**PA3 report**

1. **Cycle breaking實作所使用的資料結構與演算法**
2. **讀檔存edges與vertices資訊：**在程式中我設計了三個自定義的class，分別是Graph、vertices與edges這三個，在Graph中的data member與function主要就是負責實作整張圖的演算法還有一些helper function，vertices與edges主要就是存一些實作演算法時會用到的重要資訊，例如predecessor、adjacent list、weight等。在讀取input file的時候，我會根據讀進來的是undirected graph或是directed graph去拆分成兩個不同的case去實作(詳細方法在後面)，但讀取時紀錄資訊的方法都是先將所有vertices存進Graph中，edges的關係與權重也先存到graph中但先不要接到vertices上(也就是先讓所有vertex 的adjacent list為空)，以便後面的演算法實作。
3. **找到cycles breaking problem的solution所使用的演算法：**從這裡開始undirected graph 與directed graph的方法會不一樣，主要是針對這兩種圖各自的特性去採取相對應的措施，以下方開討論。

**Undirected graph：**

Undirected 的部分我所使用的演算法是Prim’s algorithm，因為要找到一個total weight最大acyclic graph其實就是找到這個graph的maximum spinning tree，所以這邊我藉由在一開始所存的資訊，將所有vertices的adjacent list填上，接著自己修改了PA1所寫過的MaxHeapify、HeapExtractMax與HeapIncreaseKey這些helper function，一開始先將所有的vertices都存進MaxHeap裡面，把v.d當成是MaxHeap的key，一開始初始化時先將除了source之外所有vertex的d這個property設為-101，將source的d設為0。接著，在while迴圈中每次都extract max一次將v.d最大的那個vertex取出MaxHeap，然後看這個vertex的adjacent list，藉由HeapIncreaseKey更新這些vertex的d。此外，加入一個判斷條件，若是這時該vertex已不在MaxHeap中(已經在MST中)，則代表會形成一個cycle，這時不把這個edge放進MST之餘，就把這個edge記錄下來，最後存在這個vector中的所有edges與total cost就是我們所要的答案。

**Directed graph：**

Directed 的部分我所使用的演算法是Depth first search，一開始我也是先引用了PA1所寫過的QuickSort，把所有的edges先全部由大排到小sort過一次，然後再根據這個順序一條一條edges插入到graph之中，假設這時要插入的edge是(u,v)，就先對v做dfs\_visit()，看v能不能走到u，若是v可以走到u，代表當(u,v)這條edges被插入到graph當中的話，就會形成cycle，因此就不能把這條插入到graph之中，反而應該把這條edge記錄下來，因為它就是其中一條breaking cycles的edge，若是v不能走到u，代表當(u,v)這條edges被插入到graph當中的話不會形成cycle。最後當每一條edges照順序被檢查能否插入過後，就能得到我們所要的答案。

1. **Cycle breaking實作時的發現**

在這次的PA3中我覺得比較有趣的發現是與同組的同學答案有些微的不同，經討論過後發現是因為Quicksort不是一個unstable的sorter，而且Partition的時候是randomize去取要與其他資料比對的值，所以同樣weight的edges排列順序就會與本來input的順序不同，而且每次跑的時候順序也都不同，所以會造成每個graph會固定出現幾個cost，不過這些答案都是符合acyclic的條件。舉例來說，在public\_cace\_7.in這個case的結果就會有-576、-1860、-745、-1485這些total cost的結果，而同組同學用stable sorter去做只會得出-576這個結果。除此之外，在public\_cace\_8.in這個case的結果有-27837、-27756、-27606、-27785、-27932、-27459這些結果，而同組同學只會得出-27606這個結果。由此而猜測應為Quicksort randomization的關係。