Module Bank-1-SEC-A

1. A robot must deliver identical packages to locations A and B. Assume that it is holding both packages to start with. The environment is represented as a grid of squares, some of which are free, and the robot can move into them, but some of them are occupied by walls. The robot can move into neighbouring squares and can pick up and drop packages if they are in the same square as the robot.
   a. Formulate this problem as a search problem, specifying the state space, action space and goal test.
   b. Suggest admissible heuristic for this domain.
   c. Now consider the case where the robot does not start with the packages, but it has to pickup a package from location X to deliver to location A and a package from location Y to deliver to B. What is an appropriate state space for this problem.

2. Your goal is to navigate a robot out of a maze. The robot starts in the center of the maze facing north. You can turn the robot to face north, east, south, or west. You can direct the robot to move forward a certain distance, although it will stop before hitting a wall.

   a. Formulate this problem. How large is the state space?
   b. In navigating a maze, the only place we need to turn is at the intersection of two or more corridors. Reformulate this problem using this observation. How large is the state space now?
   c. From each point in the maze, we can move in any of the four directions until we reach a turning point, and this is the only action we need to do. Reformulate the problem using these actions. Do we need to keep track of the robot's orientation now?
   d. In our initial description of the problem we already abstracted from the real world, restricting actions, and removing details. List three such simplifications we made.

3. There is a farmer who wishes to cross a river but he is not alone. He also has a goat, a wolf, and a cabbage along with him. There is only one boat available which can support the farmer and either of the goat, wolf, or the cabbage. So, at a time, the boat can have only two objects (farmer and one other). But the problem is, if the goat and wolf are left alone (either in the boat or onshore), the wolf will eat the goat. Similarly, if the Goat and cabbage are left alone, then goat will eat the cabbage. The farmer wants to cross the river with all three of his belongings: goat, wolf, and cabbage. What strategy he should use to do so?

   a. Formulate the problem precisely, making only those distinctions necessary to ensure a valid solution. Draw a diagram of the complete state space.
   b. Implement and solve the problem optimally using an appropriate search algorithm. Is it a good idea to check for repeated states?
   c. Why do you think people have a hard time solving this puzzle, given that the state space is so simple?

4.Consider a grid-based maze where the goal is to find the shortest path from the start cell (S) to the goal cell (G) using the A* algorithm. The grid contains walls (#) that cannot be traversed. Each movement from one cell to an adjacent cell (up, down, left, right) has a cost of 1.

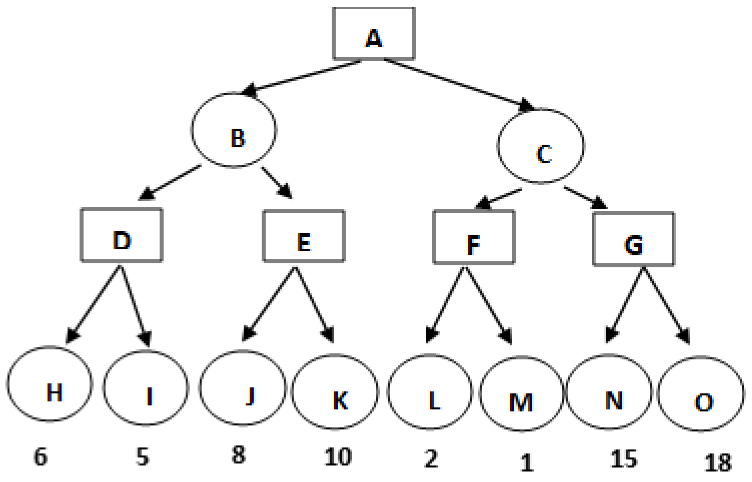
Grid Representation:

```
S . . . . . . . . . .
. # # . . . . . . . .
. . . # . . . . . . .
. # . . . . . . . . .
. . . . # . . . . . .
. . # . . . # . . . .
. . # # # . # . . . .
. . . . . . . . . . .
. . . . # # . . . . .
. . . . . . . # . . .
. . . . . . . . . G .
```

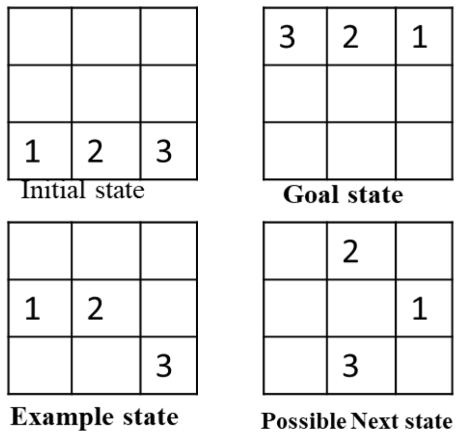
In the grid above, the start cell is represented by 'S', the goal cell is represented by 'G', the walls are represented by '#', and the empty cells are represented by '.'.

Your task is to implement the A* algorithm to find the shortest path from the start cell to the goal cell, considering the walls as obstacles.

5.Given the following search tree, apply the alpha-beta pruning algorithm to it and show the search tree that would be built by this algorithm. Make sure that you show where the alpha and beta cuts are applied, and which parts of the search tree are pruned as a result.



6.Consider n x n chess board, n pawns numbered 1,2, . . . n are initially placed in the bottom row such that pawn I is at position (I,1). The goal is to move the pawns to the top row but in reverse order, so that pawn I ends up in position (n-i+1,n). On each time step, each of the n pawns can move one square left, right, up, down or stay put. But if a pawn stays put, an adjacent pawn may hop over it. Two pawns cannot occupy the same square. An example with n=3 is illustrated below.



i. What is roughly size of the state space.
ii. What is approximately the branching factor.
iii. Suppose that pawn I is at (xi,yi). Write a non-trivial admissible heuristic hi, for the number of moves it will require to get its goal location(n-i+1), assuming there are no other pawns on the board.

7. You have been tasked with creating a cleaning agent that can navigate and clean a set of rooms in a house. The agent will be equipped with a vacuum cleaner and must navigate through the house to clean all the rooms. To simplify the problem, assume that the house is a grid of rooms, each identified by its (x, y) coordinates. The agent can move in four directions - up, down, left, and right - and can clean a room by moving to it and performing a cleaning action. Your task is to implement the cleaning agent in Python using an appropriate algorithm, such as depth-first search or A* search.

Here are the details of the scenario:

The house is a 10x10 grid of rooms, numbered (0,0) to (9,9).
Each room can be in one of two states - dirty or clean.
The agent starts in room (0,0), which is always clean.
The agent must clean all the dirty rooms in the house.
The agent can move one step at a time in any of the four directions - up, down, left, or right.
The agent can clean a room by moving to it and performing a cleaning action.
The agent can only move through doorways between adjacent rooms.
The agent has a limited battery life and can only clean a certain number of rooms before needing to recharge.

To complete this task, you will need to implement the following functions:

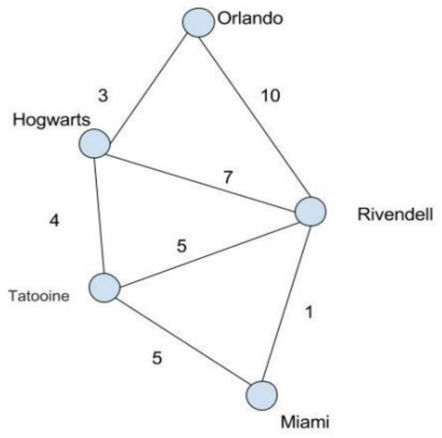
- get_dirty_rooms(): This function should return a list of the coordinates of all the dirty rooms in the house.
- get_neighbors(room): This function should return a list of the coordinates of all the neighboring rooms of the given room.
- clean_room(room): This function should clean the given room and return True if the cleaning action is successful.

- move_agent(from_room, to_room): This function should move the agent from the from_room to the to_room and return True if the movement is successful.

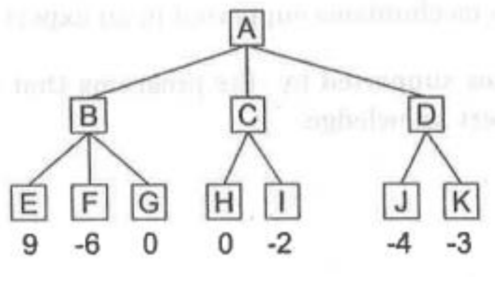
  You can use any appropriate algorithm to implement the cleaning agent, such as depth-first search, breadth-first search, or A* search.

  Once you have implemented the cleaning agent in Python, you can test it on different houses and evaluate its performance. Draw State space diagram for each move.

8.Consider the map below. Let us assume that you want to go from Miami to Orlando. Trace the path planning process using uniform cost graph search. At each step, show the currently expanded node, the fringe and the closed set. The format should be: Step 79: Expanded: Dagobah:10, Fringe={Coruscant:34, Hoth:45}, Closed={Miami:0, …} Where in the Planet:value, the value shows the current cost.



9.Consider a two-player game in which the minimax search procedure is used to compute the best moves for the first player. Assume a static evaluation function that returns values ranging from -10 to 10, with 10 indicating a win for the first player and -10 a win for the second player. Assume the following game tree in which the static scores are from the first player's point of view. Suppose the first player is the maximizing player and needs to make the next move. What move should be chosen at this point? Can the search be optimized?



10. The Alpha-Beta pruning algorithm in a specific game scenario and compare its performance with the Min-Max algorithm.

a. Explain the evaluation function used to assign scores to game states and how it influences the search process.
b. Explore possible extensions or variations of Alpha-Beta pruning that could further improve its efficiency or accuracy.
c. Consider a game of Tic-Tac-Toe where the AI algorithm uses Alpha-Beta pruning.

The game tree is represented by a search tree with each node representing a possible game state. The AI's objective is to find the optimal move by searching the game tree. You can illustrate the application of Alpha-Beta pruning by showing the pruning process at each level of the search tree. Demonstrate how the algorithm eliminates certain branches of the tree based on the calculated alpha and beta values, effectively reducing the number of nodes explored.By comparing the number of nodes explored with and without Alpha-Beta pruning, you can showcase the efficiency improvement achieved by the algorithm.