In [1]:
```
# importing the library
from PIL import Image
import matplotlib.pyplot as plt
import numpy as np
```

In [2]:
```
# read image
image=Image.open(r'C:\Users\hp\Downloads/tree.jpg')
image.show()
plt.imshow(image)
```

Out[2]:
```
<matplotlib.image.AxesImage at 0x20adf6097f0>
```

In [41]:
```
# image watermark
size = (70, 120)
crop_image = image.copy()
crop_image.thumbnail(size)

# add watermark
copied_image = image.copy()
copied_image.paste(crop_image, (180, 10))
plt.imshow(copied_image)
```

Out[41]:
```
<matplotlib.image.AxesImage at 0x20ae2a34430>
```

In [3]:
```
import cv2
import glob
import os
```

In [5]:
```
logo = cv2.imread(r'C:\Users\hp\Downloads/tree.jpg')
```

In [7]:
```
h_logo = logo.shape
print(h_logo)
```

```
(100, 150, 3)
```

In [8]:
```
w_logo = logo.shape
print(w_logo)
```

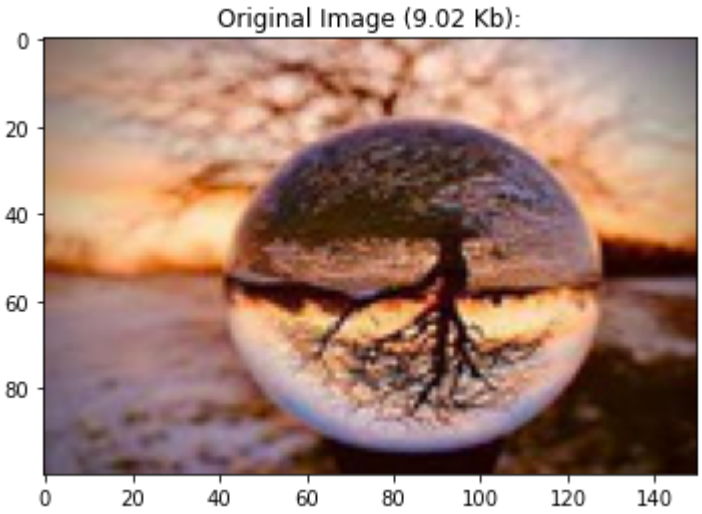
```
(100, 150, 3)
```

In [12]:
```
logo = cv2.imread(r'C:\Users\hp\Downloads/wm.jpg')
```

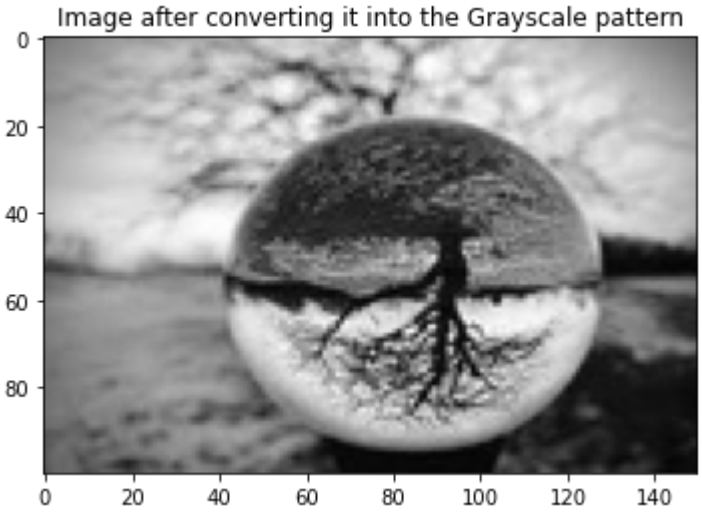
In [16]:
```
import os
import numpy as np
from PIL import Image
import matplotlib.pyplot as plt

path = r'C:\Users\hp\Downloads/tree.jpg'
img = Image.open(path)
s = float(os.path.getsize(path))/1000
print("Size(dimension): ",img.size)
plt.title("Original Image (%0.2f Kb):" %s)
plt.imshow(img)
```

```
Size(dimension):  (150, 100)
```

