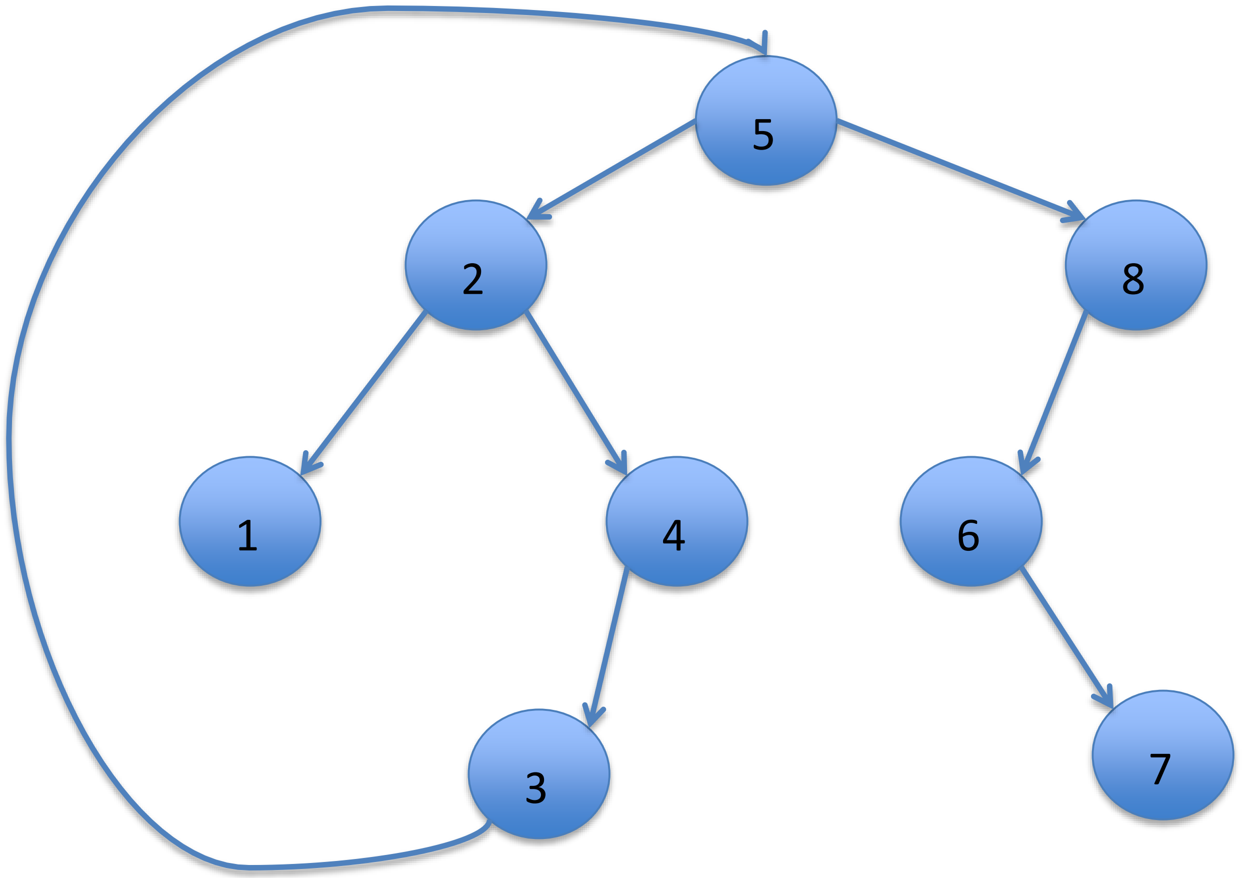
What if our trees are overgrown

- We have been implicitly assuming that there are no “loops” in our trees, i.e. that a child has one parent, and that no node is the parent of a child closer to the root
- What if we relax this constraint?
- Generalization is called a graph
  - Lots of great graph search problems
  - For now, we can think about ways to support search for binary trees that might have loops

An example “tree”



Searching these “trees”

- What happens if we run depth first search on this?

An infinite loop in many cases when item present, and always if item not present

- What happens if we run breadth first search on this?

Inefficient as repeats nodes, but still works if item present, infinite loop if not present

Avoiding loops

```
def DFSBinaryNoLoop(root, fcn):
    stack = [root]
    seen = []
    while len(stack) > 0:
        print 'at node ' + str(queue[0].getValue())
        if fcn(stack[0]):
            return True
        else:
            temp = stack.pop(0)
            seen.append(temp)
            if temp.getRightBranch():
                if not temp.getRightBranch() in seen:
                    stack.insert(0, temp.getRightBranch())
            if temp.getLeftBranch():
                if not temp.getLeftBranch() in seen:
                    stack.insert(0, temp.getLeftBranch())
    return False
```