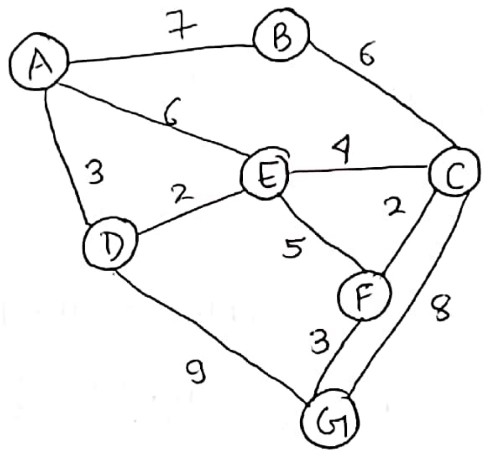
Assignment-1

Name-Yunika Upadhayaya
ID- 1001631183

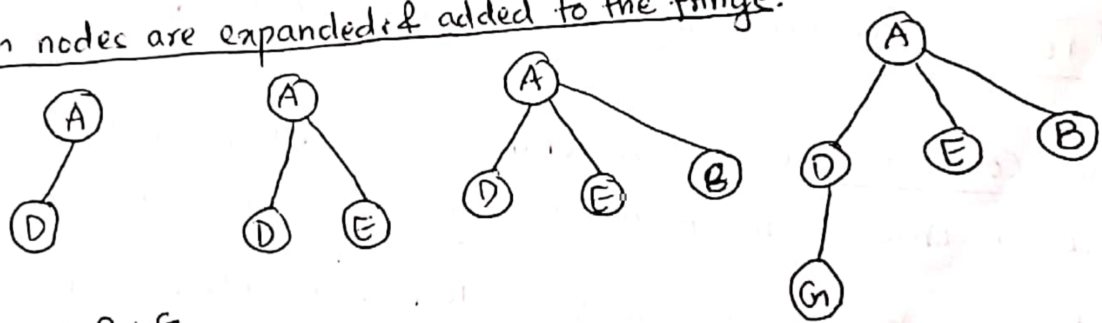
Task-2



Solution :

Breadth First Search (BFS) : In BFS, we search through every breadth. We search through all the nodes of the parent node then we move to the next node.

Order in which nodes are expanded & added to the fringe:

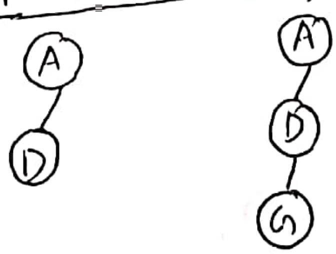


BFS = A→D→E→B→G

$\boxed{\text{ie BFS = ADEBG}}$

Depth First Search (DFS) : In DFS, we search through every depth. We search through one nodes of the parent node and keep on exploring the same node in the same depth.

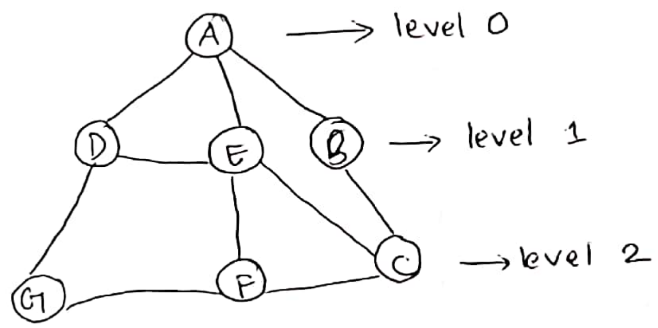
Order in which nodes are expanded and added to the fringe:



DFS = A→D→G

$\boxed{\text{i.e. DFS = ADG}}$

Iterative Deepening (IDS) search: In iterative deepening search, we search through each level. Search through each level is first done in depth and then breadth.



