Домашнее задание №11

Асланов А.Б., ИУ9-21М

Заданы координаты точек.
a=[0.6, 1.9], c=[2.7, 2.0]

1. Выполнить агломеративную кластеризацию single-link, complete-link и показать её на рисунке.
2. Посмотреть, как дальше пойдёт кластеризация с применением метода k-means.

Кластеризация с применением k-means

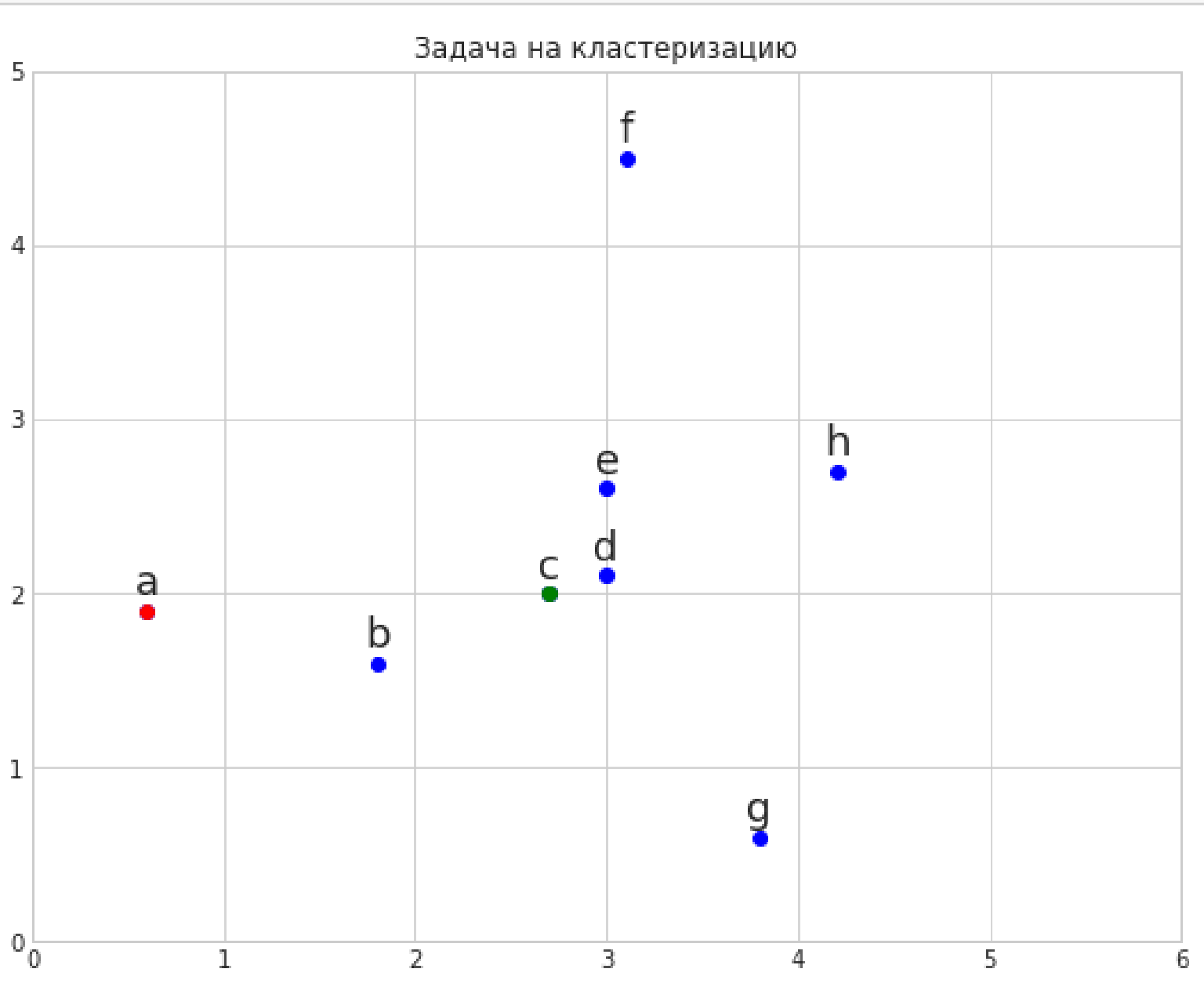
Заданы координаты точек.
a=[0.6, 1.9], c=[2.7, 2.0]

