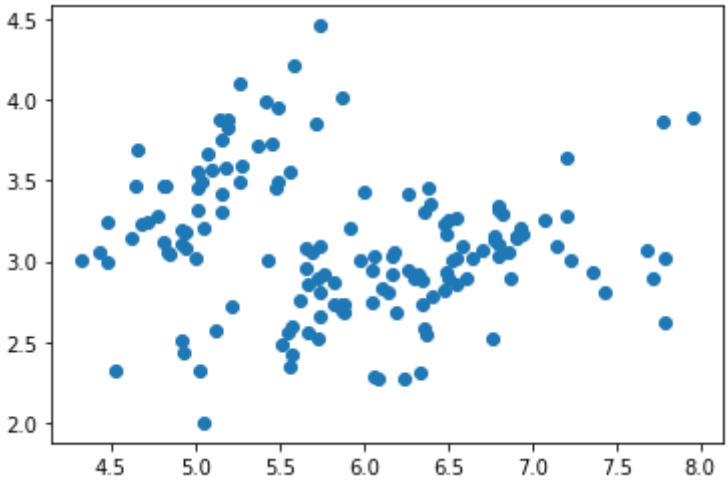
In [12]:
```
import numpy as np
import mltools as ml
import matplotlib.pyplot as plt
iris = np.genfromtxt("data/iris.txt",delimiter=None)
X = iris[:,0:2]
plt.plot(X[:,0],X[:,1], 'o')
```

Out[12]:
```
[<matplotlib.lines.Line2D at 0x11ad99910>]
```




1a. I think there is a cluster in the top left corner in the region around X = 4-6 and Y = 3-4.5. I also think there is a cluster in the bottom left region in the few points between X = 4.5 - 5.25. The next cluster would be between X = 5.5-7.5 and Y = 2.3 - 3.5. There would be another cluster in the right region around X = 7.5-8.0 and Y = 2.5 - 3.0. The last cluster would be the two points in the top right corner.

1b.

k = 2

In [154]:
```
z1,c1,sumd1 = ml.cluster.kmeans(X,2)
z2,c2,sumd2 = ml.cluster.kmeans(X,2,max_iter = 50)
z3,c3,sumd3 = ml.cluster.kmeans(X,2,max_iter = 200)
print(sumd1)
print(sumd2)
print(sumd3)
```

```
57.87966196118197
57.87966196118197
57.877648396983034
```

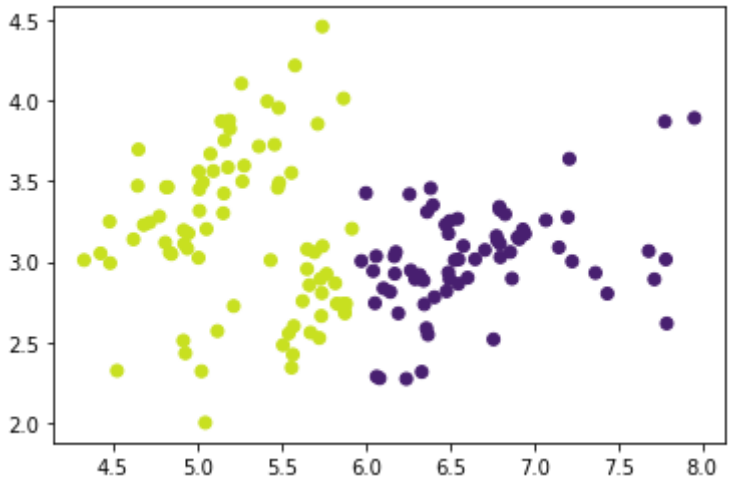
In [155]:
```
z4,c4,sumd4 = ml.cluster.kmeans(X,2, init = 'farthest')
z5,c5,sumd5 = ml.cluster.kmeans(X,2,max_iter = 50, init = 'farthest')
z6,c6,sumd6 = ml.cluster.kmeans(X,2,max_iter = 200, init = 'farthest')
print(sumd4)
print(sumd5)
print(sumd6)
```

```
57.87966196118197
57.877648396983034
57.877648396983034
```

In [156]:
```
z7,c7,sumd7 = ml.cluster.kmeans(X,2, init = 'k++')
z8,c8,sumd8 = ml.cluster.kmeans(X,2,max_iter = 50, init = 'k++')
z9,c9,sumd9 = ml.cluster.kmeans(X,2,max_iter = 200, init = 'k++')
print(sumd7)
print(sumd8)
print(sumd9)
```

```
57.87966196118197
57.877648396983034
57.877648396983034
```

In [157]:
```
ml.plotClassify2D(None,X,z3)
```




```
k = 5
```

In [162]:
```
z1,c1,sumd1 = ml.cluster.kmeans(X,5)
z2,c2,sumd2 = ml.cluster.kmeans(X,5,max_iter = 50)
z3,c3,sumd3 = ml.cluster.kmeans(X,5,max_iter = 200)
print(sumd1)
print(sumd2)
print(sumd3)
```

```
20.856963620246418
21.31426068787647
25.416974979837697
```

