Error and stuff that cost me points on previous tp to not do again

TP01

Exercice 1

1.e : il fallait réutiliser une image avec les 3 chanels, pas en gris

4 d. The difference between adding noise and `skimage.util.random_noise` - is due to clip operation. i.e. in numpy function values can go above [0, 1]

Exercice 2

2.a : il est possible de mieux faire pour ne pas avoir des traits verticaux

But était de faire un gradient de `uint8` black to white de 125×500.

```
    np.linspace(start=0, stop=255, num=500) # only one line use np.repeat to get 125 ot
    a = np.linspace(start=0, stop=255, num=500)
    b = np.repeat(a, 125, axis=0)
    c = np.reshape(np.repeat(a, 125, axis=0), (125, 500))
    print(b.shape)
    print(c.shape)
```

```
(62500,)
(125, 500)
```

2.d : pareil

Exercice 5

5.c) Using a binomial with n = 1 is the same as using a Bernoulli distribution. (`function np.random.choice()`)

- **np.random.choice()** is the function for Bernoulli distrib.
- Your result only adds salt to the image (or pepper, depending on your luck) This is because of the line :

```
img[mask] = gen_salt_or_paper()
```

- on the left-hand-side, you correctly selected all pixels that you want to modify
- on the right-hand-side, your function returns a single number.
- when numpy tries to assign a single number to an array of numbers, it uses broadcasting, thus repeats this number to every location of the array.

Note: For the next time, if you notice something strange in your results, do not hesitate to commentate, so we see that you understand that there is a problem.

TP03

Code ok, mais il ya un autre problème:;

```
mse = np.mean((img1 - img2)**2)
psnr = 10* np.log(img.max() / mse)
```

- Problème si img1 et img2 sont pas les 2 dans [0, 1] ou les 2 dans [| 0, 255 |]
- ⚠️ **Si `img1` et `img2` sont des uint8 et qu'on fait `img1 - img2` (e.g. 10 - 15) ⇒ UNDERFLOW!** ⚠️
- Mettre `img.max()` dans psnr car ça prend direct 1 si [0, 1] et 255 si [0, 255]