108321020 黃證瑋
108321033 吳明騰

- 程式說明
  - Bellman-ford
    - 程式一開始先讓使用者輸入節點數和路徑數，接著再輸入路徑(ex:3 5 3 節點 3 到節點 5 cost 3)，再來將節點 1 設為起點(cost 0)，節點 1 到其他節點的距離設為 inf(999999)，接著進入迴圈後先將現在起點到各節點的 cost 狀態 copy 下來，再對每條路徑進行比較(ex:3 5 3 ,如果起點到節點 3 再加 cost 3 到 5 的距離小於起點到節點 5 的話就更新路線)，之後再檢查更新過後的路徑是否與一開始 copy 的狀態相同，如果一樣就跳出迴圈不一樣就繼續，最後再顯示最短路徑結果。
  - Dijkstra
    - 輸入每個 node 的 vertices 與哪幾個 node 連接，如果有相連就是 1 否則為 0，以及 cost 如果沒有相連就記為 0，接著向外尋找最短路徑，每次尋找都以最少的 cost，如果已有其他更小的 cost 就更新其路線，直到所有的 node 都有相連，最後在顯示最短路徑的結果。
- 執行結果與說明
  - Bellman-ford
    - 輸出結果為設節點 1 為起點(cost 0)再顯示出起點到各節點的最少 cost(最短路徑)。
  - Dijkstra
    - 輸入一 topology
    - 輸出 a 到其他節點的最短路徑，和它的 routing table
- 遭遇的困難與解決方法
  - Bellman-ford
    - 一開始不太清楚 Bellman-ford 跟 Dijkstra 的差別，上網查詢才明白前者是每次都從起點開始演算，後者是從起點向外擴散。
  - Dijkstra
    - 輸出的結果要什方式存取或輸出?
      - 我們模仿 routing table 的存取方式，把 Destination 、Cost 、Next Router 存取下來
      - 直接輸出 a 到其他節點的最短路徑，這比較方便人們了解。