Artificial Intelligence Assignment 1 Report

Submitted by:
Name: **Aditya Mishra**
Roll No. : **M20MA201**

Submitted To:
Dr. Pratik Mazumder

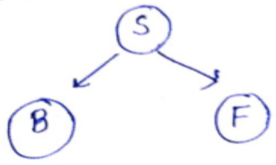
Given : NxN matrix and search for an element e, Start checking checking only from the last column of the first row.
Cost Function : The cost for moving up or down is 2 from one cell to the next and the cost for moving left or right is 1 from one cell to the next.

	0	1	2
0	3 (A)	4 (B)	5 (S)
1	2 (C)	99 (D)	1 (F)
2	12 (e)	34 (H)	9 (J)

a) Using Depth First Search (DFS)

Using the approach DFS : (Alphabetical order matters)

Step 1



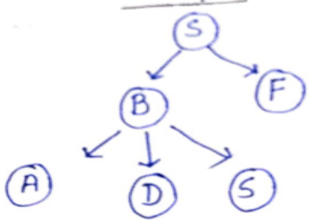
Fringe

~~S~~

S → B

S → F

Step 2



Fringe

~~S~~

~~S → B~~

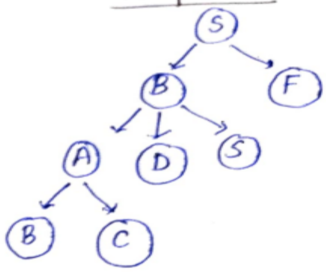
S → F

S → B → A

S → B → D

S → B → S

Step 3



Fringe

~~S~~

~~S → B~~

S → F

~~S → B → A~~

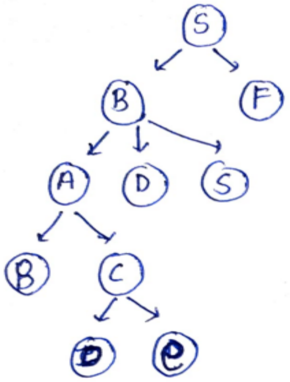
S → B → D

S → B → S

S → B → A → B

S → B → A → C

Step - 4



Fringe

~~S~~

~~S → B~~

S → F

~~S → B → A~~

S → B → D

S → B → S

S → B → A → B

~~S → B → A → C~~

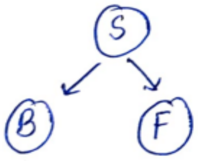
$\boxed{S \rightarrow B \rightarrow A \rightarrow C \rightarrow E}$

Final. Solution is $\boxed{S \rightarrow B \rightarrow A \rightarrow C \rightarrow E}$

Using Breadth First Search (BFS)

Using the Approach BFS

Step 1



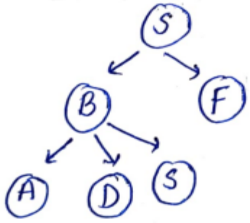
fringes

~~S~~

S → B

S → F

Step-2



Fringes

~~S~~

~~S → B~~

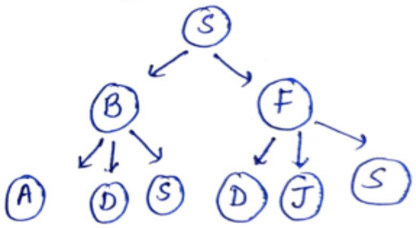
S → F

S → B → A

S → B → D

S → B → S

Step - 3



Fringes

~~S~~

~~S → B~~

~~S → F~~

S → B → A

S → B → D

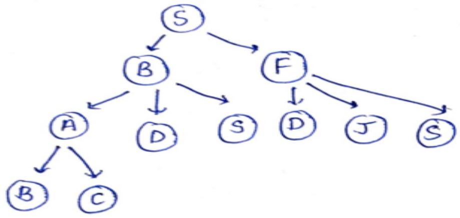
S → B → S

S → F → D

S → F → J

S → F → S

Step - 4



Fringes

~~S~~

~~S → B~~

~~S → F~~

~~S → B → A~~

S → B → D̂

S → B → S

S → F → D

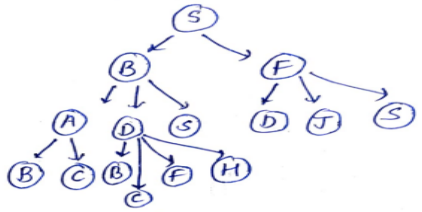
S → F → J

S → F → S

S → B → A → B

S → B → A → C

Step - 5



Fringes

~~S~~

~~S → B~~

~~S → F~~

~~S → B → A~~

~~S → B → D~~

S → B → S

S → F → D

S → F → J

S → F → S

S → B → A → B

S → B → A → C

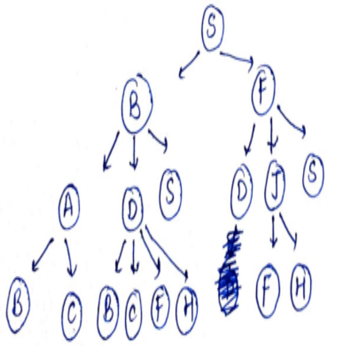
S → B → D → B

S → B → D → C

S → B → D → F

S → B → D → H

Step- 6



fringes

~~S~~

~~S → B~~

~~S → F~~

~~S → B → A~~

~~S → B → D~~

S → B → S

S → F → D

~~S → F → J~~

S → F → S

S → B → A → B

S → B → A → C

S → B → D → B

S → B → D → C

S → B → D → F

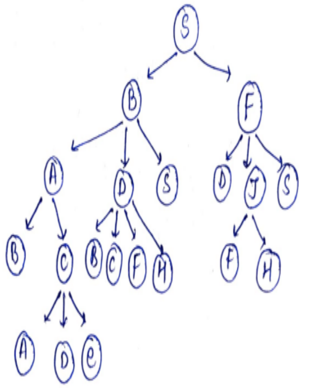
S → B → D → H

S → F → J → F

S → F → J → H

Since S → B → S is going to repeated so it was not extended further similarly for S → f → D and S → f → S are not extended further.

Step- 7



Since S → B → A → B has repeated extended node of B, so it was not extended.

Final result is $\boxed{S \to B \to A \to C \to e}$

fringes

~~S~~

~~S → B~~

~~S → F~~

~~S → B → A~~

~~S → B → D~~

S → B → S

S → F → D

~~S → F → J~~

S → F → S

S → B → A → S

~~S → B → A → C~~

S → B → D → B

S → B → D → C

S → B → D → F

S → B → D → H

S → F → J → F

S → F → J → H

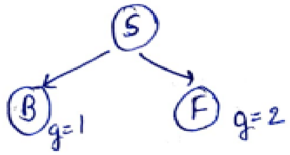
S → B → A → C → A

S → B → A → C → D

$\boxed{S \to B \to A \to C \to e}$

Uniform Cost Search (UCS)

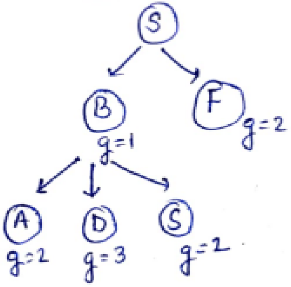
Step- 1



Fringes

~~S~~
S → B
S → F

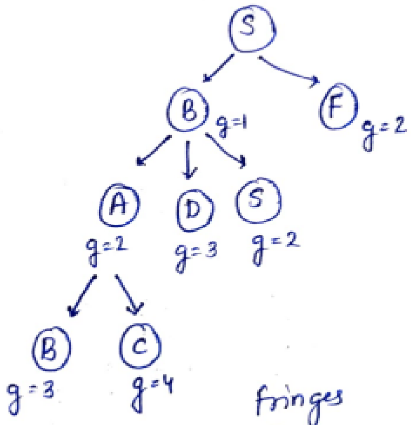
Step - 2



Fringes

~~S~~
~~S → B~~
S → f
S → B → A
S → B → D
S → B → S

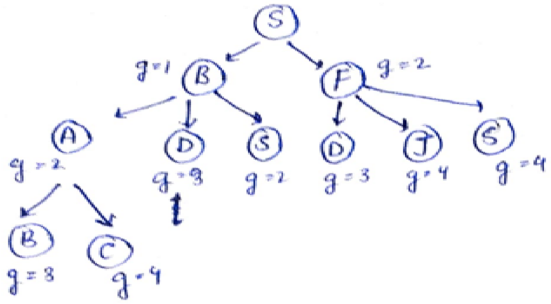
Step - 3



fringes

~~S~~
~~S → B~~
S → f
~~S → B → A~~
S → B → D
S → B → S
S → B → A → B
S → B → A → C

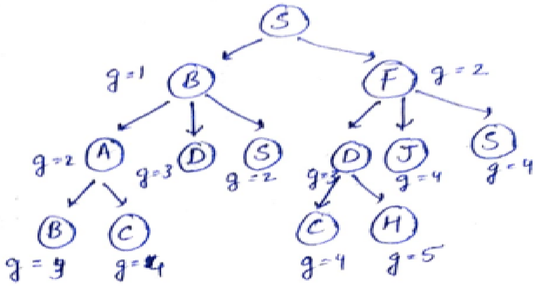
Step- 4



Fringes

~~S~~
~~S → B~~
~~S → F~~
~~S → B → A~~
S → B → D
S → B → S
S → B → A → B
S → B → A → C
S → F → D
S → F → J
S → F → S

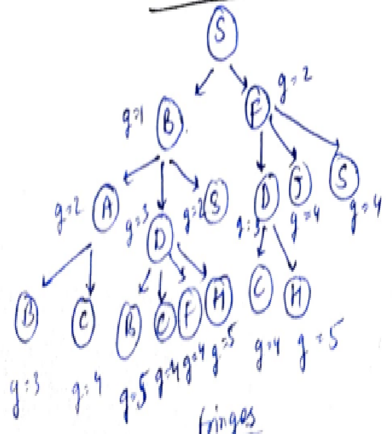
Step- 5



Fringes

~~S~~
~~S → B~~
~~S → f~~
~~S → B → A~~
S → B → D
S → B → S
S → B → A → B
S → B → A → C
~~S → f → D~~
S → f → J
S → f → S
S → F → D → C
S → f → D → H

Step - 6

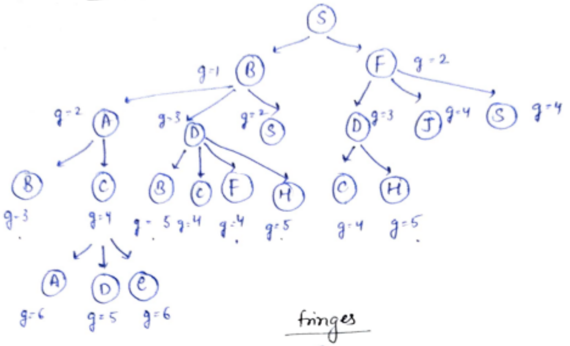


fringes

~~S~~
~~S→B~~
~~S→F~~
~~S→B→A~~
~~S→B→D~~
S→B→S
S→B→A→B
S→B→A→C
~~S→F→D~~
S→f→J
S→F→J
S→F→D→C
S→F→D→H

S→B→D→B
S→B→D→C
S→B→D→F
S→B→D→H

Step - 7



fringes

~~S~~
~~S→B~~
~~S→f~~
~~S→B→A~~
~~S→B→D~~
S→B→S
S→B→A→B
~~S→B→A→C~~
~~S→f→D~~
S→F→J
S→f→D→C
S→f→D→H
S→B→D→B
S→B→D→C
S→B→D→f
S→B→D→H
S→B→A→C→A
S→B→A→C→D
$\boxed{S \to B \to A \to C \to e}$

final Result $\boxed{S \to B \to A \to c \to e}$

b) Heuristic Functions: 1) sum of horizontal and vertical distance from the current cell to the element.

Solution : h(A) = 2, h(B) = 3, h(C) = 1, h(D) = 2, h(e) = 0, h(F) = 3, h(H) = 1, h(J) = 2

Heuristic Functions: 2) diagonal distance from the current cell to the element.

Solution : h(A) = 2, h(B) = √(5) , h(C) = 1, h(D) = √(2), h(e) = 0, h(F) =√(5), h(H) = 1, h(J) = 2

i) Check Admissibility :

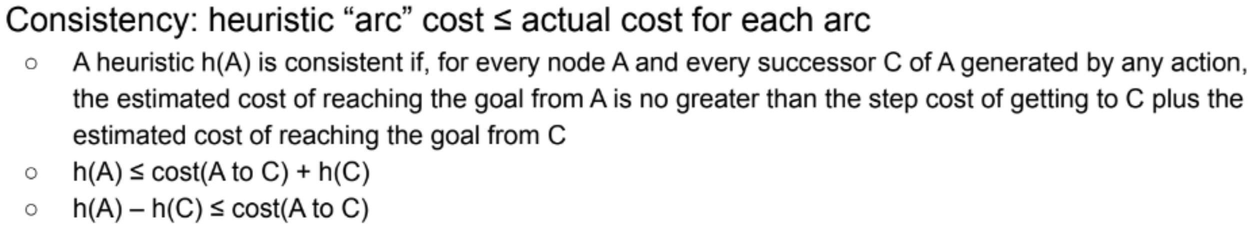
A heuristic *h* is *admissible* (optimistic) if:

$$0 \leq h(n) \leq h^*(n)$$

Where $h^*(n)$ is the true cost to a nearest goal

By This Definition for each node **h is admissible.**

ii) Check Consistency:

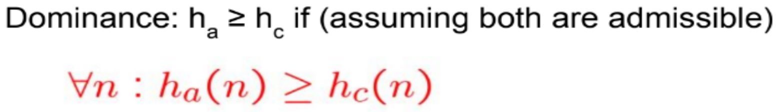


Consistency: heuristic “arc” cost ≤ actual cost for each arc

- A heuristic h(A) is consistent if, for every node A and every successor C of A generated by any action, the estimated cost of reaching the goal from A is no greater than the step cost of getting to C plus the estimated cost of reaching the goal from C
- h(A) ≤ cost(A to C) + h(C)
- h(A) – h(C) ≤ cost(A to C)

By the above definition we are finding that there is **Consistency.**

ii) Also check which heuristic function dominates the other :



Dominance: $h_a \geq h_c$ if (assuming both are admissible)

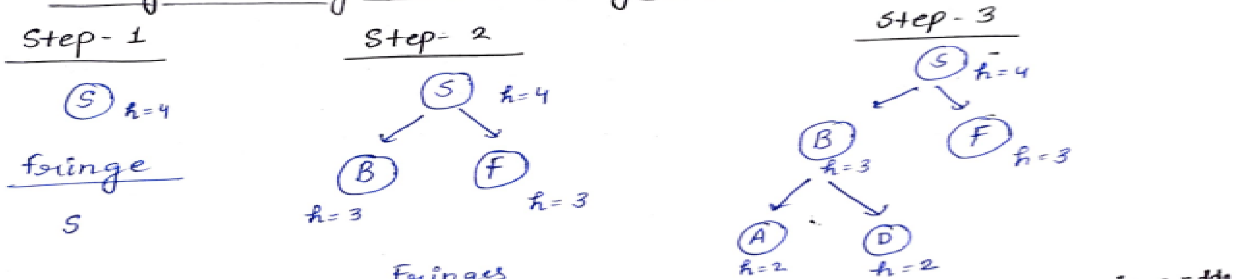
$$\forall n : h_a(n) \geq h_c(n)$$

By the above definition we are finding that Heuristics **1 is Dominating.**

---

c)Greedy Search Algorithm

Using Greedy Search Algorithm :-



fringe

S

Fringes

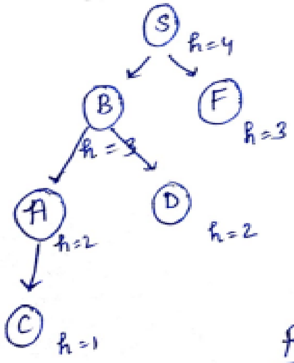
~~S~~
S → B
S → F

Note:- Repeation of previous node is restricted here.

Fringes

~~S~~
~~S → B~~
S → F
S → B → A
S → B → D

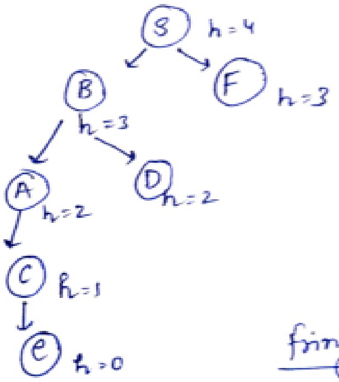
Step-4



fringes

~~S~~
~~S → B~~
S → F
~~S → B → A~~
S → B → D
S → B → A → C

Step - 5



fringes

~~S~~
~~S → B~~
S → F
~~S → B → A~~
S → B → D
~~S → B → A → C~~
S → B → A → C → e

final result $\boxed{S \to B \to A \to C \to e}$

A* Search Algorithm

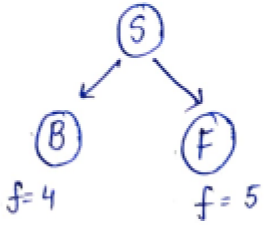
Using the approach of A* search algorithm:-

Step 1

fringes

S

Step-2

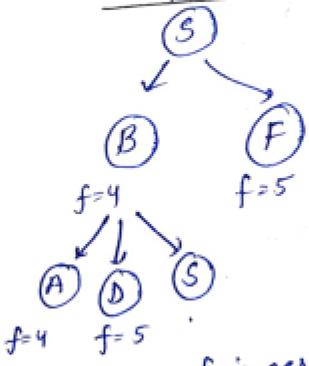


Note: $f = g + h$

fringes

~~S~~
S → B
S → F

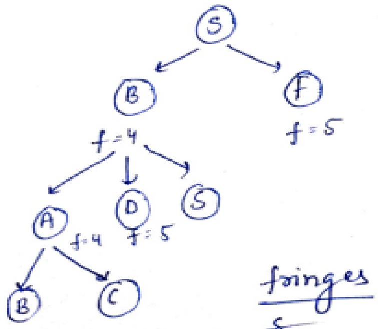
Step-3



fringes

~~S~~
~~S → B~~
S → F
S → B → A
S → B → D
S → B → S

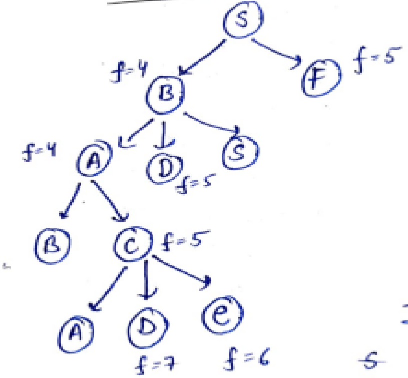
Step-4



fringes

~~S~~
~~S → B~~
S → F
~~S → B → A~~
S → B → D
S → B → S
S → B → A → B
S → B → A → C

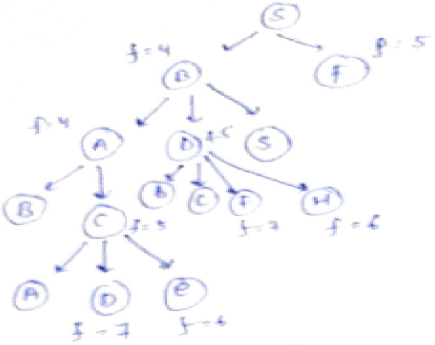
Step- 5



fringes

~~S~~
~~S → B~~
S → F
~~S → B → A~~
S → B → D
S → B → S
S → B → A → B
~~S → B → A → C~~
S → B → A → C → A
S → B → A → C → D
S → B → A → C → e

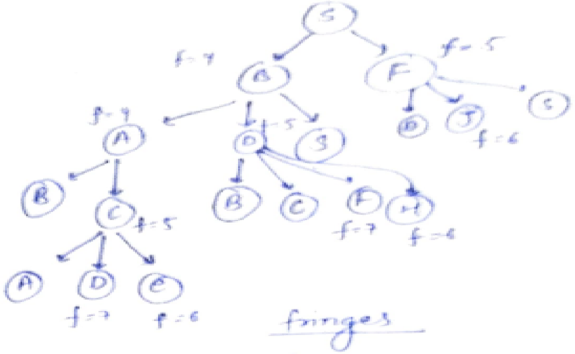
Step - 6



fringes

~~S~~
~~S → B~~
S → F
~~S → B → A~~
~~S → B → D~~
S → B → A → B
~~S → B → A → C~~
S → B → A → C → A
S → B → A → C → D
S → B → A → C → e
S → B → D → B
S → B → D → C
S → B → D → F
S → B → D → H

Step - 7



fringes

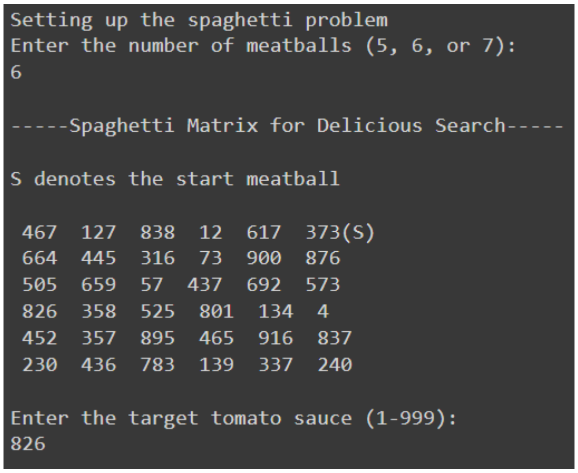
~~S~~
~~S → B~~
~~S → F~~
~~S → B → A~~
~~S → B → D~~
S → B → A → B
~~S → B → A → C~~
S → B → A → C → A
S → B → A → C → D
$\boxed{S \to B \to A \to C \to e}$
S → B → D → B
S → B → D → C
S → B → D → F
S → B → D → H
S → F → D
S → F → J
S → F → S

final result is S → B → A → C → e

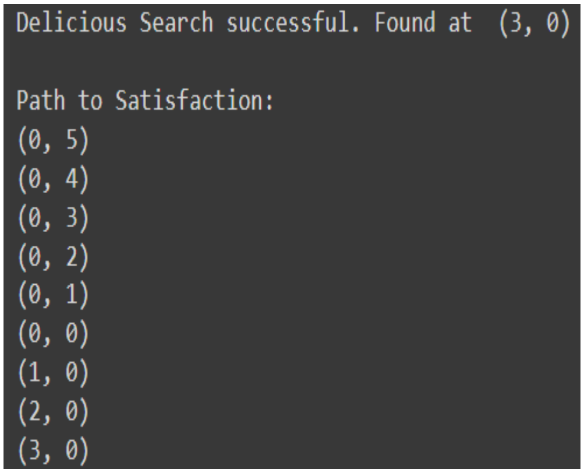
d) Given : **input N and e**. N denotes the size of the **NxN matrix and N can be either 5,6, or 7**. **e** denotes the number being searched. All other entries of the matrix should be filled with **random natural numbers between 1 and 1000**. The position of the start state will be the last column of the first row. The output of the program should be the path taken to reach the element e from the start state, i.e. all states on the path from start to the target. After each step, your code should print the current state. Each state should be represented by its row and column number.

Using Depth First Search (DFS)

Results are as :

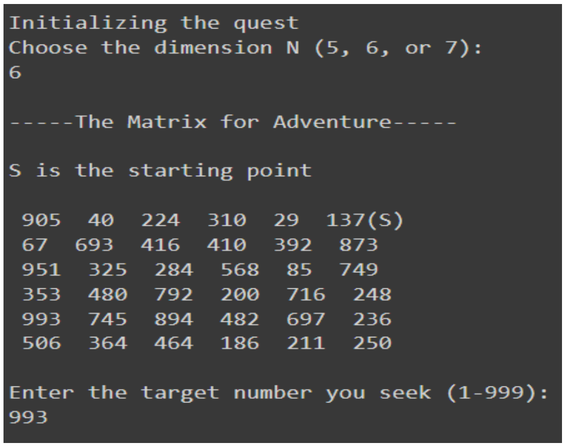


```
Setting up the spaghetti problem
Enter the number of meatballs (5, 6, or 7):
6

-----Spaghetti Matrix for Delicious Search-----

S denotes the start meatball

 467  127  838  12  617  373(S)
 664  445  316  73  900  876
 505  659  57  437  692  573
 826  358  525  801  134  4
 452  357  895  465  916  837
 230  436  783  139  337  240

Enter the target tomato sauce (1-999):
826
```

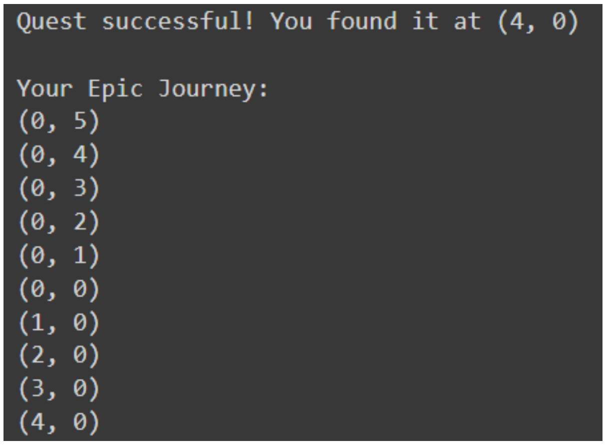


```
Delicious Search successful. Found at  (3, 0)


Path to Satisfaction:
(0, 5)
(0, 4)
(0, 3)
(0, 2)
(0, 1)
(0, 0)
(1, 0)
(2, 0)
(3, 0)
```

Using Breadth First Search (BFS)

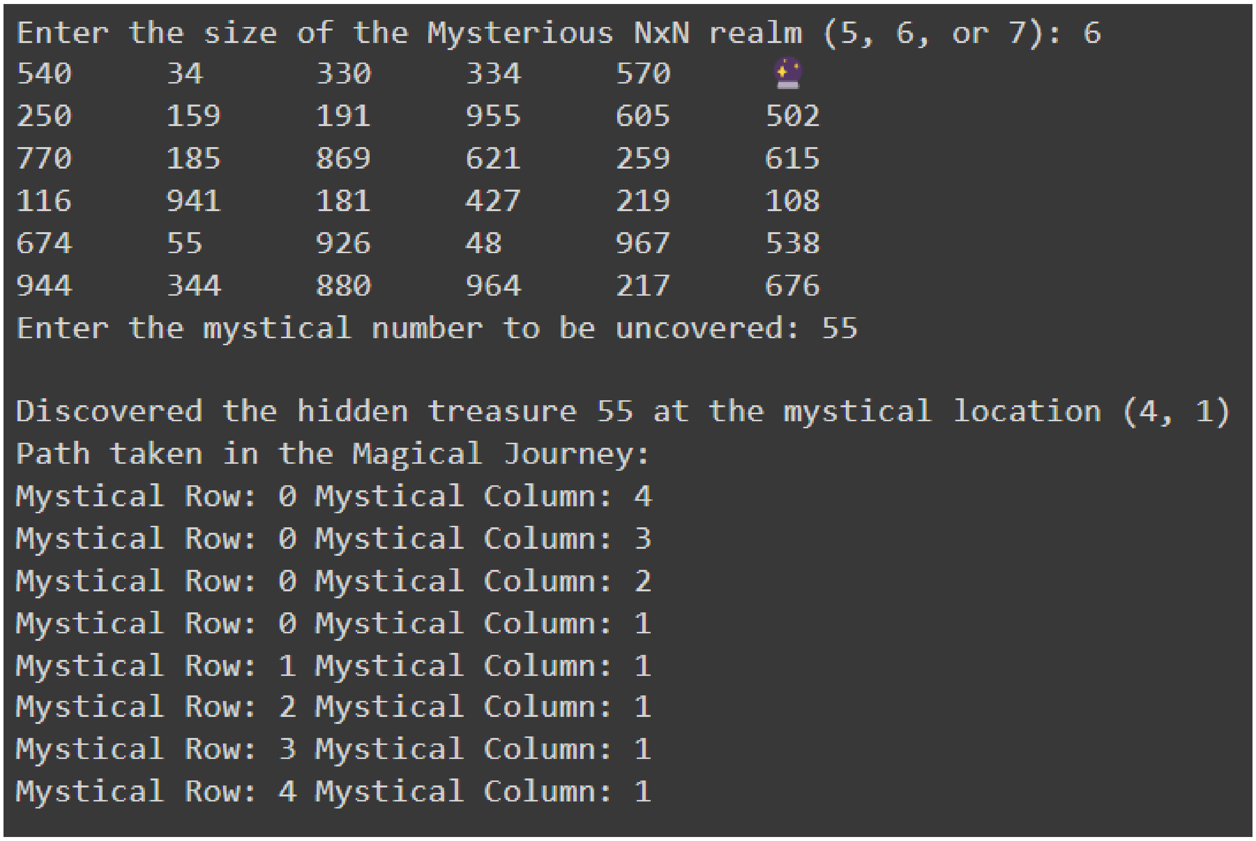


```
Initializing the quest
Choose the dimension N (5, 6, or 7):
6

-----The Matrix for Adventure-----

S is the starting point

 905  40  224  310  29  137(S)
 67  693  416  410  392  873
 951  325  284  568  85  749
 353  480  792  200  716  248
 993  745  894  482  697  236
 506  364  464  186  211  250

Enter the target number you seek (1-999):
993
```



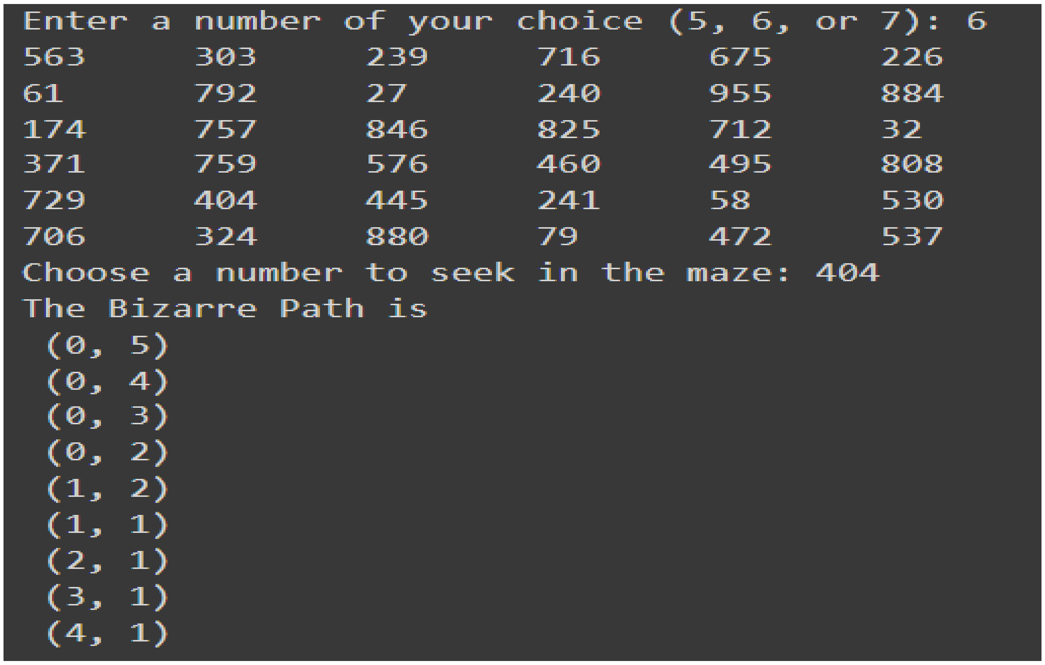
```
Quest successful! You found it at (4, 0)

Your Epic Journey:
(0, 5)
(0, 4)
(0, 3)
(0, 2)
(0, 1)
(0, 0)
(1, 0)
(2, 0)
(3, 0)
(4, 0)
```

Uniform Cost Search (UCS)



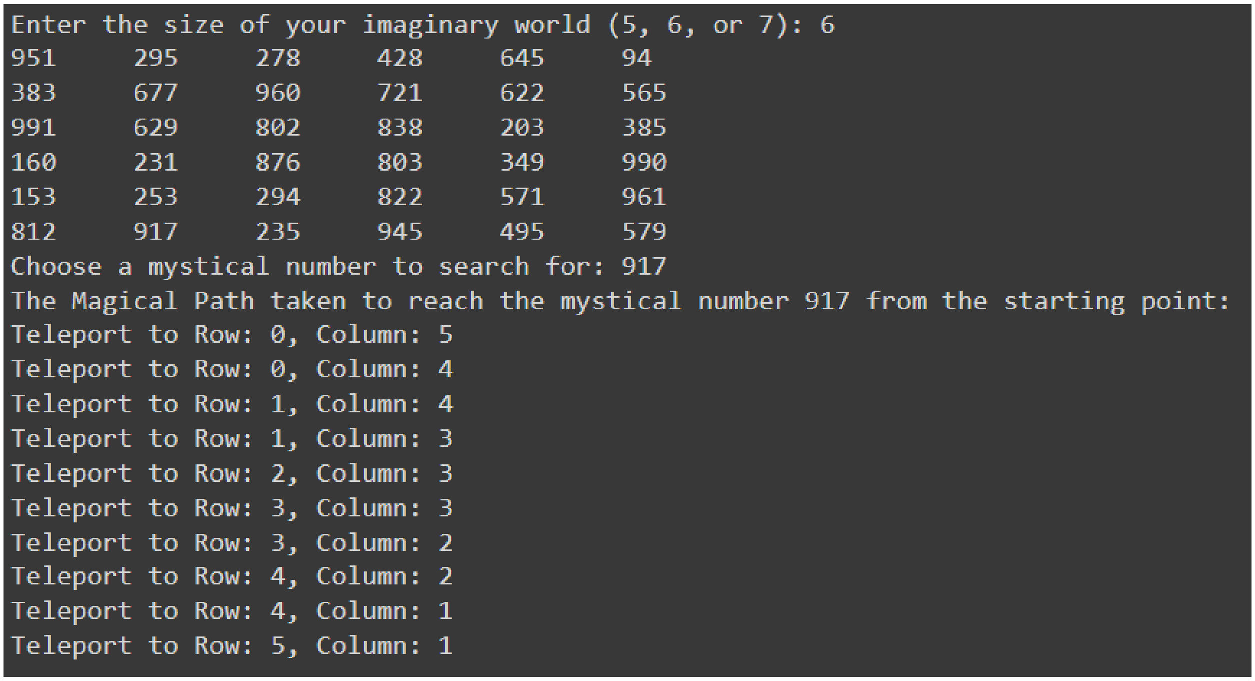
```
Enter the size of the Mysterious NxN realm (5, 6, or 7): 6
540     34      330     334     570     🔮
250     159     191     955     605     502
770     185     869     621     259     615
116     941     181     427     219     108
674     55      926     48      967     538
944     344     880     964     217     676
Enter the mystical number to be uncovered: 55

Discovered the hidden treasure 55 at the mystical location (4, 1)
Path taken in the Magical Journey:
Mystical Row: 0 Mystical Column: 4
Mystical Row: 0 Mystical Column: 3
Mystical Row: 0 Mystical Column: 2
Mystical Row: 0 Mystical Column: 1
Mystical Row: 1 Mystical Column: 1
Mystical Row: 2 Mystical Column: 1
Mystical Row: 3 Mystical Column: 1
Mystical Row: 4 Mystical Column: 1
```

Greedy Search Algorithm



```
Enter a number of your choice (5, 6, or 7): 6
563     303     239     716     675     226
61      792     27      240     955     884
174     757     846     825     712     32
371     759     576     460     495     808
729     404     445     241     58      530
706     324     880     79      472     537
Choose a number to seek in the maze: 404
The Bizarre Path is
 (0, 5)
 (0, 4)
 (0, 3)
 (0, 2)
 (1, 2)
 (1, 1)
 (2, 1)
 (3, 1)
 (4, 1)
```

A* Search Algorithm



```
Enter the size of your imaginary world (5, 6, or 7): 6
951     295     278     428     645     94
383     677     960     721     622     565
991     629     802     838     203     385
160     231     876     803     349     990
153     253     294     822     571     961
812     917     235     945     495     579
Choose a mystical number to search for: 917
The Magical Path taken to reach the mystical number 917 from the starting point:
Teleport to Row: 0, Column: 5
Teleport to Row: 0, Column: 4
Teleport to Row: 1, Column: 4
Teleport to Row: 1, Column: 3
Teleport to Row: 2, Column: 3
Teleport to Row: 3, Column: 3
Teleport to Row: 3, Column: 2
Teleport to Row: 4, Column: 2
Teleport to Row: 4, Column: 1
Teleport to Row: 5, Column: 1
```