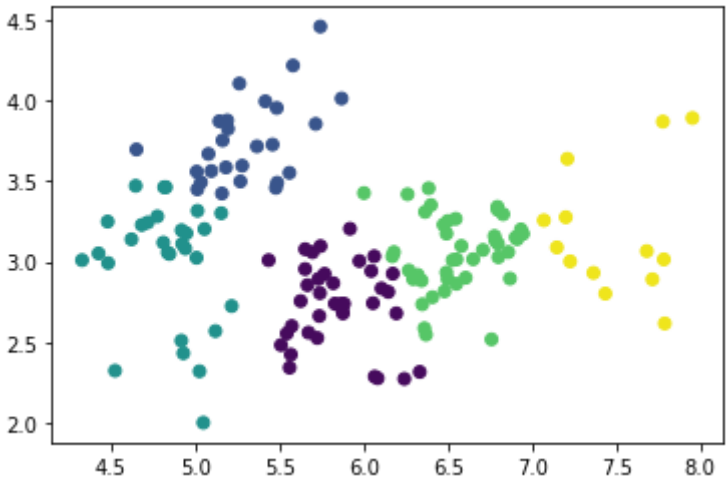
In [163]:
```
z4,c4,sumd4 = ml.cluster.kmeans(X,5, init = 'farthest')
z5,c5,sumd5 = ml.cluster.kmeans(X,5,max_iter = 50, init = 'farthest')
z6,c6,sumd6 = ml.cluster.kmeans(X,5,max_iter = 200, init = 'farthest')
print(sumd4)
print(sumd5)
print(sumd6)
```

```
20.954630196254048
20.954630196254048
21.0902063018713
```

In [164]:
```
z7,c7,sumd7 = ml.cluster.kmeans(X,5, init = 'k++')
z8,c8,sumd8 = ml.cluster.kmeans(X,5,max_iter = 50, init = 'k++')
z9,c9,sumd9 = ml.cluster.kmeans(X,5,max_iter = 200, init = 'k++')
print(sumd7)
print(sumd8)
print(sumd9)
```

```
23.438643757297644
26.637478142770384
20.895826480459537
```

In [165]:
```
ml.plotClassify2D(None,X,z1)
```




```
k = 20
```

In [166]:
```
z1,c1,sumd1 = ml.cluster.kmeans(X,20)
z2,c2,sumd2 = ml.cluster.kmeans(X,20,max_iter = 50)
z3,c3,sumd3 = ml.cluster.kmeans(X,20,max_iter = 200)
print(sumd1)
print(sumd2)
print(sumd3)
```

```
5.05885116387195
5.131089320716877
4.640029381225508
```

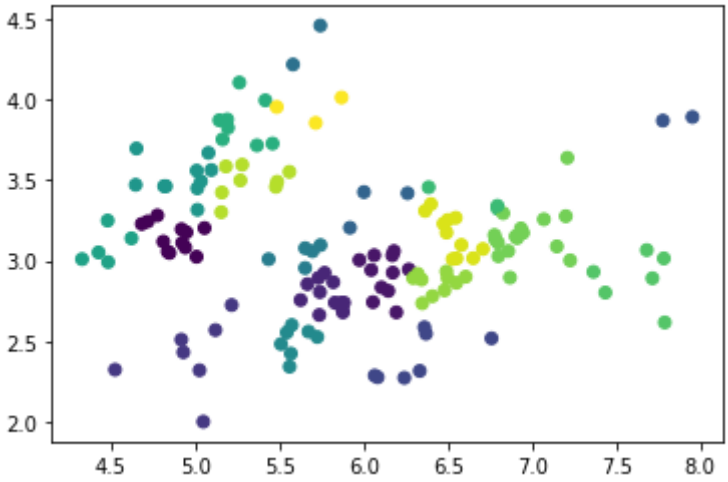
In [167]:
```
z4,c4,sumd4 = ml.cluster.kmeans(X,20, init = 'farthest')
z5,c5,sumd5 = ml.cluster.kmeans(X,20,max_iter = 50, init = 'farthest')
z6,c6,sumd6 = ml.cluster.kmeans(X,20,max_iter = 200, init = 'farthest')
print(sumd4)
print(sumd5)
print(sumd6)
```

```
4.770864849599144
4.859307078741392
4.737349777304152
```

In [168]:
```
z7,c7,sumd7 = ml.cluster.kmeans(X,20, init = 'k++')
z8,c8,sumd8 = ml.cluster.kmeans(X,20,max_iter = 50, init = 'k++')
z9,c9,sumd9 = ml.cluster.kmeans(X,20,max_iter = 200, init = 'k++')
print(sumd7)
print(sumd8)
print(sumd9)
```

```
4.984783027060488
4.775167158521336
4.833196231664706
```

In [169]:
```
ml.plotClassify2D(None,X,z3)
```



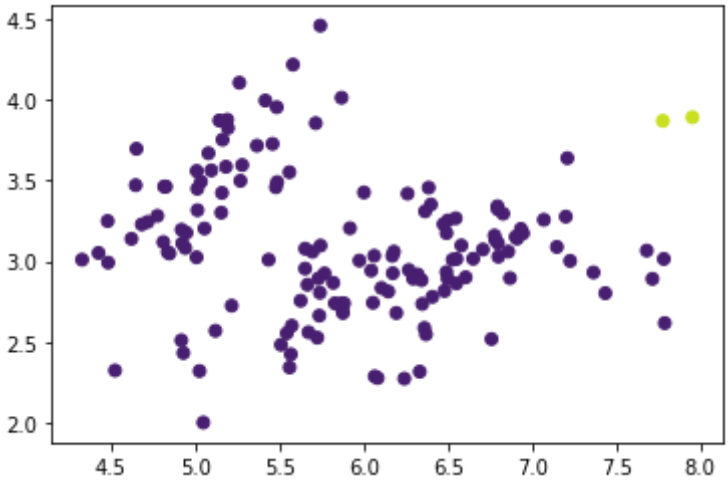

```
1c.
```

```
k = 2
```

In [175]:
```
z, join = ml.cluster.agglomerative(X,2, method = 'min')
z2, join2 = ml.cluster.agglomerative(X,2, method = 'max')
```

