Lab 6 二维数组

Task1 :小黄与它的罐子番外—不要踩小黄

考察：循环与二维数组

小黄隐蔽在n*n的棋盘上。我们不断踩上棋盘上的方格。如果踩到了小黄隐蔽的棋盘格子，则输掉游戏；如果没有小黄的方格被踩到，则会出现0～8之间的整数，表示包含小黄的相邻方格和对角相邻方格的数目。

如果我们不断踩到没有小黄的方格，直到只有m个小黄藏匿的方格没有被踩到，则获胜。

请编写一个程序，输入游戏进行的信息，输出相应的网格

输入：

输入的第一行给出一个正整数n（n<=10）。接下来的n行描述小黄的位置，每行用n个字符表示一行的内容：句号表示一个方格没有小黄，而星号表示这个方格有小黄。然后n行给出n个字符：被踩到的位置用x标识，未被踩到的地方用句号（.）标识。

输出：

输出给出网格，每个方格被填入适当的值。如果被踩到的地方没有小黄，则给出从0～8之间的值；如果有一个小黄被踩到，则所有有小黄的位置都用一个星号标识，其他方格用一个句号标识，游戏结束。

样例输入	样例输出
8	001…..
…**..*	0013….
……*.	0001….
….*…	00011…
……..	00001…
……..	00123…
…..*..	001…..
…**.*.	00123…
…..*..	
xxx…..	
xxxx….	
xxxx….	
xxxxx…	
xxxxx…	
xxxxx…	
xxx…..	
xxxxx…	

Task2 （两个矩阵相乘）编写程序计算两个矩阵相乘。

考察：二维数组，循环

编写一个测试程序，提示用户输入两个n*n的矩阵，然后显示它们的乘积。下面是一个运行示例：

为了使矩阵 a 和矩阵 b 相乘，矩阵 a 的列数必须与矩阵 b 的行数相同，并且两个矩阵的元素要具有相同或兼容的数据类型。加入矩阵 c 是相乘的结果，而 a 的列数数 n，那么每个元素 c_{ij} 就是 $a_{i1} \times b_{1j} + a_{i2} \times b_{2j} + \cdots + a_{in} \times b_{nj}$。例如，对于两个 3×3 的矩阵 a 和 b，c 有：

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \times \begin{bmatrix} b_{11} & b_{12} & b_{13} \\ b_{21} & b_{22} & b_{23} \\ b_{31} & b_{32} & b_{33} \end{bmatrix} = \begin{bmatrix} c_{11} & c_{12} & c_{13} \\ c_{21} & c_{22} & c_{23} \\ c_{31} & c_{32} & c_{33} \end{bmatrix}$$

这里的 $c_{ij} = a_{i1} \times b_{1j} + a_{i2} \times b_{2j} + a_{i3} \times b_{3j}$。

Enter size: 3 [enter]
Enter matrix1: 1 2 3 4 5 6 7 8 9[enter]
Enter matrix2: 0 2 4 1 4.5 2.2 1.1 4.3 5.2[enter]
The multiplication of the matrices is
1 2 3　　0 2.0 4.0.　　5.3 23.9 24
4 5 6　*　1 4.5 2.2　= 11.6 56.3 58.2
7 8 9　*　1.1 4.3 5.2　17.9 88.7 92.4

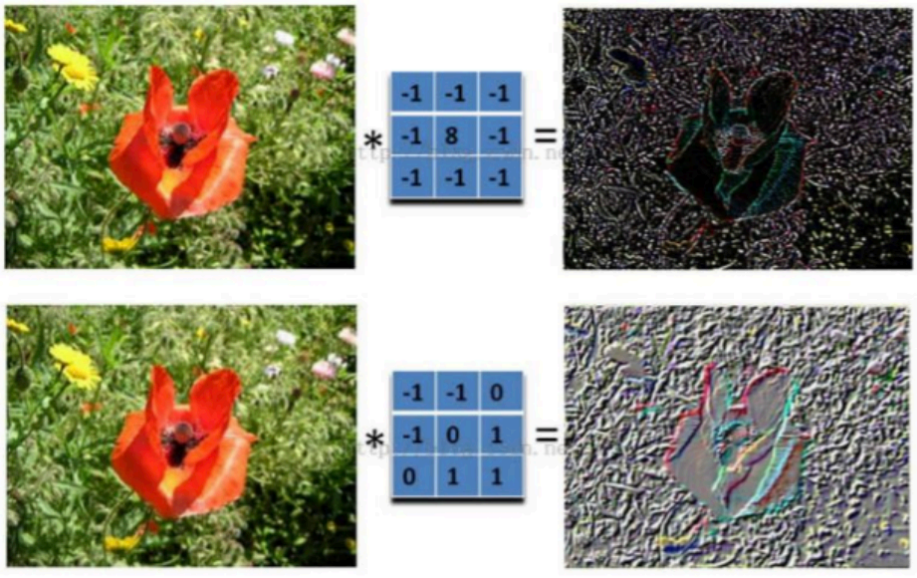
Task 3 尝试卷积

什么是卷积?

对图像（不同的数据窗口数据）和滤波矩阵（一组固定的权重：因为每个神经元的多个权重固定，所以又可以看做一个恒定的滤波器filter）做**内积**（逐个元素相乘再求和）的操作就是所谓的『卷积』操作，也是卷积神经网络（CNN）的名字来源。

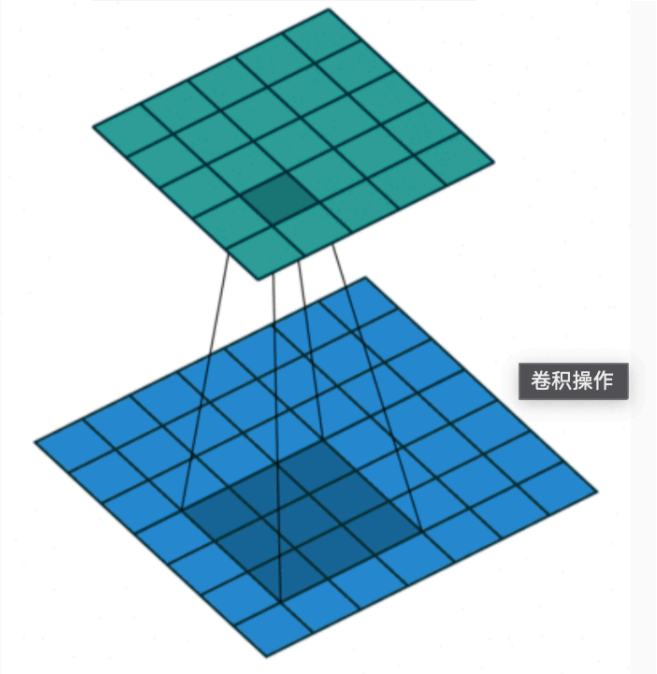
现在，卷积神经网络已经被广泛应用在图像识别的领域。

具体来说，如下图所示，左边是图像输入，中间部分就是滤波器filter（带着一组固定权重的神经元），不同的滤波器filter会得到不同的输出数据，比如颜色深浅、轮廓。相当于如果想提取图像的不同特征，则用不同的滤波器filter，提取想要的关于图像的特定信息：颜色深浅或轮廓。



在本任务中，我们尝试最基础的卷积操作。

其操作如下图所示，即使用卷积核对输入数据进行遍历处理，每次移动一步，卷积核与被处理数据对应位置相乘并求和便得到对应输出。模拟一个简单的卷积操作，输入第一行为M n两个数字，代表被处理数据以及卷积核的大小分别为M*M和n*n，下面M行为被处理数据，之后n行为卷积核，经过卷积处理后的输出大小应为 (M-n+1) * (M-n+1) 。



下面是一个输入示例：

```
5 2
1 4 2 3 6
4 2 5 1 7
3 6 9 4 7
3 6 8 5 2
4 6 3 7 4
```

1 –1
–1 1
其对应的输出为：
–5 5 –5 3
5 0 –1 –3
0 –1 2 –6
–1 –5 7 0