In [1]:
```
%matplotlib inline
import matplotlib.pyplot as plt
plt.style.use('seaborn-whitegrid')
import numpy as np
import pandas as pd
```

In [2]:
```
def plot_graph_properties(centroids, color_map, title='Задача на кластеризацию'):
    """Вспомогательная функция для построения графиков"""
    plt.xlim([0, 6])
    plt.ylim([0, 5])
    plt.title(title)
    for i in centroids.keys():
        plt.scatter(*centroids[i], color=color_map[i])
    for x, y, label in zip(xs, ys, labels):
        plt.annotate(label,
                     (x, y), # point to label
                     size=18,
                     textcoords="offset points", # how to position the text
                     xytext=(0, 7), # distance from text to points (x, y)
                     ha='center') # horizontal alignment can be left, right or center
    plt.show()
    return None


# Инициализация
xs = [0.6, 1.8, 2.7, 3.0, 3.0, 3.1, 3.8, 4.2]
ys = [1.9, 1.6, 2.0, 2.1, 2.6, 4.5, 0.6, 2.7]
labels = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h']
centroids = {'a':[0.6, 1.9], 'c':[2.7, 2.0]} # Координаты центров кластеров
color_map = {'a': 'r', 'c': 'g'} # центрам кластеров будут соответствовать красный и зеленый цвета

# Определяем центры кластеров
plt.figure(figsize=(9, 7))
plt.scatter(xs, ys, marker='o', c='b')
plot_graph_properties(centroids, color_map)
```


Задача на кластеризацию

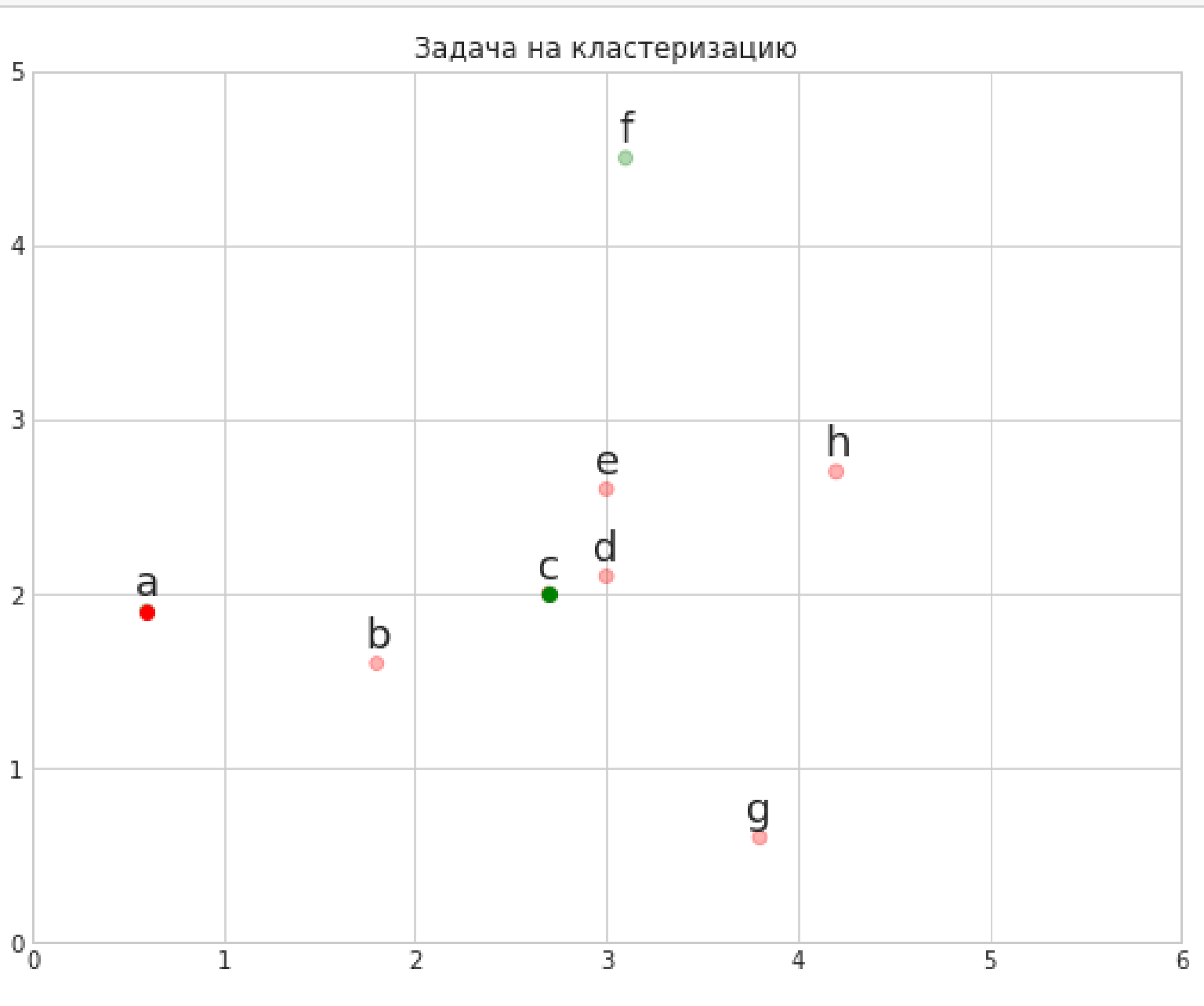
In [3]:
```
df = pd.DataFrame({'x': xs, 'y': ys})


def assignment(df, centroids):

    for i in ['a', 'c']:
        x1 = df['x']
        x2 = df['y']
        y1 = centroids[i][0]
        y2 = centroids[i][1]
        cos_dist = (x1*y1 + x2*y2) / (np.sqrt(x1**2+x2**2)*np.sqrt(y1**2+y2**2))
        df['distance_from_{}'.format(i)] = cos_dist
    # среди столбцов с расстояниями либо до кластера a, либо до кластера c - выбираем наименьший
    df['closest'] = df[['distance_from_a', 'distance_from_c']].idxmin(axis=1)
    df['closest'] = df['closest'].map(lambda x: x.split('_')[-1]) # пишем, какому кластеру присваи
вается точка
    df['color'] = df['closest'].map(lambda x: color_map[x])

    return df


df = assignment(df, centroids)
plt.figure(figsize=(9, 7))
plt.scatter(xs, ys, marker='o', c=df['color'], alpha=0.3)
plot_graph_properties(centroids, color_map)
```