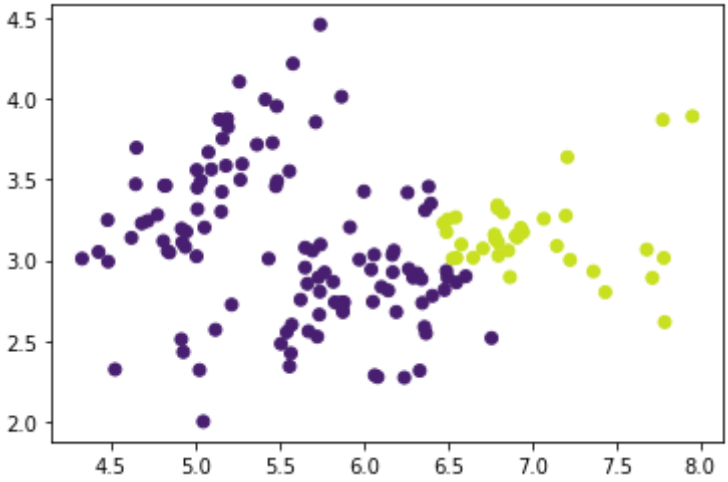
In [176]:
```
print("single linkage")
ml.plotClassify2D(None,X,z)
```

```
single linkage
```




In [177]:
```
print("complete linkage")
ml.plotClassify2D(None,X,z2)
```

```
complete linkage
```

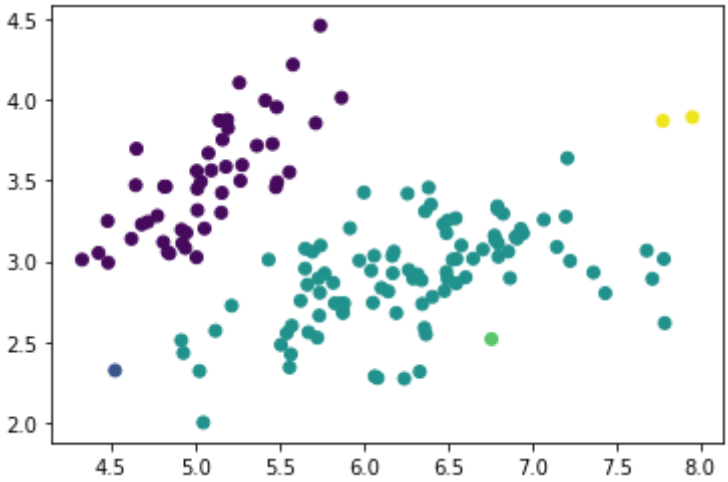



```
k = 5
```

In [178]:
```
z, join = ml.cluster.agglomerative(X,5, method = 'min')
z2, join2 = ml.cluster.agglomerative(X,5, method = 'max')
```

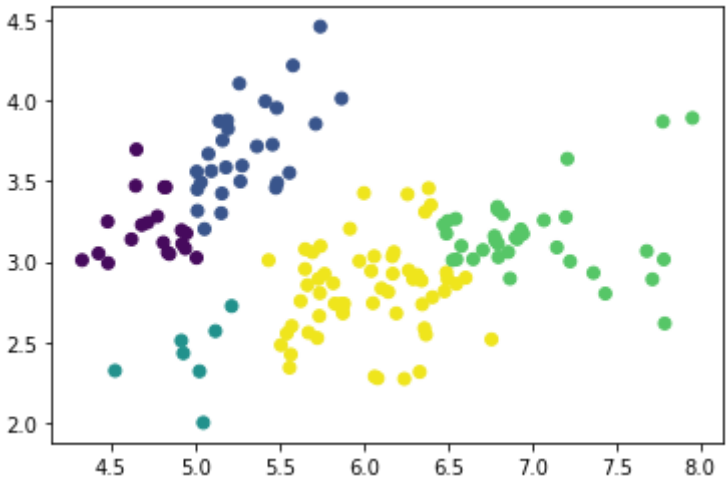
In [179]:
```
print("single linkage")
ml.plotClassify2D(None,X,z)
```

```
single linkage
```




In [180]:
```
print("complete linkage")
ml.plotClassify2D(None,X,z2)
```

```
complete linkage
```

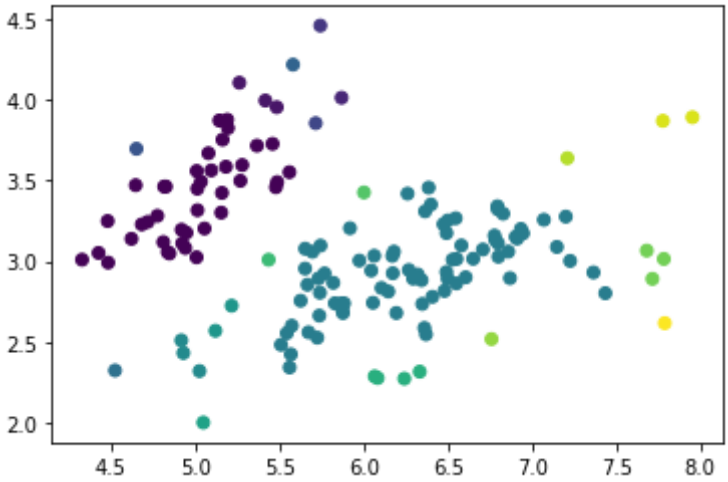



```
k = 20
```

In [181]:
```
z, join = ml.cluster.agglomerative(X,20, method = 'min')
z2, join2 = ml.cluster.agglomerative(X,20, method = 'max')
```

