Team C – Chue Zhang, Eric Mai

**Program 1 – Pacman**

For program 1, the goal is to guide Pacman through a maze to both reach a particular location and to collect food efficiently. We implemented the 4 search algorithms: Depth First Search, Breadth First Search, Uniform Cost Search, and A\* Search to help achieve this goal.

In Depth First Search, we do not make use of any heuristics and we search all the way down one path until we are at a dead end before backtracking and looking at the next path. We make use of a stack data structure, which is first in last out, to keep track of nodes that still need to be explored. Additionally, we make use of a list called visited to keep track of nodes that are already visited. The nodes that are in the visited list no longer need to be considered.

In Breadth First Search, we do not make use of any heuristics and we search the shallowest nodes in the search tree first before increasing the distance. We make use of a queue data structure, which is first in first out, to keep track of nodes that still need to be explored. Additionally, we make use of a list called visited to keep track of nodes that are already visited. The nodes that are in the visited list no longer need to be considered.

In Uniform Cost Search or Branch and Bound Search we need to keep track of the costs between nodes. We make use of a priority queue data structure, which acts like the queue data structure but there is a priority associated with each element in the queue. An element with a high priority is served before an element with a low priority. To calculate the priority of a node, we need to consider the combined cost of the getting to that node from a node and the previous cost of getting to the node. The nodes with higher priority get expanded first. Additionally, we make use of a list called visited to keep track of nodes that are already visited. The nodes that are in the visited list no longer need to be considered.

In A\* Search also need to consider the costs between nodes as well as a heuristic measurement. Like Uniform Cost Search, we make use of a priority queue data structure. To calculate the cost of getting from node A to node B, we need to consider the combined cost of the getting to node B from node A as well as the previous cost of getting to node A. To calculate the heuristic cost, we need to consider the combined cost as well as the heuristic function. The priority of a node is determined based on the heuristic cost. Heuristic formula which is also known as the Manhattan heuristic goes as the following

For question 5, we had to make a search agent to looks for all 4 corners using BFS algorithm. This essentially is just changing up the goal of the BFS so that instead of search for one destination, it will have to search for 4. Whenever we hit a corner, we can just add it into a list then we check for goal state which is a list of 4 corners.