BST

Wednesday, May 31, 2017 4:18 PM

BINARY SEARCH TREES

Problem Solving with Computers-I

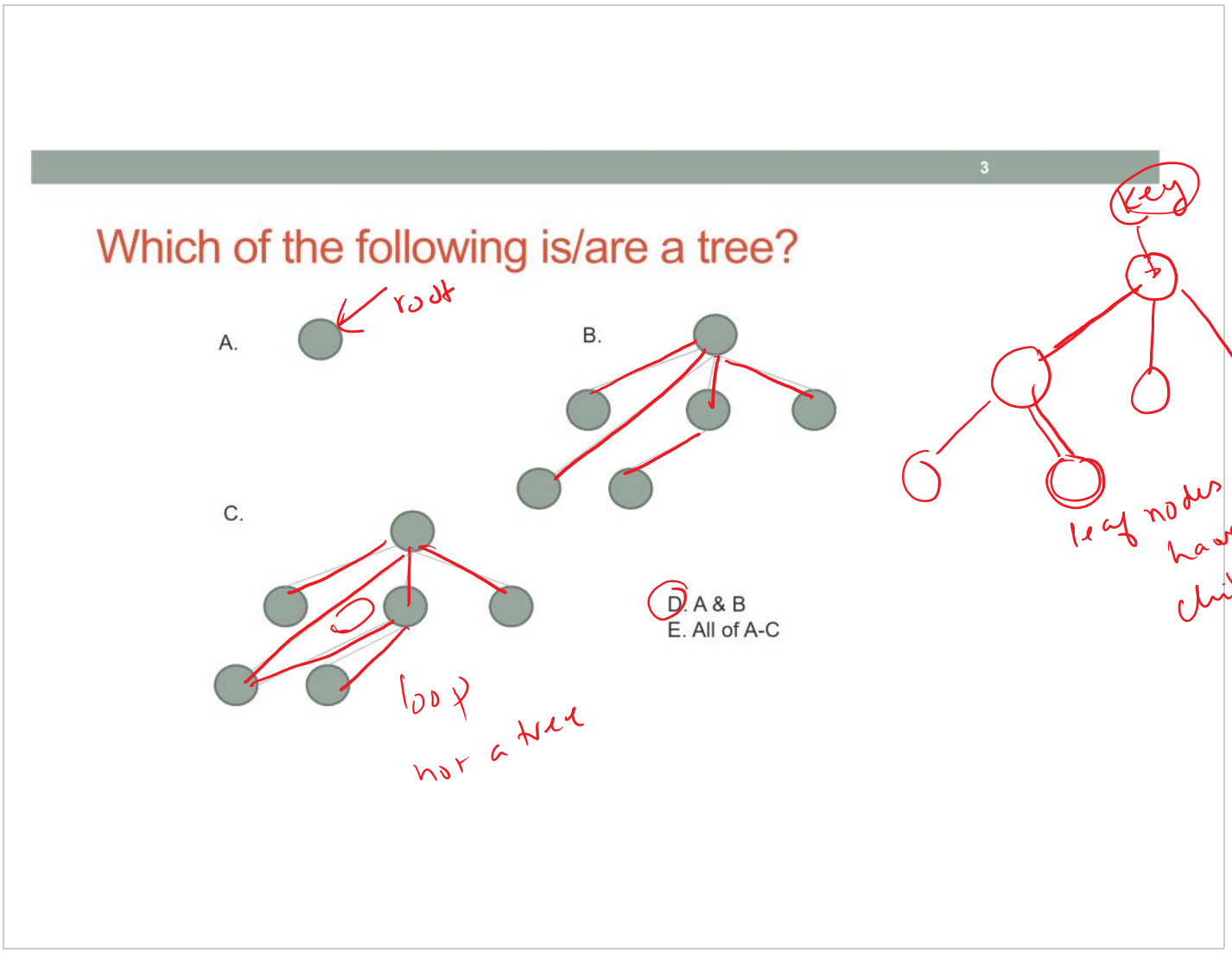
https://ucsb-cs24-sp17.github.io/

Imposter panel: Tomorrow Thurs (06/01), 12:30pm to 1:50pm, HFH 1132

Come hear faculty, grad students and undergrad alumni talk about their careers and how they dealt with feeling like an Imposter!

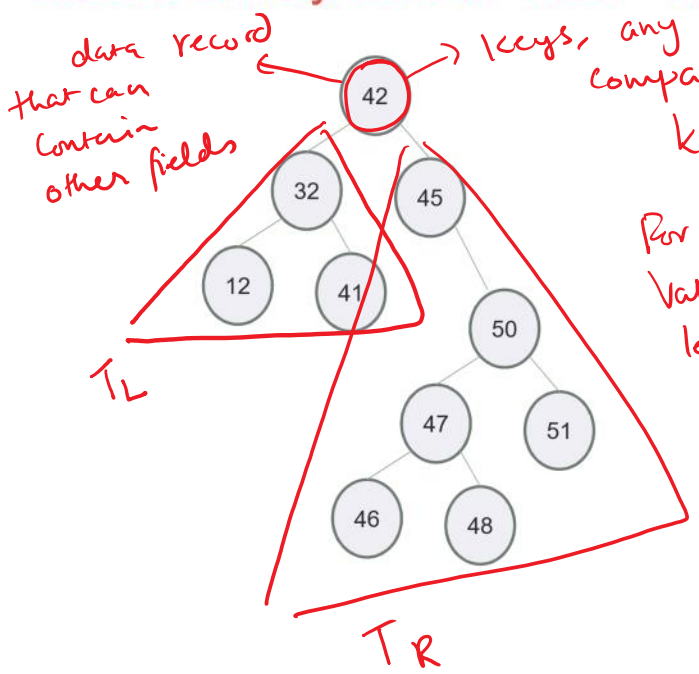
Please RSVP : https://goo.gl/forms/ttvzHNPWAZ0GCPA92

Wednesday, May 31, 2017 4:18 PM



Wednesday, May 31, 2017 4:18 PM

Lab08: Binary Search Tree – What is it?



Key: ← ordering is available on the keys

For any node in a BST

Value of all nodes in the left subtree ≤ value of the node

$T_L \leq$ value of node

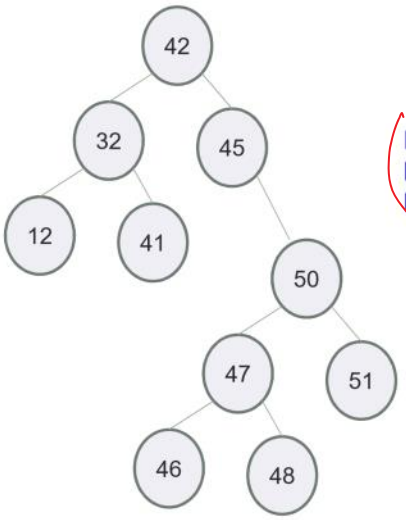
value of node $< T_R$

What are the numbers in the nodes?

Wednesday, May 31, 2017 4:18 PM

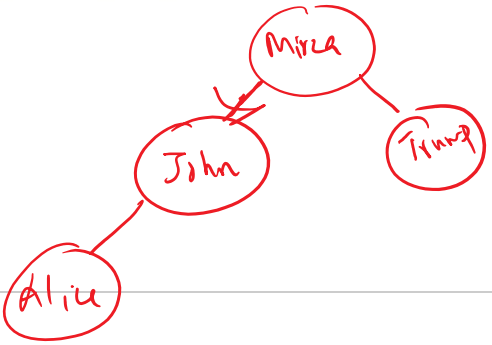
5

Binary Search Tree – What is it?