Out[16]:
```
<matplotlib.image.AxesImage at 0x1c7f64d8040>
```

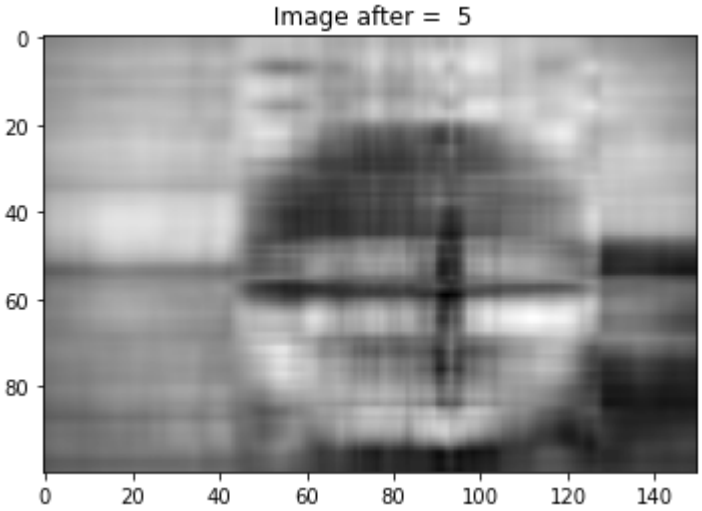


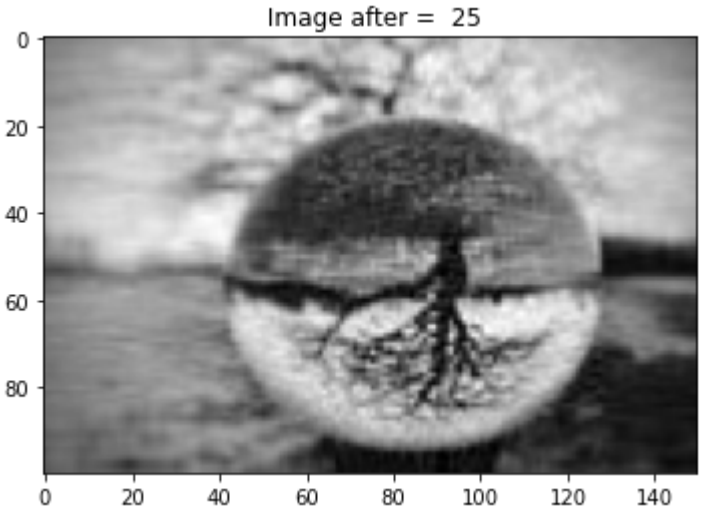
In [17]:
```
imggray = img.convert('LA')
imgmat = np.array( list(imggray.getdata(band = 0)), float)
imgmat.shape = (imggray.size[1], imggray.size[0])
imgmat = np.matrix(imgmat)
plt.figure()
plt.imshow(imgmat, cmap = 'gray')
plt.title("Image after converting it into the Grayscale pattern")
plt.show()
```

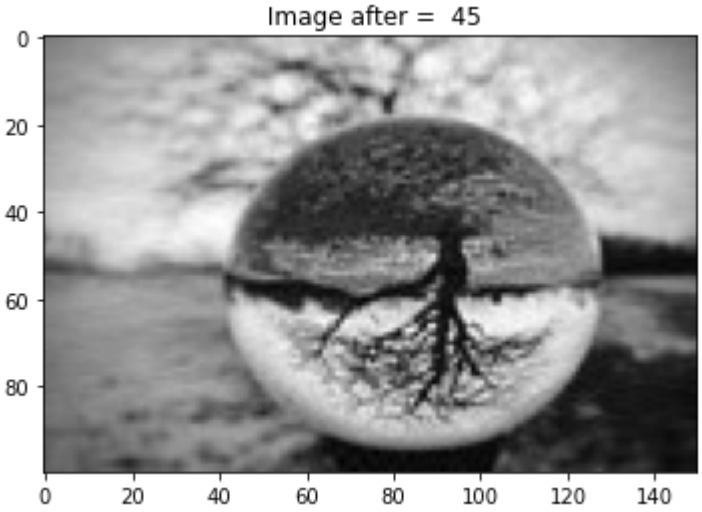


In [18]:
```
print("After compression: ")
U, S, Vt = np.linalg.svd(imgmat) #single value decomposition
for i in range(5, 51, 20):
    cmpimg = np.matrix(U[:, :i]) * np.diag(S[:i]) * np.matrix(Vt[:i,:])
    plt.imshow(cmpimg, cmap = 'gray')
    title = " Image after =  %s" %i
    plt.title(title)
    plt.show()
    result = Image.fromarray((cmpimg ).astype(np.uint8))
result.save('compressed.jpg')
```

