PIL Intro - Basic PIL Processing

Basic PIL operations for image processing.

In [1]:
```
import PIL
from PIL import Image
from IPython.display import display
ifile = "Lenna.png"
image = Image.open(ifile)
```

Make a copy of the image to use for modification. Note the conversion to RGB when the image is being opened.

In [2]:
```
image.save("Lenna-copy.png")
image = Image.open("Lenna-copy.png").convert('RGB')
```

Now show the image.

In [3]:
```
display(image)
```

This image will be used to make one larger image named `gradimg` which will show the gradient of various effects from PIL's `ImageEnhance` module. To view the `gradimg` easily the copy will be scaled to half size. Each image with increased effect will be stored in the list `imglist` and modifed with an enhancer object. Let's start with brightness.

In [4]:
```
from PIL import ImageEnhance

# rescale image
image.resize((256,256))

# create image modifing object
enhancer = ImageEnhance.Brightness(image)

# make image gradients
imgbuf = []
for i in range(3, 8):
    imgbuf.append(enhancer.enhance(i/5))
```

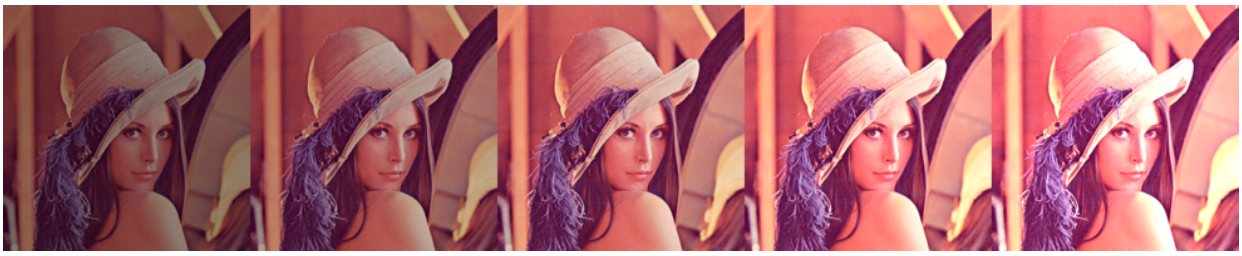
Now create a new image that stitches the 5 above into one. The `Image.new()` method requires a mode (RGB in this case) and size, which will be retrieved from the first image of `gradimg`. The images will be offset (i.e. `xpos`) by the width after each paste.

In [5]:
```
# reference image
imgref = imgbuf[0]

# create new image with 5 images from imgbuf
gradimg = Image.new(imgref.mode, (imgref.width*5, imgref.height))
xpos = 0
for img in imgbuf:
    gradimg.paste(img, (xpos, 0))
    xpos += imgref.width
```

Now, print the result!

In [6]:
```
display(gradimg)
```



This can be done easily for other enhance methods, such as contrast.

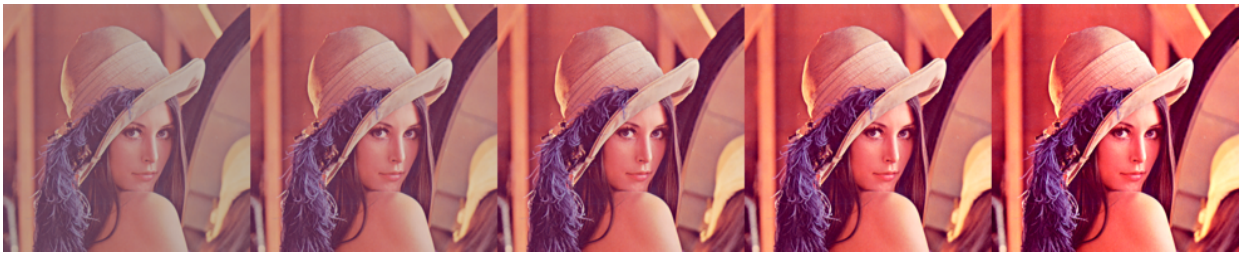
In [7]:
```
# create image modifing object
enhancer = ImageEnhance.Contrast(image)

# make image gradients
imgbuf = []
for i in range(3, 8):
    imgbuf.append(enhancer.enhance(i/5))

# reference image
imgref = imgbuf[0]

# create new image with 5 images from imgbuf
gradimg = Image.new(imgref.mode, (imgref.width*5, imgref.height))
xpos = 0
for img in imgbuf:
    gradimg.paste(img, (xpos, 0))
    xpos += imgref.width

# display image
display(gradimg)
```



Or sharpness.