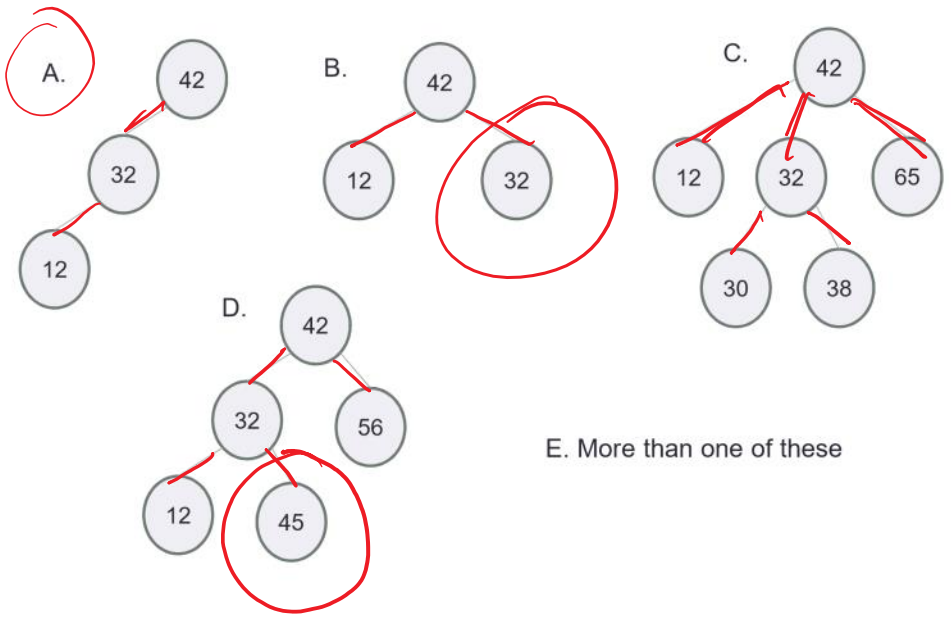
For any node,
Keys in node's left subtree <= Node key
Node key < Keys in node's right subtree

Do the keys have to be integers?



Wednesday, May 31, 2017 4:19 PM

Which of the following is/are a binary search tree?



E. More than one of these

Wednesday, May 31, 2017 4:19 PM

7

Binary Search Trees

- What are the operations supported?
- What are the running times of these operations?
- How do you implement the BST i.e. operations supported by it?

Wednesday, May 31, 2017 4:19 PM

8

Binary Search Trees

- What is it good for?
  - If it satisfies a special property i.e. Balanced, you can think of it as a dynamic version of the sorted array

everything that is supported by a sorted array + Fast insert & delete

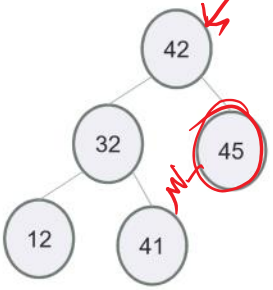
⇝ (under some conditions)

Wednesday, May 31, 2017 4:19 PM

Under the hood: Searching an element in the BST

To search for element with key k

1. Start at the root
2. If k=key(root), found key, stop.
3. Else If k< key(root), recursively search the left subtree: T_L
   Else recursively search the right subtree: T_R



Search for 41.
Now search for 43.

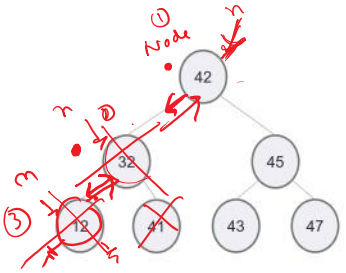
Wednesday, May 31, 2017 4:19 PM

10

Traversing the BST

Different methods of tree traversal:

- In order traversal
- Pre order traversal
- Post order traversal




```
struct Node {
    int data;
    Node * left;
    Node * right;
    Node * parent;
}
```

```
void inorder (Node *n) {
    if (n == 0)
        return;
    inorder (n->left);
    cout << n->data << " ";
    inorder (n->right);
}
```

```
void preorder (Node *n) {
    if (n) {
        cout << n->data << " ";
        preorder (n->left);
        preorder (n->right);
    }
}
```

12 32 41 42 43 45 47

O(N)

Wednesday, May 31, 2017 4:19 PM

11

BST, with templates:

```
template<typename Data>

class BSTNode {
public:
  BSTNode<Data>* left;
  BSTNode<Data>* right;
  BSTNode<Data>* parent;
  Data const data;

  BSTNode( const Data & d ) :
     data(d) {
    left = right = parent = 0;
  }

};
```

Wednesday, May 31, 2017 4:19 PM

BST, with templates:

```
template<typename Data>

class BSTNode {
public:
  BSTNode<Data>* left;
  BSTNode<Data>* right;
  BSTNode<Data>* parent;
  Data const data;

  BSTNode( const Data & d ) :
     data(d) {
    left = right = parent = 0;
  }

};
```

How would you create **a BSTNode object** on the runtime stack?