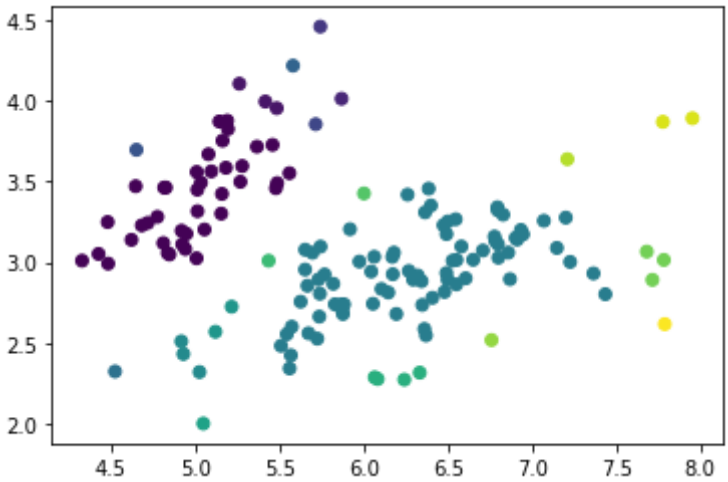
In [182]:
```
print("single linkage")
ml.plotClassify2D(None,X,z)
```

```
single linkage
```




In [183]:
```
print("single linkage")
ml.plotClassify2D(None,X,z)
```

```
single linkage
```

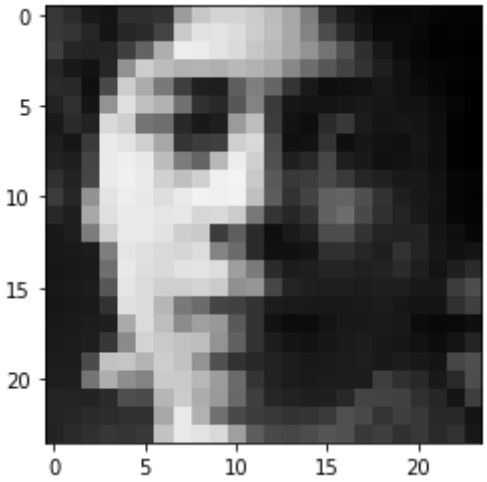



```
1d.
```

```
2a.
```

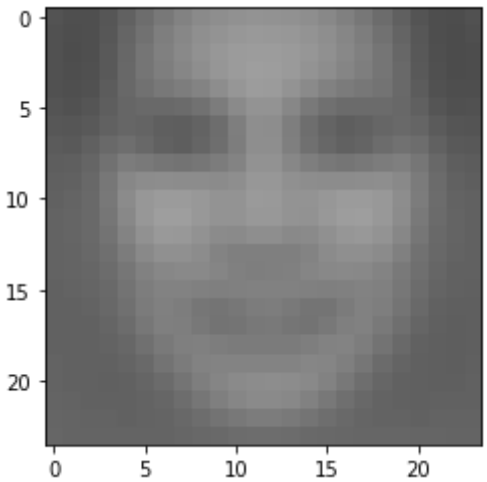
In [187]:
```
X = np.genfromtxt("data/faces.txt", delimiter=None) # load face dataset
plt.figure()
# pick a data point i for display
img = np.reshape(X[4,:],(24,24)) # convert vectorized data to 24x24 image p
plt.imshow( img.T , cmap="gray",vmin=0,vmax=255) #
```

Out[187]: <matplotlib.image.AxesImage at 0x11b0a2be0>




In [239]:
```
mean = []
for image in range(len(X[0])):
    mean.append(np.mean(X[:,image]))
newX = []
for image in X:
    newX.append(np.subtract(image,mean))
img = np.reshape(mean,(24,24)) # convert vectorized data to 24x24 image pat
plt.imshow( img.T , cmap="gray",vmin=0,vmax=255) #
print(np.array(mean).shape)
```

(576,)




2b.

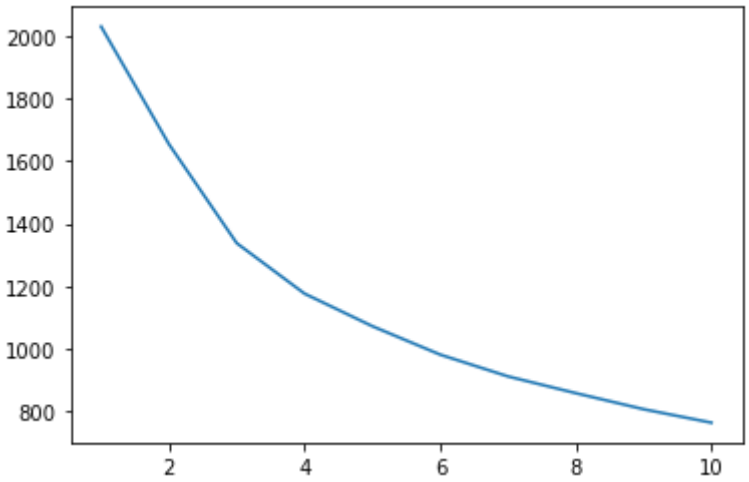
In [229]:
```
import scipy.linalg
U,s,Vh = scipy.linalg.svd(newX, full_matrices=False)
W = U.dot(np.diag(s))
print(W.shape)
print(Vh.shape)
```

```
(4916, 576)
(576, 576)
```

2c.

In [231]:
```
k = [1,2,3,4,5,6,7,8,9,10]
mean2 = []
for i in k:
    Xhat = np.dot(W[:,:i],Vh[:i,:])
    mean2.append(np.mean((newX - Xhat)**2 ))
plt.plot(k,mean2)
```

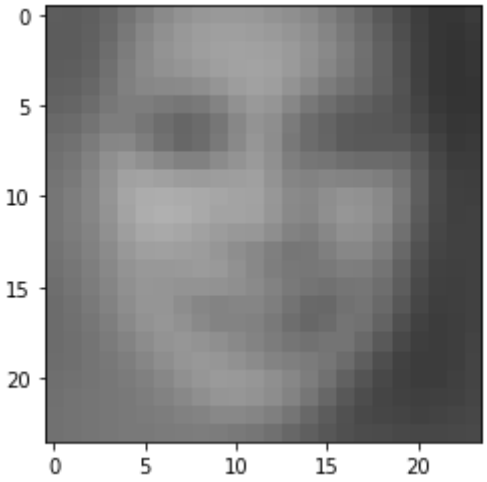
Out[231]:
```
[<matplotlib.lines.Line2D at 0x12a4011c0>]
```

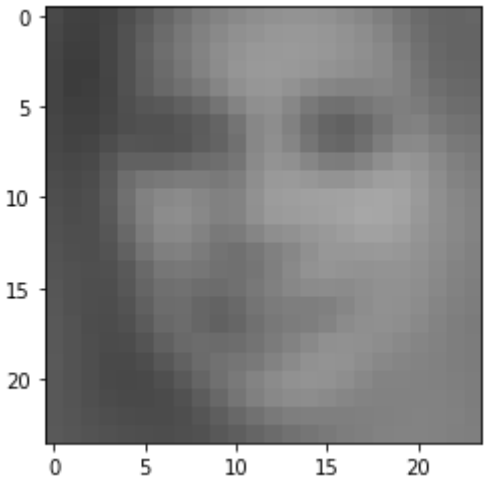



2d.

In [267]:
```
#for j in range(1,6):
alpha = 2*np.median(np.abs(W[:,2]))
dir1 = mean + (alpha * Vh[2,:])
dir2 = mean - (alpha * Vh[2,:])
img1 = np.reshape(dir1,(24,24))
img2 = np.reshape(dir2,(24,24))
plt.imshow( img1.T , cmap="gray",vmin=0,vmax=255)
plt.figure()
plt.imshow( img2.T , cmap="gray",vmin=0,vmax=255)
```

Out[267]:
```
<Figure size 432x288 with 0 Axes>
```



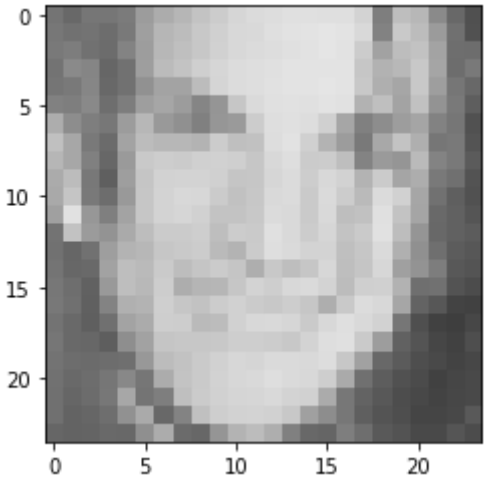

```
<Figure size 432x288 with 0 Axes>
```

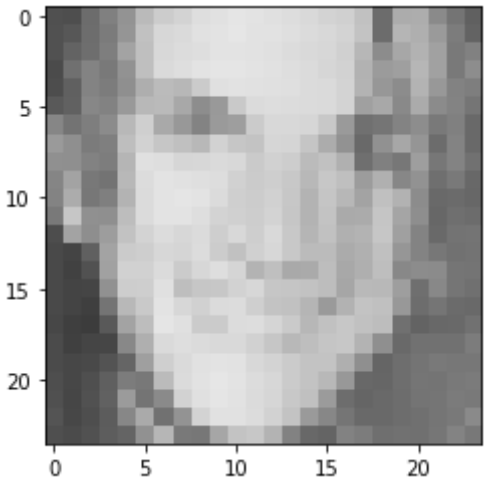
2e.

