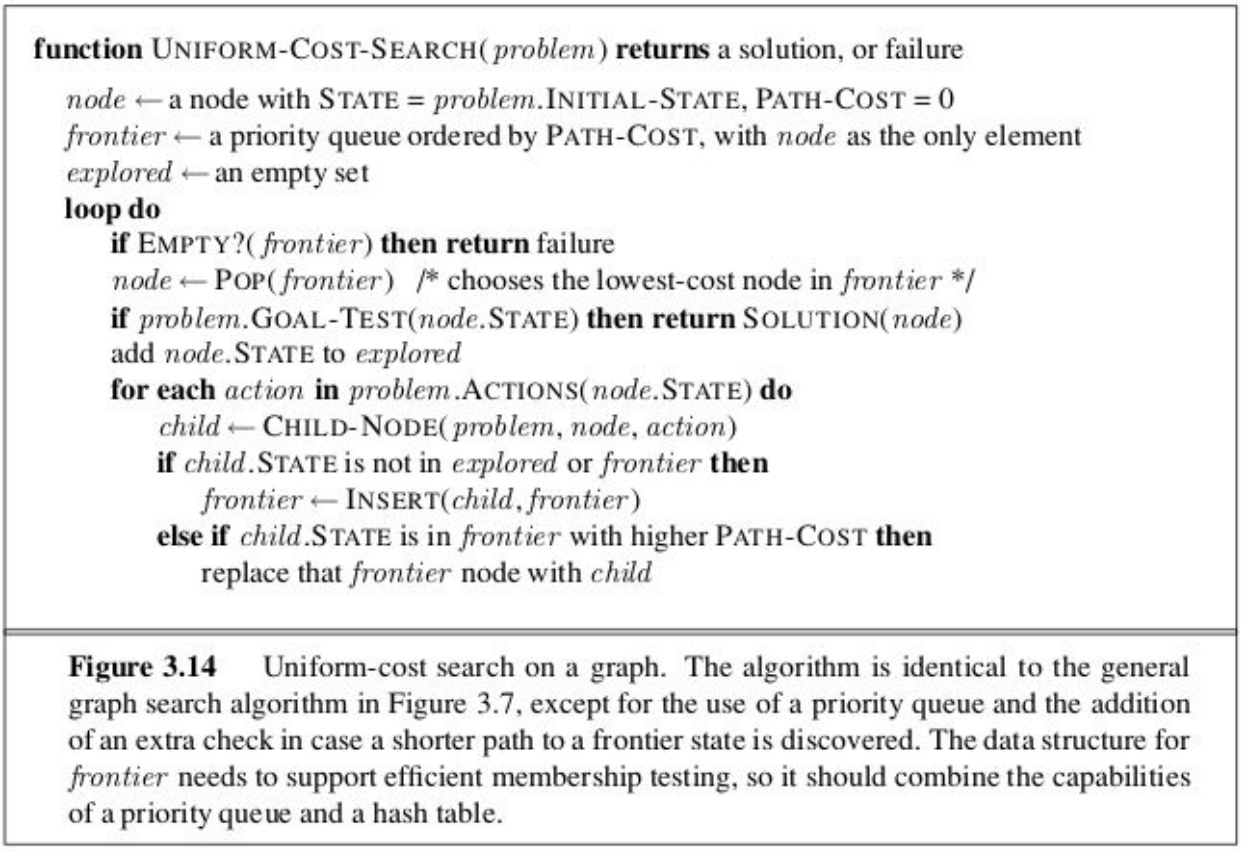
Project 1 notes:

The first project covers algorithms that are presented in the textbook. Starting on page 81 of the Russel and Norvig text you can find pseudocode for all the search algorithms that the project covers.



```
function UNIFORM-COST-SEARCH(problem) returns a solution, or failure
  node ← a node with STATE = problem.INITIAL-STATE, PATH-COST = 0
  frontier ← a priority queue ordered by PATH-COST, with node as the only element
  explored ← an empty set
  loop do
    if EMPTY?(frontier) then return failure
    node ← POP(frontier)  /* chooses the lowest-cost node in frontier */
    if problem.GOAL-TEST(node.STATE) then return SOLUTION(node)
    add node.STATE to explored
    for each action in problem.ACTIONS(node.STATE) do
      child ← CHILD-NODE(problem, node, action)
      if child.STATE is not in explored or frontier then
        frontier ← INSERT(child, frontier)
      else if child.STATE is in frontier with higher PATH-COST then
        replace that frontier node with child
```

Figure 3.14 Uniform-cost search on a graph. The algorithm is identical to the general graph search algorithm in Figure 3.7, except for the use of a priority queue and the addition of an extra check in case a shorter path to a frontier state is discovered. The data structure for *frontier* needs to support efficient membership testing, so it should combine the capabilities of a priority queue and a hash table.

Figure 3.14 Russel and Norvig pg 84

In the Uniform-Cost-Search algorithm above note that when checking to add a node to the frontier, it first checks to see if that node is not already in the explored set or the frontier queue. This will come into play when implementing your searching algorithms.

The autograder considers a node visited when problem.getSuccessors(node) is called.

The algorithm presented in the textbook for iterative deepening uses a recursive implementation. You can also implement this algorithm iteratively using a loop instead of recursive function calls. Think about which method you believe will be easier for you.