BAN 210: Workshop 4

In [76]:

```
#Import Libraries
import numpy as np
import matplotlib.pyplot as plt
import random
```

In [77]:

```
# Import Dataset
X = np.array([[1, 2], [1, 4], [2, 3], [5, 7], [6, 8], [7, 9]])
```

In [78]:

```
# Initialize centroids randomly
k = 2

initial_indices = random.sample(range(len(X)), k)
centroids = X[initial_indices]

print("Initial Centroids:")
print(centroids)
```

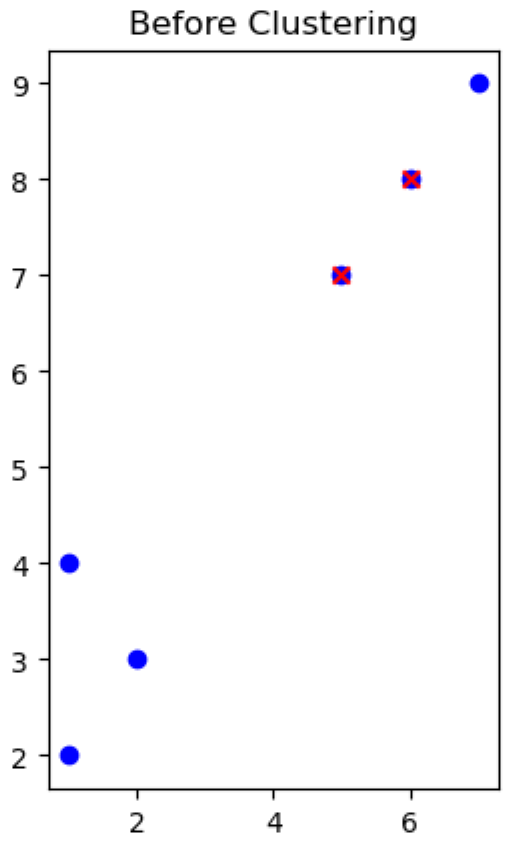
```
Initial Centroids:
[[5 7]
 [6 8]]
```

In [79]:

```
# Plot the clusters before clustering
plt.subplot(1, 2, 1)
plt.scatter(X[:, 0], X[:, 1], color='blue') # plot orginal data points in blue
plt.scatter(centroids[:, 0], centroids[:, 1], color='red', marker='x') # initial centroids plot in red x
plt.title("Before Clustering")
```

Out[79]:

```
Text(0.5, 1.0, 'Before Clustering')
```

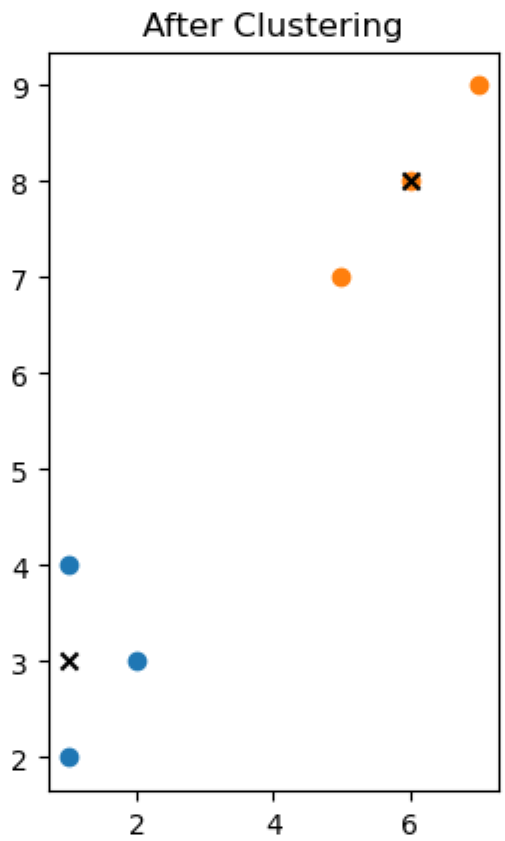


In [80]:

```
# Run the algorithm for 5 iterations
max_iter = 5

# Compute the Euclidean distance from each point to c

for i in range(max_iter):
    print("Iteration", i + 1)
    distances = []
    for idx, c in enumerate(centroids):
        d = np.linalg.norm(X - c, axis=1)
        distances.append(d)
        print(f"Distances from centroid {idx} ({c}): {d}")

# Assign Each Point to the Nearest Centroid
    distances = np.array(distances).T
    labels = np.argmin(distances, axis=1)

# Update the centroids to be the mean of all points assigned to it
    for j in range(k):
        cluster_points = X[labels == j]
        if len(cluster_points) > 0:
            new_centroid = np.mean(cluster_points, axis=0)
            print("  Cluster", j + 1, "points:", cluster_points.tolist())
            print("  New centroid", j + 1, ":", new_centroid.tolist())
            centroids[j] = new_centroid

    print()
```

```
Iteration 1
Distances from centroid 0 ([5 7]): [6.40312424 5.         5.         0.         1.41421356 2.82842712]
Distances from centroid 1 ([6 8]): [7.81024968 6.40312424 6.40312424 1.41421356 0.         1.41421356]
  Cluster 1 points: [[1, 2], [1, 4], [2, 3], [5, 7]]
  New centroid 1 : [2.25, 4.0]
  Cluster 2 points: [[6, 8], [7, 9]]
  New centroid 2 : [6.5, 8.5]

Iteration 2
Distances from centroid 0 ([2 4]): [2.23606798 1.         1.         4.24264069 5.65685425 7.07106781]
Distances from centroid 1 ([6 8]): [7.81024968 6.40312424 6.40312424 1.41421356 0.         1.41421356]
  Cluster 1 points: [[1, 2], [1, 4], [2, 3]]
  New centroid 1 : [1.3333333333333333, 3.0]
  Cluster 2 points: [[5, 7], [6, 8], [7, 9]]
  New centroid 2 : [6.0, 8.0]

Iteration 3
Distances from centroid 0 ([1 3]): [1.         1.         1.         5.65685425 7.07106781 8.48528137]
Distances from centroid 1 ([6 8]): [7.81024968 6.40312424 6.40312424 1.41421356 0.         1.41421356]
  Cluster 1 points: [[1, 2], [1, 4], [2, 3]]
  New centroid 1 : [1.3333333333333333, 3.0]
  Cluster 2 points: [[5, 7], [6, 8], [7, 9]]
  New centroid 2 : [6.0, 8.0]

Iteration 4
Distances from centroid 0 ([1 3]): [1.         1.         1.         5.65685425 7.07106781 8.48528137]
Distances from centroid 1 ([6 8]): [7.81024968 6.40312424 6.40312424 1.41421356 0.         1.41421356]
  Cluster 1 points: [[1, 2], [1, 4], [2, 3]]
  New centroid 1 : [1.3333333333333333, 3.0]
  Cluster 2 points: [[5, 7], [6, 8], [7, 9]]
  New centroid 2 : [6.0, 8.0]

Iteration 5
Distances from centroid 0 ([1 3]): [1.         1.         1.         5.65685425 7.07106781 8.48528137]
Distances from centroid 1 ([6 8]): [7.81024968 6.40312424 6.40312424 1.41421356 0.         1.41421356]
  Cluster 1 points: [[1, 2], [1, 4], [2, 3]]
  New centroid 1 : [1.3333333333333333, 3.0]
  Cluster 2 points: [[5, 7], [6, 8], [7, 9]]
  New centroid 2 : [6.0, 8.0]
```

```
In [81]: # Plot the clusters after clustering
plt.subplot(1, 2, 2)
for j in range(k):
    plt.scatter(X[labels == j, 0], X[labels == j, 1])
plt.scatter(centroids[:, 0], centroids[:, 1], color='black', marker='x')
plt.title("After Clustering")
plt.show()

print("Final Centroids:")
print(centroids)
```


After Clustering

```
Final Centroids:
[[1 3]
 [6 8]]
```

Centroid Initialization

The initial centroids were selected randomly from the existing data points, as required.

This ensures that each run may start with different centroid positions.

Iteration Explanation

The K-Means algorithm was run for 5 full iterations.

In each iteration, points were reassigned to the nearest centroid and the centroids were updated accordingly.

This process was repeated exactly five times as required by the task.