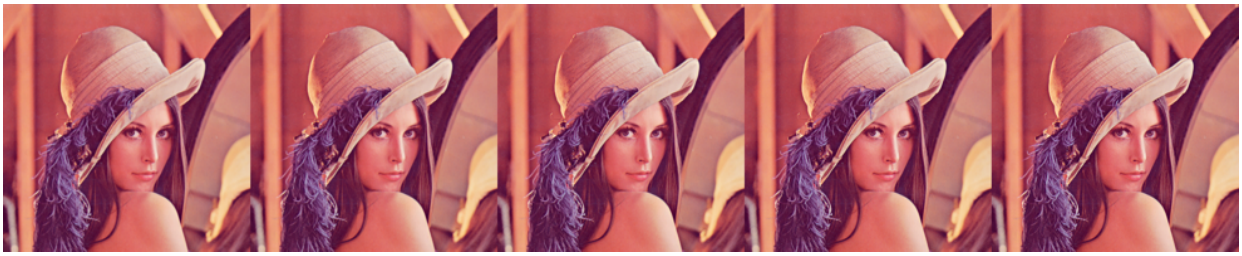
In [9]:
```
# create image modifing object
enhancer = ImageEnhance.Sharpness(image)

# make image gradients
imgbuf = []
for i in range(3, 8):
    imgbuf.append(enhancer.enhance(i/5))

# reference image
imgref = imgbuf[0]

# create new image with 5 images from imgbuf
gradimg = Image.new(imgref.mode, (imgref.width*5, imgref.height))
xpos = 0
for img in imgbuf:
    gradimg.paste(img, (xpos, 0))
    xpos += imgref.width

# display image
display(gradimg)
```



And finally, color. Here the enhance value was modified for the range $[0, 1]$.

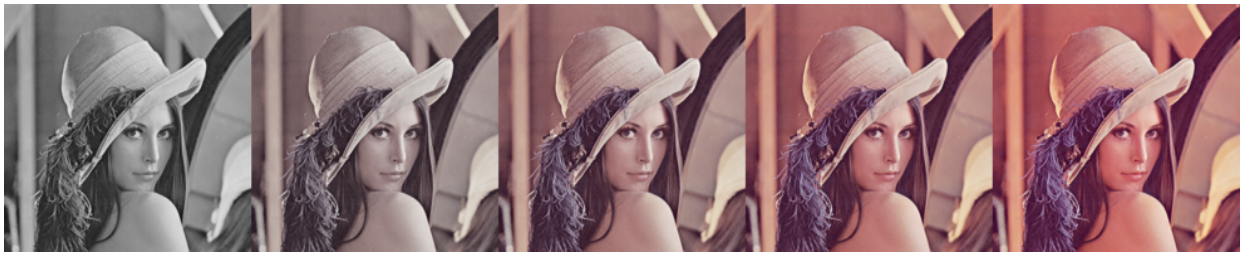
In [10]:
```
# create image modifing object
enhancer = ImageEnhance.Color(image)

# make image gradients
imgbuf = []
for i in range(0, 6):
    imgbuf.append(enhancer.enhance(i/5))

# reference image
imgref = imgbuf[0]

# create new image with 5 images from imgbuf
gradimg = Image.new(imgref.mode, (imgref.width*5, imgref.height))
xpos = 0
for img in imgbuf:
    gradimg.paste(img, (xpos, 0))
    xpos += imgref.width

# display image
display(gradimg)
```



In [ ]:
```
```