Ethan
Wong

SYOP 4

I found the concepts of Breadth-First-Search (BFS) and Depth-First-Search (DFS) to be particularly interesting and a question came to mind: what are real life scenarios that might utilize these strategies?

After thinking for a while, I realized they could be applied to more than just trees. For example, both BFS and DFS are viable methods to solve mazes. I came up with a Python algorithm that can utilize BFS to find the shortest path through a maze. The algorithm essentially checks the paths layer by layer and returns the shortest one, if such a path exists. My implementation uses a deque, which is similar to a list.

```
def BFS_path(maze):
    start, end = (1,1), (len(maze)-2, len(maze[0])-2)
    q = deque([("", start)])
    visit = set()
    g = maze_to_graph(maze)          # This function translates the
    while q:                           maze into a graphical representation
        curr, path = q.popleft()       with a dictionary
        if curr = end:
            return path
        if curr in visit:
            continue
        visit.add(curr)
        for d, n in g[curr]:
            q.append((path + d, n))
    return "No path found"
```

DFS for finding the shortest path in a maze follows a similar structure. The algorithm instead fully investigates each path before moving on. Once all paths have been investigated, a path is returned. This algorithm is generally worse than BFS though because it doesn't guarantee the best solution.

```
def DFS_path(maze):
    start, end = (1,1), (len(maze)-2, len(maze[0]-2)
    s = deque([["", start)])
    visit = set()
    g = maze_to_graph(maze)
    while s:
        curr, path = stack.pop()
        if curr = goal:
            return path
        if curr in visit:
            continue
        visit.add(curr)
        for d, n, in g[curr]:
            s.append((path + d, n))
    return "No path found"
```