**VGG-19 model**

def load\_vgg(layer\_names):

  vgg = tf.keras.applications.VGG19(include\_top=False, weights='imagenet')

  vgg.trainable = False

  content\_layers = ['block5\_conv2']

  style\_layers = ['block1\_conv1',

                  'block2\_conv1',

                  'block3\_conv1',

                  'block4\_conv1',

                  'block5\_conv1']

  num\_content\_layers = len(content\_layers)

  num\_style\_layers = len(style\_layers)

  outputs = [vgg.get\_layer(name).output for name in layer\_names]

  model = Model([vgg.input], outputs)

  return model

**Optimizer**

optimizer = tf.optimizers.Adam(learning\_rate=0.01, beta\_1=0.99, epsilon=1e-1)

**Style and content weights**

style\_weight=1e-2

content\_weight=1e-1

**Content and loss function**

style\_loss = tf.add\_n([tf.reduce\_mean((style\_outputs[name]-style\_targets[name])\*\*2)

content\_loss = tf.add\_n([tf.reduce\_mean((content\_outputs[name]-content\_targets[name])\*\*2)

**Training**

start = time.time()

epochs = 100

step = 0

for i in range(epochs):

  step = step + 1

  train\_step(image)

  print(".", end = '', flush = True)

print("\n")

print("Successfully completed {}".format(step) + " epochs.")

end = time.time()

print("Total time: {}".format(end-start))

display.display(tensor\_to\_image(image))