Depth-Limited-Search

- For Iterative Deepening, you need a depth-limited search
  That is depth-first-search that stops searching at a given depth:

```
function depth-limited-search(problem, maxDepth) returns a path
   frontier = [start]
   depthList[start] = 0
   cameFrom = {start ← null}
   Loop:
        if frontier is empty, then return FAIL
        node = remove-choice (frontier)
        if not depthList[node] > maxDepth
            if goaltest(node), then
                return reconstruct-path (cameFrom, start, end)
            for next in actions
                if not in visisted or in frontier
                     add next to frontier
                     add depthList[next] = depthList[node]+1
                     add next ← node to cameFrom
```

→ **As we are always stop searching at a given depth, we do not need to store visited nodes (unless you are concerned with efficiency)**