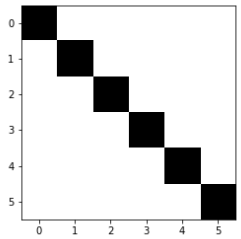
Random Image Slash

A 6x6 black-and-white image is represented as a Numpy array `x` as in the following,

```
>>> import numpy as np
>>> x = np.eye(6)
```

Note that this is not a grayscale or color image for which there would be three dimensions (e.g., 6 x 6 x 3). This can easily be visualized using Matplotlib's `imshow` function, as in the following:

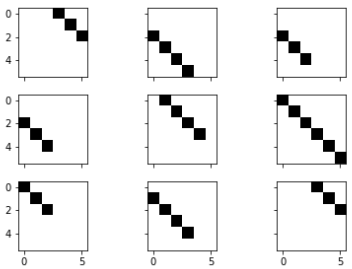
```
>>> from matplotlib.pylab import subplots, cm >>> fig, ax = subplots()  >>> ax.imshow(x,cmap=cm.gray_r)
```



To debug an image processing algorithm, you have to generate a large number of exemplar training images that consist of such Numpy arrays. Each image should represent a forward or backward leaning (shown above) slash symbol. Each symbol must consist of at least two non-zero pixels and be contiguous (i.e., no gaps). For example, the longest possible slash symbol that is representable in a 6x6 image is the 6 nonzero pixel diagonal image show above (or its opposite leaning forwardslash variant).

The assignment is to write a function that can produce a uniformly random forward or backslashed image (i.e., Numpy array) of at least two non-zero pixels. Here is some code that generates the following figure,

```
fig,axs=subplots(3,3,sharex=True,sharey=True)
 for ax in axs.flatten():
     ax.imshow(gen_rand_slash(),cmap=cm.gray_r)
```



Here is the function signature: `gen_rand_slash(m=6,n=6,direction='back')`. The `direction` keyword argument can be either `back` or `forward`. The `m` is the number of rows in the image.

Note: Don't import `matplotlib` in your solutions. Only allowed external library is `numpy`.
Please put your Python code in a Python script file and upload it. Please retain your submitted source files! Remember to use all the best practices we discussed in class. You can use any module in the Python standard library, but third-party modules (e.g., Numpy, Pandas) are restricted to those **explicitly** mentioned in the problem description.

Tips:

- After you have submitted your file, do **not** use the browser back or reload buttons to navigate or open the page in multiple browser tabs, as this may cause your `attempts` to decrease unexpectedly. It may take up to thirty seconds for your code to be processed, so please be **patient**.
- If you find yourself back at the main page without any feedback or change in your `attempts` then it means that your code timed out or crashed in some unexpected way.
- Ensure that your development environment does not presume the existence of certain packages for the autograder. The autograder does not have anything other than the standard library and those third-party libraries **explicitly** named in the problem description.
- Do not leave extraneous statements in your code like test cases, print statements, or anything else besides what is needed to evaluate your submission because the the autograder will spend its limited time executing those lines, which may result in unexpected crashes or timeouts.

浏览... 未选择文件。 Upload Python source code file