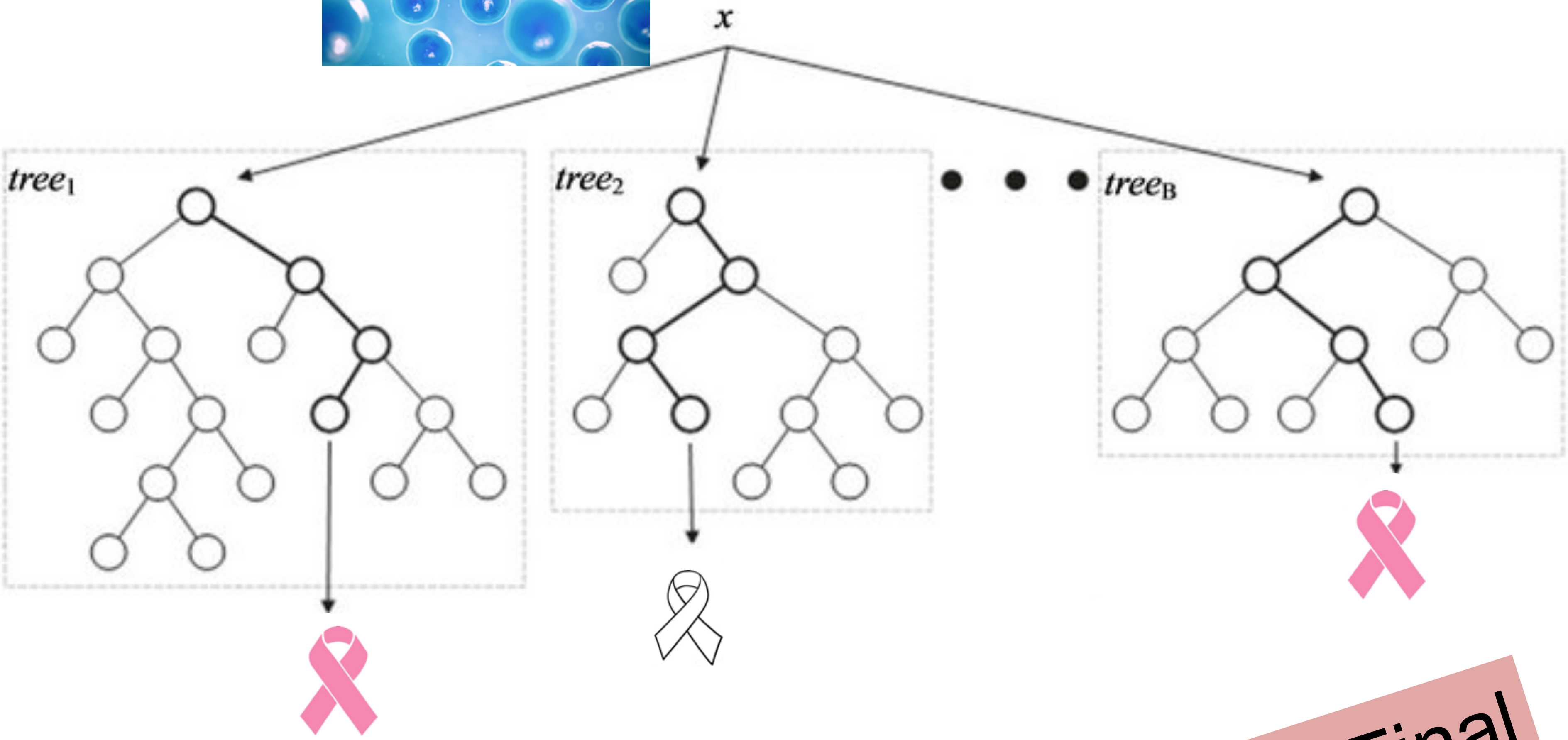
$$\begin{tikzcd}\textit{High dimensional vector} \arrow[r, "\texttt{Training}"] & \textit{Label}\end{tikzcd}$$

$$\begin{tikzcd}\textit{High dimensional vector} \arrow[r, "\texttt{Testing}"] & \textit{?}\end{tikzcd}$$

Not on Final

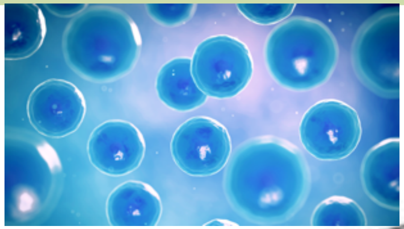
Random Forest

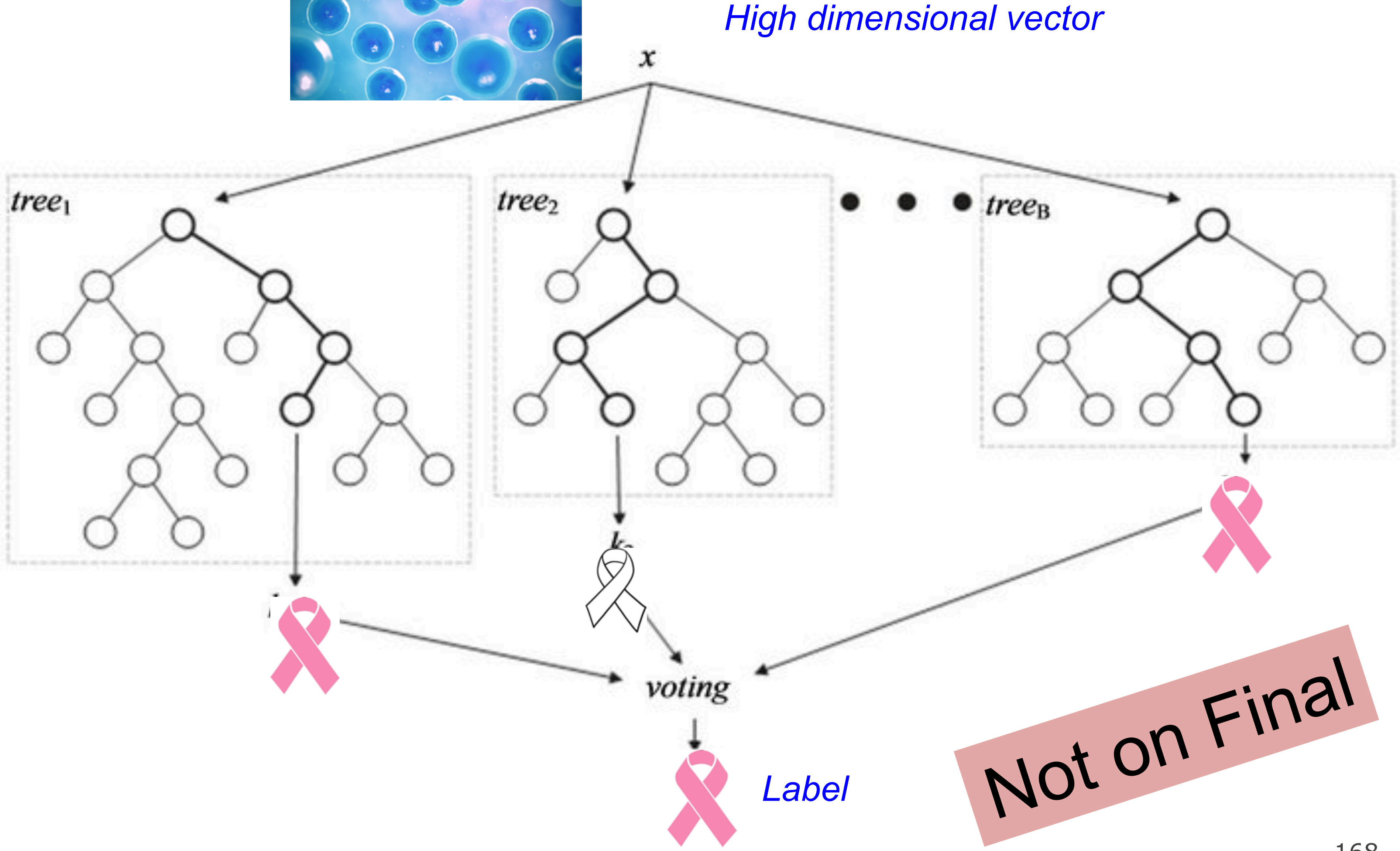




Not on Final

Random Forest





Not on Final