A.I

Assignment - 1

Task 2:-

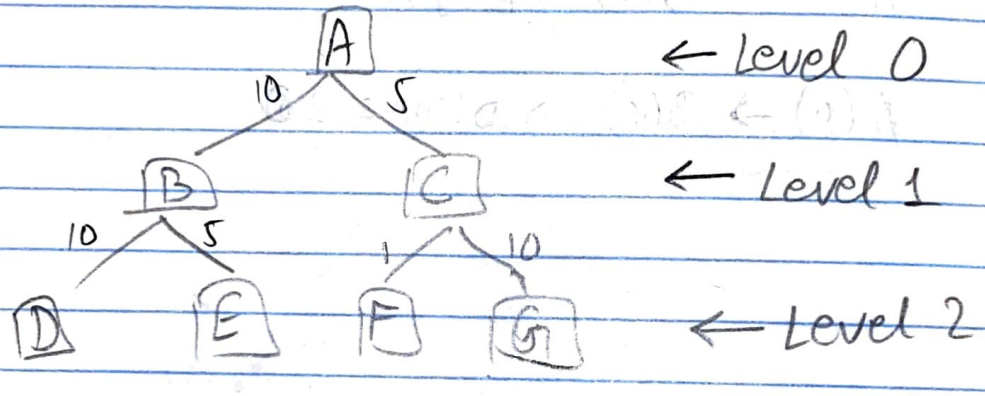
1. Breadth - first search

A, B, C, D, E, F, G

2. Depth first search

A, B, D, E, C, F, G.

3. Iterative deepening search.



Step 1:- At level 0:- A

Step 2:- At level 1 :- A, B, C

Step 3:- At level 2 :- A, B, D, E, C, F, G.

4. Uniform Cost Search:-

A(0) → C(5) → F(1) = 6.

A(0) → C(5) → F(1) → G(10) = 16

A(0) → C(5) → G(10) = 15 ✓

A(0) → B(10) → D(10) → E(5) = 25.

A(0) → B(10) → E(5) = 15

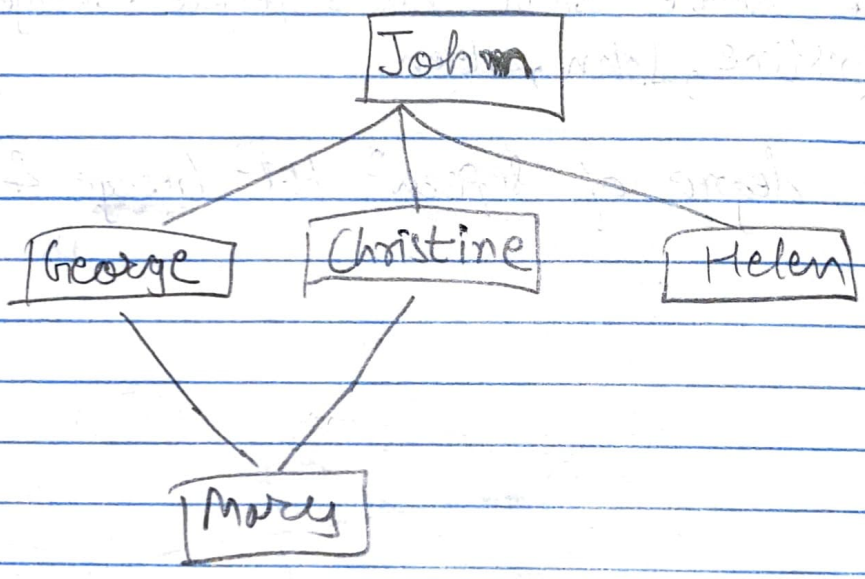
A(0) → B(10) → D(10) = 20

Task 3:-

(i). We can get correct number of following algorithms:-

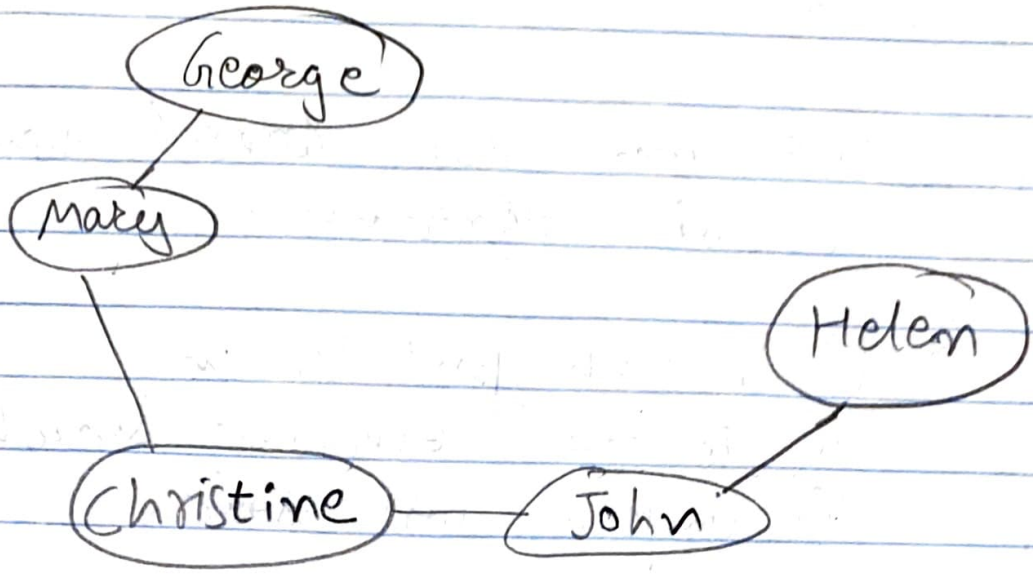
(a) (i) Broadth first search
(ii) Iterative deepening search
(iii) Uniform Cost search.

(ii)



(iii) No, the search tree nodes and SNG vertex cannot have a one to one correspondence as the tree generated from SNG will have nodes whose parents & children might repeat due to given condition that the algorithm does not try to avoid revisiting nodes. Also, there are nodes that have common connection. (I.E, If we draw tree, from perspective of George as root, Christne would be twice in the tree, once under Mary & again under John.

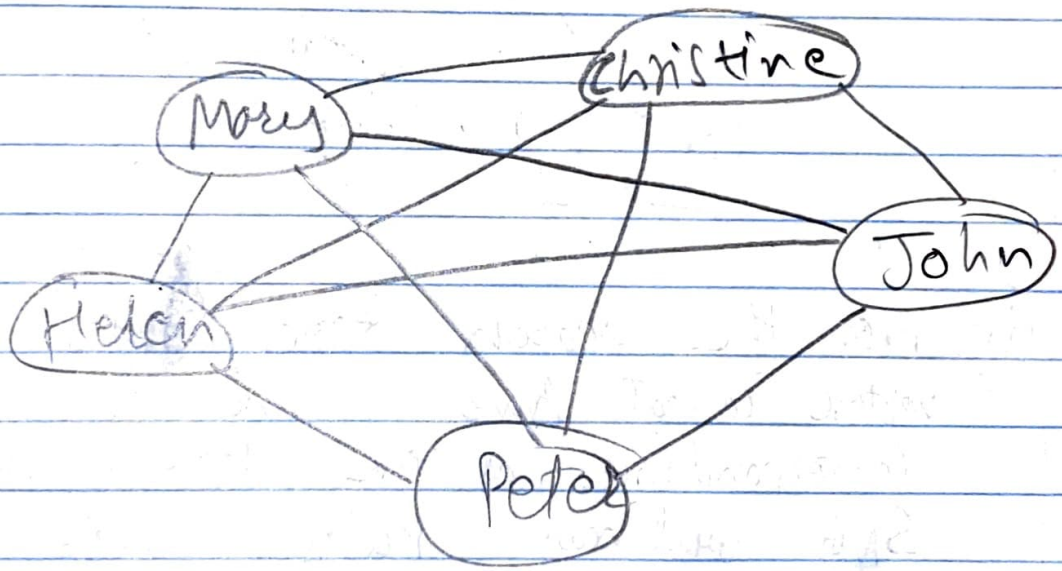
(iv).



The above SNG has 5 people : George, Mary, Christine, John, Helen.

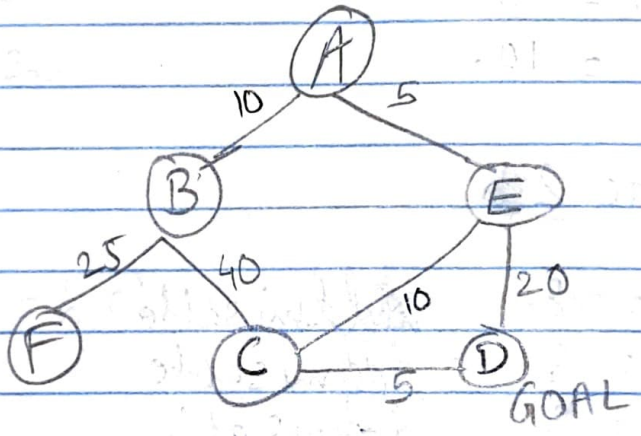
The degree of seperation bet^n George & Helen is 4.

