**Practical No 9 :- Visualize the result of the clustering and compare.**

**Name : Radhika Palve Roll No : 118**

**Batch : B4 Date :**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

import numpy as np  
import matplotlib.pyplot as plt  
from sklearn.datasets import make\_blobs  
from sklearn.cluster import KMeans, DBSCAN  
  
# Generate synthetic dataset  
X, y = make\_blobs(n\_samples=300, centers=3, cluster\_std=1.0, random\_state=42)  
  
# Apply K-Means clustering  
kmeans = KMeans(n\_clusters=3, random\_state=42)  
kmeans\_labels = kmeans.fit\_predict(X)  
  
# Apply DBSCAN clustering  
dbscan = DBSCAN(eps=0.8, min\_samples=5)  
dbscan\_labels = dbscan.fit\_predict(X)  
  
# Plot the results  
fig, axes = plt.subplots(1, 2, figsize=(12, 5))  
  
# K-Means Clustering  
axes[0].scatter(X[:, 0], X[:, 1], c=kmeans\_labels, cmap='viridis', marker='o', edgecolor='k')  
axes[0].scatter(kmeans.cluster\_centers\_[:, 0], kmeans.cluster\_centers\_[:, 1], c='red', marker='X', s=200, label="Centroids")  
axes[0].set\_title("K-Means Clustering")  
axes[0].legend()  
  
# DBSCAN Clustering  
axes[1].scatter(X[:, 0], X[:, 1], c=dbscan\_labels, cmap='viridis', marker='o', edgecolor='k')  
axes[1].set\_title("DBSCAN Clustering")  
  
plt.show()

**Output**:

