Understanding Images and Pixels

Import required libraries

In [1]:
```
import numpy as np
import cv2
import matplotlib.pyplot as plt

%matplotlib inline
```

Create dummy array

In [2]:
```
arr = np.arange(0,100,1)
```

In [3]:
```
arr.shape
```

Out[3]:
```
(100,)
```

In [4]:
```
arr
```

Out[4]:
```
array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16,
       17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33,
       34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50,
       51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67,
       68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84,
       85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99])
```

In [5]:
```
arr1 = arr.reshape(10,10)
```

In [6]:
```
arr1.shape
```

Out[6]:
```
(10, 10)
```

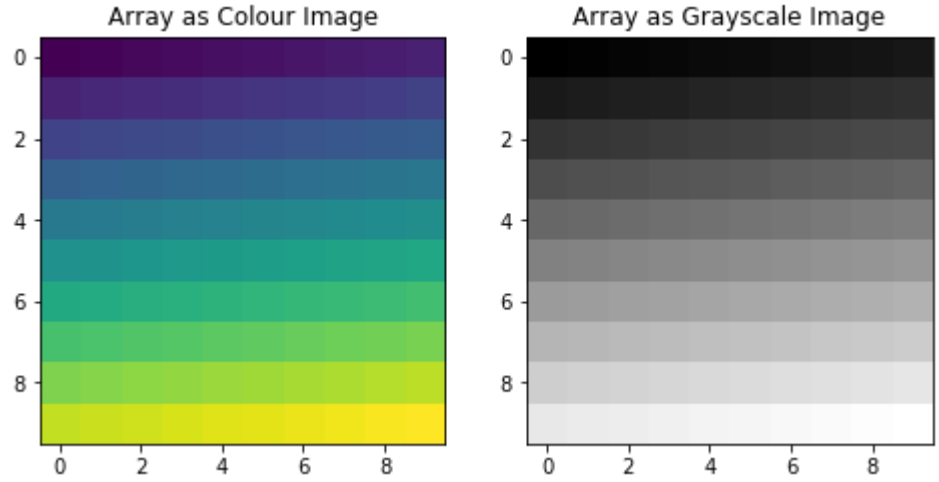
In [7]:
```
arr1
```

Out[7]:
```
array([[ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9],
       [10, 11, 12, 13, 14, 15, 16, 17, 18, 19],
       [20, 21, 22, 23, 24, 25, 26, 27, 28, 29],
       [30, 31, 32, 33, 34, 35, 36, 37, 38, 39],
       [40, 41, 42, 43, 44, 45, 46, 47, 48, 49],
       [50, 51, 52, 53, 54, 55, 56, 57, 58, 59],
       [60, 61, 62, 63, 64, 65, 66, 67, 68, 69],
       [70, 71, 72, 73, 74, 75, 76, 77, 78, 79],
       [80, 81, 82, 83, 84, 85, 86, 87, 88, 89],
       [90, 91, 92, 93, 94, 95, 96, 97, 98, 99]])
```

Visualize the array as an image

In [8]:
```
plt.figure(figsize=(8,8))
plt.subplot(1,2,1)
plt.title('Array as Colour Image')
plt.imshow(arr1)
plt.subplot(1,2,2)
plt.title('Array as Grayscale Image')
plt.imshow(arr1,cmap='gray')
```

