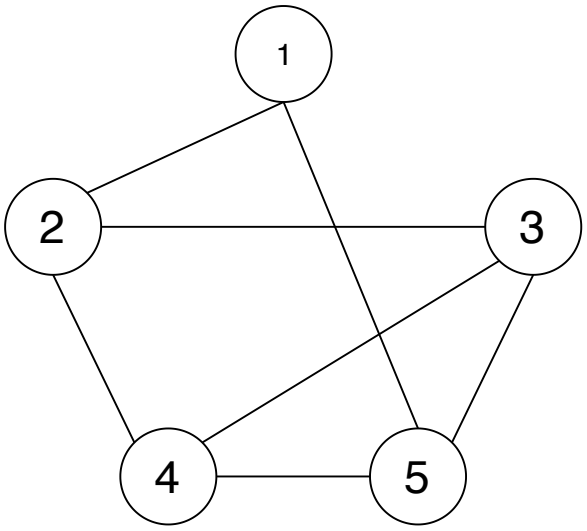
學習歷程

Breadth-First Search 是以佇列及遞迴的技巧來運算，從圖形的某個點開始，被經過的點就做上記號，接著走到的頂點所有相鄰且未經過頂點中的任意一個頂點，並做上記號，再以該點為新的起點繼續進行運算。



以頂點1為起點,與頂點1相鄰且未拜訪過的頂點2及頂點5放入佇列。

2	5			

取出頂點2,將與頂點2相鄰且未拜訪過的頂點3及頂點4放入佇列。

5	3	4		

取出頂點5,將與頂點5相鄰且未拜訪過的頂點3及頂點4放入佇列。

3	4	3	4	

取出頂點3,將與頂點3相鄰且未拜訪過的頂點4放入佇列。

4	3	3	4	

取出頂點4,將與頂點4相鄰且未拜訪過的頂點放入佇列中,各位可以發現與頂點4相鄰的頂點全部被拜訪過,所以無需再放入佇列中。

3	4	2	4	

將佇列內的值取出並判斷是否已

將佇列內的值取出並判斷是否已經走訪過了,直到佇列內無節點可走訪為止。

1	2	5	3	4

BFS & DFS的比較

BFS

廣度優先，起點的每個點都走訪一次，再換起點繼續走訪
時間複雜度為O(V+E)，空間複雜度為O(V)，V為圖的頂點數，E為邊數
優點:找出目標最小的步驟，會方便許多
缺點:占用記憶體太多，沒有目標式地搜尋

DFS

深度優先，一直往深處走，直到沒路，才往回找分岔點
時間複雜度為O(V+E)，空間複雜度為O(V)，V為圖的頂點數，E為邊數
優點:對於深度低的狀態空間來說，不但相當簡潔，所占用的空間記憶體也較少，執行上的效率也跟著快
缺點:如果深度過深的話，對電腦的負荷也相當的大。

參考資料

https://docs.google.com/presentation/d/e/2PACX-1vSYJYXUXvGAeTZ5fknxj_-EPm3zxgy4ITd
lmrXzy63Y-iZgs8uwVNmOaZlnx9fUNzsbo8kphvMTa0c4/pub?start=false&loop=false&delayms=3000&slide=id.g7a5d8b85ee_0_0
https://www.shs.edu.tw/works/essay/2017/03/2017033023453259.pdf
https://magiclen.org/dfs-bfs/
https://alrightchiu.github.io/SecondRound/graph-breadth-first-searchbfsguang-du-you-xian-sou-xun.html#code
與同學私下討論概念