1. 8- Puzzle

**Code:**

if \_\_name\_\_ == '\_\_main\_\_':

#creating instance of puzzle class

problem= EightPuzzle.Eight\_Puzzle\_Problem([3,0,7,2,8,1,6,4,5])

#creating list of serch algorithms from "AI\_search.py" file

list\_search= [AI\_search.depthFirstSearch, AI\_search.breadthFirstSearch, AI\_search.uniformCostSearch, AI\_search.astarSearch, AI\_search.greedySearch, AI\_search.iterativeDeepeningDFS]

#Applying solve() for results

solve(problem, list\_search)

**Output:**

A screenshot of a survey

Description automatically generated

A close up of a logo

Description automatically generated

1. PACMAN

**Code:**

if \_\_name\_\_ == '\_\_main\_\_':

#creating instance of puzzle class

problem= PACMAN.Pacman\_Problem(["%%%%%%","% %", "% %% %", "% %", "%%%% %","% %", "%%%%%%"], (1,1), (5,1))

#creating list of serch algorithms from "AI\_search.py" file

list\_search= [AI\_search.depthFirstSearch, AI\_search.breadthFirstSearch, AI\_search.uniformCostSearch, AI\_search.astarSearch, AI\_search.greedySearch, AI\_search.iterativeDeepeningDFS]

#Applying solve() for results

solve(problem, list\_search)

**Output:**

A picture containing bird

Description automatically generated

1. Stone Puzzle
2. **For: [O O O W W W]**

**Code:**

if \_\_name\_\_ == '\_\_main\_\_':

#creating instance of puzzle class

problem= StonePuzzle.Stone\_Puzzle(['O','O','O',' ','X','X','X'])

#creating list of serch algorithms from "AI\_search.py" file

list\_search= [AI\_search.depthFirstSearch, AI\_search.breadthFirstSearch, AI\_search.uniformCostSearch, AI\_search.astarSearch, AI\_search.greedySearch, AI\_search.iterativeDeepeningDFS]

#Applying solve() for results

solve(problem, list\_search)

**Output:**

A picture containing large, bird, flying

Description automatically generated

1. **For: [O O W W]**

**Code:**

if \_\_name\_\_ == '\_\_main\_\_':

#creating instance of puzzle class

problem= StonePuzzle.Stone\_Puzzle(['O','O',' ','X','X'])

#creating list of serch algorithms from "AI\_search.py" file

list\_search= [AI\_search.depthFirstSearch, AI\_search.breadthFirstSearch, AI\_search.uniformCostSearch, AI\_search.astarSearch, AI\_search.greedySearch, AI\_search.iterativeDeepeningDFS]

#Applying solve() for results

solve(problem, list\_search)

**Output:**

A screenshot of a cell phone

Description automatically generated

1. Farmer, Sheep, Wolf, Lettuce Puzzle

**Code:**

if \_\_name\_\_ == '\_\_main\_\_':

#creating instance of puzzle class

problem= FarmerPuzzle.Farmer\_Sheep\_Wolf\_Lettuce\_Puzzle([1,1,1,1])

#creating list of serch algorithms from "AI\_search.py" file

list\_search= [AI\_search.depthFirstSearch, AI\_search.breadthFirstSearch, AI\_search.uniformCostSearch, AI\_search.astarSearch, AI\_search.greedySearch, AI\_search.iterativeDeepeningDFS]

#Applying solve() for results

solve(problem, list\_search)

**Output:**

A screenshot of a social media post

Description automatically generated

1. Man- Woman- Children Puzzle

**Man, wife and 2 kids:**  
A man and a woman of equal weight, together with their two children, each of half their weight, wish to cross a river using a boat which can only carry the weight of one adult. How can this be done?

**Code:**

if \_\_name\_\_ == '\_\_main\_\_':

#creating instance of puzzle class

problem= ManWoman.Man\_Woman\_Children([1,1,1,1,1])

#creating list of serch algorithms from "AI\_search.py" file

list\_search= [AI\_search.depthFirstSearch, AI\_search.breadthFirstSearch, AI\_search.uniformCostSearch, AI\_search.astarSearch, AI\_search.greedySearch, AI\_search.iterativeDeepeningDFS]

#Applying solve() for results

solve(problem, list\_search)

**Output:**

A screenshot of a social media post

Description automatically generated

1. Travelling Salesman Puzzle

Case 1:



B

A



D

C

**Code:**

if \_\_name\_\_ == '\_\_main\_\_':

#creating instance of puzzle class

problem= Salesman.Travelling\_Salesman(['A'],[[0,20,42,35],[20,0,30,34],[42,30,0,12],[35,34,12,0]])

#creating list of serch algorithms from "AI\_search.py" file

list\_search= [AI\_search.depthFirstSearch, AI\_search.breadthFirstSearch, AI\_search.uniformCostSearch, AI\_search.astarSearch, AI\_search.greedySearch, AI\_search.iterativeDeepeningDFS]

#Applying solve() for results

solve(problem, list\_search)

**Output:**

A screenshot of a cell phone

Description automatically generated

Case 2:



B

A



D

C

**Code:**

if \_\_name\_\_ == '\_\_main\_\_':

#creating instance of puzzle class

problem= Salesman.Travelling\_Salesman(['A'],[[0,80,42,35],[80,0,30,34],[42,30,0,12],[35,34,12,0]])

#creating list of serch algorithms from "AI\_search.py" file

list\_search= [AI\_search.depthFirstSearch, AI\_search.breadthFirstSearch, AI\_search.uniformCostSearch, AI\_search.astarSearch, AI\_search.greedySearch, AI\_search.iterativeDeepeningDFS]

#Applying solve() for results

solve(problem, list\_search)

**Output:**

A screenshot of a social media post

Description automatically generated