JOSEPH THANKACHAN

PERSONAL NO : 961104-4877

AIM

The aim of the lab is Implement different types of searching algorithms and also implement searching algorithms in poker game

TASKS

1. **Path searching**

   A* searching algorithm

   Greedy search

   Breadth first search

   Depth first search

   Random search

2. **Poker game**

   Breadth first search

   Greedy search

Searching

In this task we find the shortest distance from starting point to end point with the help of different type of Searching.

Start point

It is the point at which the searching is started

End point

It is Define as the goal. The searching end in this point

Obstacles

It is the blocking nods in the map so we should avoid this node to find path

Finding neighbours

We take the neighbours by using x and y coordinates. Each node have FOUR NEIGHBOURS (right, left, top, bottom) similarly at the same time it check the neighbours is GOAL and neighbours INDEXVALUE is equal to 0. Then pass neighbours to another list for finding the path. if the node index is equals to 1 then the node is not considered because the node already visited.

Note: cosied the x and y coordinates value should not exceed the maximum and minimum value of map [use if condition and check example: $x < length(map), x > 0, y < length(map), y > 0$]

bottom = [x+1, y]

Top = [x-1, y]

Right = [x, y+1]

Left = [x, y-1]

Depth

It is used to calculate the cost. Depth value is Increment by 1 in each neighbour finding step

Depth = depth + 1

Visited nodes

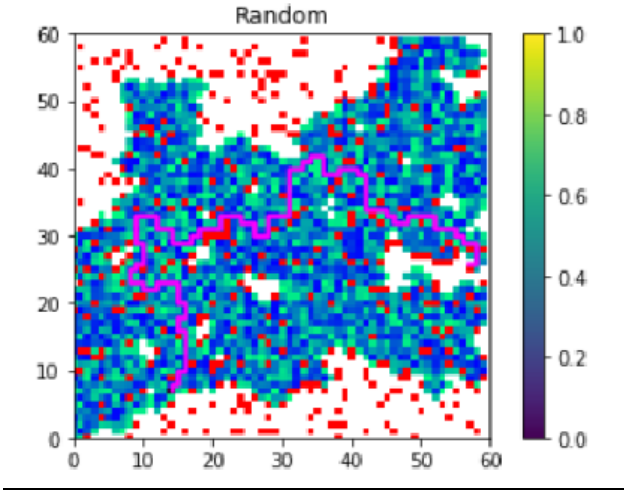
After each neighbour finding step the nodes are added to the visited node list and finally it used to find the path

Random search

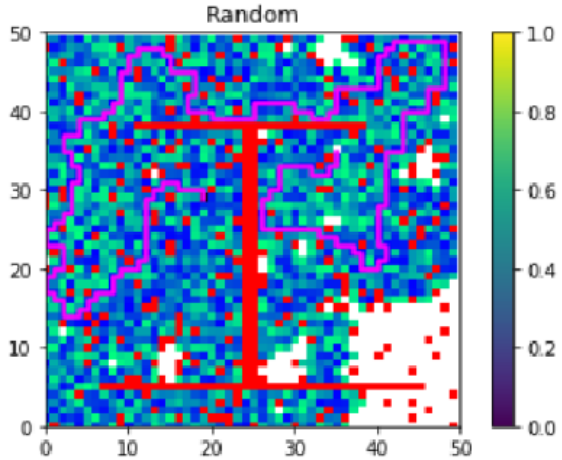
In this searching method we just search randomly and find the goal. The searching is start at the starting point and take all neighbours and choose the nest node randomly and take all nearest point then repeat the searching until get the goal.

Output:

Task 1a



Task 1b

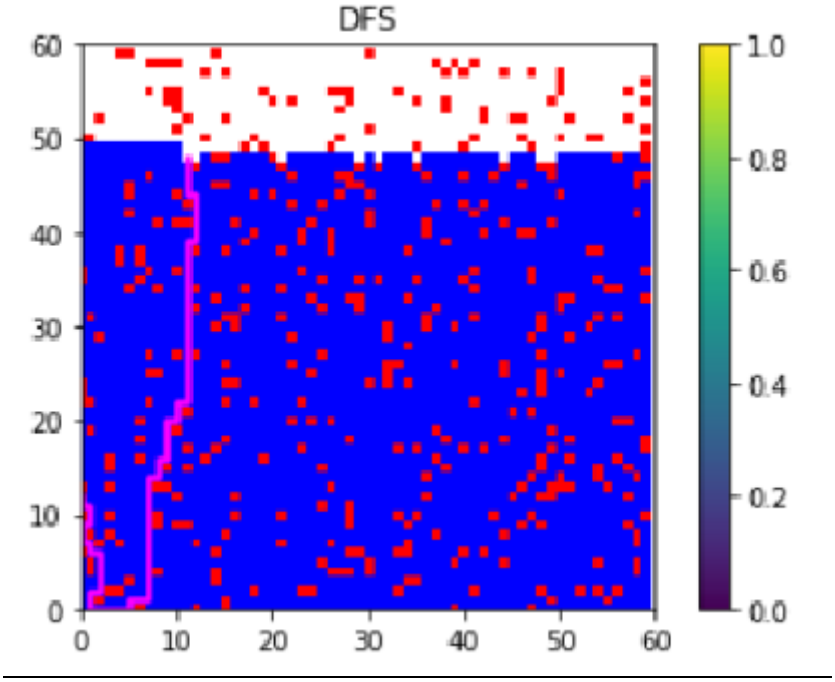


Depth first search:

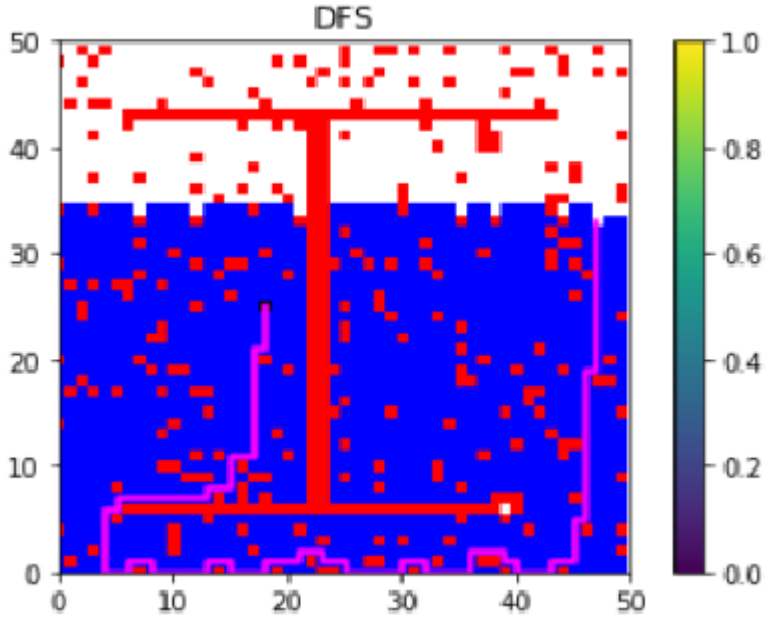
In DFS searching is done each neighbours and take the cost is equal to 1

Output:

Task 1a



Task 1b

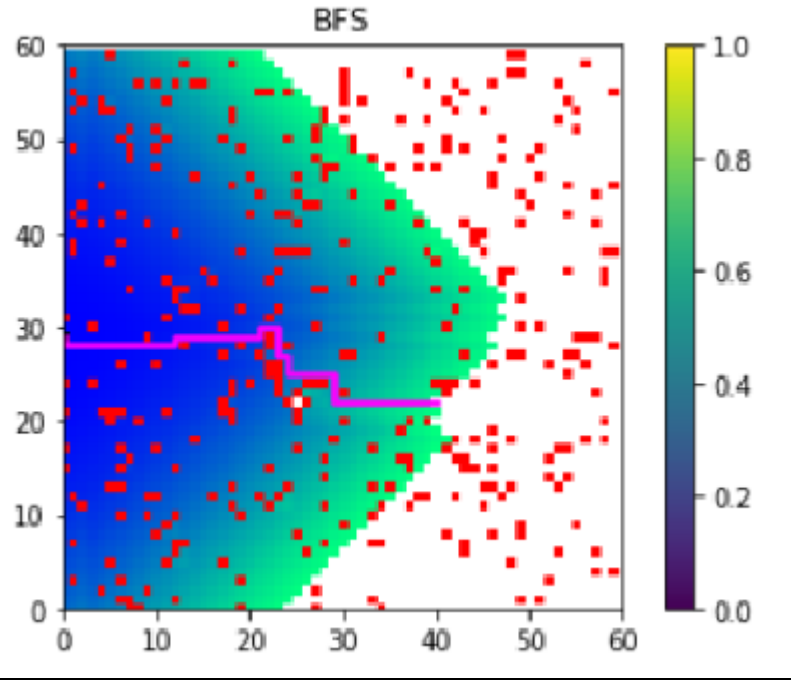


Breadth first search

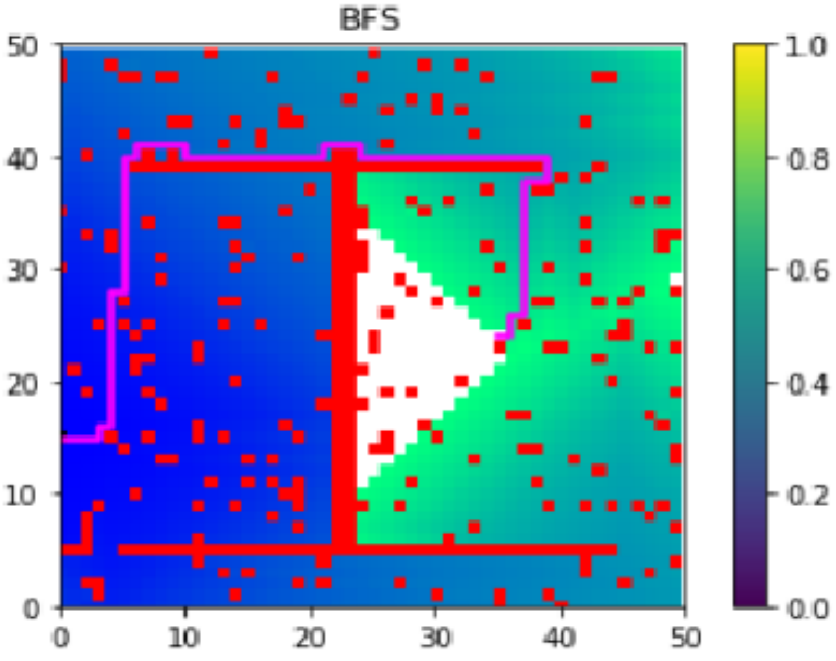
In BFS the searching is done by layer (depth) wise and add the depth value (depth =depth + 1) after the layers

Output:

Task 1a



Task 1b



Greedy search

The greedy search we used two methods to find the distance

1. Manhattan distance

2. Euclidean distance

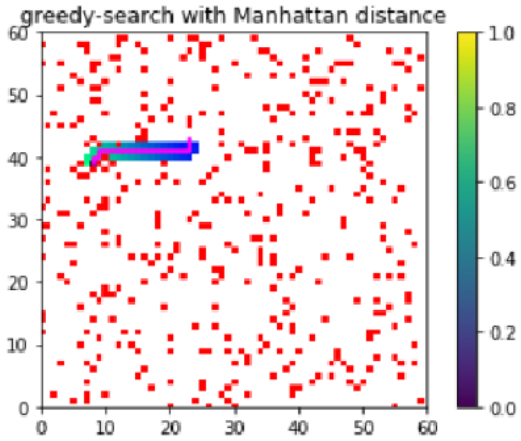
Cost = Euclidean distance/ Manhattan distance

The cost calculated by the distance and it added to priority queue then the list is sorted in the base of lowest cost. take the first value and find the neighbours.

