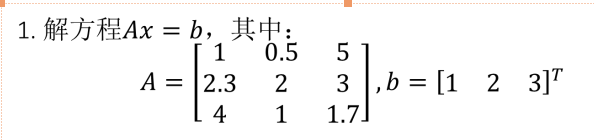
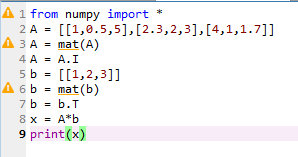
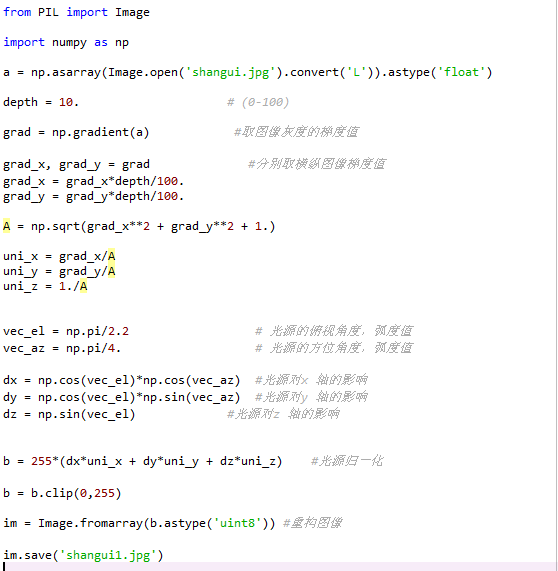
#### 课程名称：运用python语言 教师：林卫中

#### 姓名：洪佳珺 专业班级：17应用统计2班 学号：117060400213

1. 实验名称：运用python语言
2. 实验目的：利用numpy库计算矩阵；图像的手绘
3. 实验内容及步骤：

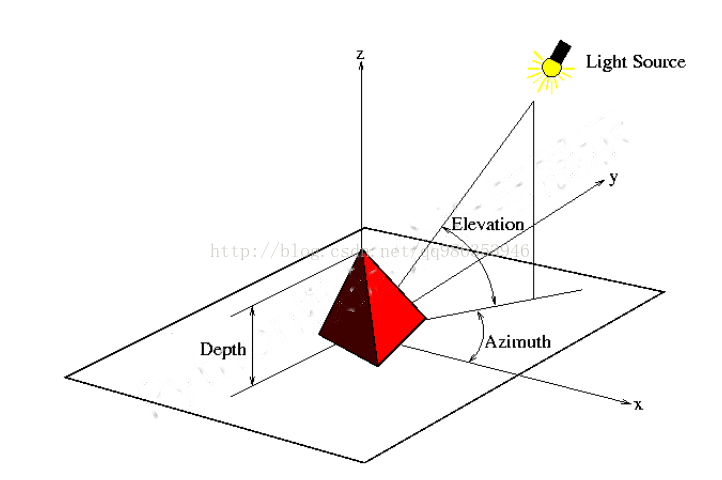






其中np.asarray(Image.open('./beijing.jpg').convert('L')).astype('float')意思是将图像以灰度图的方式打开并将数据转为float存入np中.

np.gradient(a)  是求a的梯度，返回的是二元信息，可分别赋值给grad\_x,grad\_y，将梯度按照深度等级计算并且归一化处理。

 我们还需建立光源效果：

建立模型后可分析出np.cos(vec\_el)为单位光线在地平面上的投影长度，dx,dy,dz是光源对x/y/z三方向的影响程度.

将梯度归一化

构造x和y轴梯度的三维归一化单位坐标系 A = np.sqrt(grad\_x\*\*2 + grad\_y\*\*2 +1)

梯度和光源相互作用，将梯度转化为灰度b = 255\*(dx\*uni\_x + dy\*uni\_y + dz\*uni\_z)

之前为了处理方便，我们梯度值调为0-1之间，梯度与光源相互作用后，还原为0-255，但仍然可能会有部分溢出，因此，为避免越界，将生成的灰度值剪裁至0-255区间b = b.clip(0,255)

四、实验心得：

**python的常见矩阵运算**

### 1.矩阵的创建

由一维或二维数据创建矩阵

from numpy import \*;

a1=array([1,2,3]);

a1=mat(a1);