Out[8]:
```
<matplotlib.image.AxesImage at 0x23849b94518>
```



Generate an array of random numbers

In [9]:
```
arr2 = np.random.randint(0,150,(10,10))
```

In [10]:
```
arr2.shape
```

Out[10]:
```
(10, 10)
```

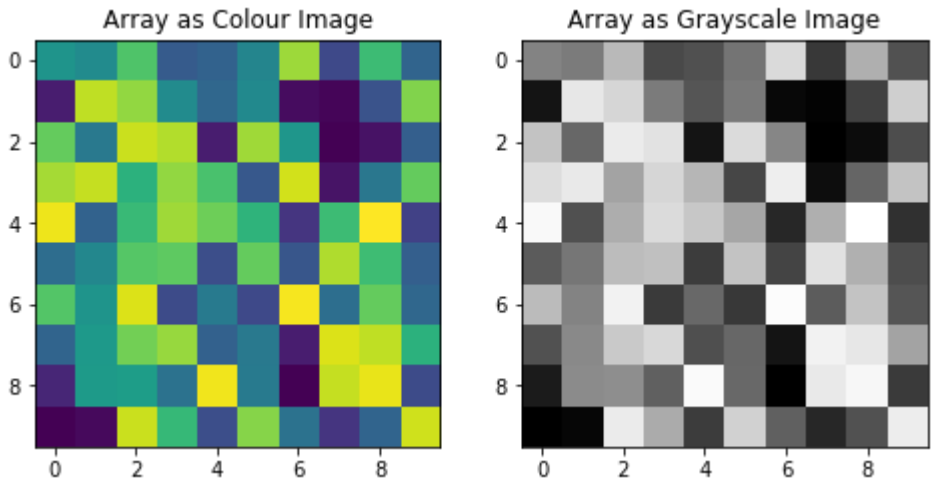
In [11]:
```
arr2
```

Out[11]:
```
array([[ 77,  72, 108,  43,  47,  68, 127,  33, 102,  48],
       [ 12, 135, 125,  72,  50,  71,   5,   2,  38, 121],
       [114,  60, 137, 132,  12, 128,  78,   0,   7,  45],
       [129, 136,  96, 125, 106,  41, 139,   8,  59, 114],
       [145,  47, 101, 128, 116,  97,  23, 102, 149,  29],
       [ 53,  69, 110, 112,  36, 114,  40, 131, 103,  45],
       [109,  77, 141,  34,  61,  33, 147,  54, 114,  50],
       [ 48,  80, 117, 126,  46,  60,  12, 141, 135,  96],
       [ 16,  81,  83,  56, 146,  61,   0, 136, 144,  34],
       [  0,   4, 137, 100,  36, 122,  56,  23,  48, 138]])
```

Visualizing an array of random numbers as an image

In [12]:
```
plt.figure(figsize=(8,8))
plt.subplot(1,2,1)
plt.title('Array as Colour Image')
plt.imshow(arr2)
plt.subplot(1,2,2)
plt.title('Array as Grayscale Image')
plt.imshow(arr2,cmap='gray')
```

Out[12]: <matplotlib.image.AxesImage at 0x23849e15c88>



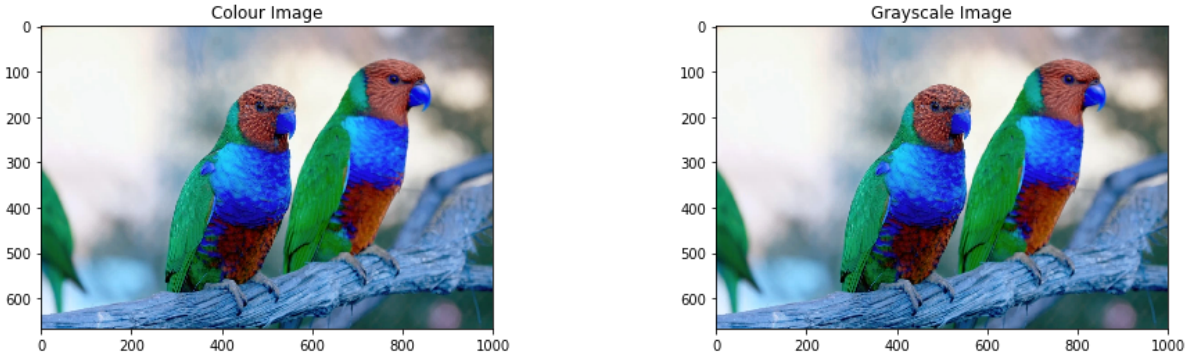
Images have pixels. Each pixel has a range of 0-(2^n -1). eg- for 8 bit image, it is 0-255. Lower the value, darker the color. Higher the value, lighter the color. eg- for grayscale black is 0 and white is 255. So, images are basically an array of numbers.

Read an image and visualize it

In [13]:
```
img_cv = cv2.imread('./Data/birds.jpg')

plt.figure(figsize=(16,4))
plt.subplot(1,2,1)
plt.title('Colour Image')
plt.imshow(img_cv)
plt.subplot(1,2,2)
plt.title('Grayscale Image')
plt.imshow(img_cv,cmap='gray')
#Since image is not converted to grayscale yet, cmap='gray' will show the same results
```

Out[13]: <matplotlib.image.AxesImage at 0x23849e8a208>



In [14]:
```
img_cv.shape
```

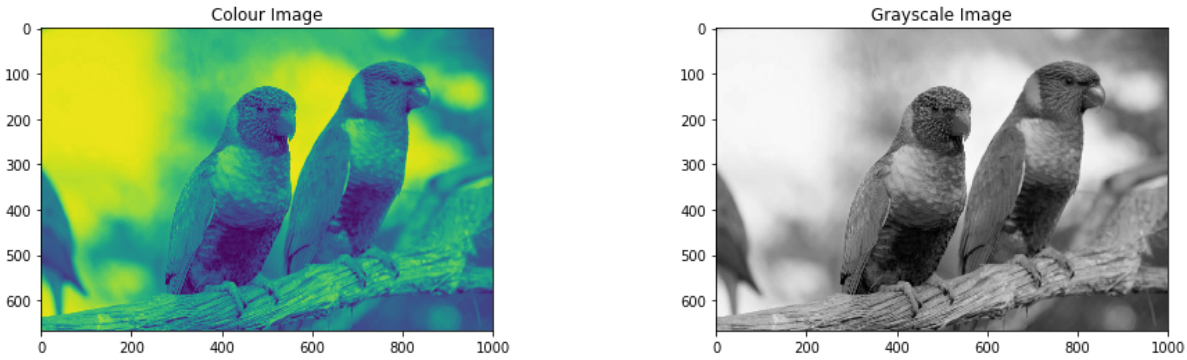
Out[14]: (667, 1000, 3)

Convert color image to grayscale

In [15]:
```
gray_img_cv =  cv2.cvtColor(img_cv,cv2.COLOR_BGR2GRAY)

plt.figure(figsize=(16,4))
plt.subplot(1,2,1)
plt.title('Colour Image')
plt.imshow(gray_img_cv)
plt.subplot(1,2,2)
plt.title('Grayscale Image')
plt.imshow(gray_img_cv,cmap='gray')
```

Out[15]: <matplotlib.image.AxesImage at 0x23849f058d0>



In [16]:
```
gray_img_cv.shape
```

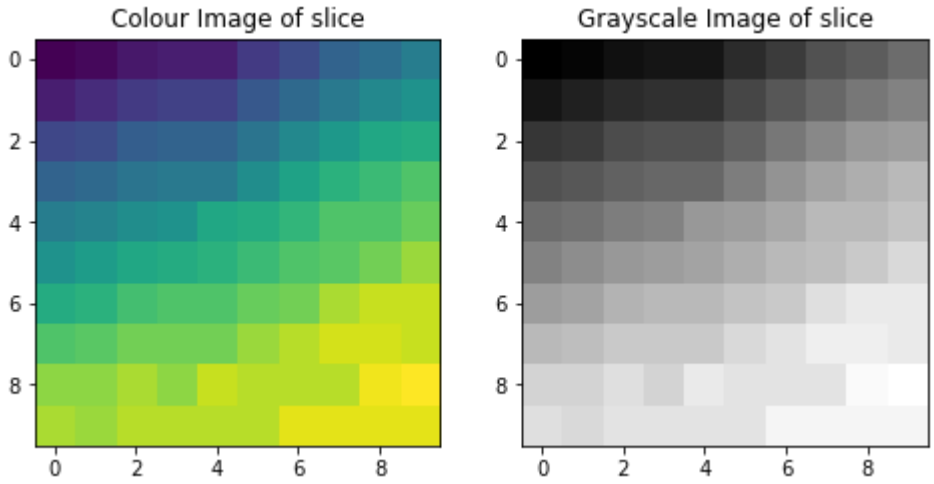
Out[16]:
```
(667, 1000)
```

Slice different parts of the image and analyze

In [19]:
```
sliced_gray = gray_img_cv[0:10,0:10]

plt.figure(figsize=(8,8))
plt.subplot(1,2,1)
plt.title('Colour Image of slice')
plt.imshow(sliced_gray)
plt.subplot(1,2,2)
plt.title('Grayscale Image of slice')
plt.imshow(sliced_gray,cmap='gray')
```

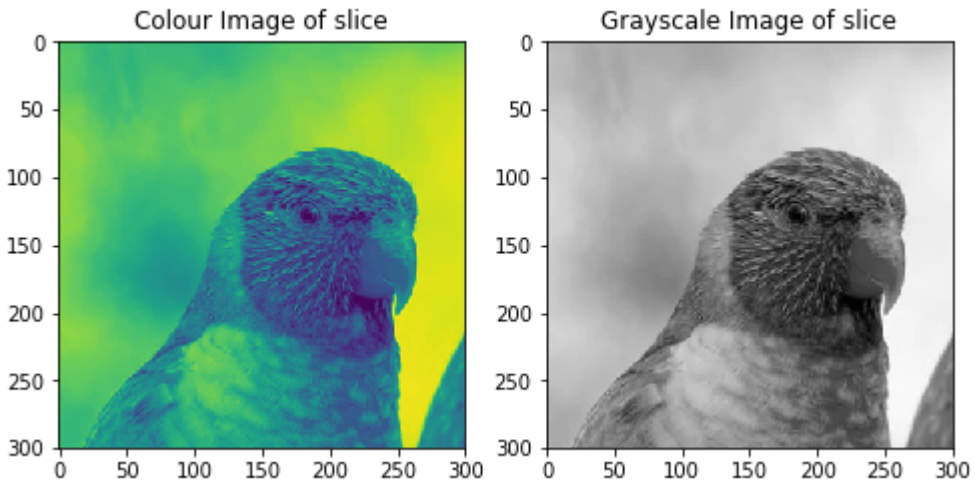
Out[19]:
```
<matplotlib.image.AxesImage at 0x2384db15588>
```



In [28]:
```
sliced_gray2 = gray_img_cv[50:350,300:600]

plt.figure(figsize=(8,8))
plt.subplot(1,2,1)
plt.title('Colour Image of slice')
plt.imshow(sliced_gray2)
plt.subplot(1,2,2)
plt.title('Grayscale Image of slice')
plt.imshow(sliced_gray2,cmap='gray')
```

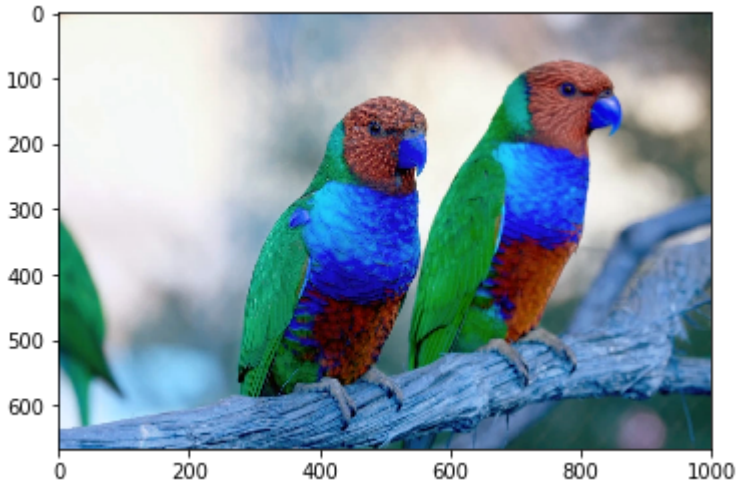
Out[28]: <matplotlib.image.AxesImage at 0x2384dd22ac8>



Resizing Images:

In [29]:
```
plt.imshow(img_cv)
```

Out[29]: <matplotlib.image.AxesImage at 0x2384dd53cf8>



In [30]:
```
img_cv.shape
```

Out[30]: (667, 1000, 3)

Resizing are of 2 types, shrinking and enlarging.

