A. 自己寫的 KMeans 演算法、用現成的 KMeans 函數寫的皆要做程式碼說明，說明得愈完整、愈清楚，分數愈高：
    a. 自己寫的 KMeans 演算法：
        i. 引入必要函式庫：引入 sklearn.datasets、numpy 和 matplotlib.pyplot 庫。分別用於生成數據集、數學運算及繪製圖形。
        ii. 使用 sklearn.datasets 的 make_blobs 函數建立資料：使用 make_blobs 函數生成一個具有 250 個樣本、2 個特徵、4 個中心的數據集。數據集的每個樣本都有兩個特徵，並以四群來建立（非真實分群）。
        iii. 實作 KMeans 演算法：首先隨機初始化群中心點，然後計算每個點到各群中心點的距離，並將每個點指派到最近群中心點。接著重新計算群中心點。如果群中心點沒有變化，則結束迴圈。
        iv. 繪製分群前的資料點散圖：使用 matplotlib.pyplot 的 scatter 函數繪製分群前的資料點散圖。在這散點圖中所有資料點都被繪製成藍色。
        v. 繪製分群後的資料點散圖：使用 matplotlib.pyplot 的 scatter 函數繪製分群後的資料點散圖。在這散點圖中每個群組的點都被繪製成不同的顏色，並且群中心點被繪製成紅色。
    b. 用現成的 KMeans 函數：
        i. 引入必要函式庫：引入 sklearn.datasets、sklearn.cluster、numpy 和 matplotlib.pyplot 函式庫。分別用於生成數據集、進行 KMeans 聚類、進行數學運算以及繪製圖形。
        ii. 使用 sklearn.datasets 的 make_blobs 函數建立資料：使用 make_blobs 函數生成一個具有 250 個樣本、2 個特徵、4 個中心的數據集。數據集的每個樣本都有兩個特徵，並以四群來建立（非真實分群）。參數設定為：
            1. n_samples=250：固定建立 250 個資料
            2. n_features=2：固定每個資料都有兩個特徵值
            3. centers=4：固定以 4 個中心來建立資料（非真實分群）
            4. shuffle=True：固定設為 True
            5. cluster_std=1.5：自行調整，滿足 >= 1。
            6. random_state=114514：自行調整
        iii. 使用 sklearn.cluster 的 KMeans 函數：使用 KMeans 函數進行聚類。該函數參數包括：
            1. n_clusters=centers：指定要將數據集分為 4 個群組。
            2. init='random'：指定要隨機初始化群中心點。
            3. n_init=10：指定運行 10 次 KMeans 演算法，每次都使用不同的群中心點初始化。最終選擇最佳的一次結果（即群內平方和最小的一次）。
            4. max_iter=300：指定每次運行 KMeans 演算法的最大迭代次數為 300。如果在 300 次迭代之內，群中心點已經不再變化，則提前結束。

5. tol=1e-04：指定群中心點變化的容忍度。如果群中心點的變化小於這個值，則認為群中心點已經不再變化。
6. random_state=0：指定隨機數生成器的種子，以確保每次運行程式碼時都能得到相同的結果。

iv. 繪製分群前的資料點散圖：使用 matplotlib.pyplot 的 scatter 函數繪製分群前的資料點散圖。在這散點圖中所有資料點都被繪製成藍色。

v. 繪製分群後的資料點散圖：使用 matplotlib.pyplot 的 scatter 函數繪製分群後的資料點散圖。在這散點圖中每個群組的點都被繪製成不同的顏色，並且群中心點被繪製成紅色。

B. 現成的 KMeans 函數請針對參數調整的原因或結果比較做說明即可，例： 為何 max_iter 設為 400，說明愈清楚分數愈高：
在 KMeans 演算法中，max_iter 參數控制算法的最大迭代次數。每一次迭代，算法都會重新計算群中心點，並將每個數據點指派到最近的群中心點。這個過程會一直重複，直到達到最大迭代次數，或者群中心點的變化小於一個預設的閾值。
max_iter 參數的設定需要在計算效率和算法性能之間取得平衡。如果 max_iter 的值設定得太大，例如 400 或更高，那麼算法可能需要花費更長時間來達到收斂，尤其是在數據集很大或者群數很多的情況下。然而，如果 max_iter 的值設定得太小，例如 100 或更低，那麼算法可能在達到最佳解之前就已經停止了。
我將 max_iter 設為 300，300 是一個相對保守的選擇，可確保演算法在合理時間內完成，同時也有足夠的機會找到好的結果。

C. 自己寫的 KMeans 演算法、用現成的 KMeans 函數寫的皆需要附上分群前的資料點散圖 (如 Fig. 1.) 以及分群後的資料點散圖 (如 Fig. 2.)，總共要附上 4 張圖。請標明清楚群中心點、每個點屬於哪一群 (不同群請以不同顏色標示)，圖旁邊需附上圖示，標明特徵空間座標， 且須標明清楚圖片是屬於哪一個程式版本的圖：

a. 自己寫的 KMeans 演算法：

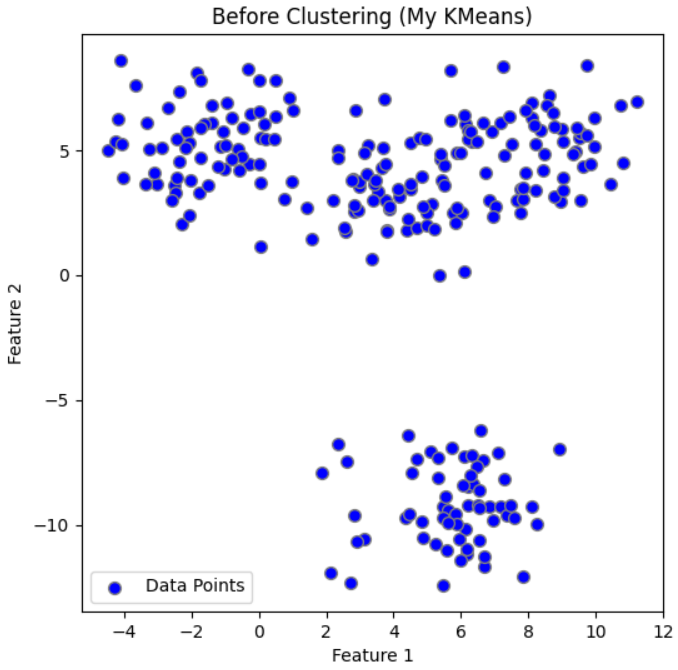

Fig. 1. 上圖為分群前的圖

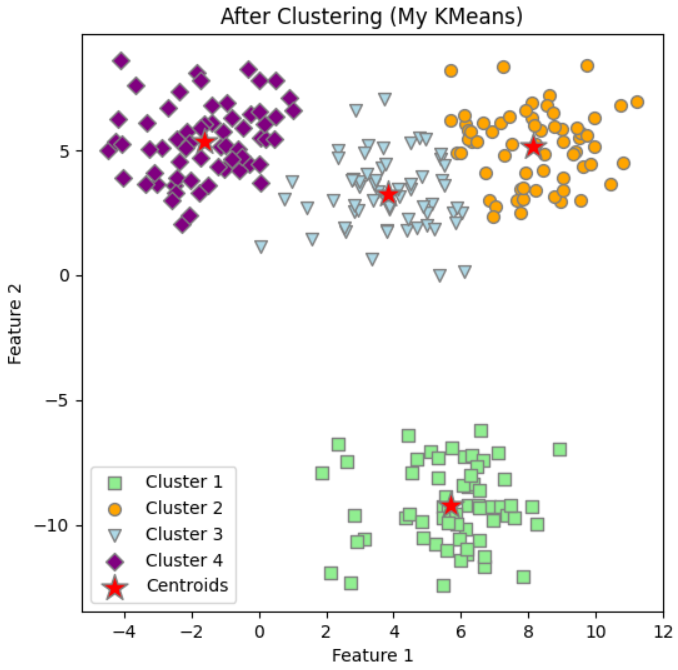

Fig. 2. 上圖為分群後的圖

b. 用現成的 KMeans 函數：

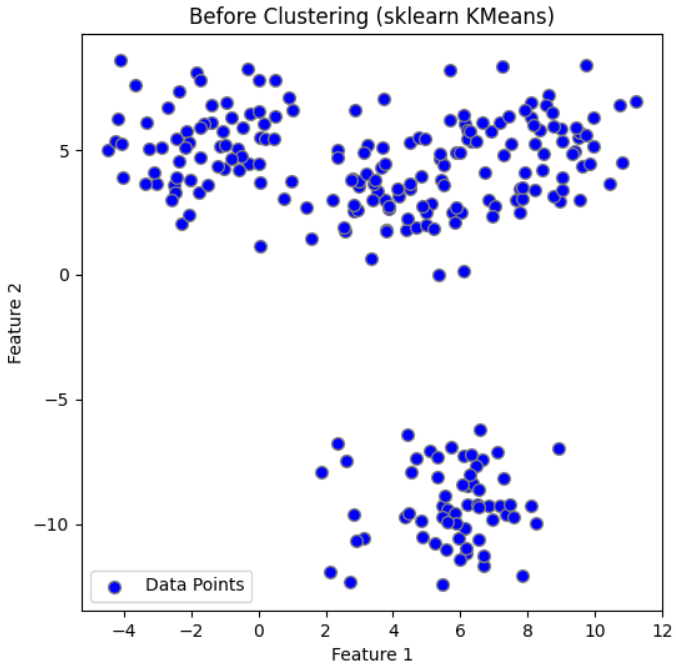

Fig. 1. 上圖為分群前的圖

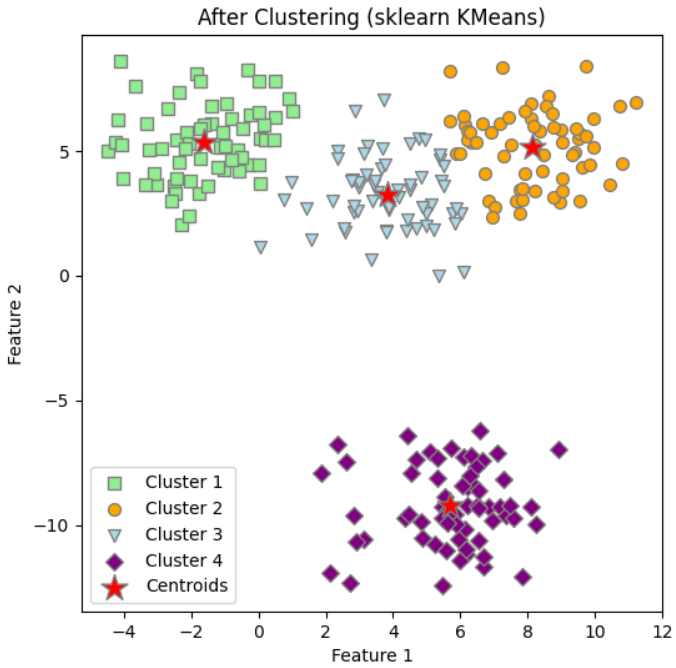

Fig. 2. 上圖為分群後的圖