Exercise on matrix arithmetic:

Construct an expression that combines the following four operations:
(1) compute a point-by-point sum of the contents of the two square patches in `image1` and `image2` shown below
(2) scale all of the individual sums by 3
(3) square all of the individual scaled sums
(4) compute the sum of all of these quantities over the entire square patch

Note: this exercise is related to a piece of the `getMatch` function described in the extra credit problem for Assignment 2!

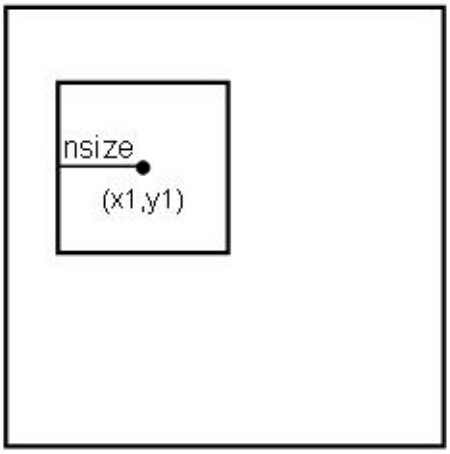

image1

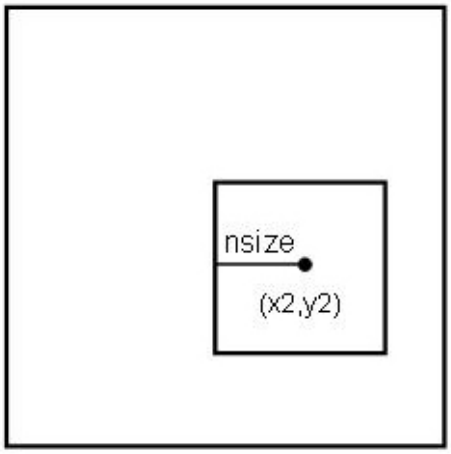

image2

Warning! Warning!

Suppose you create the following script `tempML.m` in the MATLAB editor:

```
image = uint8(zeros(10,10));
image(2:6,2:4) = 100;
image(6:9,6:9) = 200
diffIm = image(1:5,1:5) - image(6:10,6:10)
sumIm = image(1:5,1:5) + image(6:10,6:10)
```

and then execute the script in the Command Window:

```
>> tempML
image =
    0    0    0    0    0    0    0    0    0    0
    0  100  100  100    0    0    0    0    0    0
    0  100  100  100    0    0    0    0    0    0
    0  100  100  100    0    0    0    0    0    0
    0  100  100  100    0    0    0    0    0    0
    0  100  100  100    0  200  200  200  200    0
    0    0    0    0    0  200  200  200  200    0
    0    0    0    0    0  200  200  200  200    0
    0    0    0    0    0  200  200  200  200    0
    0    0    0    0    0    0    0    0    0    0
diffIm =
    0    0    0    0    0
    0    0    0    0    0
    0    0    0    0    0
    0    0    0    0    0
    0  100  100  100    0
sumIm =
  200  200  200  200    0
  200  255  255  255    0
  200  255  255  255    0
  200  255  255  255    0
    0  100  100  100    0
```

In this example, the problem can be solved by *not converting* the image to the type `uint8`, i.e., the script can begin with: `image = zeros(10,10);`

In other contexts (this is done in the `fingerprintScript.m` script for the extra credit problem for Assignment 2), you can convert an image of `uint8` values into `double` values:

```
image = double(image);
```