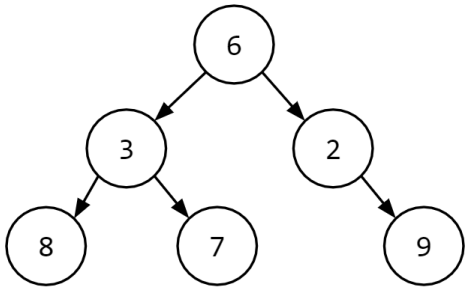
ANSWER SHEET

These problems were generated by TAs and instructors in previous semesters. They may or may not match the actual difficulty of problems on Quiz4.

Trees

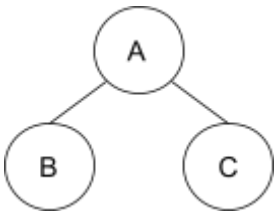
1. Write a function that calculates the height of a (possibly unbalanced) binary tree. For example, the tree below would have a height of 3.



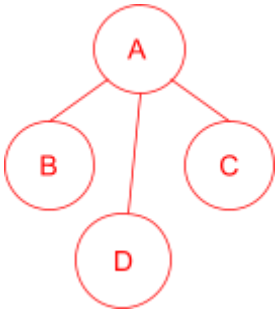
ANSWER:

```
def getHeight(tree):
    if tree == None:
        return 1
    else:
        right = getHeight(tree["right"])
        left = getHeight(tree["left"])
        return 1 + max(right, left)
```

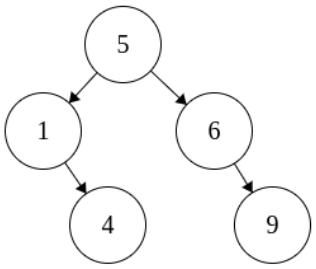
2. Add a node and edge to transform this tree from a binary tree to a general tree (i.e., it should no longer be binary).



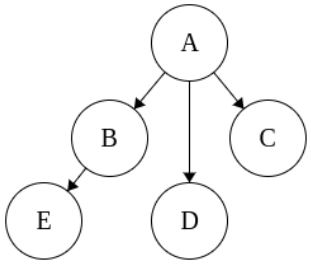
ANSWER:



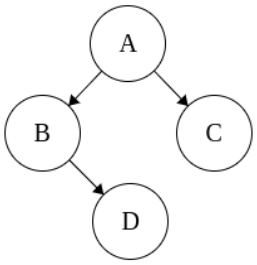
3. Label the following trees with either tree, binary tree, or binary search tree, giving the most specific term if multiple terms apply.



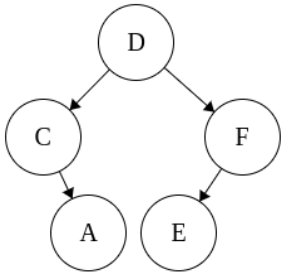
binary search tree



tree



binary tree



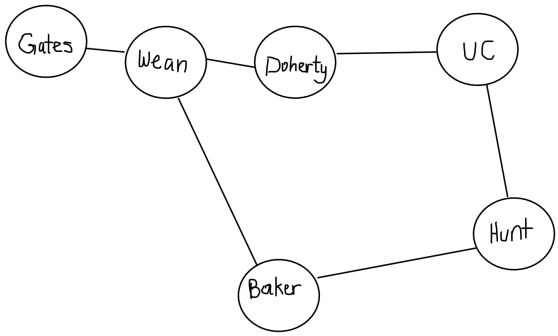
binary tree

Graphs

1. How are graphs and trees similar? How are they different?

ANSWER:
Similar: both use nodes and connect the nodes together
Different: any node can be connected to any other node in a graph; there are more restrictions with trees (hierarchical)

2. Write the dictionary that would represent the following graph.



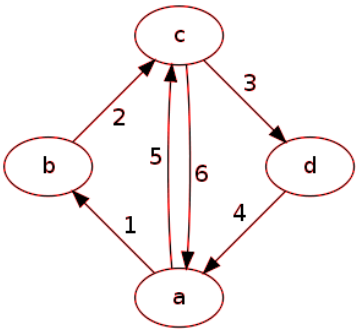
ANSWER:

```
{ "Baker"   : [ "Hunt", "Wean" ],
  "Hunt"    : [ "Baker", "UC" ],
  "Doherty" : [ "Wean", "UC" ],
  "Gates"   : [ "Wean" ],
  "UC"      : [ "Doherty", "Hunt" ],
  "Wean"    : [ "Baker", "Doherty", "Gates" ] }
```