创建常见的矩阵

data1=mat(zeros((3,3)));#创建一个3\*3的零矩阵，矩阵这里zeros函数的参数是一个tuple类型(3,3)

data2=mat(ones((2,4)));#创建一个2\*4的1矩阵，默认是浮点型的数据，如果需要时int类型，可以使用dtype=int

data3=mat(random.rand(2,2));#这里的random模块使用的是numpy中的random模块，random.rand(2,2)创建的是一个二维数组，需要将其转换成#matrix

data4=mat(random.randint(10,size=(3,3)));#生成一个3\*3的0-10之间的随机整数矩阵，如果需要指定下界则可以多加一个参数

data5=mat(random.randint(2,8,size=(2,5));#产生一个2-8之间的随机整数矩阵

data6=mat(eye(2,2,dtype=int));#产生一个2\*2的对角矩阵

a1=[1,2,3];

a2=mat(diag(a1));#生成一个对角线为1、2、3的对角矩阵

### 2.常见的矩阵运算

****1. 矩阵相乘****

a1=mat([1,2]);

a2=mat([[1],[2]]);

a3=a1\*a2;

#1\*2的矩阵乘以2\*1的矩阵，得到1\*1的矩阵

****2. 矩阵点乘****

矩阵对应元素相乘

a1=mat([1,1]);a2=mat([2,2]);a3=multiply(a1,a2);

矩阵点乘

a1=mat([2,2]);a2=a1\*2;

****3.矩阵求逆，转置****   
矩阵求逆

a1=mat(eye(2,2)\*0.5);

a2=a1.I;

#求矩阵matrix([[0.5,0],[0,0.5]])的逆矩阵

矩阵转置

a1=mat([[1,1],[0,0]]);

a2=a1.T;

****4.计算矩阵对应行列的最大、最小值、和。****

a1=mat([[1,1],[2,3],[4,2]]);

计算每一列、行的和

a2=a1.sum(axis=0);//列和，这里得到的是1\*2的矩阵

a3=a1.sum(axis=1);//行和，这里得到的是3\*1的矩阵

a4=sum(a1[1,:]);//计算第一行所有列的和，这里得到的是一个数值

计算最大、最小值和索引

a1.max();//计算a1矩阵中所有元素的最大值,这里得到的结果是一个数值

a2=max(a1[:,1]);//计算第二列的最大值，这里得到的是一个1\*1的矩阵

a1[1,:].max();//计算第二行的最大值，这里得到的是一个一个数值

np.max(a1,0);//计算所有列的最大值，这里使用的是numpy中的max函数

np.max(a1,1);//计算所有行的最大值，这里得到是一个矩阵

np.argmax(a1,0);//计算所有列的最大值对应在该列中的索引

np.argmax(a1[1,:]);//计算第二行中最大值对应在改行的索引

3.矩阵的分隔和合并   
矩阵的分隔，同列表和数组的分隔一致。

a=mat(ones((3,3)));b=a[1:,1:];//分割出第二行以后的行和第二列以后的列的所有元素

矩阵的合并

a=mat(ones((2,2)));b=mat(eye(2));c=vstack((a,b));//按列合并，即增加行数d=hstack((a,b));//按行合并，即行数不变，扩展列数

### 4.矩阵、列表、数组的转换

列表可以修改，并且列表中元素可以使不同类型的数据，如下：

l1=[[1],'hello',3];

numpy中数组，同一个数组中所有元素必须为同一个类型，有几个常见的属性：

a=array([[2],[1]]);

dimension=a.ndim;

m,n=a.shape;

number=a.size;//元素总个数

str=a.dtype;//元素的类型

numpy中的矩阵也有与数组常见的几个属性。   
它们之间的转换：

a1=[[1,2],[3,2],[5,2]];//列表

a2=array(a1);//将列表转换成二维数组

a3=array(a1);//将列表转化成矩阵

a4=array(a3);//将矩阵转换成数组

a5=a3.tolist();//将矩阵转换成列表

a6=a2.tolist();//将数组转换成列表

这里可以发现三者之间的转换是非常简单的，这里需要注意的是，当列表是一维的时候，将它转换成数组和矩阵后，再通过tolist()转换成列表是不相同的，需要做一些小小的修改。如下：

a1=[1,2,3];

a2=array(a1);

a3=mat(a1);

a4=a2.tolist();//这里得到的是[1,2,3]

a5=a3.tolist();//这里得到的是[[1,2,3]]

a6=(a4 == a5);//a6=False

a7=(a4 is a5[0]);//a7=True,a5[0]=[1,2,3]

矩阵转换成数值，存在以下一种情况：

dataMat=mat([1]);val=dataMat[0,0];

//这个时候获取的就是矩阵的元素的数值，而不再是矩阵的类型