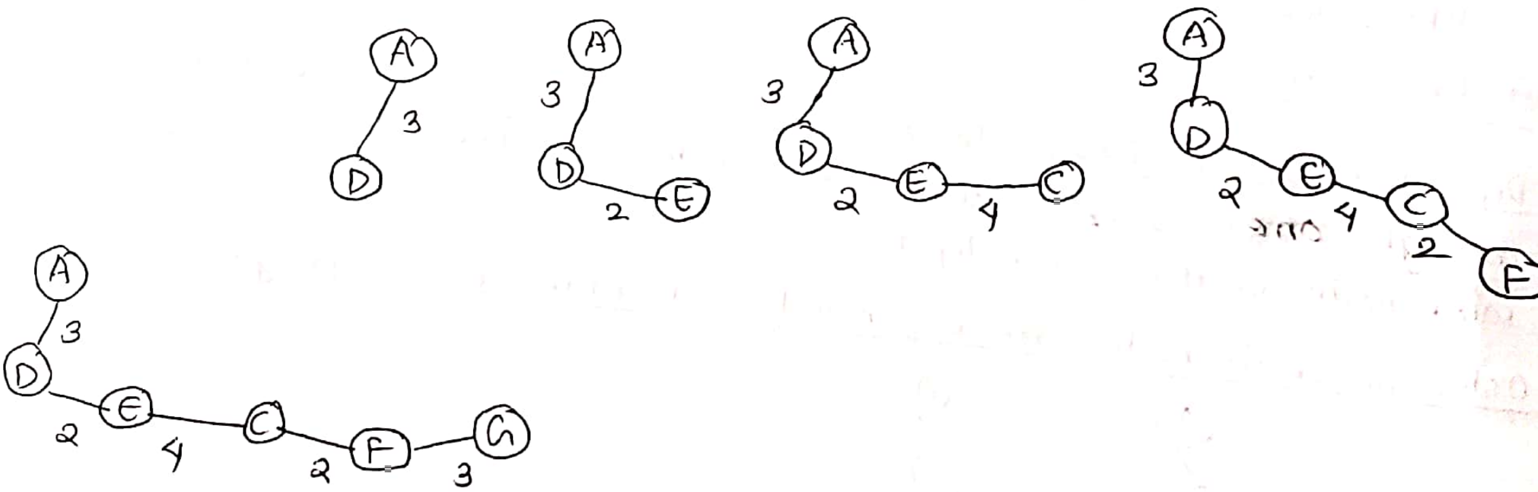
Order in which nodes are expanded and added to the fringe:

Level / Depth	Iterative Deepening Search (IDS)
0	A
1	A → D → E → B
2	A → D → G

$\boxed{\text{i.e. Iterative Deepening Search (IDS) = ADG}}$

Uniform Cost Search (UCS): In Uniform Cost Search, we find the goal node by search through the nodes having less cost price.

Orders in which nodes are expanded and added to the fringe:



i.e. UCS = A → D → E → C → F → G = ADECFG

Total Cost = 3 + 2 + 4 + 2 + 3 = 14

Task-3

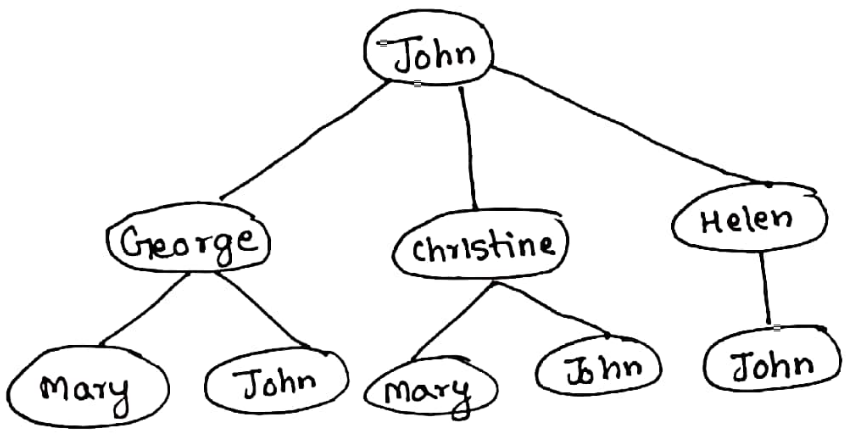
(i) a) Using BFS : Yes, we can get the expected output if the degree of branching is less.

