Module 1 - Pseudocode

Graph Based State Space Search

This pseudocode is the basis for all graph search algorithms (it's based on pseudocode in your book on p. 77). Depending on the data structure used for the frontier and whether or not the cost model is taken into account, this basic algorithm can implement both uninformed and informed search.

```
1      place initial state on frontier
2      initialize explored list
3      while frontier is not empty
4            current-state := next state on frontier
5            return path( current-state) if is-terminal( current-state)
6            children := successors( current-state)
7            for each child in children
8                  add child to frontier if not on explored or frontier
9            add current-state to explored
10     return nil
```

However, pseudocode always leaves out the details. Here are a some you'll need to piece together:

1. If the algorithm is implemented as a function, what should the arguments be? If the algorithm is implemented as a method in a class, what else will the class need to provide?

2. How should the data structure for the frontier be specified? What are the easy ways to implement the functionality of a stack, queue and priority queue in Python?

3. The path() function (line 5) doesn't show how the solution is retrieved from the current state. How should path() be implemented? What additional bookkeeping will be required? Should a State be a complex data structure?

4. is-terminal() only works if all terminal states are goal states. If there are terminal failure states we want to stop searching through that path but we do want to keep searching. What changes are required for this?

5. The successor() function is not described. How should it use T(s, a) to generate all possible successor states? How should T(s, a) be specified? How will successor() need to be different when using informed as compared to uniformed search?

6. The check at line 8 has not been specified. How does the check differ for uniformed and informed search?