UCSC Silicon Valley Extension

C Programming, Advanced

Assignment 1 : C Review
Instructor : Radhika Grover

Include test cases in all of the following programs to demonstrate that the program executes correctly. The output carries 50% of the points for that problem. No points will be awarded if the program does not compile.

Problem 1 : Find the hidden words

A 2D grid contain words hidden horizontally, vertically and diagonally. Write a program to determine if any of the words in a given list are present in the grid and the number of occurrences of each word.

A	L	P	H	A
B	O	F	L	A
B	C	P	Z	T
B	H	H	A	E
A	T	X	Y	B

For example, the input is the following list of words :

```
ALPHA
BETA
GAMMA
```

The output is a count of the frequency of occurrence of each word:

```
ALPHA 2
```

```
BETA 1
GAMMA 0
```

The test case file has the format:
10 #total number of test cases
5 # number of rows and columns in grid
3 # number of words to find
word 1
word 2
word 3
grid

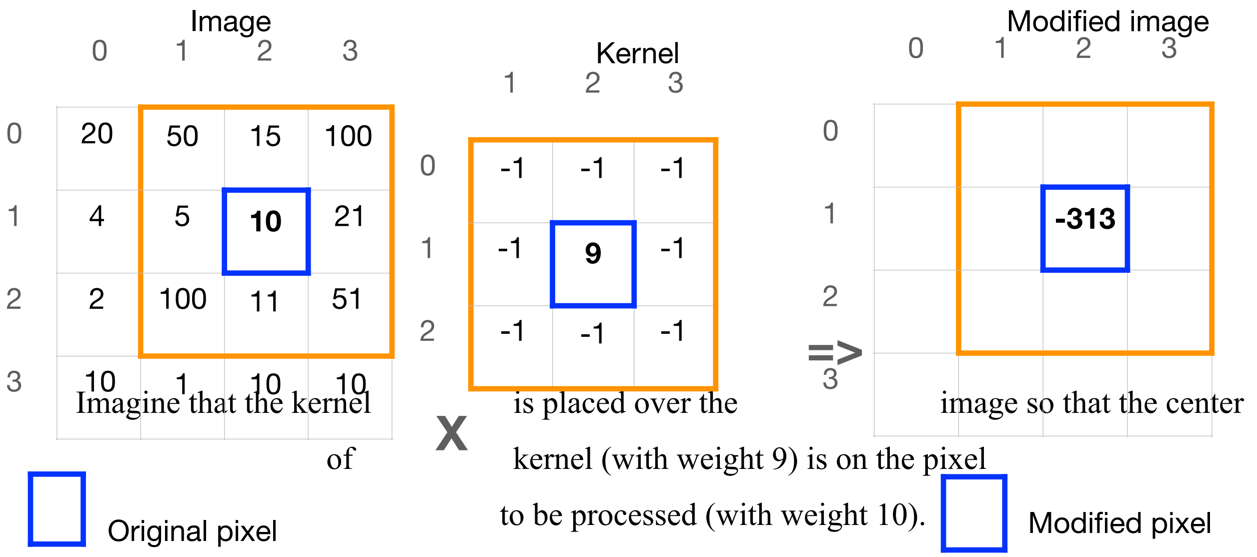
Problem 2

Convolution is widely used in image processing to perform a variety of operations, including blurring and sharpening, detecting edges, and embossing of an image. Each pixel in an image is modified using a kernel, where a kernel is a 2D array that stores weights. For example, the following 3X3 kernel stores the weights w_0, w_1, w_2, ….., w_8 each of which is a floating point number.

w_0	w_1	w_2
w_3	w_4	w_5
w_6	w_7	w_8

Kernel

An image consist of a grid of intensity values and typically one byte is used for each value. To perform the convolution operation, each pixel in an image is modified by multiplying its value and those of its eight neighbors with a kernel weight and summing them together. Black is represented by 0 and white by 255. For example, given a 4X4 image and the sharpening kernel shown below, replace each pixel by the weighted sum of its value and the neighbors.



Image

	0	1	2	3
0	20	50	15	100
1	4	5	**10**	21
2	2	100	11	51
3	10	1	10	10

Kernel

	1	2	3
0	-1	-1	-1
1	-1	**9**	-1
2	-1	-1	-1

Modified image

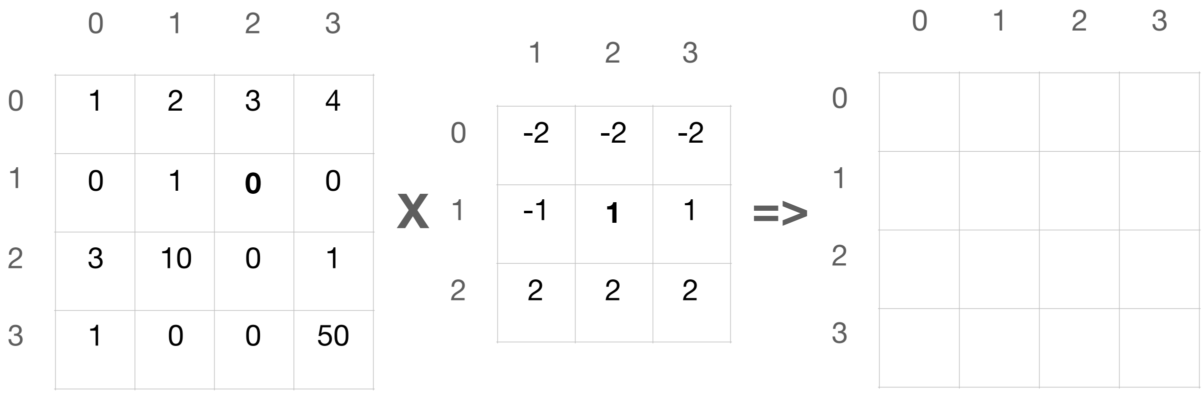
	0	1	2	3
0				
1			**-313**	
2				
3				

Imagine that the kernel is placed over the image so that the center of kernel (with weight 9) is on the pixel to be processed (with weight 10). Applying the corresponding weights to the neighboring pixels and summing them gives the value of this pixel in the modified image:

```
(50 X (-1) + 15 X (-1) + 100 X (-1) + 5 X(-1) + 10 (9) + 21 X
(-1) + 100 X (-1) + 11 X (-1) + 51 X (-1) = - 313)
```

The kernel is applied similarly to all pixels as shown below. Pixels along corners have fewer than 8 neighbors.

Example 2 :



	0	1	2	3
0	1	2	3	4
1	0	1	**0**	0
2	3	10	0	1
3	1	0	0	50

	1	2	3
0	-2	-2	-2
1	-1	**1**	1
2	2	2	2

	0	1	2	3
0				
1				
2				
3				

The test case file has 10 test cases with the format:

4 # image size

3 # kernel size

3 # number of words to find

image grid

kernel grid