b) Using DFS : No, we cannot get expected result because there might be an infinite interation between parent node and its first child if we do not keep track of visited nodes.

c) Using UCS : Yes, this will give the expected output.

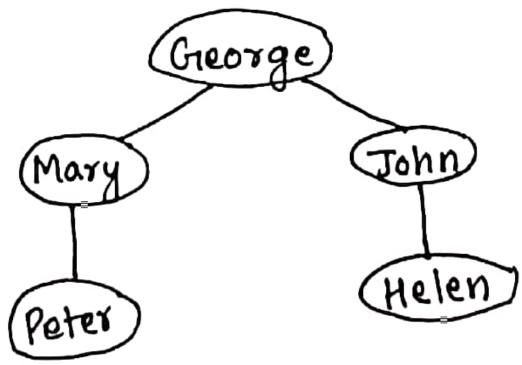
d) Using IDS : Yes, this will give the expected output if started at the large initial depth.

(ii)



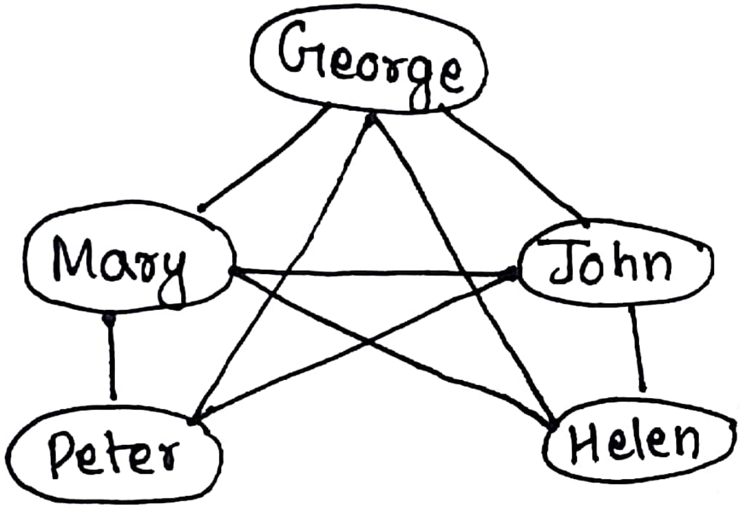
(iii) No, there is no one to one correspondence between nodes in the above search tree and the vertices in SNG. This is because one vertex "John" corresponds to multiple nodes in tree. Also, another vertex "Peter" is not a node in the tree.

(iv)



Peter and Helen has 4 degrees of separation.

(V)



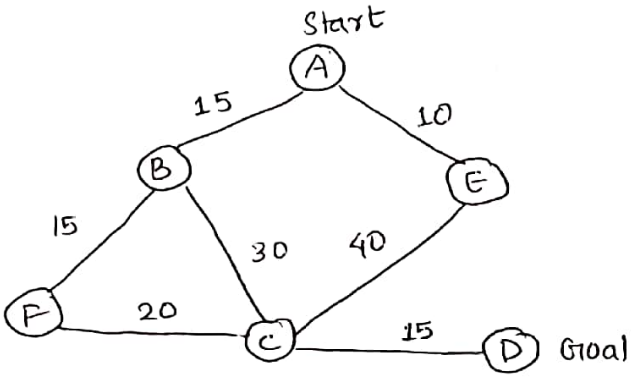
Each people have 1 degree of separation between them.

