মেট্রোপলিটন ইউনিভার্সিটি
Metropolitan University

ASSIGNMENT

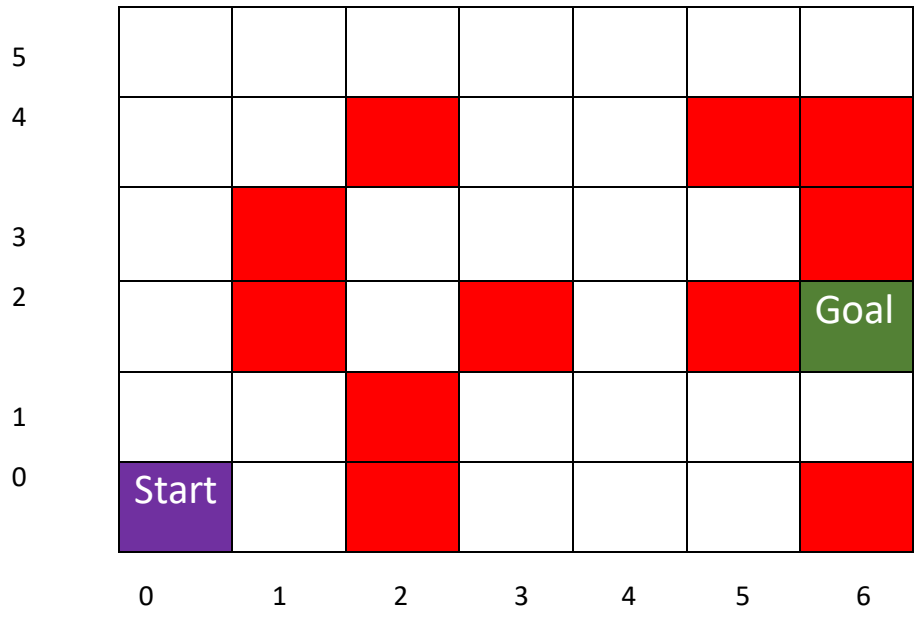
Course Code	CSE 421
Course Title	Artificial Intelligence

Date of Assignment	30 november 2020
Date of Submission	4 december 2020

Submitted to
Ranith Debnath Akash Lecturer Department of CSE Metropolitan University Sylhet-3100

Submitted by
Azizur Rahman ID: **181-115-086** Batch: 44th, Section: C Department of CSE Metropolitan University Sylhet-3100

a. Run Breadth First Search (BFS) from the start node and stop when you reach the goal. Show each step/phase of the algorithm.

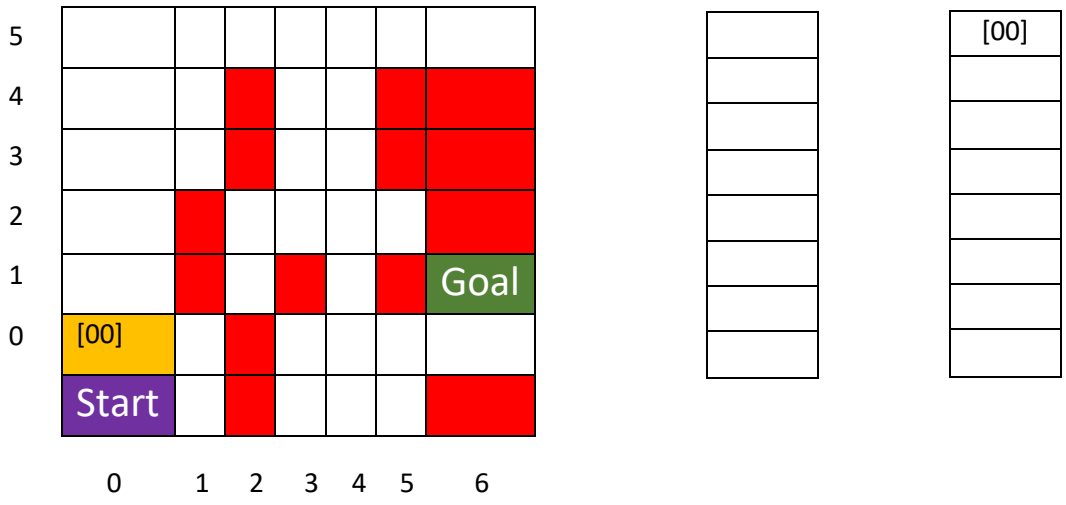


First of all, the source node is [00]

We need to push [00] in queue

The queue contains only [00]

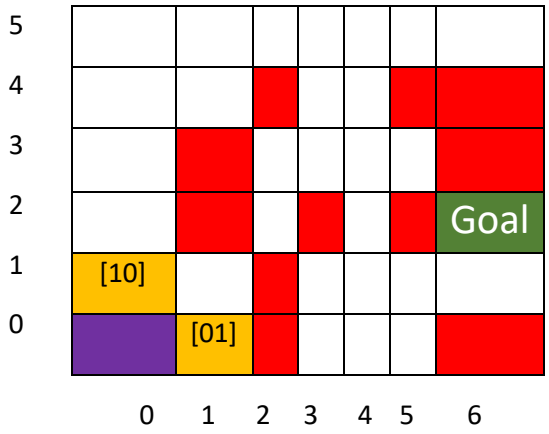
Let Start = [00]



We need to pop out [00] from queue .And it goes to visited array.

The children of [00] is [01] and [10]

The queue contains [01] and [10]



Visited

[00]

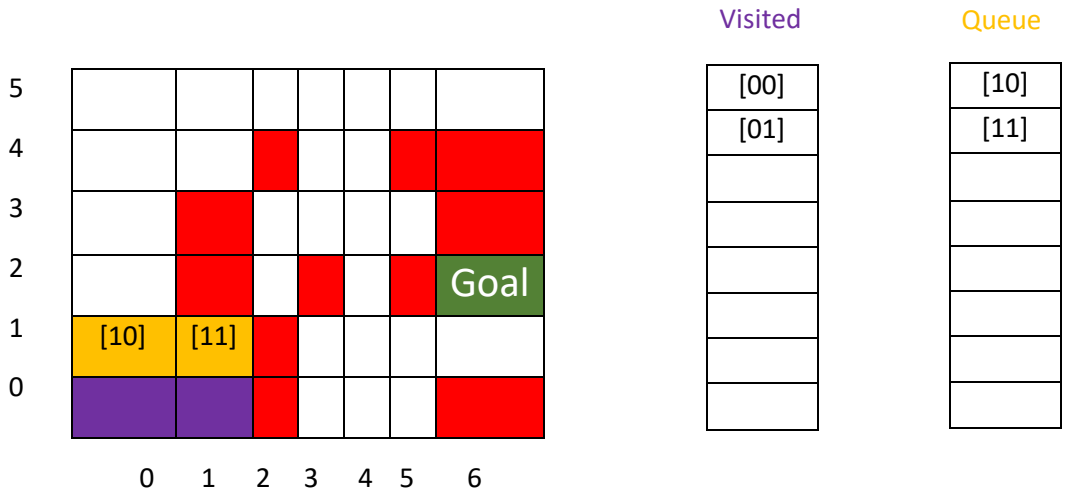
Queue

[01]
[10]

We need to pop out [01] from queue .And it goes to visited array.

The children of [01] is [11]

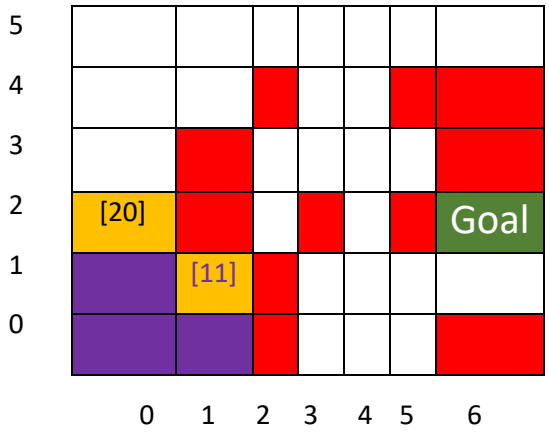
The queue contains only [10] and [11]. And visited array contains [00] and [01]



We need to pop out[10] from queue .And it goes to visited array.

The is no child of [10] grid.

The queue contains [11] and [20]



Visited

[00]
[01]
[10]

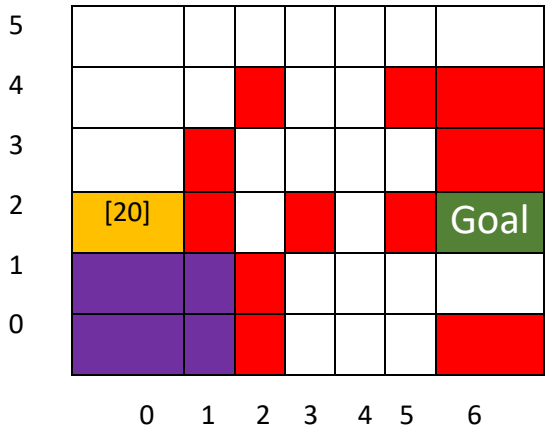
Queue

[11]
[20]

We need to pop out[11] from queue .And it goes to visited array.

The is no child of [11] grid.

The queue contains [20]



Visited

[00]
[01]
[10]
[11]

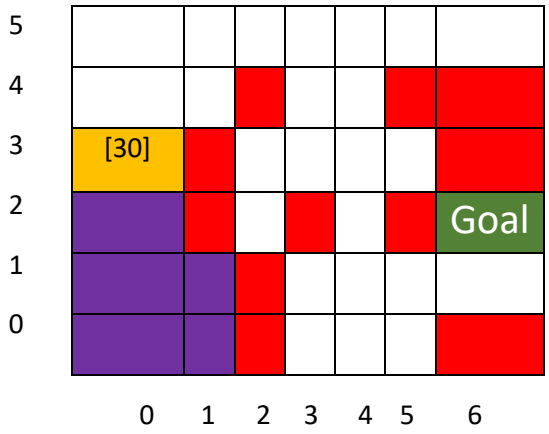
Queue

[20]

We need to pop out[20] from queue .And it goes to visited array.

The is one child of grid [20] . That is [30]

The queue contains [30]



Visited

[00]
[01]
[10]
[11]
[20]

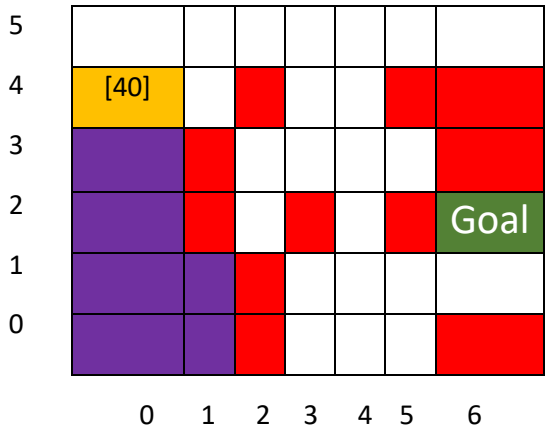
Queue

[30]

We need to pop out[30] from queue .And it goes to visited array.

The is one child of grid [30] . That is [40]

The queue contains [40]



Visited

[00]
[01]
[10]
[11]
[20]

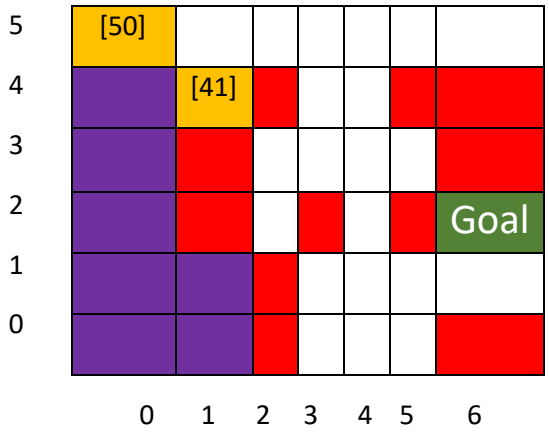
Queue

[40]

We need to pop out[40] from queue .And it goes to visited array.

There are 2 children of grid [40].

The queue contains [41] and [50].



Visited

[00]
[01]
[10]
[11]
[20]
[40]

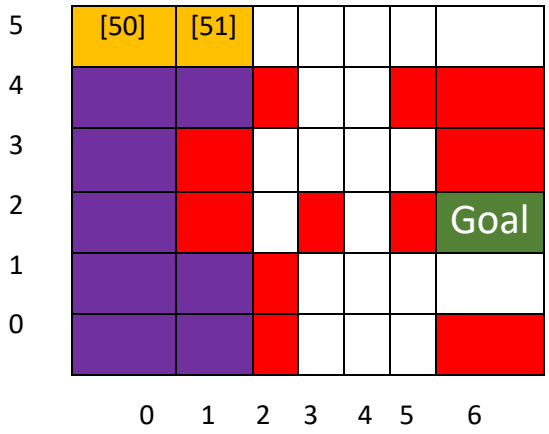
Queue

[41]
[50]

We need to pop out[41] from queue .And it goes to visited array.

There is 1 children of grid [41].

The queue contains [50] and [51].



Visited

[00]
[01]
[10]
[11]
[20]
[40]
[41]

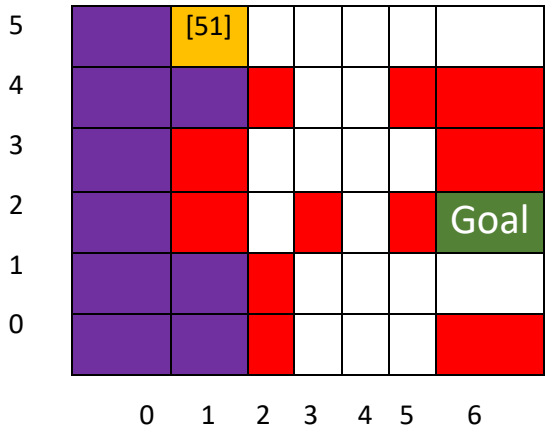
Queue

[50]
[51]

We need to pop out[50] from queue .And it goes to visited array.

There is no children of grid [50]

The queue contains [51].



Visited

[00]
[01]
[10]
[11]
[20]
[40]
[41]
[50]

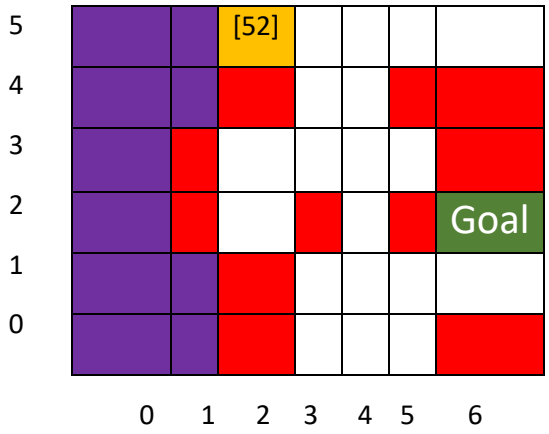
Queue

[51]

We need to pop out[51] from queue .And it goes to visited array.

Grid [51] child is [52]

The queue contains [52].



Visited

[00]
[01]
[10]
[11]
[20]
[40]
[41]
[50]
[51]

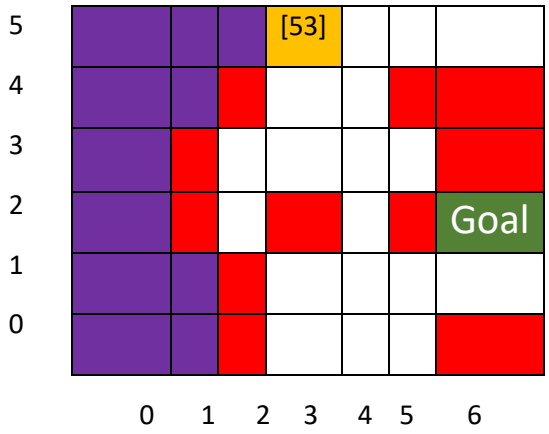
Queue

[52]

We need to pop out[52] from queue .And it goes to visited array.