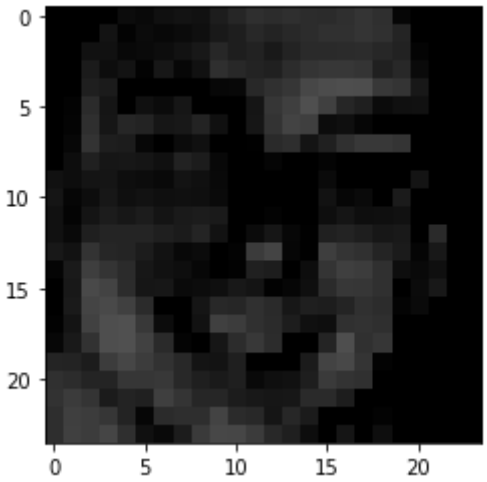
K = 5

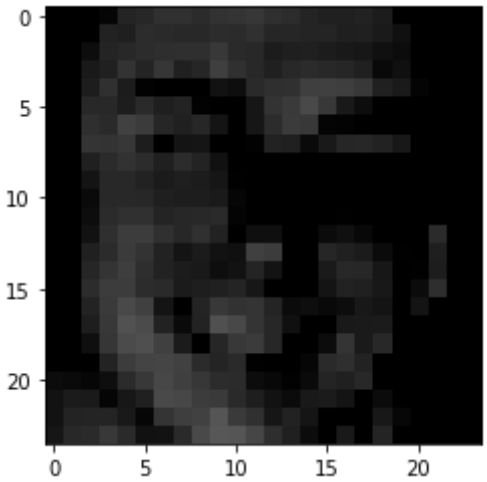
In [305]:
```
alpha = 2*np.median(np.abs(W[:,5]))
dir1 = X[500] + (alpha * Vh[5,:])
dir2 = X[500] - (alpha * Vh[5,:])
dir3 = newX[600] + (alpha * Vh[5,:])
dir4 = newX[600] - (alpha * Vh[5,:])
img1 = np.reshape(dir1,(24,24))
img2 = np.reshape(dir2,(24,24))
img3 = np.reshape(dir3,(24,24))
img4 = np.reshape(dir4,(24,24))
plt.imshow( img1.T , cmap="gray",vmin=0,vmax=255)
plt.figure()
plt.imshow( img2.T , cmap="gray",vmin=0,vmax=255)
plt.figure()
plt.imshow( img3.T , cmap="gray",vmin=0,vmax=255)
plt.figure()
plt.imshow( img4.T , cmap="gray",vmin=0,vmax=255)
```

Out[305]:
```
<matplotlib.image.AxesImage at 0x12c3a9f40>
```




0
5
10
15
20
0
5
10
15
20

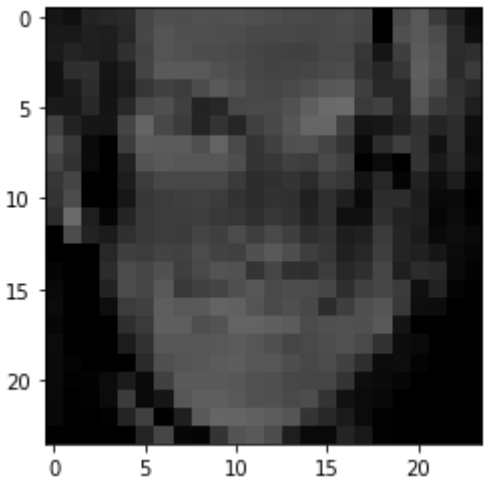

0
5
10
15
20
0
5
10
15
20

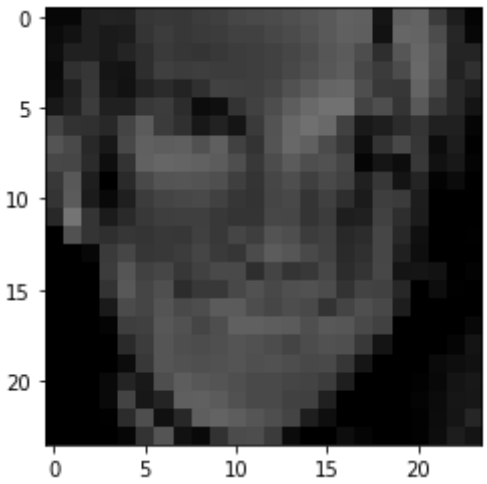

0
5
10
15
20
0
5
10
15
20

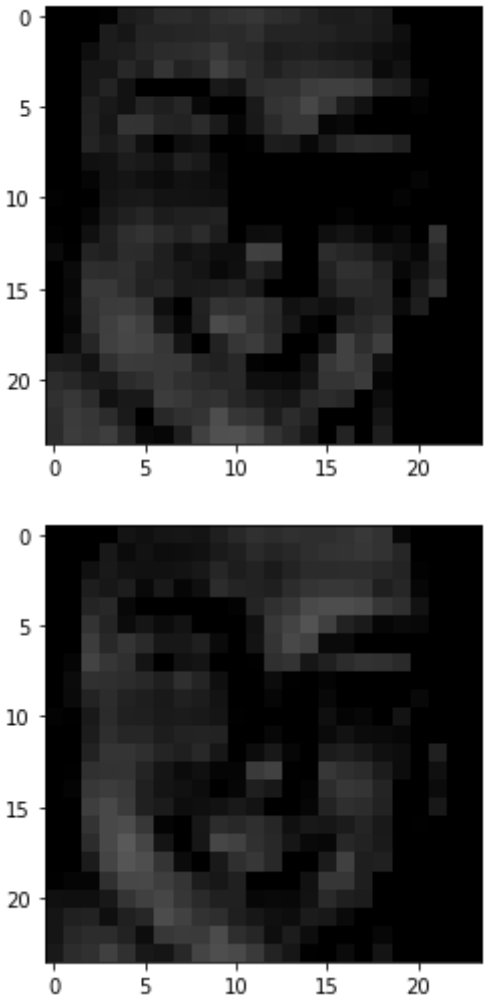
```
K = 10
```

In [306]:
```
alpha = 2*np.median(np.abs(W[:,10]))
dir1 = newX[500] + (alpha * Vh[10,:])
dir2 = newX[500] - (alpha * Vh[10,:])
dir3 = newX[600] + (alpha * Vh[10,:])
dir4 = newX[600] - (alpha * Vh[10,:])
img1 = np.reshape(dir1,(24,24))
img2 = np.reshape(dir2,(24,24))
img3 = np.reshape(dir3,(24,24))
img4 = np.reshape(dir4,(24,24))
plt.imshow( img1.T , cmap="gray",vmin=0,vmax=255)
plt.figure()
plt.imshow( img2.T , cmap="gray",vmin=0,vmax=255)
plt.figure()
plt.imshow( img3.T , cmap="gray",vmin=0,vmax=255)
plt.figure()
plt.imshow( img4.T , cmap="gray",vmin=0,vmax=255)
```

Out[306]:
```
<matplotlib.image.AxesImage at 0x12c4e0fd0>
```



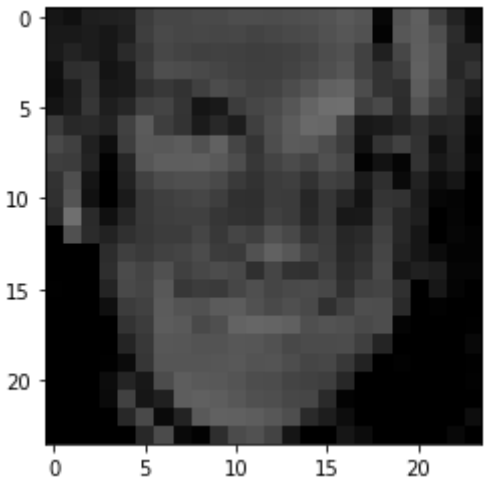



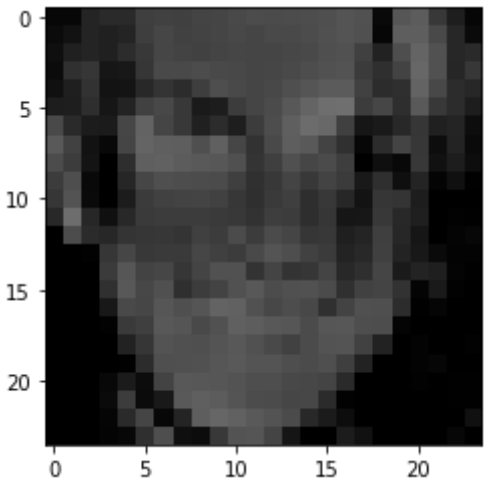


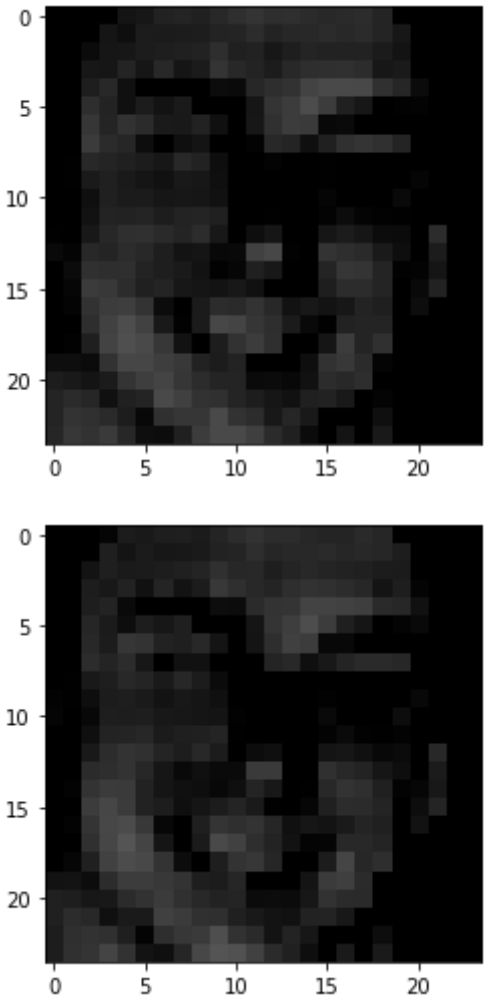

```
K = 50
```

In [307]:
```
alpha = 2*np.median(np.abs(W[:,50]))
dir1 = newX[500] + (alpha * Vh[50,:])
dir2 = newX[500] - (alpha * Vh[50,:])
dir3 = newX[600] + (alpha * Vh[50,:])
dir4 = newX[600] - (alpha * Vh[50,:])
img1 = np.reshape(dir1,(24,24))
img2 = np.reshape(dir2,(24,24))
img3 = np.reshape(dir3,(24,24))
img4 = np.reshape(dir4,(24,24))
plt.imshow( img1.T , cmap="gray",vmin=0,vmax=255)
plt.figure()
plt.imshow( img2.T , cmap="gray",vmin=0,vmax=255)
plt.figure()
plt.imshow( img3.T , cmap="gray",vmin=0,vmax=255)
plt.figure()
plt.imshow( img4.T , cmap="gray",vmin=0,vmax=255)
```