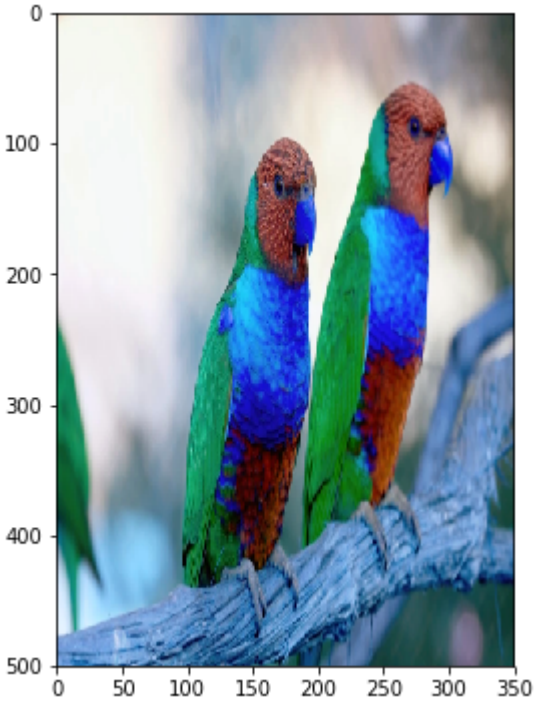
Shrink image:

In [31]:
```
shrunk_img_cv = cv2.resize(img_cv,(350,500),cv2.INTER_AREA)
shrunk_img_cv.shape
```

Out[31]: (500, 350, 3)

In [38]:
```
plt.figure(figsize=(10,6))
plt.imshow(shrunk_img_cv)
```

Out[38]: <matplotlib.image.AxesImage at 0x2384e1f3a90>



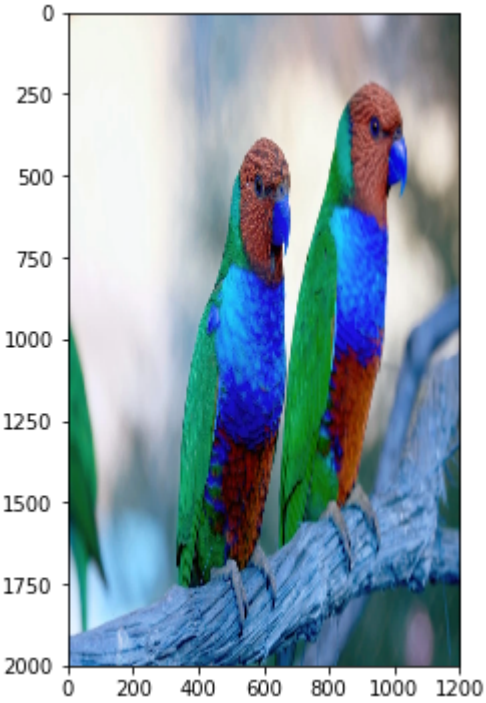
Enlarge Image:

In [35]:
```
enlarge_img_cv = cv2.resize(img_cv,(1200,2000),cv2.INTER_CUBIC)
enlarge_img_cv.shape
#For enlarging images from small size, INTER_CUBIC is better than INTER_AREA
```

Out[35]: (2000, 1200, 3)

In [41]:
```
plt.figure(figsize=(10,6))
plt.imshow(enlarge_img_cv)
```

Out[41]: <matplotlib.image.AxesImage at 0x2384e1cd6a0>



Enlarging image results in insertion of more no of duplicate values, hence image might become blurred

In [ ]:
```
```