1. Write a program in the language of your choice to read a binary image and count the number of blobs in it, by the method explained in Shapiro and Stockman, Section 1.2.3.

A **binary image** is an image whose pixel values are all either ‘off’ or ‘on.’ To make visualization with standard image display programs easier, images for you to work on have ‘off’ pixels represented with zeros and ‘on’ pixels represented with the value 255, and are stored as GIF images (more on opening images below).

A **blob** is the converse of a hole: A blob is a set of ‘on’ pixels surrounded by ‘off’ pixels. As a result, the meanings of “external” and “internal” corners, as defined in Shapiro and Stockman, are switched.

You may assume that the blobs are all interior, four-connected, and simply connected.

A blob is **interior** if it does not touch the boundaries of the image. So all pixels in the first and last row and in the first and last column of the image can be assumed to be ‘off.’

Blobs are **four-connected** if no two of them touch each other other diagonally. The following is an example of an image with two blobs that are *not* four-connected because of the tenuous, diagonal contact between pixels in row 3, column 3, and in row 4, column 4:

```
0 0 0 0 0 0 0 0
0 1 1 0 0 0 0 0
0 1 1 0 0 0 0 0
0 0 0 1 1 1 1 0
0 0 0 1 1 0 0 0
0 0 0 1 1 1 1 0
0 0 0 0 0 0 0 0
```

Blobs are **simply connected** if they do not contain ‘off’ pixels. For instance, the blob in the following image is *not* simply connected, because it contains an ‘inner lake’ of ‘off’ pixels:

```
0 0 0 0 0 0 0 0
0 0 1 1 1 1 1 0
0 1 1 0 0 1 1 0
0 0 1 1 1 1 1 0
0 0 0 0 0 0 0 0
```

It is highly recommended that you work in MATLAB, where images can be opened with commands of the following format:

```
img = imread('imagename.gif');
```

In MATLAB you can convert an image with values 0 and 255 into a logical image with *false* for zero and *true* for 255 by the statement

```
img = img > 0;
```

If you work in any other language, please find a library you can use to open GIF files. Do not write that part of the code yourself. The class resources web page links to the OpenCV library for C++ programming, and to the ImageJ library for Java code. If you do not want to work in Matlab, it is your responsibility to download and install those libraries.

The images provided with this assignment are small, so you can ignore efficiency considerations in this assignment. We will talk about that in class.

(a) **Hand in the main part of your source code.** The main part is just a function `blobCount` that takes an image and outputs the number of blobs in it. You need *not* check if the image complies with the assumptions. Do *not* hand in the rest of your code (how you open images, how you call `blobCount`, etc).

(b) **State what your code returns when run on the images small.gif and nuclei.gif provided on the homework web page.**

(c) **Modify your code to handle blobs that are not interior.** Describe your modification, implement it, hand in the additional code only, and state where you put it within the original `blobCount` function. A good modification is simple.

(d) **State the result of running your modified code on the images small.gif, nuclei.gif, and boundary.gif provided on the homework web page.**

(e) **How much harder do you think it would be to handle blobs that are not simply connected?** You need *not* implement a method for doing so. Just describe a method in general terms if you think it would require a simple change. Alternatively, explain in general terms why you think the modification would be complex.