In [19]:

```
pip install pygad

```

```
Requirement already satisfied: pygad in c:\users\dell e5490\anaconda3\lib\site-packages (3.0.1)
Requirement already satisfied: cloudpickle in c:\users\dell e5490\anaconda3\lib\site-packages (from pygad) (2.0.0)
Requirement already satisfied: matplotlib in c:\users\dell e5490\anaconda3\lib\site-packages (from pygad) (3.7.0)
Requirement already satisfied: numpy in c:\users\dell e5490\anaconda3\lib\site-packages (from pygad) (1.23.5)
Requirement already satisfied: contourpy>=1.0.1 in c:\users\dell e5490\anaconda3\lib\site-packages (from matplotlib->pygad) (1.0.
5)
Requirement already satisfied: cycler>=0.10 in c:\users\dell e5490\anaconda3\lib\site-packages (from matplotlib->pygad) (0.11.0)
Requirement already satisfied: fonttools>=4.22.0 in c:\users\dell e5490\anaconda3\lib\site-packages (from matplotlib->pygad) (4.2
5.0)
Requirement already satisfied: kiwisolver>=1.0.1 in c:\users\dell e5490\anaconda3\lib\site-packages (from matplotlib->pygad) (1.4.
4)
Requirement already satisfied: packaging>=20.0 in c:\users\dell e5490\anaconda3\lib\site-packages (from matplotlib->pygad) (22.0)
Requirement already satisfied: pillow>=6.2.0 in c:\users\dell e5490\anaconda3\lib\site-packages (from matplotlib->pygad) (9.4.0)
Requirement already satisfied: pyparsing>=2.3.1 in c:\users\dell e5490\anaconda3\lib\site-packages (from matplotlib->pygad) (3.0.
9)
Requirement already satisfied: python-dateutil>=2.7 in c:\users\dell e5490\anaconda3\lib\site-packages (from matplotlib->pygad)
(2.8.2)
Requirement already satisfied: six>=1.5 in c:\users\dell e5490\anaconda3\lib\site-packages (from python-dateutil>=2.7->matplotlib-
>pygad) (1.16.0)
Note: you may need to restart the kernel to use updated packages.
```

In [20]:

```python
import numpy
import matplotlib.pyplot
import pygad
```

In [21]:

```
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 [22]:

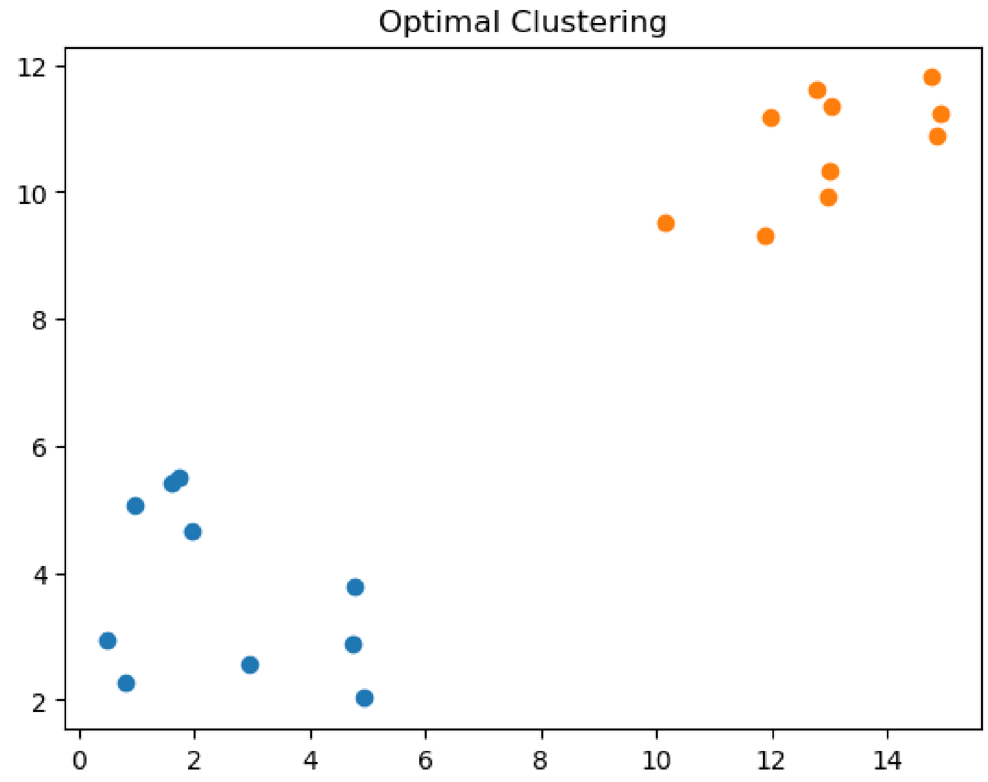
```
c1 = numpy.array([cluster1_x1, cluster1_x2]).T
c2 = numpy.array([cluster2_x1, cluster2_x2]).T
data = numpy.concatenate((c1, c2), axis=0)
data
```

Out[22]:

```
array([[ 1.7494081 ,  5.5103293 ],
       [ 1.60890576,  5.42008744],
       [ 0.82449641,  2.27309535],
       [ 4.77793117,  3.79491509],
       [ 4.92888418,  2.02957057],
       [ 4.75850745,  2.8817934 ],
       [ 0.95653592,  5.0816579 ],
       [ 2.9477697 ,  2.58118379],
       [ 0.49180158,  2.93371916],
       [ 1.95010561,  4.67021634],
       [11.97825879, 11.17385021],
       [14.76285864, 11.79930468],
       [13.03667382, 11.3448926 ],
       [14.85712163, 10.89071152],
       [14.92604189, 11.24595704],
       [13.01924418, 10.33448889],
       [12.99149357,  9.92618232],
       [12.76845324, 11.62178504],
       [10.1476546 ,  9.5219723 ],
       [11.90027031,  9.32491814]])
```

In [23]:

```python
matplotlib.pyplot.scatter(cluster1_x1, cluster1_x2)
matplotlib.pyplot.scatter(cluster2_x1, cluster2_x2)
matplotlib.pyplot.title("Optimal Clustering")
matplotlib.pyplot.show()
```



In [24]:

```python
def euclidean_distance(X, Y):
    return numpy.sqrt(numpy.sum(numpy.power(X - Y, 2), axis=1))
```

In [29]:

```
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
```

In [34]:

```
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 [35]:

```
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 [36]:

```
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))
```

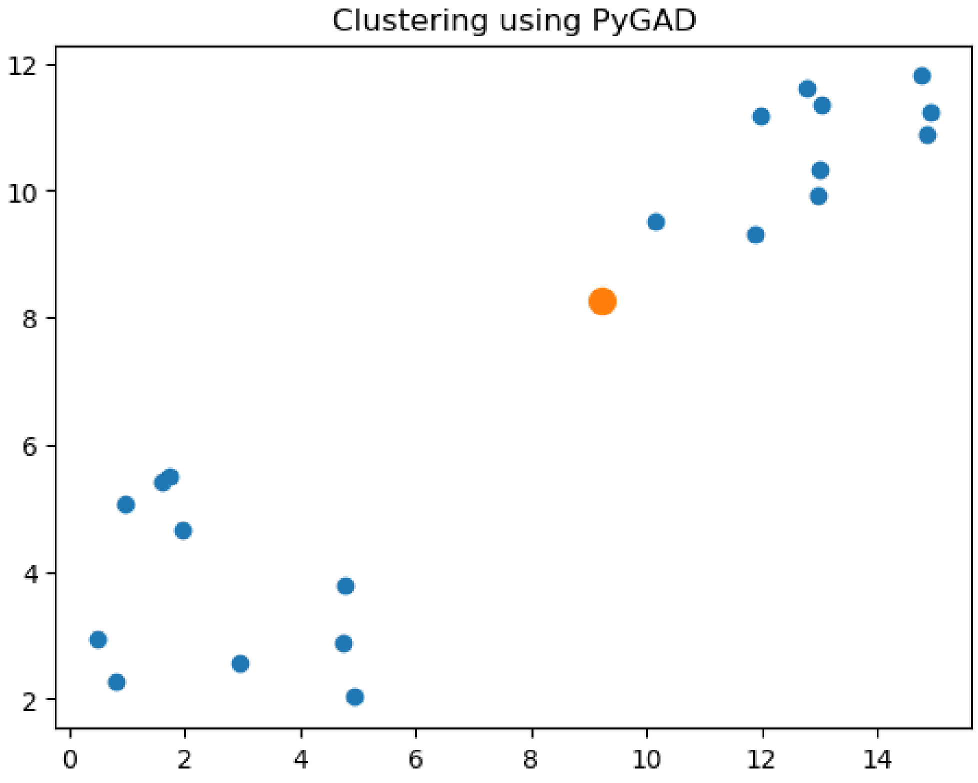
```
Best solution is [ 9.23918529  8.25731961  4.86408881 -0.48284896]
Fitness of the best solution is 0.007761560011998536
Best solution found after 92 generations
```

In [40]:

```
cluster_centers, all_clusters_dists, cluster_indices, clusters, clusters_sum_dist=cluster_data(best_solution, best_solution_idx)
```

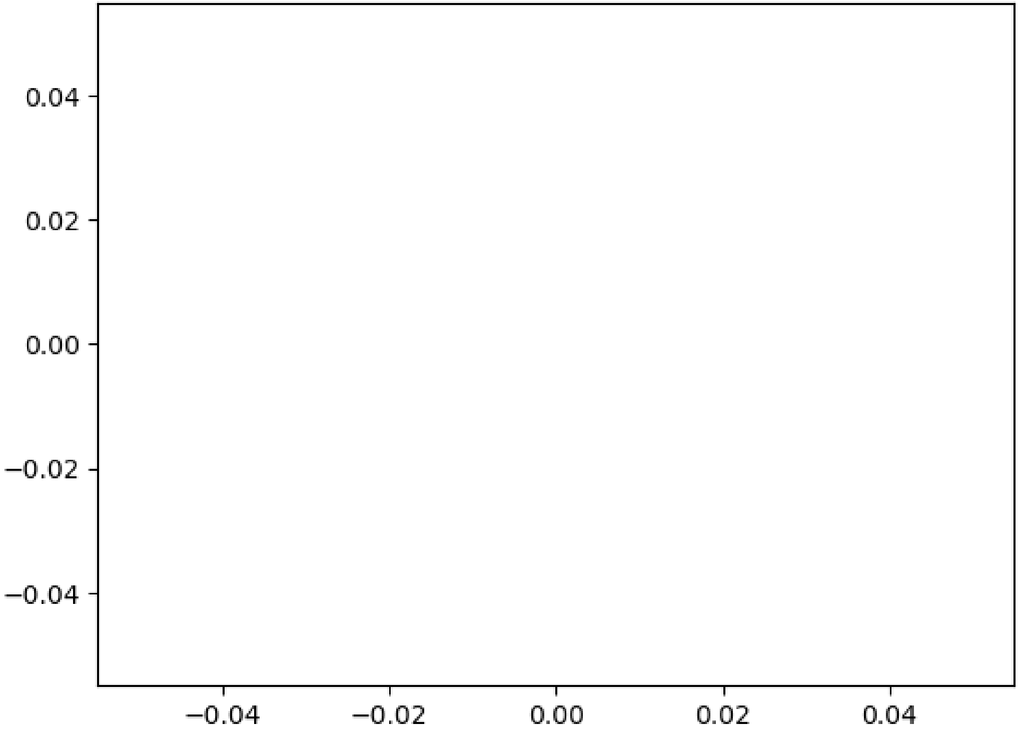
In [43]:

```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()
```



```
---------------------------------------------------------------------------
IndexError                                Traceback (most recent call last)
Cell In[43], line 5
      3 cluster_y = data[clusters[cluster_idx], 1]
      4 matplotlib.pyplot.scatter(cluster_x, cluster_y)
----> 5 matplotlib.pyplot.scatter(cluster_centers[cluster_idx, 0], cluster_centers[cluster_idx, 1], linewidths=5)
      6 matplotlib.pyplot.title("Clustering using PyGAD")
      7 matplotlib.pyplot.show()

IndexError: index 1 is out of bounds for axis 0 with size 1
```



In [ ]: