Joshua Reno
CS 4476: Intro to Computer Vision, Fall 2018
PS0

2.

a. x = np.random.permutation(1000)

Creates an array of values from {0 … 999} and randomly permutes them, returning an array of the 1000 values randomly ordered.

b. a = np.array([[1,2,3],[4,5,6],[7,8,9]])
b = a[2,:]

The first line creates three arrays of values and combines them into what is effectively a 2-D or multidimensional array. The second line returns [7, 8, 9] which is the second (and last) array and all of its values.

c. a = np.array([[1,2,3],[4,5,6],[7,8,9]])
b = a.reshape(-1)

The first line creates a 2-D or multidimensional array while the second line reshapes the array using the dimension -1 which returns a 1-D array with all the values, inferring the dimension from the existing dimension of the 2-D array.

d. f = np.random.randn(5,1)
g = f[f>0]

The first line creates an array of 5 random values (the 1 results in a 1-D array) using N(0, 1) while the second line creates an array will all values from the first that are greater than 0.

e. x = np.zeros(10)+0.5
y = 0.5*np.ones(len(x))
z=x+y

The first line creates an array of length 10 with values of 0 and adds 0.5 to all indices, resulting in an array of length 10 with all values of 0.5. The second does the same by creating an array of length 10 from the first array with all values of 1 and then multiples the array by 0.5, resulting in an array of length 10 with all values of 0.5. The final line adds the previous two arrays and gives an array of length 10 with all values of 1.

f. a = np.arange(1,100)
b = a[::-1]

The first line creates an array with values from 1 to 100. The second line reverses the array, resulting in an array with values {99 … 1}.

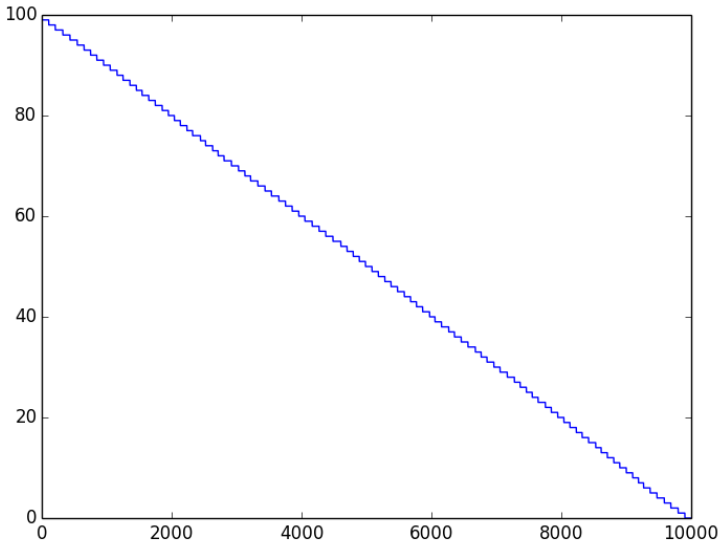
3.

```
import numpy as np
def partA(n):
        roles = []
        for i in range(n):
                roles.append(np.random.randint(1, 7))
        return roles
def partB():
        y = np.array([1, 2, 3, 4, 5, 6])
        z = y.reshape(3, 2)
        return z
def partC():
        y = np.array([1, 2, 3, 4, 5, 6])
        z = y.reshape(3, 2)
        x = z.max()
        r = np.where(z == x)
        c = r[1][0]
        r = r[0][0]
        return x, r, c
```

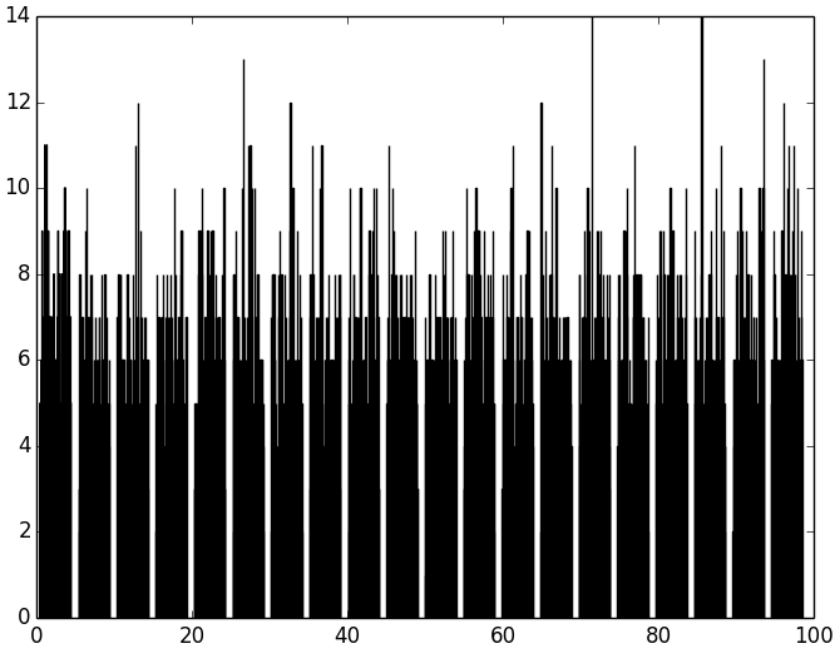
```
def partD():
        v = np.array([1, 8, 8, 2, 1, 3, 9, 8])
        x = (v == 1).sum()
        return x
```

4.

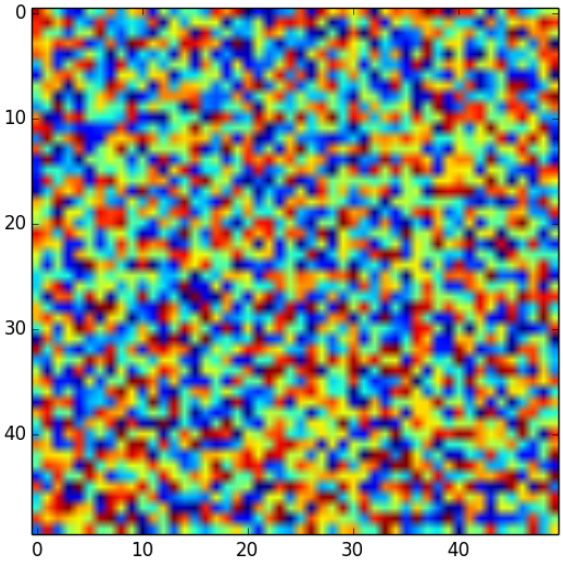
a.



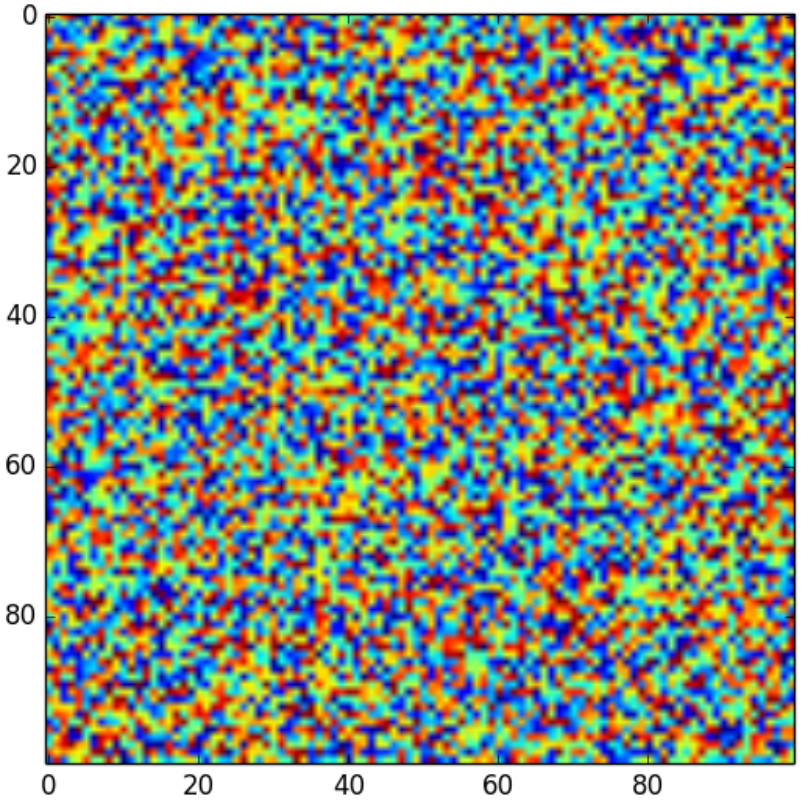
b.



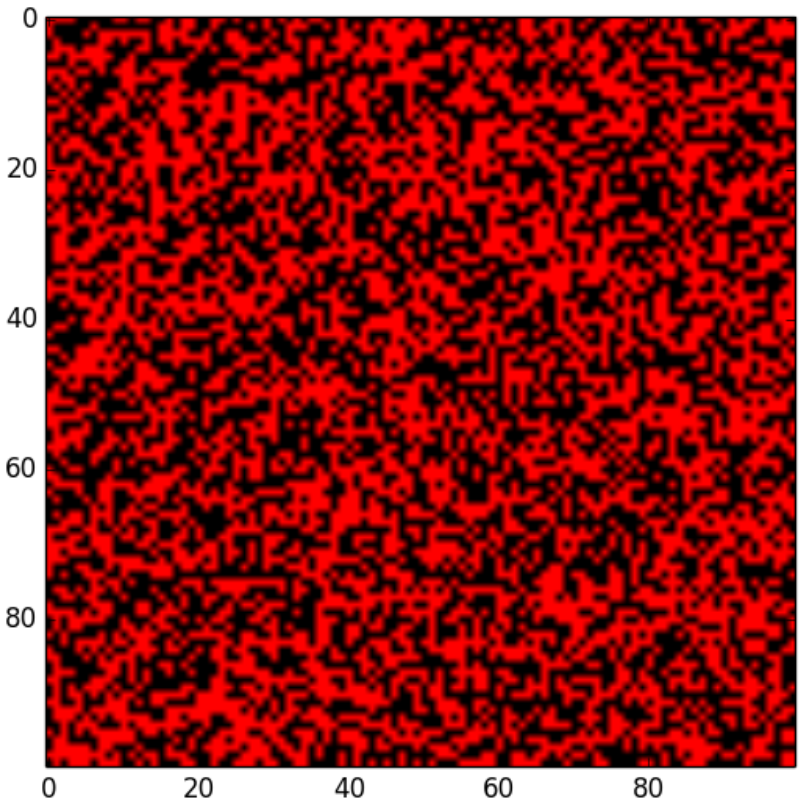
c.



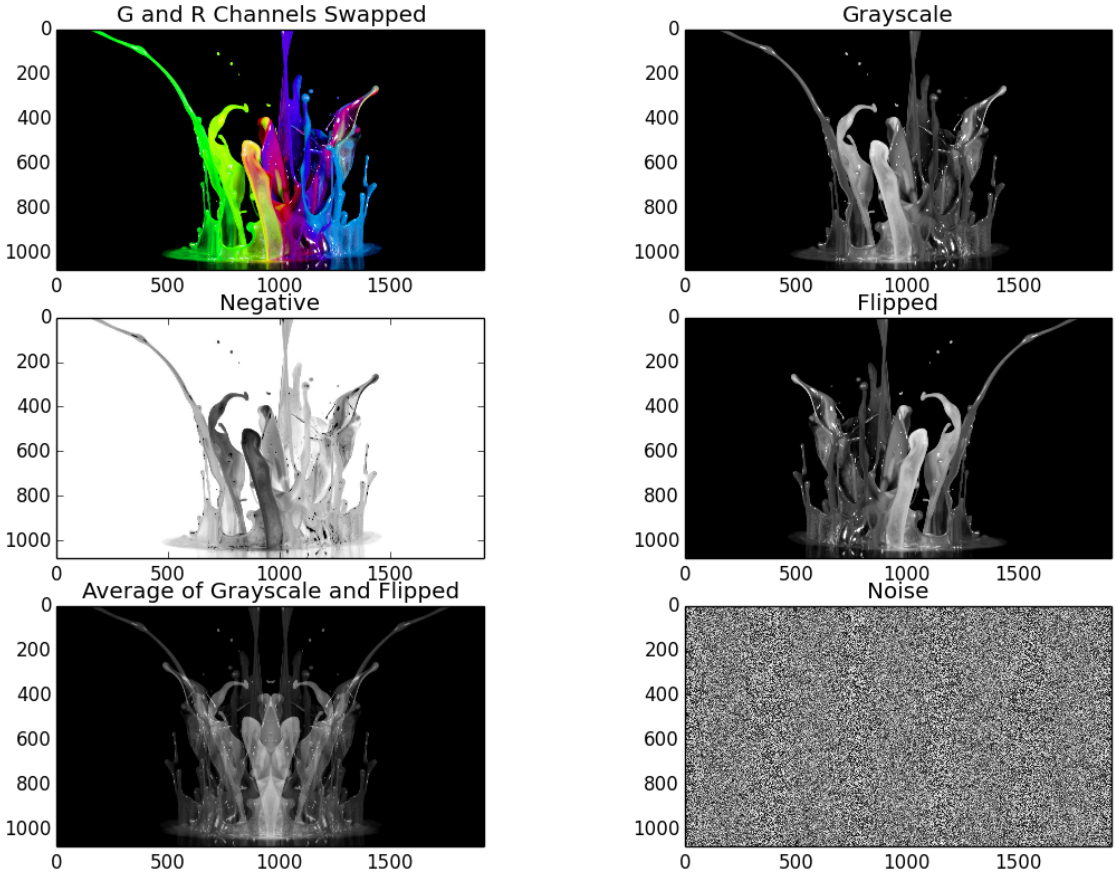
d.



e.


0
20
40
60
80
0
20
40
60
80

4.


G and R Channels Swapped
Grayscale
Negative
Flipped
Average of Grayscale and Flipped
Noise
0
200
400
600
800
1000
0
500
1000
1500