Bit Plane Slicing

In Bit-plane slicing, we divide the image into bit planes. This is done by first converting the pixel values in the binary form and then dividing it into bit planes

In [7]:
```
import numpy as np
from PIL import Image
import matplotlib.pyplot as plt
```

In [8]:
```
# reading image and converting to gray scale
img = Image.open('../images/tiger.jpg').convert('L')
# display image
img
```

Out[8]:

In [9]:
```
def bit_slicing_function(input_image):
  img = input_image.resize((400,400), Image.Resampling.LANCZOS)
  # convert to numpy array
  numpy_image = np.array(img)

  fig = plt.figure()
  fig.set_figheight(20)
  fig.set_figwidth(20)

  fig.add_subplot(3,3,1)
  plt.imshow(img, cmap='gray')
  plt.title('original image')

  #Iterate over each pixel and change pixel value to binary using
np.binary_repr() and store it in a list.
  lst = []
  for i in range(numpy_image.shape[0]):
```

```
    for j in range(numpy_image.shape[1]):
      lst.append(np.binary_repr(numpy_image[i][j] ,width=8)) # width = no. of
bits

  # We have a list of strings where each string represents binary pixel value.
To extract bit planes we need to iterate over the strings and store the
characters corresponding to bit planes into lists.
  # Multiply with 2^(n-1) and reshape to reconstruct the bit image.
  eight_bit_img = (np.array([int(i[0]) for i in lst],dtype = np.uint8) *
128).reshape(400,400)
  eight_bit_image = Image.fromarray(eight_bit_img)

  fig.add_subplot(3,3,2)
  plt.imshow(eight_bit_image, cmap='gray')
  plt.title('8 bit')


  seven_bit_img = (np.array([int(i[1]) for i in lst],dtype = np.uint8) *
64).reshape(400,400)
  seven_bit_image = Image.fromarray(seven_bit_img)

  fig.add_subplot(3,3,3)
  plt.imshow(seven_bit_image, cmap='gray')
  plt.title('7 bit')

  six_bit_img = (np.array([int(i[2]) for i in lst],dtype = np.uint8) *
32).reshape(400,400)
  six_bit_img = Image.fromarray(six_bit_img)

  fig.add_subplot(3,3,4)
  plt.imshow(six_bit_img, cmap='gray')
  plt.title('6 bit')


  five_bit_img = (np.array([int(i[3]) for i in lst],dtype = np.uint8) *
16).reshape(400,400)
  five_bit_img = Image.fromarray(five_bit_img)

  fig.add_subplot(3,3,5)
  plt.imshow(five_bit_img, cmap='gray')
  plt.title('5 bit')

  four_bit_img = (np.array([int(i[4]) for i in lst],dtype = np.uint8) *
8).reshape(400,400)
  four_bit_img = Image.fromarray(four_bit_img)

  fig.add_subplot(3,3,6)
  plt.imshow(four_bit_img, cmap='gray')
  plt.title('4 bit')


  three_bit_img = (np.array([int(i[5]) for i in lst],dtype = np.uint8) *
4).reshape(400,400)
  three_bit_img = Image.fromarray(three_bit_img)

  fig.add_subplot(3,3,7)
  plt.imshow(three_bit_img, cmap='gray')
  plt.title('3 bit')

  two_bit_img = (np.array([int(i[6]) for i in lst],dtype = np.uint8) *
```

```
2).reshape(400,400)
  two_bit_img = Image.fromarray(two_bit_img)

  fig.add_subplot(3,3,8)
  plt.imshow(two_bit_img, cmap='gray')
  plt.title('2 bit')

  one_bit_img = (np.array([int(i[7]) for i in lst],dtype = np.uint8) *
1).reshape(400,400)
  one_bit_img = Image.fromarray(one_bit_img)

  fig.add_subplot(3,3,9)
  plt.imshow(one_bit_img, cmap='gray')
  plt.title('1 bit')
```

In [10]:
```
bit_slicing_function(img)
```