Задача на кластеризацию

In [4]:
```
def update(k):
    for i in centroids.keys():
        centroids[i][0] = np.mean(df[df['closest'] == i]['x'])
        centroids[i][1] = np.mean(df[df['closest'] == i]['y'])
    return k

centroids = update(centroids)
plt.figure(figsize=(9, 7))
plt.scatter(xs, ys, color=df['color'], alpha=0.3)
plot_graph_properties(centroids, color_map, title='Задача на кластеризацию: перемещение центра кластер
а')
plt.show()
```

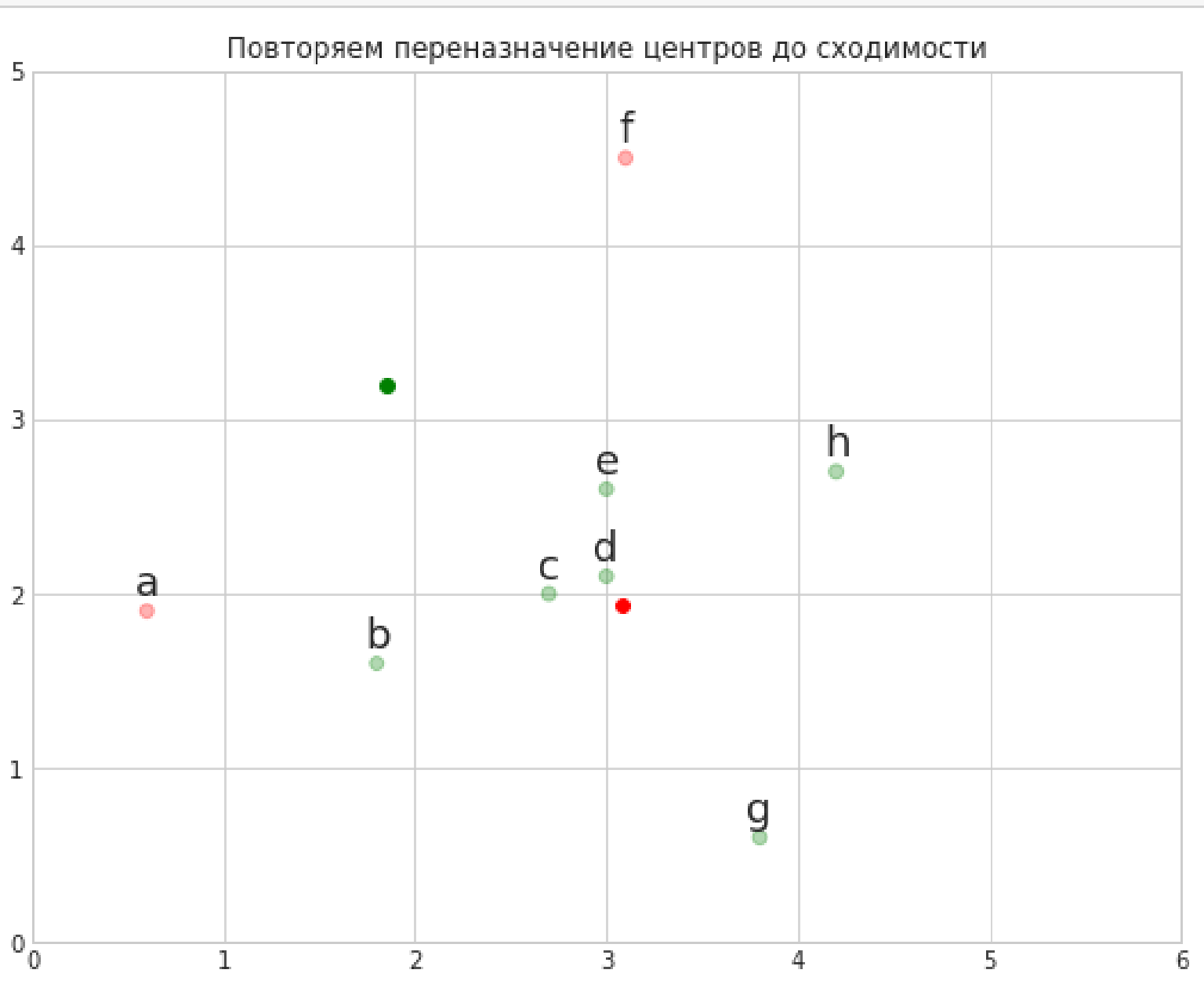

Задача на кластеризацию: перемещение центра кластера

In [5]:
```
df = assignment(df, centroids)
plt.figure(figsize=(9, 7))
plt.scatter(xs, ys, color=df['color'], alpha=0.3)
plot_graph_properties(centroids, color_map, title='Повторная итерация: переприсвоение точек кластерам')
```


Повторная итерация: переприсвоение точек кластерам

In [12]:
```
iteration_num = 0
while iteration_num < 100:
    iteration_num += 1
    closest_centroids = df['closest'].copy(deep=True)
    centroids = update(centroids)
    df = assignment(df, centroids)
    if closest_centroids.equals(df['closest']):
        break

plt.figure(figsize=(9, 7))
plt.scatter(df['x'], df['y'], color=df['color'], alpha=0.3)
