In [2]: `pip install pygad`

```
Collecting pygad
  Downloading pygad-3.0.1-py3-none-any.whl (67 kB)
                                            0.0/68.0 kB ? eta -:--:--
     ----------------------               41.0/68.0 kB 653.6 kB/s eta 0:00:01
     ------------------------------------ 68.0/68.0 kB 739.9 kB/s eta 0:00:00
Collecting cloudpickle (from pygad)
  Downloading cloudpickle-2.2.1-py3-none-any.whl (25 kB)
Requirement already satisfied: matplotlib in c:\users\user\appdata\local\programs\python\python311\lib\site-
packages (from pygad) (3.7.1)
Requirement already satisfied: numpy in c:\users\user\appdata\local\programs\python\python311\lib\site-packa
ges (from pygad) (1.24.3)
Requirement already satisfied: contourpy>=1.0.1 in c:\users\user\appdata\local\programs\python\python311\lib
\site-packages (from matplotlib->pygad) (1.0.7)
Requirement already satisfied: cycler>=0.10 in c:\users\user\appdata\local\programs\python\python311\lib\sit
e-packages (from matplotlib->pygad) (0.11.0)
Requirement already satisfied: fonttools>=4.22.0 in c:\users\user\appdata\local\programs\python\python311\li
b\site-packages (from matplotlib->pygad) (4.39.4)
Requirement already satisfied: kiwisolver>=1.0.1 in c:\users\user\appdata\local\programs\python\python311\li
b\site-packages (from matplotlib->pygad) (1.4.4)
Requirement already satisfied: packaging>=20.0 in c:\users\user\appdata\local\programs\python\python311\lib
\site-packages (from matplotlib->pygad) (23.1)
Requirement already satisfied: pillow>=6.2.0 in c:\users\user\appdata\local\programs\python\python311\lib\si
te-packages (from matplotlib->pygad) (9.5.0)
Requirement already satisfied: pyparsing>=2.3.1 in c:\users\user\appdata\local\programs\python\python311\lib
\site-packages (from matplotlib->pygad) (3.0.9)
Requirement already satisfied: python-dateutil>=2.7 in c:\users\user\appdata\local\programs\python\python311
\lib\site-packages (from matplotlib->pygad) (2.8.2)
Requirement already satisfied: six>=1.5 in c:\users\user\appdata\local\programs\python\python311\lib\site-pa
ckages (from python-dateutil>=2.7->matplotlib->pygad) (1.16.0)
Installing collected packages: cloudpickle, pygad
Successfully installed cloudpickle-2.2.1 pygad-3.0.1
Note: you may need to restart the kernel to use updated packages.
```

```
In [3]:   import numpy
          import matplotlib.pyplot
          import pygad
```

```
In [4]:   cluster1_num_samples = 10
          cluster1_x1_start = 0
          cluster1_x1_end = 5
          cluster1_x2_start = 2
          cluster1_x2_end = 6
          cluster1_x1 = numpy.random.random(size=(cluster1_num_samples))
          cluster1_x1 = cluster1_x1 * (cluster1_x1_end - cluster1_x1_start) + cluster1_x1_start
          cluster1_x2 = numpy.random.random(size=(cluster1_num_samples))
          cluster1_x2 = cluster1_x2 * (cluster1_x2_end - cluster1_x2_start) + cluster1_x2_start
          cluster2_num_samples = 10
          cluster2_x1_start = 10
          cluster2_x1_end = 15
          cluster2_x2_start = 8
          cluster2_x2_end = 12
          cluster2_x1 = numpy.random.random(size=(cluster2_num_samples))
          cluster2_x1 = cluster2_x1 * (cluster2_x1_end - cluster2_x1_start) + cluster2_x1_start
          cluster2_x2 = numpy.random.random(size=(cluster2_num_samples))
          cluster2_x2 = cluster2_x2 * (cluster2_x2_end - cluster2_x2_start) + cluster2_x2_start
```

In [5]:
```python
c1 = numpy.array([cluster1_x1, cluster1_x2]).T
c2 = numpy.array([cluster2_x1, cluster2_x2]).T
data = numpy.concatenate((c1, c2), axis=0)
data
```

Out[5]:
```
array([[ 4.17094383,  4.22984348],
       [ 3.46226484,  4.26983221],
       [ 2.23046348,  2.01892606],
       [ 2.27430265,  3.11525232],
       [ 3.42517399,  4.72979582],
       [ 0.30258189,  4.31797797],
       [ 3.70092728,  4.42451638],
       [ 2.38067899,  3.32796332],
       [ 0.10491735,  5.71060254],
       [ 2.32120854,  4.18250625],
       [14.41853806,  9.21083733],
       [14.95824219, 10.28229812],
       [13.4297114 ,  8.43200205],
       [14.1233841 , 11.96657437],
       [13.7304333 , 11.54244065],
       [10.14797515,  8.85073911],
       [10.17727036,  9.31310712],
       [11.20819667, 10.08596482],
       [12.17015706,  9.6888654 ],
       [12.43394282,  9.12667018]])
```

In [6]:
```python
matplotlib.pyplot.scatter(cluster1_x1, cluster1_x2)
matplotlib.pyplot.scatter(cluster2_x1, cluster2_x2)
matplotlib.pyplot.title("Optimal Clustering")
matplotlib.pyplot.show()
```


Optimal Clustering
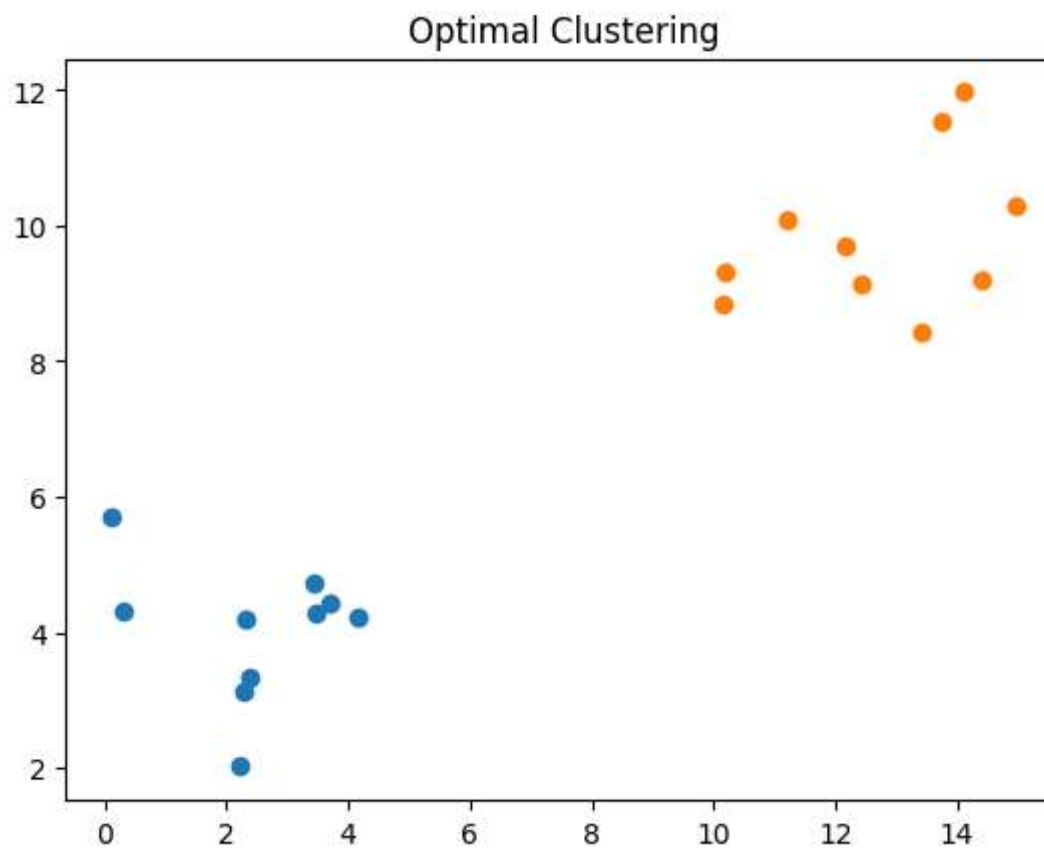
```python
In [11]: def cluster_data(solution, solution_idx):
             global num_cluster, data
             feature_vector_length = data.shape[1]
             cluster_centers = []
             all_clusters_dists = []
             clusters = []
             clusters_sum_dist = []
             for clust_idx in range(num_clusters):
                 cluster_centers.append(solution[feature_vector_length*clust_idx:feature_vector_length*(clust_idx+1)])
                 cluster_center_dists = euclidean_distance(data, cluster_centers[clust_idx])
                 all_clusters_dists.append(numpy.array(cluster_center_dists))
             cluster_centers = numpy.array(cluster_centers)
             all_clusters_dists = numpy.array(all_clusters_dists)
             cluster_indices = numpy.argmin(all_clusters_dists, axis=0)
             for clust_idx in range(num_clusters):
                 clusters.append(numpy.where(cluster_indices == clust_idx)[0])

                 if len(clusters[clust_idx]) == 0:
                     clusters_sum_dist.append(0)
                 else:
                     clusters_sum_dist.append(numpy.sum(all_clusters_dists[clust_idx, clusters[clust_idx]]))
                     clusters_sum_dist = numpy.array(clusters_sum_dist)
             return cluster_centers, all_clusters_dists, cluster_indices, clusters, clusters_sum_dist
```

```python
In [12]: def fitness_func(ga_instance,solution, solution_idx):
             _, _, _, _, clusters_sum_dist = cluster_data(solution, solution_idx)
