Iterative Deepening Search

In [1]:

```
graph = {
    'a' : ['b', 'c', 'e'],
    'b' : ['d', 'f'],
    'c' : ['g','a'],
    'e' : ['f'],
    'f' : ['e'],
}

def IDDFS(root, goal):
    depth = 0
    while True:
        print ("LOOPING AT DEPTH %i " % (depth))
        result = DLS(root, goal, depth)
        print ("RESULT: %s, GOAL: %s" % (result, goal))
        if result == goal:
            return result
        depth = depth +1

def DLS(node, goal, depth):
    print ("NODE: %s, GOAL %s, DEPTH: %i" % (node, goal, depth))
    if depth == 0 and node == goal:
        print( "GOAL FOUND ,RETURN TO")
        return node
    elif depth > 0:
        print ("LOOPING THROUGH CHILD NODES: %s" % (graph.get(node, [])))


        for child in graph.get(node, []):
            if goal == DLS(child, goal, depth-1):
                return goal

IDDFS('a', 'g')
```

```
LOOPING AT DEPTH 0
NODE: a, GOAL g, DEPTH: 0
RESULT: None, GOAL: g
LOOPING AT DEPTH 1
NODE: a, GOAL g, DEPTH: 1
LOOPING THROUGH CHILD NODES: ['b', 'c', 'e']
NODE: b, GOAL g, DEPTH: 0
NODE: c, GOAL g, DEPTH: 0
NODE: e, GOAL g, DEPTH: 0
RESULT: None, GOAL: g
LOOPING AT DEPTH 2
NODE: a, GOAL g, DEPTH: 2
LOOPING THROUGH CHILD NODES: ['b', 'c', 'e']
NODE: b, GOAL g, DEPTH: 1
LOOPING THROUGH CHILD NODES: ['d', 'f']
NODE: d, GOAL g, DEPTH: 0
NODE: f, GOAL g, DEPTH: 0
NODE: c, GOAL g, DEPTH: 1
LOOPING THROUGH CHILD NODES: ['g', 'a']
NODE: g, GOAL g, DEPTH: 0
GOAL FOUND ,RETURN TO
RESULT: g, GOAL: g
```

Out[1]:

```
'g'
```

In [ ]:

In [ ]:

```

```