(Vi) We can make sure by following ways:

↳ Keep note of the nodes visited and expanded.

↳ we should not add the node to the fringe if it is already visited before.

Task4



Answer:

A heuristic is admissible if the values corresponding to the nodes in the heuristic should be less than or equal to the minimum cost path from the corresponding node to the destination node.
The non-admissible node is circled in the table below:

Node	Min. Cost to D	Heuristic-1	Heuristic-2	Heuristic-3	Heuristic-4
h(A)	AECD = 10+40+15 = 65	1	70	35	0
h(B)	BFCD = 15 +20+15 = 50	50	70	30	0
h(C)	CD = 15	15	70	20	0
h(D)	D = 0	0	70	5	0
h(E)	ECD = 40+15 = 55	10	70	0	0
h(F)	FBAECD = 15+15+10+40+15 = 95	0	70	30	0
Is Admissible ?		Yes	No	NO	yes

Now changing the Heuristic to make it admissible:

a) No change for Heuristic-1.

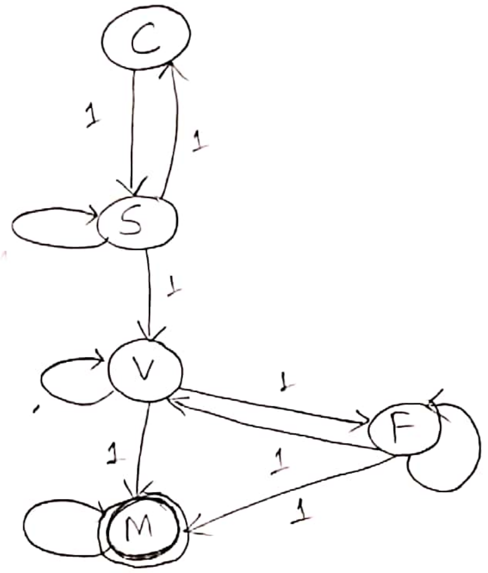
b) Heuristic-2 : h(A) = 65
h(B) = 50
h(C) = 15
h(D) = 0
h(E) = 55
h(F) = 70

c) Heuristic-3 : h(A) = 35
h(B) = 30
h(C) = 15
h(D) = 0
h(E) = 0
h(F) = 30

d) No change for Heuristic 4.

Task-5

Graph based on the information of the question:



City = C
Suburb = b
Village = V
Farm = F
Mountain = M

On the basis of graph above, we can have following value of Heuristics:

$H(C) = 3$

$H(S) = 2$

$H(V) = 1$

$H(F) = 1$

$H(M) = 0$

Task-6

For figure - 5 & 6 :

- Step cost = 1
- Euclid distance $(d) = \sqrt{(x_2-x_1)^2+(y_2-y_1)^2}$ where d is the distance between (x_1,y_1) & (x_2,y_2), which can be used to calculate heuristics.
- A* algorithm uses $f(x) = g(x) + h(x)$ where $g(x)$ is the cummulative cost and $h(x)$ is the heuristic cost, to find best route.
- Greedy algorithm finds the best path using smallest heuristics value.

In figure-5 :

Greedy search performs better than or the same as A*, depending on the start and end states is true. If start node is (0,0) and lets say our destination is (2,1). Then, path using greedy algorithm is $(0,0) \rightarrow (1,0) \rightarrow (1,1) \rightarrow (2,1)$ in which we reach our destination in 4^{th} iteration. For, A* algorithm path would be $(0,0) \rightarrow (1,0) \rightarrow (2,0) \rightarrow (2,1)$, but we reach our destination in 5^{th} iteration. The nodes in fringe in greedy would be less compared to A* algorithm.

In figure-6 :

Greedy search always performs worse than or the same as A*, depending on the start and end states. Greedy might fail in figure 6, because it might get stuck between two nodes which won't be able to give our destination node. A* algorithm would still find the destination node, depending on the start and end state