(v).



(vi) As we know, 1 node is 1 KB & 1 million KB is 1 GB. We can modify BFS to avoid ~~re-visiting~~ re-visiting nodes as to reduce generation of duplicate nodes. This ensures that even in the worst case scenario we might explore all nodes but without creating duplicates and also wouldn't exceed the memory capacity.

Task 4:-



$h^*(n)$ for A → 20, A → E → C → D
$h^*(n)$ for B → 30, B → A → E → C → D
$h^*(n)$ for C → 5, C → D
$h^*(n)$ for D → 0, D → D
$h^*(n)$ for E → 15, E → C → D
$h^*(n)$ for F → 55, F → B → A → E → C → D

Heuristic 1 : If $h(n) \leq h^*(n)$, addmissible.

$h(A) = 5$ Addmisable. 20.
$h(B) = 40$ not Addmisable 30
$h(C) = 10$ not Addmisable 5.
$h(D) = 0$ Addmisable
$h(E) = 10$ Addmisable
$h(F) = 0$ Addmisable

$\therefore h(B)$ & $h(C)$ = not Addmisable. So we can replace their values.

$\therefore h^*(B) = 40$
$h^*(C) = 10.$

Heuristic 2 :-

$h(A) = 8$ Addmisable.
$h(B) = 5$ Addmisable
$h(C) = 3$ Addmisable
$h(D) = 5$ not Addmisable
$h(E) = 5$ Addmisable
$h(F) = 0$ Addmisable.

Here, $h(D)$ is not Addmisable. So we can replace its value.

$h^*(D) = 5$

Heuristic 3 :-

$h(A) = 35$		Not Admisable
$h(B) = 30$		Admisable
$h(C) = 20$		Not Admisable
$h(D) = 0$		Admisable
$h(E) = 0$		Admisable
$h(F) = 50$		Admisable

Here, $h(A)$ & $h(C)$ are not Admisable.

$h^*(A) = 35$
$h^*(C) = 20$

Heuristic 4 :-

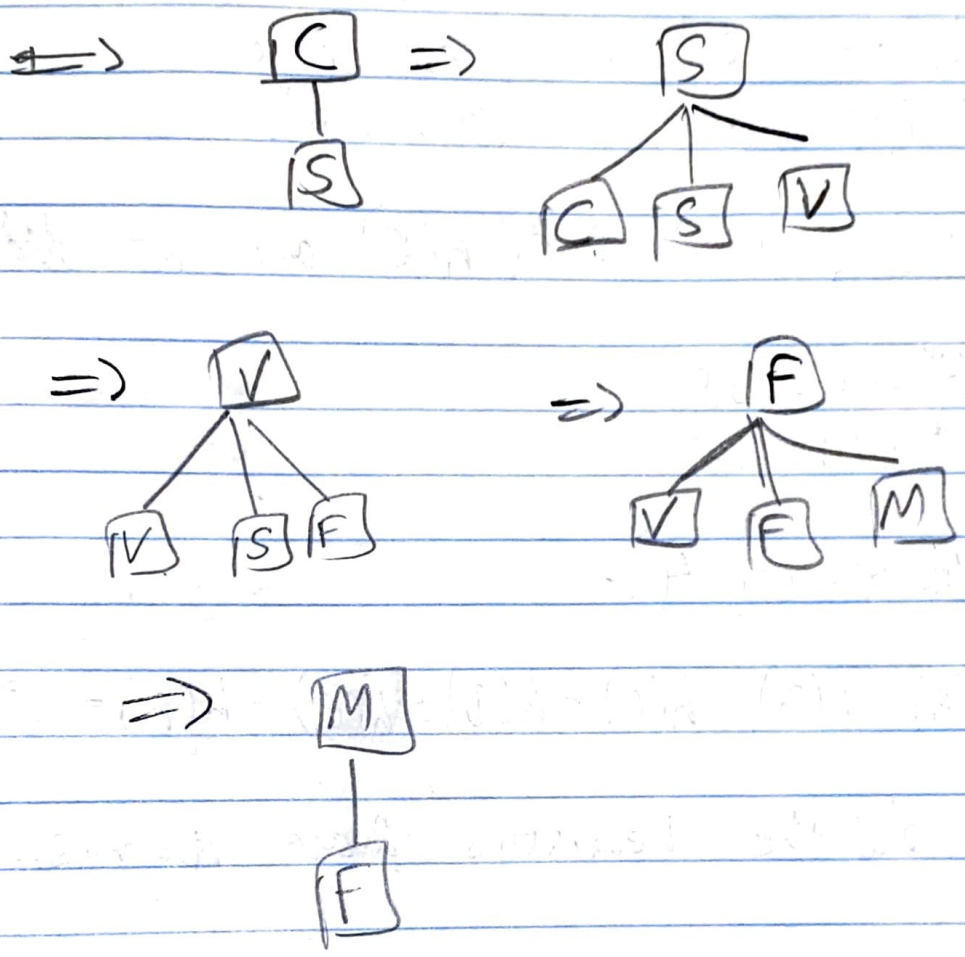
$h(A) = h(B) = h(C) = h(D) = h(E) = h(F) = 0$.