After compression:







In [19]:
```
import pywt
cA, cD = pywt.dwt([1, 0, 0, 1], 'db1')
print (cA)
print (cD)
```

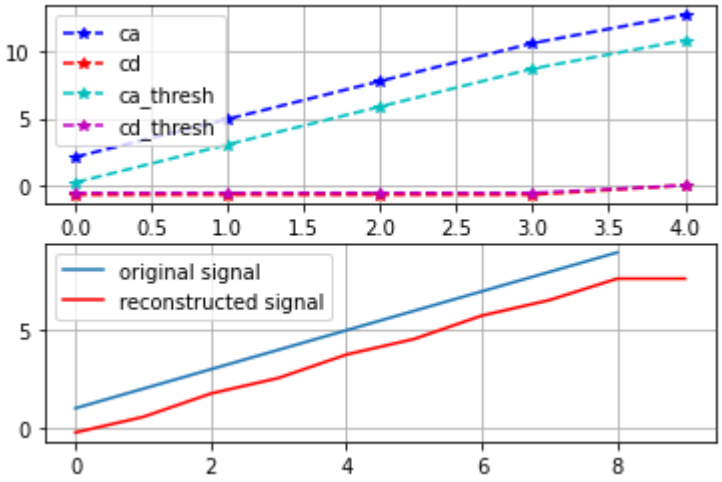
```
[0.70710678 0.70710678]
[ 0.70710678 -0.70710678]
```

In [20]:
```
import pywt
coef = pywt.wavedec([1, 2, 3, 4], 'db1', level=2)
cA, cD, cE = coef
print (cA)
print (cD)
print (cE)
```

```
[5.]
[-2.]
[-0.70710678 -0.70710678]
```

In [22]:

```
import pywt
import matplotlib.pyplot as plt
import numpy as np

ts = [1,2,3,4,5,6,7,8,9]

(ca, cd) = pywt.dwt(ts,'haar')

cat = pywt.threshold(ca, np.std(ca)/2, mode='soft')
cdt = pywt.threshold(cd, np.std(cd)/2, mode='soft')

ts_rec = pywt.idwt(cat, cdt, 'haar')

plt.close('all')

plt.subplot(211)
# Original coefficients
plt.plot(ca, '--*b')
plt.plot(cd, '--*r')
# Thresholded coefficients
plt.plot(cat, '--*c')
plt.plot(cdt, '--*m')
plt.legend(['ca','cd','ca_thresh', 'cd_thresh'], loc=0)
plt.grid('on')

plt.subplot(212)
plt.plot(ts)
# plt.hold('on')
plt.plot(ts_rec, 'r')
plt.legend(['original signal', 'reconstructed signal'])
plt.grid('on')
plt.show()
```



In [24]:
```
import numpy as np
from scipy.linalg import svd
"""Singular Value Decomposition"""
# define a matrix
X = np.array([[1,0,1,0], [0,1,0,1]])
print(X)
# perform SVD
U, singular, V_transpose = svd(X)
# print different components
print("U: ",U)
print("Singular array",singular)
print("V^{T}",V_transpose)
```

```
[[1 0 1 0]
 [0 1 0 1]]
U:  [[1. 0.]
 [0. 1.]]
Singular array [1.41421356 1.41421356]
V^{T} [[ 0.70710678  0.          0.70710678  0.        ]
 [-0.          0.70710678  0.          0.70710678]
 [-0.70710678  0.          0.70710678  0.        ]
 [ 0.         -0.70710678  0.          0.70710678]]
```

In [ ]: