**Project 1: CSE 537 - Artificial Intelligence**

**Question 1: DFS Implementation**

**Methodology Used: <explain here>**

Execution Details Set1: python pacman.py -l tinyMaze -p SearchAgent

**Function:** DepthFirstSearch

**Problem Type:** PositionSearchProblem

**Total Cost for the Path Found:** 10

**Search Nodes Expanded:** 15

Execution Details Set2: python pacman.py -l mediumMaze -p SearchAgent

**Function:** DepthFirstSearch

**Problem Type:** PositionSearchProblem

**Total Cost for the Path Found:** 130

**Search Nodes Expanded:** 146

**Execution Details Set3:** python pacman.py -l bigMaze -z .5 -p SearchAgent

**Function:** DepthFirstSearch

**Problem Type:** PositionSearchProblem

**Total Cost for the Path Found:** 210

**Search Nodes Expanded:** 390

**Question 2: BFS Implementation**

**Methodology Used:** <explain here>

**Execution Details Set1:** python pacman.py -l mediumMaze -p SearchAgent -a fn=bfs

**Function:** BreadthFirstSearch

**Problem Type:** PositionSearchProblem

**Total Cost for the Path Found:** 68

**Search Nodes Expanded:** 269

**Execution Details Set2:** python pacman.py -l bigMaze -p SearchAgent -a fn=bfs -z .5

**Function:** BreadthFirstSearch

**Problem Type:** PositionSearchProblem

**Total Cost for the Path Found:** 210

**Search Nodes Expanded:** 620

Note: python eightpuzzle.py works correctly with the our generic algorithmic implementation.

**Varying Cost Function related problems:**

MediumDottedMaze & MediumScaryMazes are used – here the cost function is varied taking into consideration dangerous steps & food rich areas.

**Question 3: Uniform Cost Search:**

Methodology Used: < explain here>

**Execution Set 1:** python pacman.py -l mediumMaze -p SearchAgent -a fn=ucs

**Function:** UniformCostSearch (ucs)

**Problem Type:** PositionSearchProblem

**Total Cost for the Path Found:** 68

**Search Nodes Expanded:** 269

**Execution Set 2:** python pacman.py mediumDottedMaze -p StayEastSearchAgent

**Total Cost for the Path Found:** 1

**Search Nodes Expanded:** 186

<explaination about the results???>

**Execution Set 3:** python pacman.py -l mediumScaryMaze -p StayWestSearchAgent

**Total Cost for the Path Found:** 17183894840

**Search Nodes Expanded:** 169

<explaination about the results???>

**A\* Search:**

**Question 4: A\* Search**

Methodology Used: <explain here>

Execution: python pacman.py -l bigMaze -z .5 -p SearchAgent -a fn=astar,heuristic=manhattanHeuristic

**Function:** astar and manhattanHeuristic

**Problem Type:** PositionSearchProblem

**Total Cost for the Path Found:** 210

**Search Nodes Expanded:** 549

(vs) – Comparison with ucs – astar gives a better optimal solution:

Execution: python pacman.py -l bigMaze -z .5 -p SearchAgent -a fn=ucs

**Function:** UCS

**Problem Type:** PositionSearchProblem

**Total Cost for the Path Found:** 210

**Search Nodes Expanded:** 620

Here search nodes expanded is 620 compared to 549 in astar.

**Finding All Corners:**

**Finding whether all the four corners have been reached.**

**Question 5:**

**Methodology Used: <explain here>**

Execution Set: python pacman.py -l tinyCorners -p SearchAgent -a fn=bfs,prob=CornersProblem

**Function:** BFS

**Problem Type:** CornersProblem

**Total Cost for the Path Found:** 29

**Search Nodes Expanded:** 277

Execution Set: python pacman.py -l mediumCorners -p SearchAgent -a fn=bfs,prob=CornersProblem

**Function:** BFS

**Problem Type:** CornersProblem

**Total Cost for the Path Found:** 107

**Search Nodes Expanded:** 2006

<results explanation>

**Question 6: Heuristic for Corners Problem in cornersheuristic**

<Heuristic explanation>

Execution Details:

python pacman.py -l mediumCorners -p SearchAgent –a fn=aStar,prob=CornersProblem,heuristic=cornersHeuristic

**Function:** astart and cornersHeuristic

**Problem Type:** CornersProblem

**Total Cost for the Path Found:** 107

**Search Nodes Expanded:** 709

Question 7: Eating all the dots:

Checking of the code with testSearch for FoodSearchProblem.

Execution Set:

python pacman.py -l testSearch -p AStarFoodSearchAgen

(or)

python pacman.py -l testSearch -p SearchAgent -a fn=astar,prob=FoodSearchProblem,heuristic=foodHeuristic

**Function:** astart and FoodHeuristic

**Problem Type:** FoodSearchProblem

**Total Cost for the Path Found:** 7

**Search Nodes Expanded:** 12

**UCS algorithm with tinySearch Layout – very Slow:**

**Execution Set:**

python pacman.py -l testSearch -p SearchAgent -a fn=ucs,prob=FoodSearchProblem

**Function:** UCS

**Problem Type:** FoodSearchProblem

**Total Cost for the Path Found: 2**7

**Search Nodes Expanded:** 5057

**Question 7:**

**Methodology used to solve the problem:**

The foods are visualized to be on the circumference of the convex hull. The convex hull is constructed taking into consideration several foods and such that no food falls outside convex hull but there might be food inside convex hull. Then nearest food ( in terms of the location) present on the convex hull from the pacman is considered – that food is traversed and so on.

Execution Set: Our Heuristic expands within 7000 nodes!!....

python pacman.py -l trickySearch -p SearchAgent -a fn=astar,prob=FoodSearchProblem,heuristic=foodHeuristic

**Function:** astart and foodHeuristic

**Problem Type:** FoodSearchProblem

**Total Cost for the Path Found:** 60

**Search Nodes Expanded:** 6945

We just expand 6945 nodes to find optimal path buth whereas with null heuristic/bfs the nodes expanded are 16688.

By,

**Team Members:**