3. Draw a directed graph with weights based on the given dictionary:

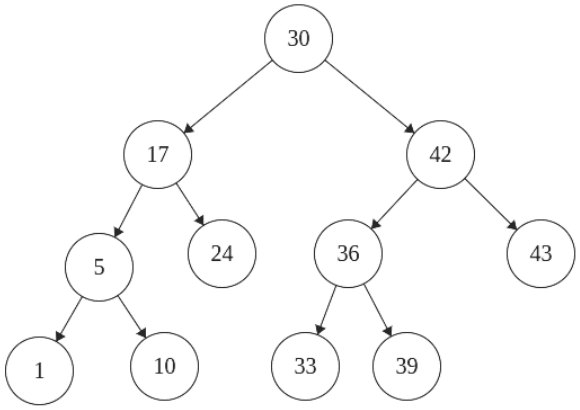
```
graph = { "a" : [ ["b", 1], ["c", 5] ],
          "b" : [ ["c", 2] ],
          "c" : [ ["a", 6], ["c", 3] ],
          "d" : [ ["a", 4] ] }
```

ANSWER:



Search Algorithms II

1. Consider the binary search tree below. What nodes would you visit while searching the tree for the value 33?



ANSWER:
Nodes visited: 30, 42, 36, 33

2. Given a description of a search algorithm, identify A) what kind of data structure is being searched, and B) what the name of that search algorithm is. Each algorithm is searching a data structure for the value `item`.

A: If the node's value equals `item`, return `True`. If `item` is less than the node's value, recurse on the left child and return the result; otherwise, recurse on the right child and return the result.

ANSWER:
Data structure: binary search tree
Algorithm: binary search

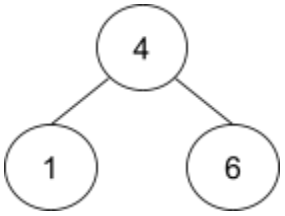
B: Go through each value sequentially, starting from the beginning. If you reach a value that equals `item`, return `True`. If you run out of values to search, return `False`.

ANSWER:
Data structure: list
Algorithm: linear search

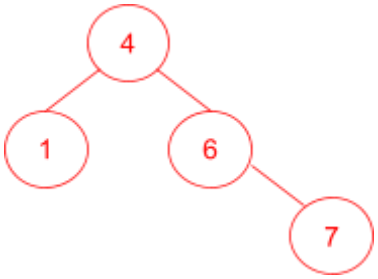
C: Begin with the start node in the to-visit list. While there are still nodes left to visit, check if the next node on the to-visit list equals `item`, and return `True` if it does. Then check if it has been visited before. If it has not, add all of the nodes connected to that node to the **end** of the to-visit list. If the to-visit list becomes empty, return `False`.

ANSWER:
Data structure: graph
Algorithm: breadth-first search

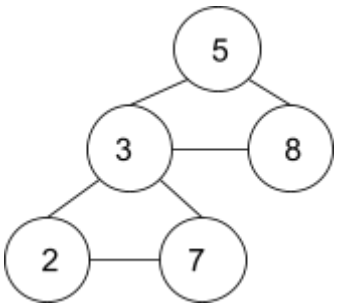
3. Add a node with value 7 to the tree so that it remains a binary search tree.



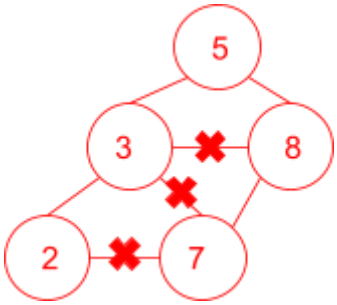
ANSWER:



4. Draw an **X** over each edge that must be removed and draw **a dotted line** for any edges that must be added to make this graph a binary search tree.



ANSWER:



Tractability

1. State True/False for the following questions and explain the answer.
   a. If we have an algorithm to solve a problem then we have a tractable solution.

ANSWER:
False. We can have an algorithm to solve an intractable problem.

   b. Any problem that runs in O(n^k) is intractable.

ANSWER:
False. n^k is polynomial. Intractable runtimes are: 2^n, k^n, and n!

   c. If any problem that can be solved in polynomial time can be verified in polynomial time, then we have proved P=NP.

ANSWER:
False. We already know every problem solved in polynomial time (every problem in P) is verifiable in polynomial time. This tells us P is a subset of NP.

2. Is exam scheduling a tractable problem? If yes, explain why / how you know. If not, explain how we still get all of our exams scheduled.

ANSWER:
No, exam scheduling is intractable. When we have these large, intractable problems, we find approximate solutions, and they just take a long time to generate.

3. For each question, check the box next to the correct answer.

☐ True or ☐ False	NP means “Not P” so everything not in P is in NP.
☐ True or ☐ False	If every problem in NP can be solved in polynomial time, then P=NP
☐ True or ☐ False	A problem is intractable if it can be solved but it may take too long to practically get that answer.
☐ True or ☐ False	All problems in NP are intractable to verify.