Out[307]:
```
<matplotlib.image.AxesImage at 0x12ae09d30>
```



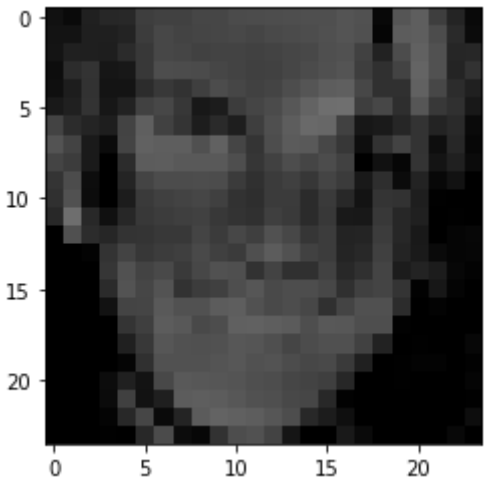



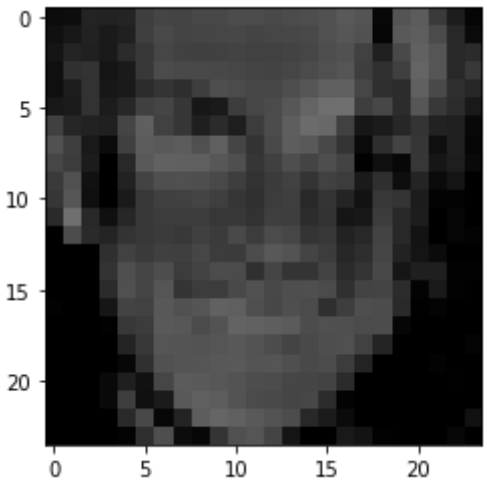


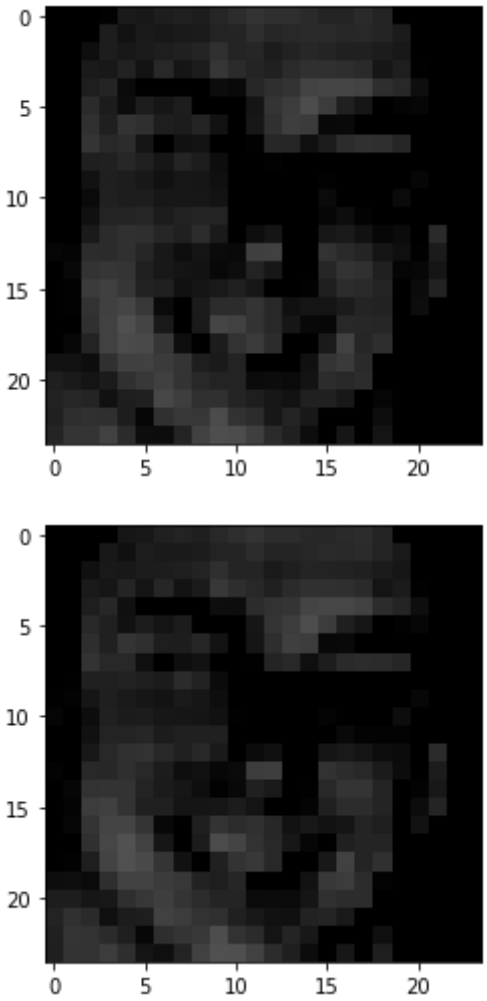

In [ ]:
```
K = 100
```

In [308]:
```
alpha = 2*np.median(np.abs(W[:,100]))
dir1 = newX[500] + (alpha * Vh[100,:])
dir2 = newX[500] - (alpha * Vh[100,:])
dir3 = newX[600] + (alpha * Vh[100,:])
dir4 = newX[600] - (alpha * Vh[100,:])
img1 = np.reshape(dir1,(24,24))
img2 = np.reshape(dir2,(24,24))
img3 = np.reshape(dir3,(24,24))
img4 = np.reshape(dir4,(24,24))
plt.imshow( img1.T , cmap="gray",vmin=0,vmax=255)
plt.figure()
plt.imshow( img2.T , cmap="gray",vmin=0,vmax=255)
plt.figure()
plt.imshow( img3.T , cmap="gray",vmin=0,vmax=255)
plt.figure()
plt.imshow( img4.T , cmap="gray",vmin=0,vmax=255)
```

Out[308]:
```
<matplotlib.image.AxesImage at 0x12ac71790>
```





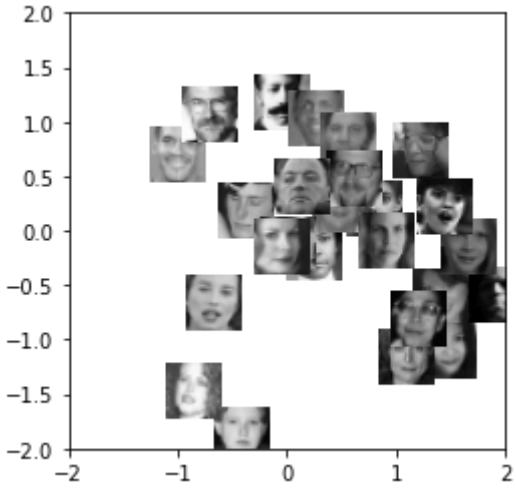

```
2f.
```

In [281]:
```
idx = [1,5,10,3,8,159,210,589,75,92,1000,39,4902,3524,68,938,382,190,1230,8

import mltools as ml
import mltools.transforms

coord,params = ml.transforms.rescale( W[:,0:2] ) # normalize scale of "W" l
plt.figure();
for i in idx:
# compute where to place image (scaled W values) & size
    loc = (coord[i,0],coord[i,0]+0.5, coord[i,1],coord[i,1]+0.5)
    img = np.reshape( X[i,:], (24,24) ) # reshape to square
    plt.imshow( img.T , cmap="gray", extent=loc, vmin=0,vmax=255 ) # draw e
    plt.axis( (-2,2,-2,2) )
```

```
25
```




```
Statement of Collaboration: I did not discuss this homework with anyone
```