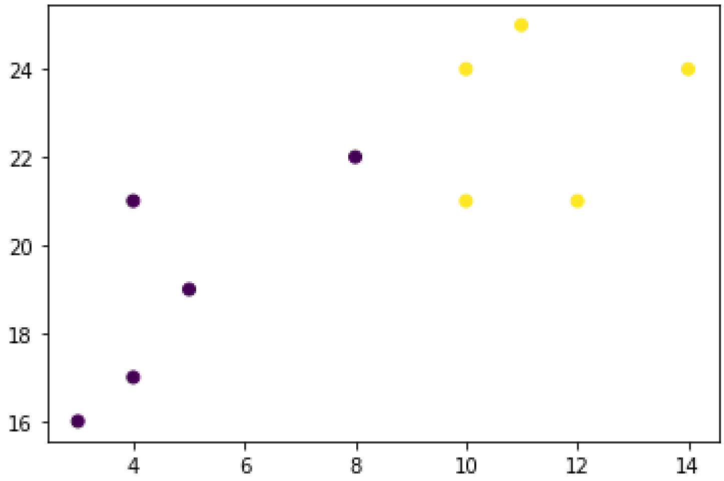
In [11]:

```
import matplotlib.pyplot as plt

x = [4, 5, 10, 4, 3, 11, 14 , 8, 10, 12]
y = [21, 19, 24, 17, 16, 25, 24, 22, 21, 21]
classes = [0, 0, 1, 0, 0, 1, 1, 0, 1, 1]

plt.scatter(x, y, c=classes)
plt.show()
```



In [12]:

```
#Now we fit the KNN algorithm with K=1:

#Turn the input features into a set of points:

from sklearn.neighbors import KNeighborsClassifier
data = list(zip(x, y))
knn = KNeighborsClassifier(n_neighbors=1)

knn.fit(data, classes)
```

Out[12]:

```
KNeighborsClassifier(n_neighbors=1)
```

In [13]:

```
print(data)
```

```
[(4, 21), (5, 19), (10, 24), (4, 17), (3, 16), (11, 25), (14, 24), (8, 22),
(10, 21), (12, 21)]
```

In [14]:

```
new_x = 8
new_y = 21
new_point = [(new_x, new_y)]

prediction = knn.predict(new_point)
```

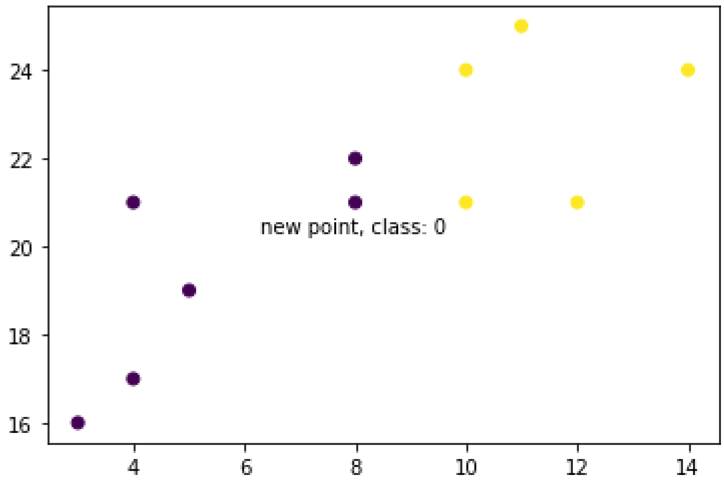
In [15]:

```
print(prediction)
```

```
[0]
```

In [16]:

```
plt.scatter(x + [new_x], y + [new_y], c=classes + [prediction[0]])
plt.text(x=new_x-1.7, y=new_y-0.7, s=f"new point, class: {prediction[0]}")
plt.show()
```



In [17]:

```
#Now we do the same thing, but with a higher K value which changes the prediction:

knn = KNeighborsClassifier(n_neighbors=5)

knn.fit(data, classes)

prediction = knn.predict(new_point)
```

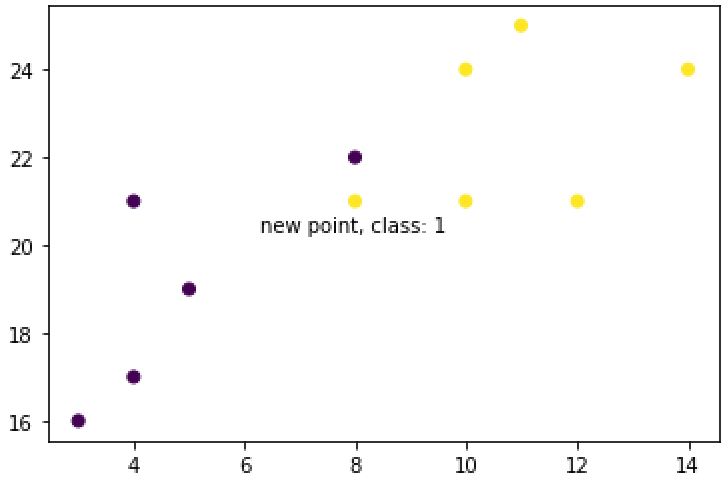
In [18]:

```
print(prediction)
```

```
[1]
```

In [19]:

```
plt.scatter(x + [new_x], y + [new_y], c=classes + [prediction[0]])
plt.text(x=new_x-1.7, y=new_y-0.7, s=f"new point, class: {prediction[0]}")
plt.show()
```



In [ ]:

```

```