plot_graph_properties(centroids, color_map, title='Повторяем переназначение центров до сходимости')
```


Повторяем переназначение центров до сходимости

Перепроверка результатов с использованием NLTK

In [7]:
```
import nltk
from nltk.cluster import KMeansClusterer
from sklearn import cluster

X = list(zip(xs, ys))
X = np.array([list(value) for value in X])

NUM_CLUSTERS = 2
kclusterer = KMeansClusterer(NUM_CLUSTERS, distance=nltk.cluster.util.cosine_distance, repeats
=100)
assigned_clusters = kclusterer.cluster(X, assign_clusters=True)
print(list(zip(labels, assigned_clusters)))
```

```
[('a', 1), ('b', 0), ('c', 0), ('d', 0), ('e', 0), ('f', 1), ('g', 0), ('h', 0)]
```

Агломеративная кластеризация

In [8]:
```
%matplotlib inline
import matplotlib.pyplot as plt
plt.style.use('seaborn-whitegrid')
import numpy as np
import pandas as pd
```

In [9]:
```
# Инициализация
xs = [0.6, 1.8, 2.7, 3.0, 3.0, 3.1, 3.8, 4.2]
ys = [1.9, 1.6, 2.0, 2.1, 2.6, 4.5, 0.6, 2.7]
labels = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h']
centroids = {'a':[0.6, 1.9], 'c':[2.7, 2.0]} # Координаты центров кластеров
color_map = {'a': 'r', 'c': 'g'} # центрам кластеров будут соответствовать красный и зеленый цвета
```

In [10]:
```
from sklearn.cluster import AgglomerativeClustering

X = list(zip(xs, ys))
cluster = AgglomerativeClustering(n_clusters=2, affinity='cosine', linkage='single')
cluster.fit_predict(X)
X = np.array([list(value) for value in X])

plt.figure(figsize=(9, 7))
plt.scatter(X[:, 0], X[:, 1], c=cluster.labels_, cmap='plasma')

plt.xlim([0, 6])
plt.ylim([0, 5])
plt.title('Агломеративная кластеризация (single-link)')
for x, y, label in zip(xs, ys, labels):
    plt.annotate(label,
                 (x, y), # point to label
                 size=18,
                 textcoords="offset points", # how to position the text
                 xytext=(0, 7), # distance from text to points (x, y)
                 ha='center') # horizontal alignment can be left, right or center
plt.show()
```

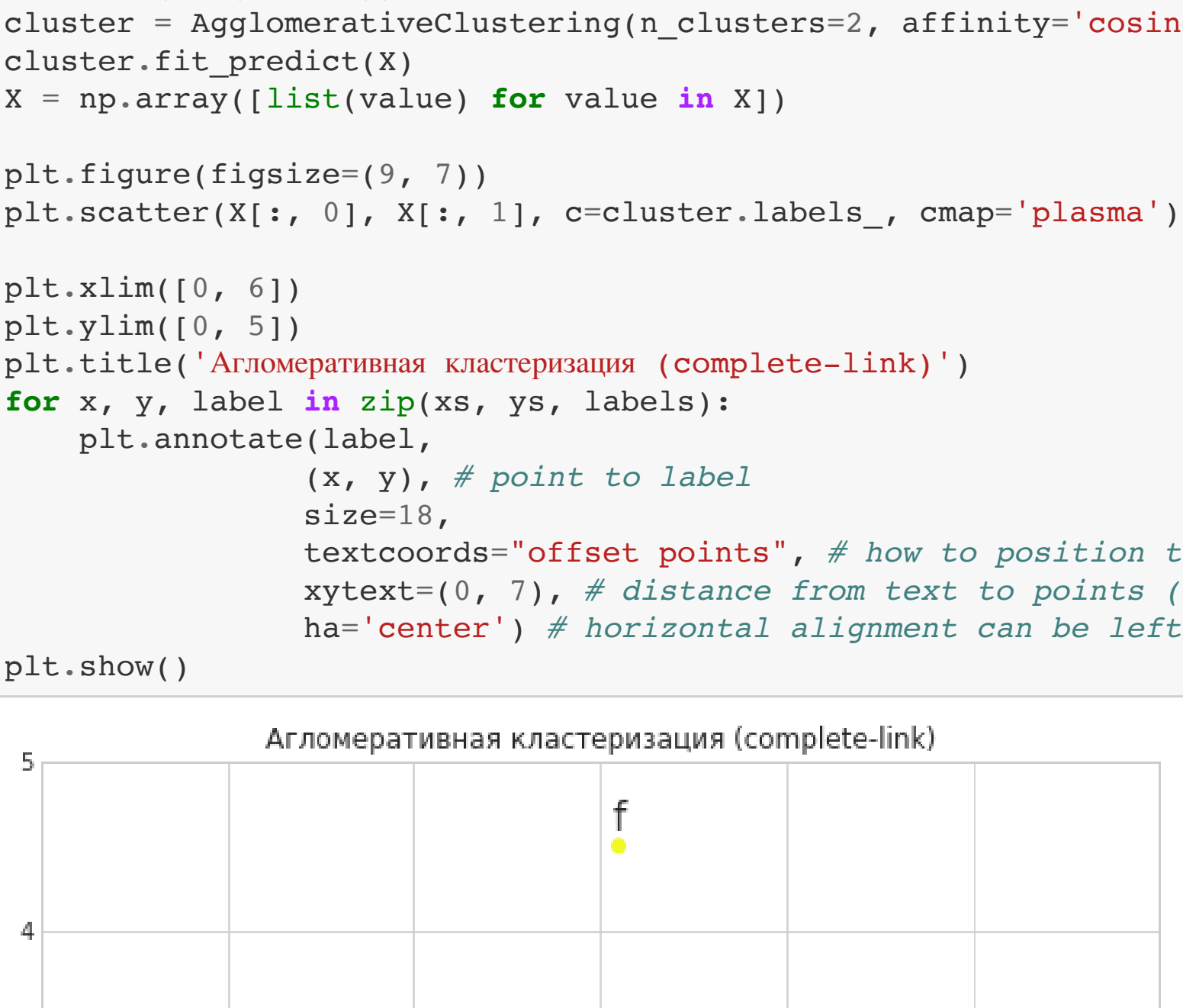

Агломеративная кластеризация (single-link)

