TC2011 - Uninformed Search Lab

- Which heuristics did you use for the A* algorithm?

```
// calculates the heuristic value of the current node
int Node::calculateHeuristic(Node *goal_node) {
  int heuristic = 0;
  // first heuristic: iterate through all the state and if the curr_state and the goal_node->state are the same continue, otherwise increase the value of the heuristic. if the goal_node->state is an X, ignore that platform and look for the next ones.
  for (int i = 0; i < state[0].size(); i++) {
    for (int j = 0; j < state.size(); j++) {
      // if goal_node has X , skip
      if (goal_node->state[j][i] == 'X') break;
      if (state[j][i] == goal_node->state[j][i]) continue;
      heuristic++;
    }
  }
  // second heuristic: if state and goal_node have the same height continue, otherwise add 1 to the heuristic.
  for (int i = 0; i < heights.size(); i++) {
    if (heights[i] == goal_node->heights[i]) continue;
    heuristic++;
  }
  return heuristic;
}
```

- Test your program with a couple of different problems. Increase the size of the problem to test the limits of your program. Make a table comparing **how many nodes are searched** to find the answer for each problem. For this table, you should compare a number of different problems (at least 3) to avoid a statistical bias. Which of the three algorithms (UCS, A *with consistent and and A* with an inconsistent heuristic) searches the least nodes and which one take the most?

Input 1:
10
(A, D, H, N, P); (B, L, T); (C, R, C)
(A, C); X; X
*****UCS******
nodes searched: 3781
11
(0, 1); (0, 1); (0, 1); (0, 1); (2, 0)
time: 285.903
******Astar*******

nodes searched: 164
11
(0, 1); (0, 1); (0, 1); (0, 1); (2, 0)
time: 12.692
******Astar_inconsistent**********
nodes searched: 5959
14
(0, 1); (0, 2); (2, 1); (0, 1); (0, 1); (2, 0)
time: 48.957

Input 2:
5
(A, B); (); (E, F, G)
(B, E); X; X
*****UCS******
nodes searched: 13837
15
(0, 2); (0, 1); (2, 0); (2, 1); (2, 1); (2, 0)
time: 594.068
******Astar*******
nodes searched: 2517
15
(2, 1); (2, 1); (2, 1); (0, 1); (0, 2); (1, 0); (1, 0)
time: 124.951
******Astar_inconsistent**********
nodes searched: 7914
18
(2, 1); (2, 1); (2, 0); (0, 1); (0, 1); (0, 2); (1, 0); (1, 0)
time: 308.096

Input 3:
4
(A, B, C); (K, L); (); (T)
X; X; (C, B, A); (L, T)
*****UCS******
nodes searched: 454553
16
(0, 2); (0, 2); (3, 2); (1, 3); (2, 3); (0, 2)
time: 34419.8
******Astar*******
nodes serched: 2430
16
(0, 2); (3, 2); (1, 3); (2, 3); (0, 2); (0, 2)
time: 215.554

******Astar_inconsistent**********
nodes searched: 113577
17
(0, 2); (0, 1); (1, 2); (0, 2); (3, 2); (1, 3); (2, 3)
time: 7661.55

We can clearly see that the A* is the best for finding an answer to this problem, the A*_inconsistent sometimes gave the wrong answer.

- Why does this happen?

.....

- Which algorithms are optimal? Why?

.....

- In your opinion, what are the benefits of simpler algorithms versus more complex ones?

.....