A. BSTNode n(10);
B. BSTNode<int> n;
C. BSTNode<int> n(10);
D. BSTNode<int> n = new BSTNode<int>(10);
E. More than one of these will work

{ } syntax OK too

Wednesday, May 31, 2017 4:19 PM

13

BST, with templates:

```
template<typename Data>

class BSTNode {
public:
  BSTNode<Data>* left;
  BSTNode<Data>* right;
  BSTNode<Data>* parent;
  Data const data;

  BSTNode( const Data & d ) :
     data(d) {
    left = right = parent = 0;
  }

};
```

How would you create a **pointer** to BSTNode with integer data?

A. BSTNode* nodePtr;
B. BSTNode<int> nodePtr;
C. BSTNode<int>* nodePtr;

Wednesday, May 31, 2017 4:19 PM

14

BST, with templates:

```
template<typename Data>

class BSTNode {
public:
  BSTNode<Data>* left;
  BSTNode<Data>* right;
  BSTNode<Data>* parent;
  Data const data;

  BSTNode( const Data & d ) :
     data(d) {
    left = right = parent = 0;
  }

};
```

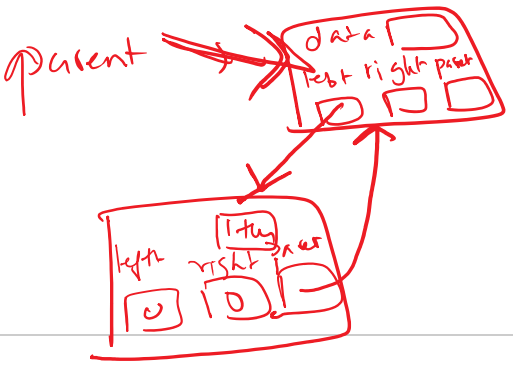
Complete the line of code to create a new BSTNode object with int data on the heap and assign nodePtr to point to it.

```
BSTNode<int>* nodePtr
```

Wednesday, May 31, 2017 4:19 PM

Working with a BST

```
template<typename Data>
class BST {

private:

  /** Pointer to the root of this BST, or 0 if the BST is empty */
  BSTNode<Data>* root;

public:

  /** Default constructor. Initialize an empty BST. */
  BST() : root(nullptr){  }

  void insertAsLeftChild(BSTNode<Data>* parent, const Data & item)
  {
     // Your code here
  }
```



Wednesday, May 31, 2017 4:19 PM

16

Working with a BST: Insert

```
void insertAsLeftChild(BSTNode<Data>* parent, const Data & item)
  {
     // Your code here
  }
```

Which line of code correctly inserts the data item into the BST as the left child of the parent parameter.

A. `parent.left = item;`
B. `parent->left = item;`
C. `parent->left = BSTNode(item);`
D. `parent->left = new BSTNode<Data>(item);`
E. `parent->left = new Data(item);`

parent→left → parent = parent;

Wednesday, May 31, 2017 4:19 PM

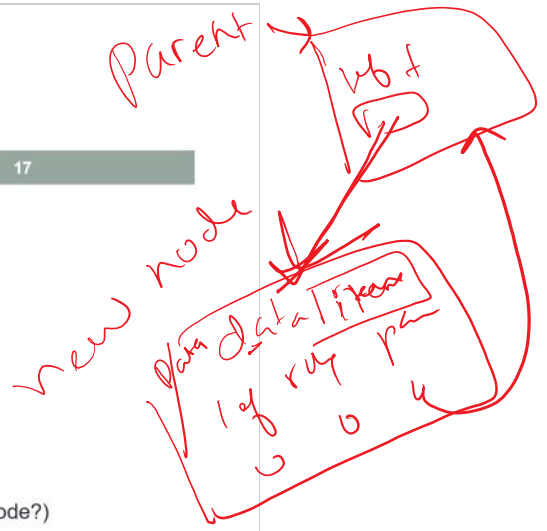
17

Working with a BST: Insert

```
void insertAsLeftChild(BSTNode<Data>* parent, const Data & item)
{
    parent->left = new BSTNode<Data>(item);
}
```

Is this function complete? (i.e. does it to everything it needs to correctly insert the node?)

A. Yes. The function correctly inserts the data
B. No. There is something missing.



Wednesday, May 31, 2017 4:19 PM

Working with a BST: Insert

```
void insertAsLeftChild(BSTNode<Data>* parent, const Data & item)
{

    parent->left = new BSTNode<Data>(item);


}
```