             fitness = 1.0 / (numpy.sum(clusters_sum_dist) + 0.00000001)
             return fitness
```
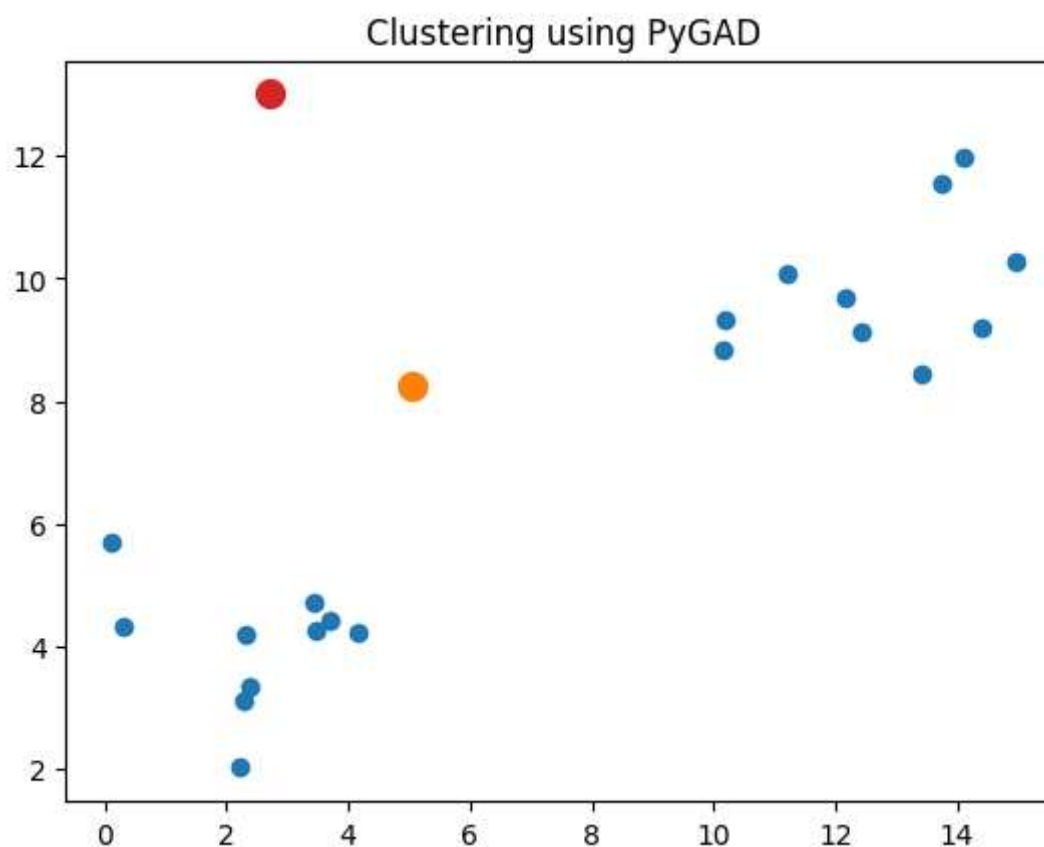
In [15]:
```python
def euclidean_distance(X, Y):
    return numpy.sqrt(numpy.sum(numpy.power(X - Y, 2), axis=1))
num_clusters = 2
num_genes = num_clusters * data.shape[1]
ga_instance = pygad.GA(num_generations=100,
                       sol_per_pop=10,
                       num_parents_mating=5,
                       init_range_low=-6,
                       init_range_high=20,
                       keep_parents=2,
                       num_genes=num_genes,
                       fitness_func=fitness_func,
                       suppress_warnings=True)
ga_instance.run()
```

In [20]:
```python
best_solution, best_solution_fitness, best_solution_idx = ga_instance.best_solution()
print("Best solution is {bs}".format(bs=best_solution))
print("Fitness of the best solution is {bsf}".format(bsf=best_solution_fitness))
print("Best solution found after {gen} generations".format(gen=ga_instance.best_solution_generation))
```

```
Best solution is [ 5.06254353  8.23716173  2.72143849 13.00801582]
Fitness of the best solution is 100000000.0
Best solution found after 0 generations
```

In [24]:
```python
cluster_centers, all_clusters_dists, cluster_indices, clusters, clusters_sum_dist= cluster_data(best_solution.
```

In [26]:
```python
for cluster_idx in range(num_clusters):
    cluster_x = data[clusters[cluster_idx], 0]
    cluster_y = data[clusters[cluster_idx], 1]
    matplotlib.pyplot.scatter(cluster_x, cluster_y)
    matplotlib.pyplot.scatter(cluster_centers[cluster_idx, 0], cluster_centers[cluster_idx, 1], linewidths=5)
matplotlib.pyplot.title("Clustering using PyGAD")
matplotlib.pyplot.show()
```

Clustering using PyGAD

In [ ]: