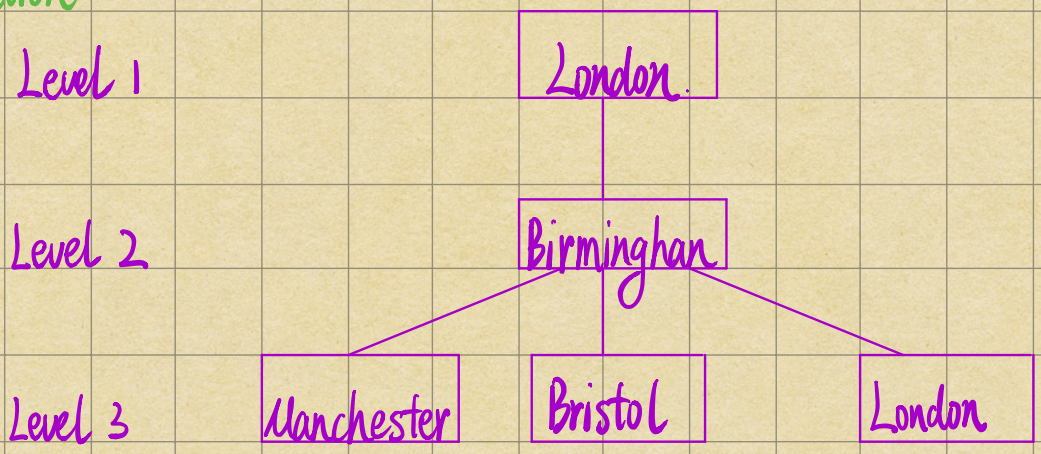
Task 2:

1° Draw the first three levels of the search tree starting from London.

Solution:



2° Solution:

1' Breadth-First Search:

(Dresden), (Leipzig, Berlin), (Magdeburg, Nuremburg)

level 1 level 2 level 3

2' Depth-First Search:

Dresden, Berlin, Hamburg, Luebeck, Bremen.

level 1 level 2 level 3 level 4 level 4

3' Iterative Deepening Search:

Limit = 0 : Dresen

Limit = 1 : Dresen, Leipzig, Berlin

Limit = 2 : Dresen, Leipzig, Nuremburg, Magdeburg, Berlin.

Limit = 3 : Dresen, Leipzig, Nuremburg, Munich, Struttgart.

Limit = 4 : Dresen, Leipzig, Nuremburg, Munich, Struttgart.

4' Uniform-cost Search: city (cost)

Dresen (0), Leipzip (119), Berlin (204), Magdeburg (119+125=244)

Nuremburg (119+263=382)

Task 3:

1° Solution:

BFS (Breadth-First Search) and UCS (Uniform-cost Search) and IDS (Iterative deepen search). can guarantee finding the correct number required. Because those three methods are complete according to the textbook. In addition: UCS (Uniform cost search) is applicable as long as we consider the "cost" among the nodes are iterative steps.

2° Solution:

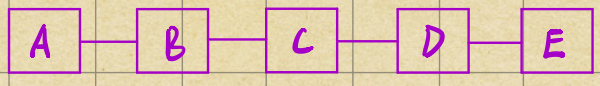
It depends on the levels of the search tree and the starting point.

1' If we have no limitation of the levels of tree, we can draw that tree because the the graph is a connected graph

2' We can not draw a tree when { the levels of the search tree is 3
starting point is Mary or John. or Helen or Peter

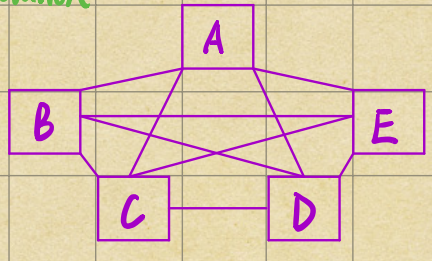
In other words, only Christine and George is available when the max level of search tree is set to 3.

3° Solution:



The degree of A and E is 4

4° Solution:



5° Solution:

We can just pop out the visited nodes that can form a loop to save memory.
Because the degrees in the sub-connected graph are fixed, so we can delete them safely.

$h^*(A)=35$ $h^*(B)=20$ $h^*(C)=10$ $h^*(D)=0$ $h^*(E)=30$ $h^*(F)=40$

Task 4:

Overall, the h^* of each node is

$h^*(A)=17$, $h^*(B)=14$, $h^*(C)=10$, $h^*(D)=12$,

$h^*(E)=7$, $h^*(F)=4$, $h^*(G)=0$, we need to make sure $h(n) \leq h^*(n)$, as a result

1° Heuristic 1:

Inadmissable, modify $h(A), h(B), h(F), h(G)$

$h(A)=17$, $h(B)=14$, $h(F)=7$, $h(G)=0$

2° Heuristic 2:

Inadmissable, modify $h(A), h(B), h(C), h(D), h(E), h(F), h(G)$

$h(A)=17$, $h(B)=14$, $h(C)=10$, $h(D)=12$, $h(E)=7$, $h(F)=4$, $h(G)=0$

3° Heuristic 3:

Inadmissable, modify $h(G)$

$h(G)=0$

4° Heuristic 4:

Admissable

5° Heuristic 5:

Admissable

Task 5: Solution

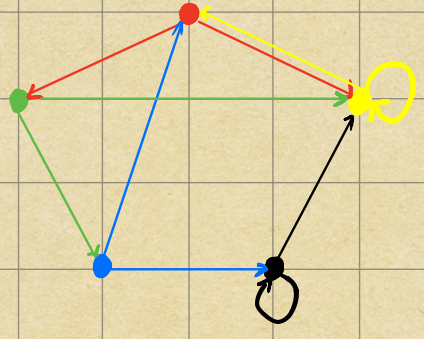
$h(red)=3$

$h(green)=2$

$h(blue)=1$

$h(yellow)=4$

$h(black)=0$



Task 6 : Solution

(a): According to the problem, the lower bounds of the required memory is 100 KB

∴ No method can guarantee that never use more than 100 KB.

(b) Consider the space complexity of each algorithm:

Breadth-first search: keep every node in memory $O(b^{d+1})$

Uniform-cost search: keep nodes with $g \leq$ cost of optimal solution. $O(b^{\lceil C^*/\varepsilon \rceil})$

Depth-first Search: linear space $O(bm)$

Iterative deepening search: linear space $O(bd)$

b: maxium braning factor

d: depth of least-cost

m: maxium depth of state.

Because all initial states, the shortest solution is at most 208.

∴ $b=4$, $m=\log_4 208=4$

We found that DFS and IDS

which have linear space complexity can be used.