Grid [52] child is [53]

The queue contains [53].



Visited

[00]
[01]
[10]
[11]
[20]
[40]
[41]
[50]
[51]
[52]

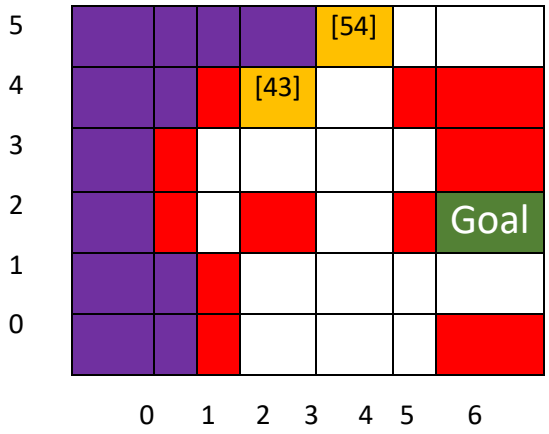
Queue

[53]

We need to pop out[53] from queue .And it goes to visited array.

Grid [53] children are [43] and [54]

The queue contains [43] and [54]



Visited

[00]
[01]
[10]
[11]
[20]
[40]
[41]
[50]
[51]
[52]
[53]

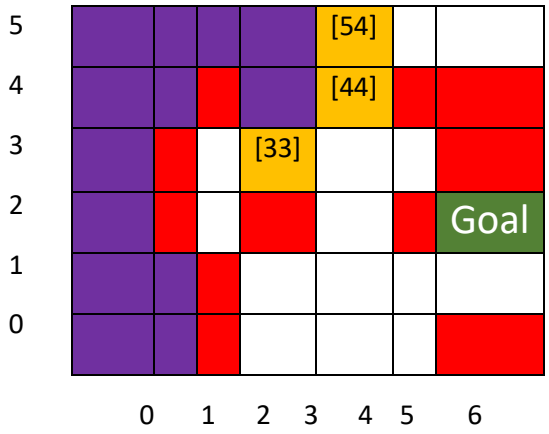
Queue

[43]
[54]

We need to pop out[43] from queue .And it goes to visited array.

Grid [53]'s children are [33] and [44]

The queue contains [33] and [44]



Visited

[00]
[01]
[10]
[11]
[20]
[40]
[41]
[50]
[51]
[52]
[53]
[43]

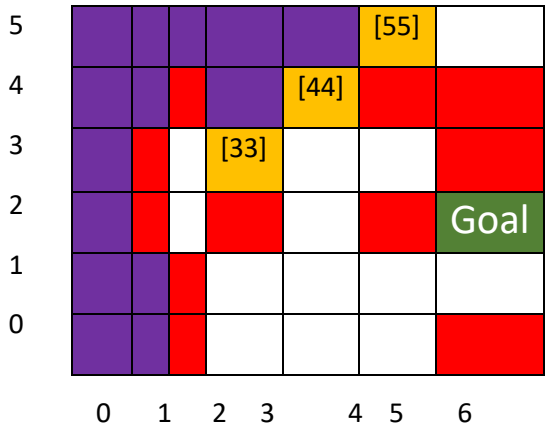
Queue

[54]
[33]
[44]

We need to pop out[54] from queue .And it goes to visited array.

Grid [53]'s is [55]

The queue contains [33], [44] and [55]



Visited

[00]
[01]
[10]
[11]
[20]
[40]
[41]
[50]
[51]
[52]
[53]
[43]
[54]

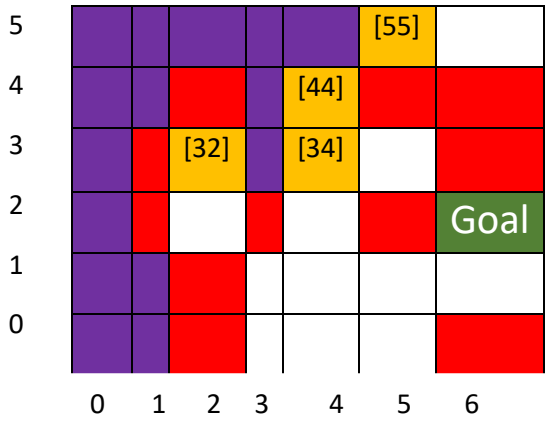
Queue

[33]
[44]
[55]

We need to pop out[33] from queue .And it goes to visited array.

Grid [53]'s children are [32] and [34]

The queue contains [44], [55], [32] and [34]



Visited

[00]
[01]
[10]
[11]
[20]
[40]
[41]
[50]
[51]
[52]
[53]
[43]
[54]
[33]

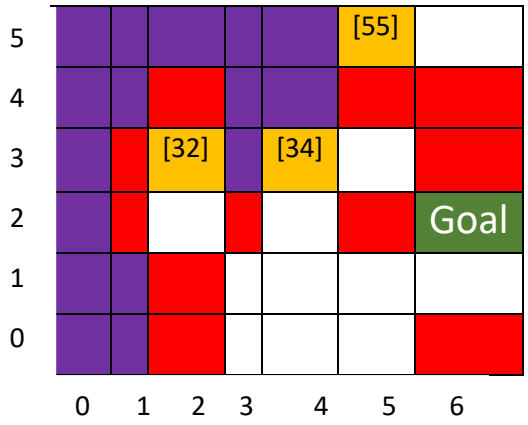
Queue

[44]
[55]
[32]
[34]

We need to pop out[44] from queue .And it goes to visited array.

Grid [44] got one child

The queue contains [55], [32] and [34]



Visited

[00]
[01]
[10]
[11]
[20]
[40]
[41]
[50]
[51]
[52]
[53]
[43]
[54]
[33]
[44]

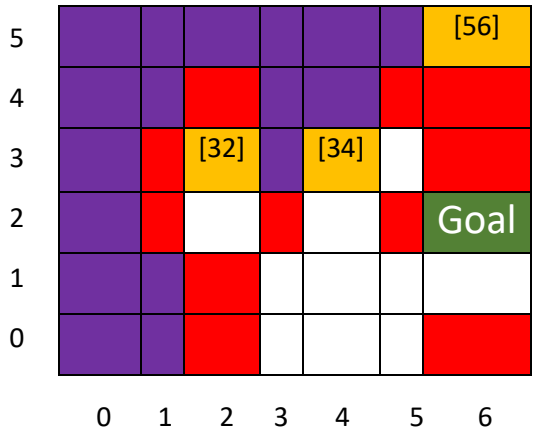
Queue

[55]
[32]
[34]

We need to pop out[55] from queue .And it goes to visited array.

Grid [55] got one child

The queue contains [32] ,[34] and [56]



Visited

[00]
[01]
[10]
[11]
[20]
[40]
[41]
[50]
[51]
[52]
[53]
[43]
[54]
[33]
[44]
[55]

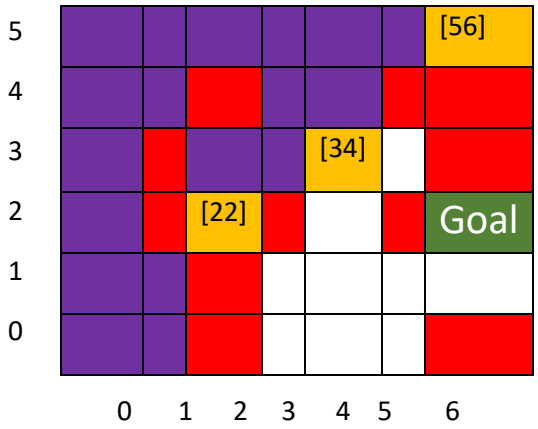
Queue

[32]
[34]
[56]

We need to pop out [32] from queue .And it goes to visited array.

Grid [55] got one child

The queue contains [34], [56] and [22]



Visited

[00]
[01]
[10]
[11]
[20]
[40]
[41]
[50]
[51]
[52]
[53]
[43]
[54]
[33]
[44]
[55]
[32]

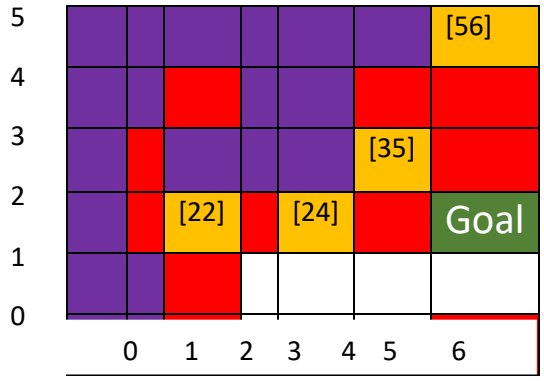
Queue

[34]
[56]
[22]

We need to pop out [34] from queue .And it goes to visited array.

Grid [55] got 2 children

The queue contains [56] ,[22] , [24] and [35]



Visited

[00]
[01]
[10]
[11]
[20]
[40]
[41]
[50]
[51]
[52]
[53]
[43]
[54]
[33]
[44]
[55]
[32]
[34]

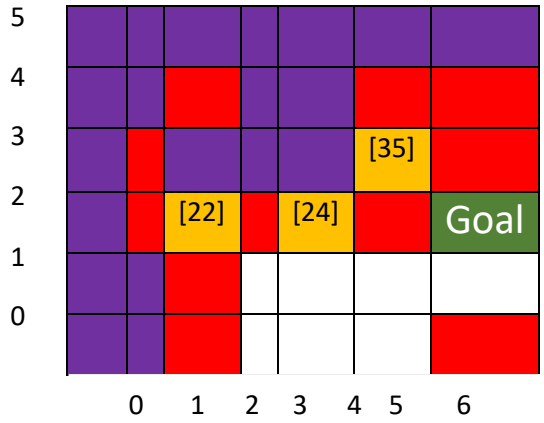
Queue

[56]
[22]
[24]
[35]

We need to pop out [56] from queue .And it goes to visited array.

Grid [55] got no children

The queue contains [22] , [24] and [35]



Visited

[00]
[01]
[10]
[11]
[20]
[40]
[41]
[50]
[51]
[52]
[53]
[43]
[54]
[33]
[44]
[55]
[32]
[34]
[56]

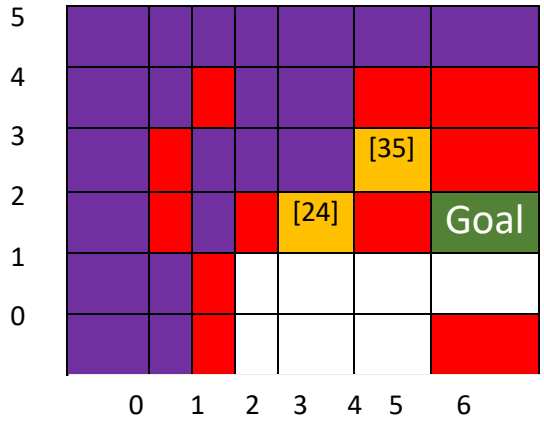
Queue

[22]
[24]
[35]

We need to pop out [22] from queue .And it goes to visited array.

Grid [55] got no children

The queue contains [24] and [35]



Visited

[00]
[01]
[10]
[11]
[20]
[40]
[41]
[50]
[51]
[52]
[53]
[43]
[54]
[33]
[44]
[55]
[32]
[34]
[56]
[22]

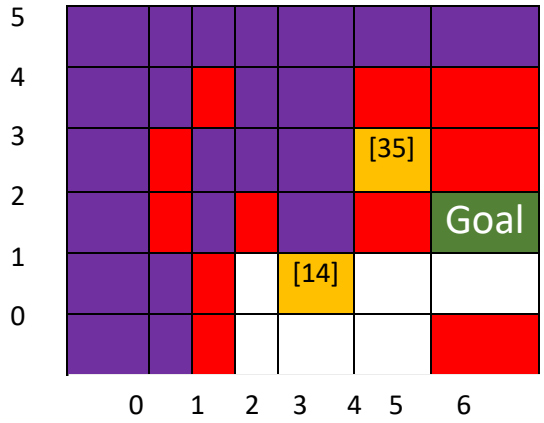
Queue

[24]
[35]

We need to pop out [24] from queue .And it goes to visited array.

Grid [55] got one child

The queue contains [35] and [14]



Visited

[00]
[01]
[10]
[11]
[20]
[40]
[41]
[50]
[51]
[52]
[53]
[43]
[54]
[33]
[44]
[55]
[32]
[34]
[56]
[22]
[24]

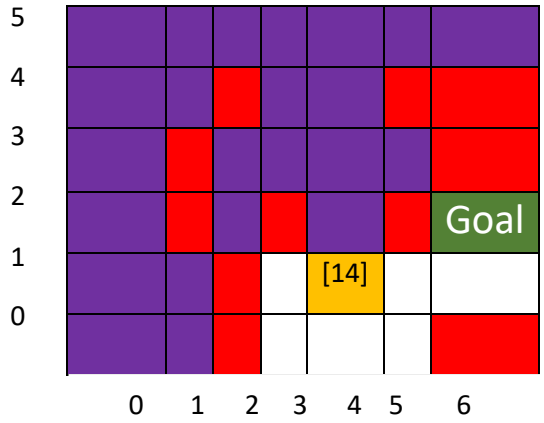
Queue

[35]
[14]

We need to pop out [35] from queue .And it goes to visited array.

Grid [35] got no child

The queue contains [14]



Visited

[00]
[01]
[10]
[11]
[20]
[40]
[41]
[50]
[51]
[52]
[53]
[43]
[54]
[33]
[44]
[55]
[32]
[34]
[56]
[22]
[14]
[35]

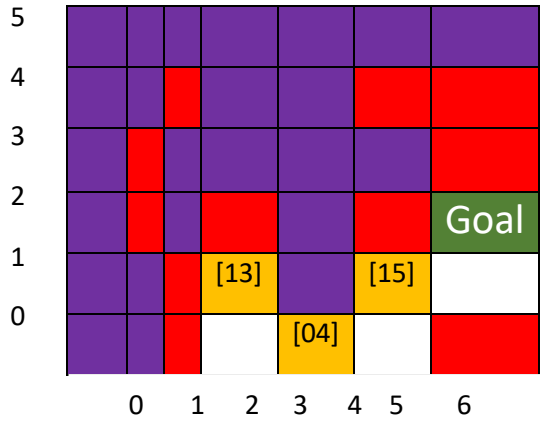
Queue

[14]

We need to pop out [14] from queue .And it goes to visited array.

Grid [14] got three children

The queue contains [04], [13] and [15]



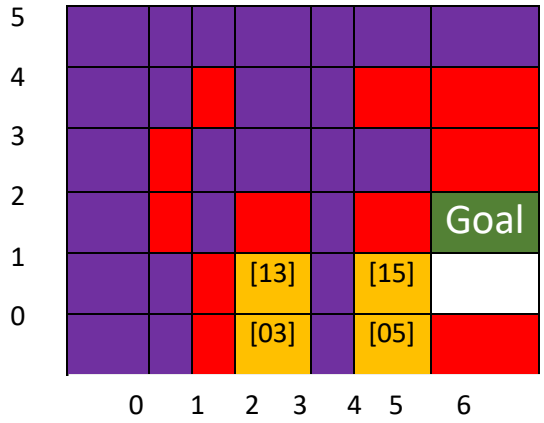
Visited
[00]
[01]
[10]
[11]
[20]
[40]
[41]
[50]
[51]
[52]
[53]
[43]
[54]
[33]
[44]
[55]
[32]
[34]
[56]
[22]
[14]
[35]
[14]

Queue
[04]
[13]
[15]

We need to pop out [04] from queue .And it goes to visited array.

Grid [04] got two children

The queue contains [13], [15], [03] and [05]



Visited

[00]
[01]
[10]
[11]
[20]
[40]
[41]
[50]
[51]
[52]
[53]
[43]
[54]
[33]
[44]
[55]
[32]
[34]
[56]
[22]
[35]
[14]
[04]

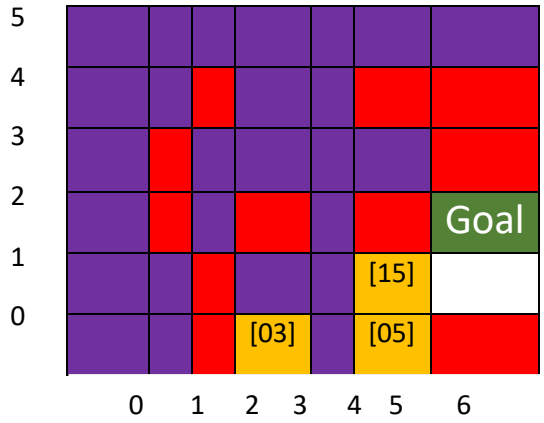
Queue

[13]
[15]
[03]
[05]

We need to pop out [13] from queue .And it goes to visited array.

Grid [13] got no children

The queue contains [15], [03] and [05]



Visited

[00]
[01]
[10]
[11]
[20]
[40]
[41]
[50]
[51]
[52]
[53]
[43]
[54]
[33]
[44]
[55]
[32]
[34]
[56]
[22]
[35]
[14]
[04]
[13]

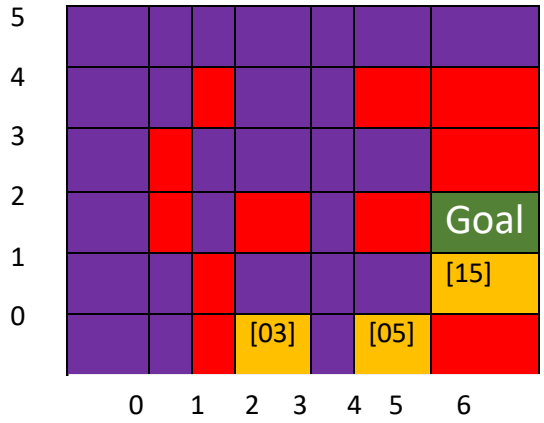
Queue

[15]
[03]
[05]

We need to pop out [15] from queue .And it goes to visited array.

Grid [15] got one children

The queue contains [03], [05] and [16]



Visited

[00]
[01]
[10]
[11]
[20]
[40]
[41]
[50]
[51]
[52]
[53]
[43]
[54]
[33]
[44]
[55]
[32]
[34]
[56]
[22]
[35]
[14]
[04]
[13]
[15]

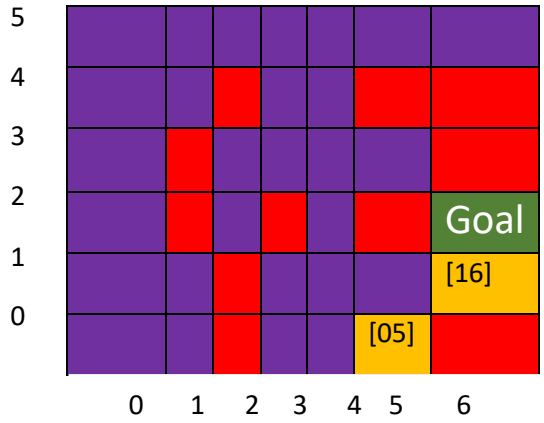
Queue

[03]
[05]
[16]

We need to pop out [03] from queue .And it goes to visited array.

Grid [03] got no children

The queue contains [05] and [16]



Visited

[00]
[01]
[10]
[11]
[20]
[40]
[41]
[50]
[51]
[52]
[53]
[43]
[54]
[33]
[44]
[55]
[32]
[34]
[56]
[22]
[35]
[14]
[04]
[13]
[15]
[03]

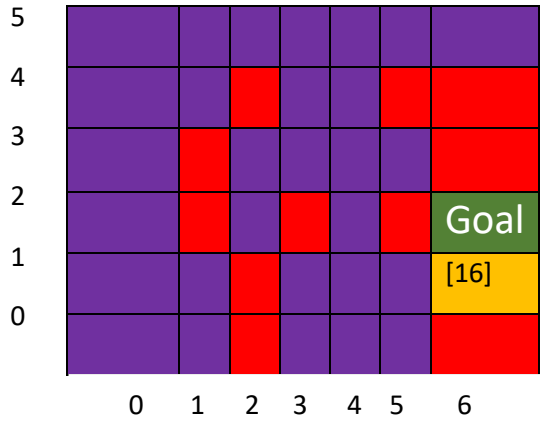
Queue

[05]
[16]

We need to pop out [05] from queue .And it goes to visited array.

Grid [05] got no children

The queue contains [16]



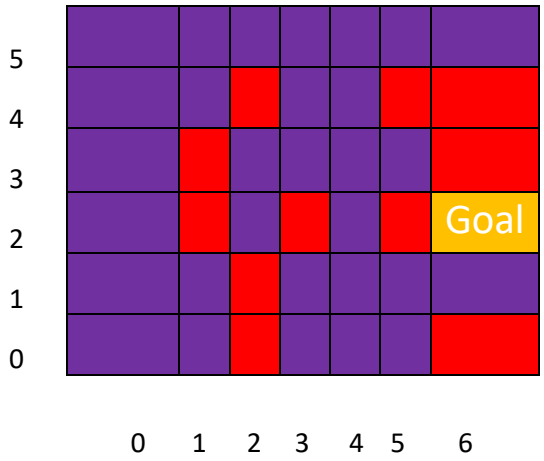
Visited

[00]
[01]
[10]
[11]
[20]
[40]
[41]
[50]
[51]
[52]
[53]
[43]
[54]
[33]
[44]
[55]
[32]
[34]
[56]
[22]
[35]
[14]
[04]
[13]
[15]
[03]
[05]

Queue

[16]

we have reached the goal point.



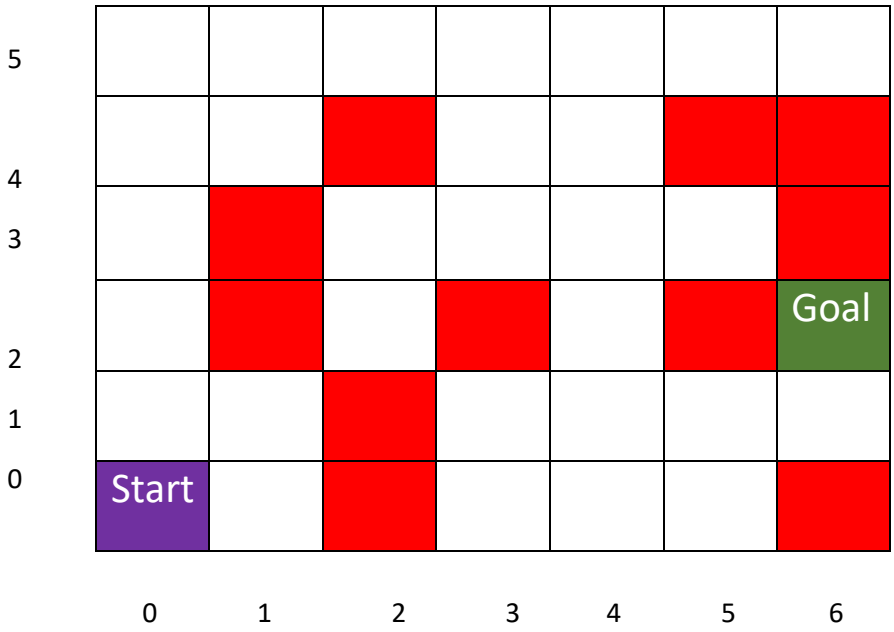
Visited

[00]
[01]
[10]
[11]
[20]
[40]
[41]
[50]
[51]
[52]
[53]
[43]
[54]
[33]
[44]
[55]
[32]
[34]
[56]
[22]
[35]
[14]
[04]
[13]
[15]
[03]
[05]
[16]

Queue

[26]

2. Run Depth FIrst Search (DFS) from the start node and stop when you reach the goal. Show each step/phase of the algorithm



Starting node is [00] and the goal is [26]. We will reach the goal node by using dfs.we move left, right, up and down

5	[5,0]	[5,1]	[5,2]	[5,3]	[5,4]	[5,5]	[5,6]
	[4,0]	[1,4]		[4,3]	[4,4]		
4	[3,0]		[3,2]	[3,3]	[3,4]		
	[2,0]		[2,2]		[2,4]		Goal
3	[1,0]			[1,3]	[1,4]	[1,5]	[1,6]
	Start			[0,3]	[0,4]	[0,5]	
2							
1							
0							
	0	1	2	3	4	5	6

[4,4]	
[3,4]	
[2,2]	
[3,2]	
[3,3]	Goal
[4,3]	[1,6]
[5,3]	[1,5]
[5,2]	[0,5]
[1,4]	[1,3]
[5,1]	[0,3]
[5,0]	[0,4]
[4,0]	[1,4]
[3,0]	[2,4]
[2,0]	[5,6]
[1,0]	[5,5]
Start	[5,4]

[5,0]	[5,1]	[5,2]	[5,3]	[5,4]	[5,5]	[5,6]
[4,0]	[1,4]		[4,3]	[4,4]		
[3,0]		[3,2]	[3,3]	[3,4]		
[2,0]		[2,2]		[2,4]		Goal
[1,0]			[1,3]	[1,4]	[1,5]	[1,6]
Start			[0,3]	[0,4]	[0,5]	

Shortest Path:

Visited

a. Compare BFS and DFS and write the differences between them.

BFS(Breadth First Search)	DFS(Depth First Search)
BFS uses Queue to find the shortest path.	DFS uses Stack to find the shortest path.
BFS is better when target is closer to Source.	DFS is better when target is far from source.
As BFS considers all neighbour so it is not suitable for decision tree used in puzzle games.	DFS is more suitable for decision tree. As with one decision, we need to traverse further to augment the decision. If we reach the conclusion, we won.
BFS is slower than DFS.	DFS is faster than BFS.

Time Complexity of BFS = O(V+E) where V is vertices and E is edges.	Time Complexity of DFS is also O(V+E) where V is vertices and E is edges