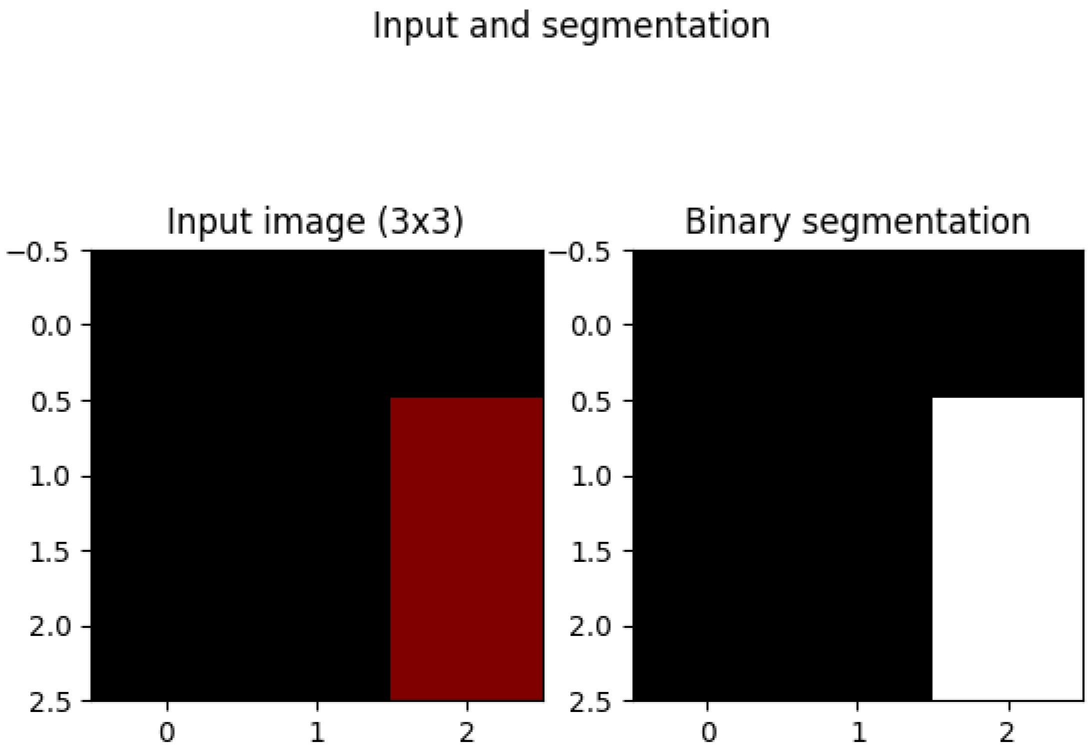
In [ ]:

```python
# Author: Jordan Malof
# Date: 2022.12.07
# Version: 2

#Import your class (it should be in same directly as this file)
import segmentationClass

#IMport numpy, and plotting package
import numpy as np
from matplotlib import pyplot as plt

#Instantiate an object for your class.
obj = segmentationClass.segmentationClass()

## Create a simple test image
# The image has two red pixels, and other pixels are zero-valued
I = np.zeros([3,3,3]);
I[2,2,0]=128;
I[1,2,0]=128;


#Set segmentation object properties
obj.x_a = np.array([2,2]);  # Foreground pixel coordinate
obj.x_b = np.array([0,0]);  # Background pixel coordinate
obj.p0 = 1;                # Edge capacities between neighboring pixels

# Segment the image
# This method and its I/O are needed in your implementaiton
t = obj.segmentImage(I);

# Plot the results
fig, axs = plt.subplots(1,2)
fig.suptitle('Input and segmentation')
axs[0].imshow(I.astype(np.uint8), interpolation='nearest')
axs[0].set_title("Input image (3x3)")
# The matrix 't' is binary, but it is helpful to scale the values to be 0 or 255
#  when displaying with imshow
axs[1].imshow(255*t.astype(np.uint8), interpolation='nearest')
axs[1].set_title("Binary segmentation")
plt.show()
```



In [ ]:

```
# Create adjacency list for the image
# This method takes an image as input and returns
# an adjacency list (python dictionary).  This method is used in my inside
# my implementaiton of segmentImage.
# Note: You are not required to have this function or this particular I/O.
A = obj.createAdjacencyListFromImage(I);

# Convert adjacency list to matrix
# In my segmentation software, I work with adjacency lists,
# and therefore I needed to convert my adjacency list to an adjacency matrix
# Note: You are not required to have this function or this I/O
Am = obj.adjacencyListToMatrix(A);

# Display adjacency matrix for pixels at location (0,0) and (1,0)
# In a 3x3 image, this corresponds to rows 0 and 3 in an adjacency matrix
# You are *required* to display an adjacency matrix for these two pixels, although
# the precise way in which you do it is up to you.
# Note: the last two columns of my adjacency matrix represent a source and target n
#   In this matrix, non-zero values represent edge capacities.
#
# Note: You may alternatively dispaly adjacency list output instead of an adjacency
#     as long as the contents are *clearly* explained
print(Am[[0,3],:])
```

```
[[  0.   1.   0.   1.   1.   0.   0.   0.   0.   0. 442.]
 [  1.   1.   0.   0.   1.   0.   1.   1.   0.   0. 442.]]
```