∴ All the heuristics are Admisable.

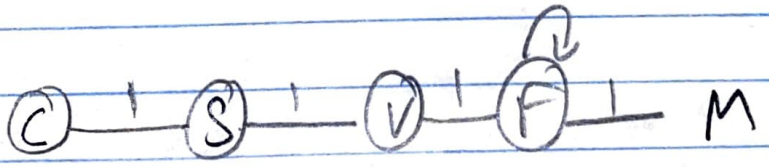
Task 5 :-

→ Given :-

City = C Village = V Mountain = M.
Suburb = S. Farmland = F.



⇒ According to the given statements :-



⇒ Therefore :-

h(n) :-

h(c) = 4, h(v) = 2. h(m) = 0.
h(s) = 3, h(f) = 1

∴ thus h(m) is goal state.

Task 6 :-

→ Here, the greedy search algorithm is represented as h(n) which is Euclidian distance.

⇒ A* algorithm is represented as f(n).

∴ $f(n) = g(n) + h(n)$

· where, h(n) is euclidian distance and g(n) is minimum distance between nodes.

↳ From, figure 5, we assume that it starts from (5,0) and ends on (5,3).

∴ Heuristic values of following items :-

h(5,0) = 6.9
h(4,0) = 6.8
h(6,0) = 6.7
h(5,1) = 6.6
h(4,1) = 6.5
h(6,1) = 6.4
h(5,2) = 6.3
h(4,2) = 6.2
h(6,2) = 6.1
h(5,3) = 0.

→ Greedy Algorithm:- Start node (5,0) it has 3 ways from which (5,1) is shortest value where (5,1) = 6.6.

Now, it has four options & it will select (5,2) or its shortest h(5,2) = 6.3

Similarly, now it will select the shortest path having the shortest value.

where h(5,3) = 0. which is goal node.

⇒ A* algorithm:- It works similarly to greedy algorithm and also it will follow the same path & result will also be same as greedy algo.

→ Here, we can say that greedy search algo performs better or same as A* algorithm.

→ Solution for fig 6:-

Considering, Start node = (3,1)
End node (3,2)

$h(3,1) = 1$
$h(3,0) = 2$
$h(2,0) = 2.23$
$h(4,0) = 2.23$

Considering (3,1) as start node, in greedy search, the algo will have just one option. (i.e) $h(3,0)$ with value 2.

In iteration 2, the node (3,0) will have three options namely n(2,0), h(4,0) and (3,0) with values of (2.23, 2.23 & 1).

The algorithm will select h(3,1) and go there.

→ A* algorithm will find f(n) (i.e).

$f(n) = h(n) + g(n)$.

Here, $f(3,1) = 1 + 0$
$= 1$.

$f(3,0) = 2+1 = 3$.
$f(4,0) = 2.23 + 2 = 4.23$
$f(4,1) = 1.4 + 3 = 4.4$.
$f(4,2) = 1+4 = 5$
$f(3.2) = 0+5 = 5$

The A* algorithm will follow the following path $(3,1) \rightarrow (3,0) \rightarrow (4,0) \rightarrow (4,1) \rightarrow (4,2) \rightarrow (3,2)$ as it has the lowest f(n) value.

At (4,1) for Eg. the algo has 3 options. (4,0) (5,1) & (4,2) with values 4.23. (2.23+4) = 6.23. and 5.

Here, it will not go back to (4,0) as it has already visited it. out of (5,1) and (4,2) it will go to (4,2) as it has the lower $g(n)$ value.

∴ Greedy search always performs worse than or same as A* depending on start & end state.