In [11]:
```
from sklearn.cluster import AgglomerativeClustering

X = list(zip(xs, ys))
cluster = AgglomerativeClustering(n_clusters=2, affinity='cosine', linkage='complete')
cluster.fit_predict(X)
X = np.array([list(value) for value in X])

plt.figure(figsize=(9, 7))
plt.scatter(X[:, 0], X[:, 1], c=cluster.labels_, cmap='plasma')

plt.xlim([0, 6])
plt.ylim([0, 5])
plt.title('Агломеративная кластеризация (complete-link)')
for x, y, label in zip(xs, ys, labels):
    plt.annotate(label,
                 (x, y), # point to label
                 size=18,
                 textcoords="offset points", # how to position the text
                 xytext=(0, 7), # distance from text to points (x, y)
                 ha='center') # horizontal alignment can be left, right or center
plt.show()
```

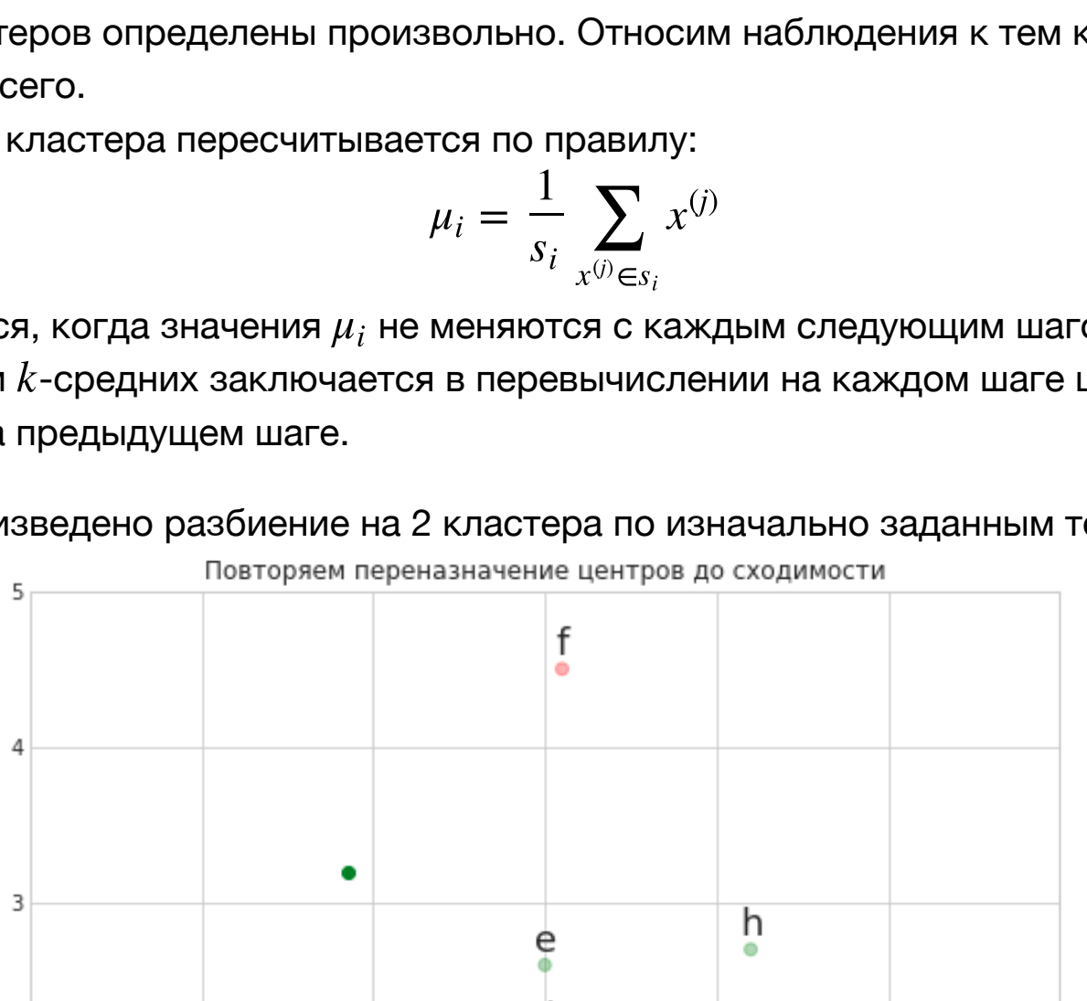

Агломеративная кластеризация (complete-link)

Выводы

Метод k-средних:

- Число кластеров определяется заранее (в данной задаче было 2 кластера);
- Изначально центры кластеров определены произвольно. Относим наблюдения к тем кластерам, чьё среднее (центроид) к ним ближе всего.
- Затем центроид каждого кластера пересчитывается по правилу:
$$\mu_i = \frac{1}{s_i}\sum_{x^{(j)}\in s_i} x^{(j)}$$
- Алгоритм останавливается, когда значения μ_i не меняются с каждым следующим шагом.
  Таким образом, алгоритм k-средних заключается в перевычислении на каждом шаге центроида для каждого кластера, полученного на предыдущем шаге.

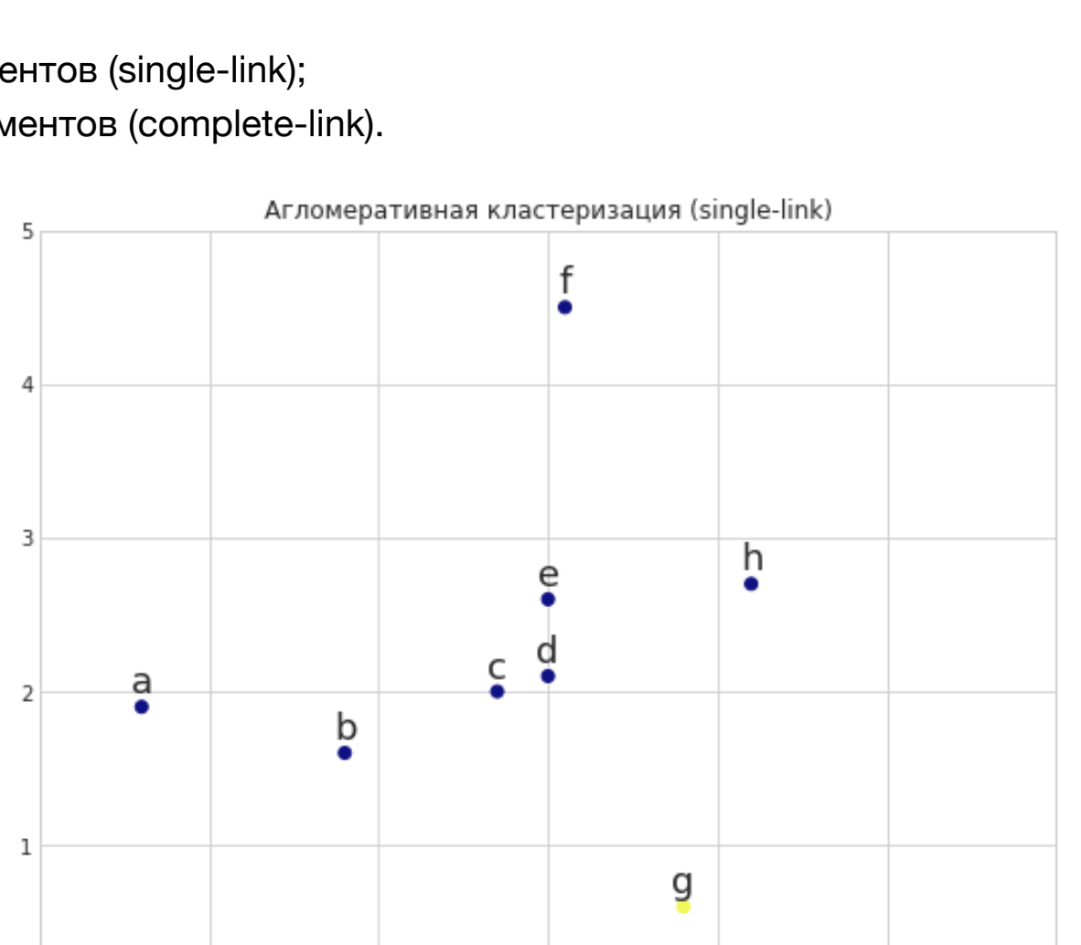
Для данной задачи было произведено разбиение на 2 кластера по изначально заданным точкам a, c.

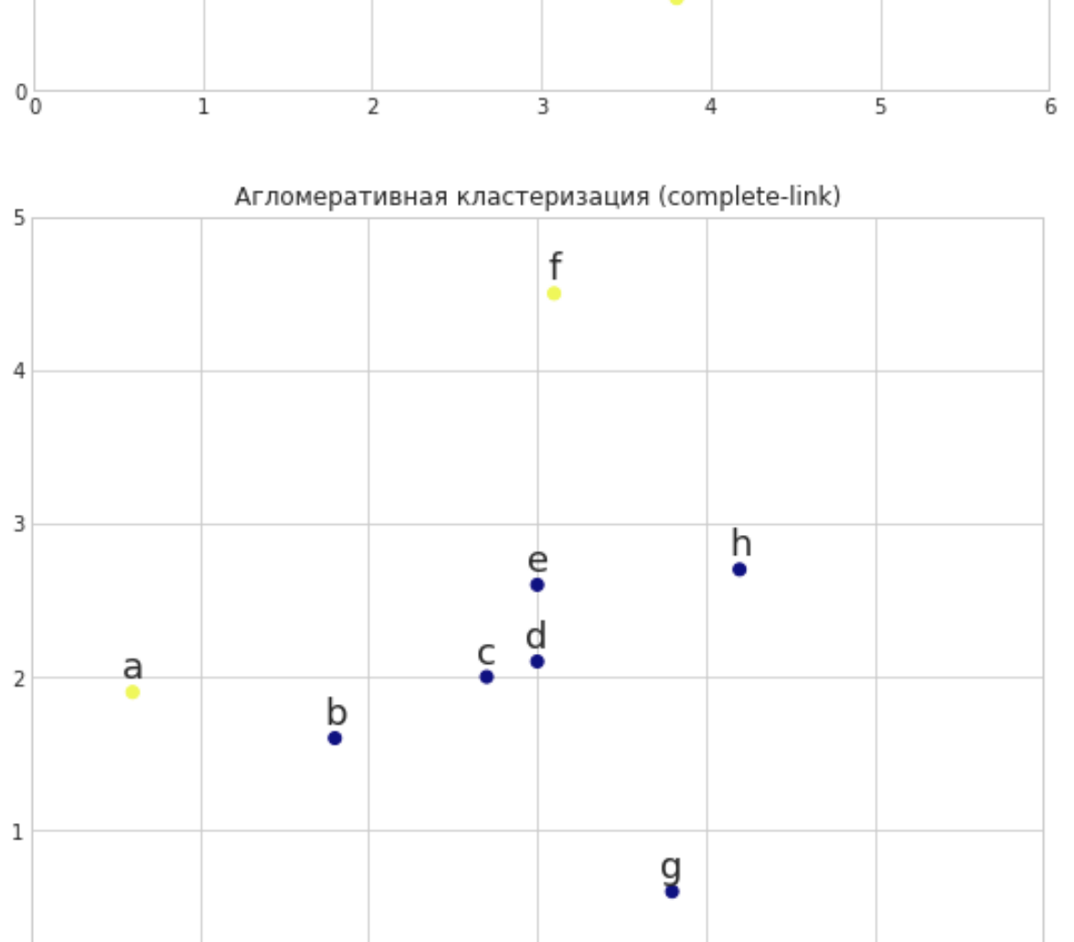

Повторяем переназначение центров до сходимости

Агломеративная кластеризация

Это подмножество иерархической кластеризации, где главная идея основывается на том, что каждый объект - это кластер, и с каждой итерацией мы сливаем в дальнейшем эти кластеры друг с другом. Объединяются точки (документы), которые ближе друг к другу (по косинусному расстоянию). Слияния происходит по сходству:

- наиболее похожих документов (single-link);
- наболее *не*похожих документов (complete-link).


Агломеративная кластеризация (single-link)


Агломеративная кластеризация (complete-link)