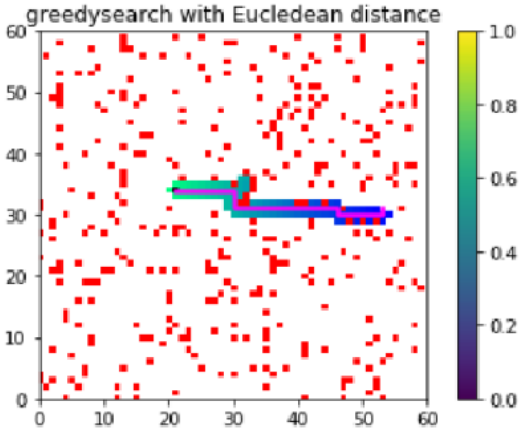
Output:

Task 1a

Greedy search with manhattan distance

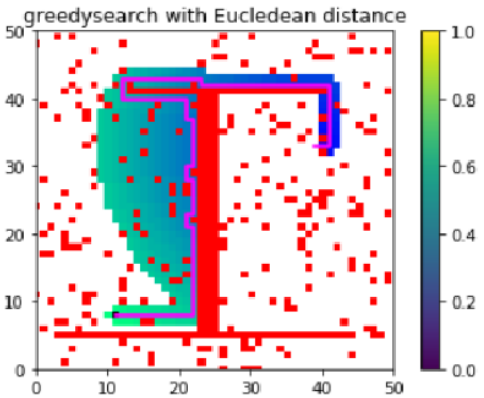


Greedy search with eucledean distance

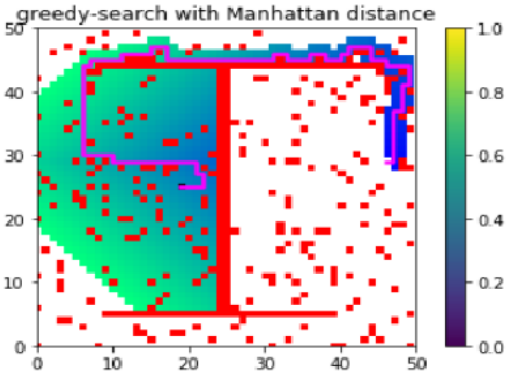


Task 1b

Greedy search with manhattan distance



Greedy search with eucledean distance



A*search

The A* search we used two methods to find the distance

1. Manhattan distance

2. Euclidean distance

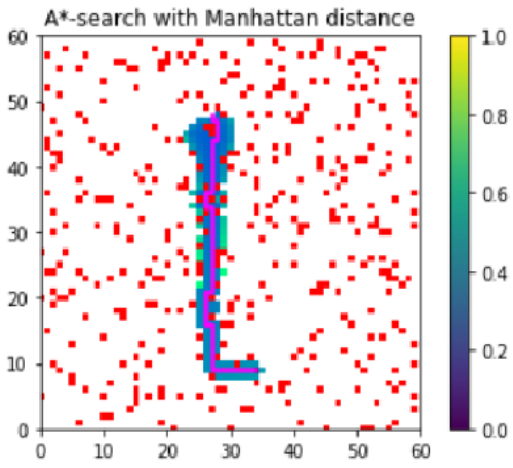
Cost = Manhattan distance/ Euclidean distance + depth

The cost calculated by take the sum of distance and depth. it added to priority queue then the list is sorted in the base of lowest cost. take the first value and find the neighbours.

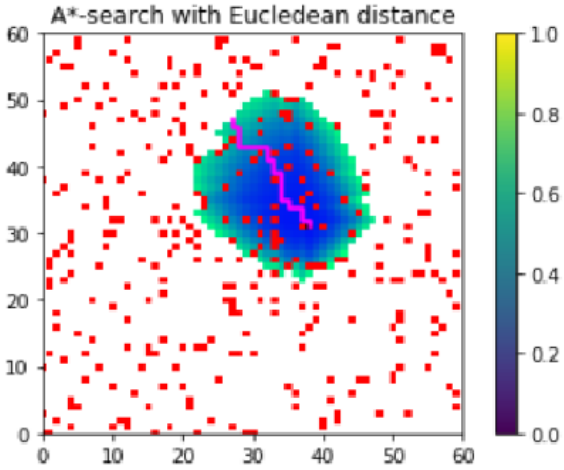
Output:

Task 1a

A* with manhattan distance

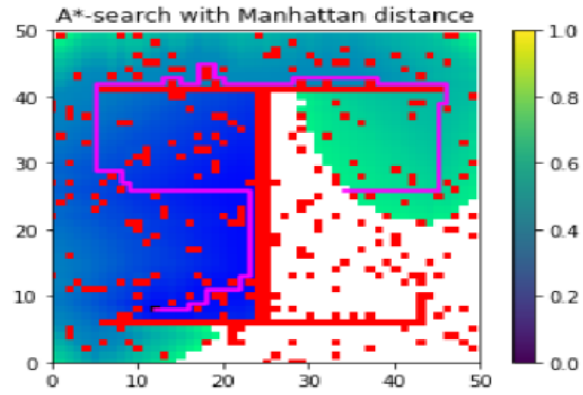


A* search with eucledean distance

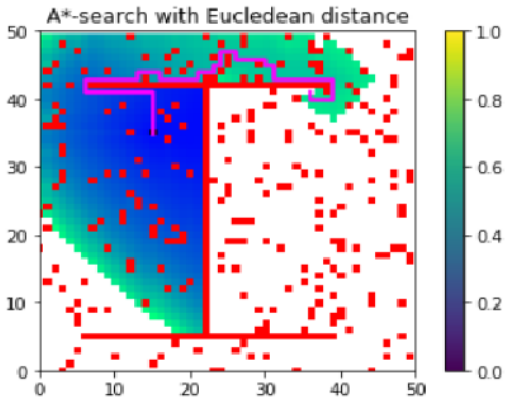


Task 1b

A* with manhattan distance



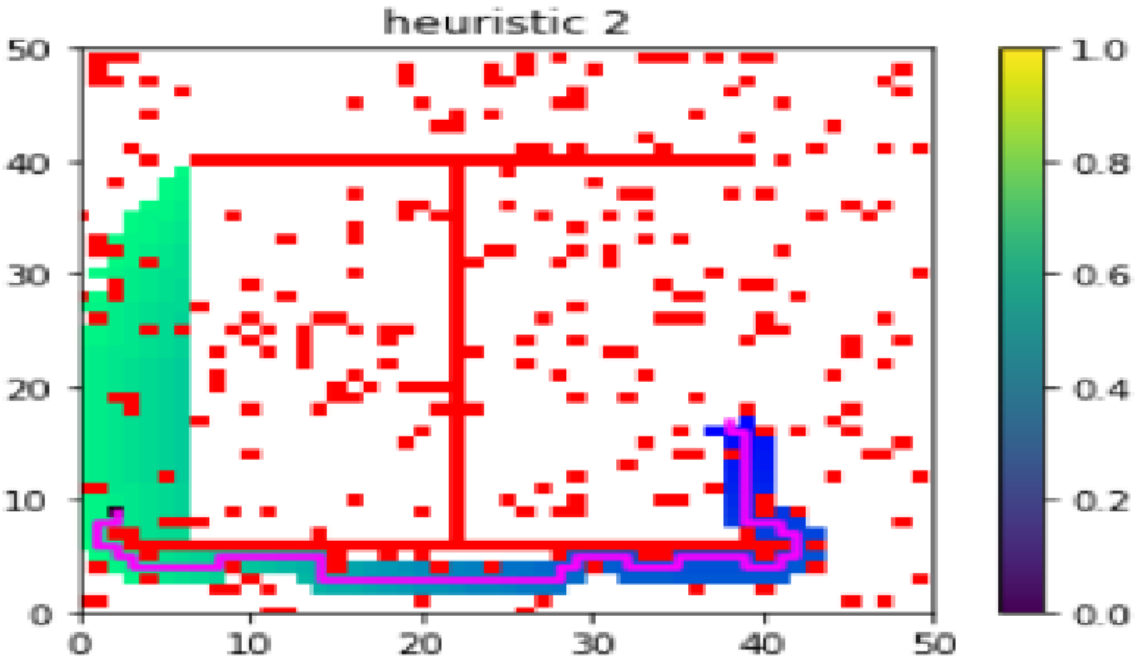
A* search with eucledean distance



A* With heuristic

In this program add the heuristic in the A* searching to avoid the searching inside the obstacle. When we get the neighbours inside the obstacle then stop and search another node. It can easy to understand from the output graph.

Output:



Task 2

conditions

hand have only 4 cards

our get 400 points

depth first search

search the condition by take the different states in the given program. after the each search the depth will added one (depth = ne.nn_current_hand+1) when we get the true condition then stop search and return the value. Get the out put in the hand 4 and total states 101

greedy search

in the greedy search the given conditions are search from list get the output in the second hand and 91 total states