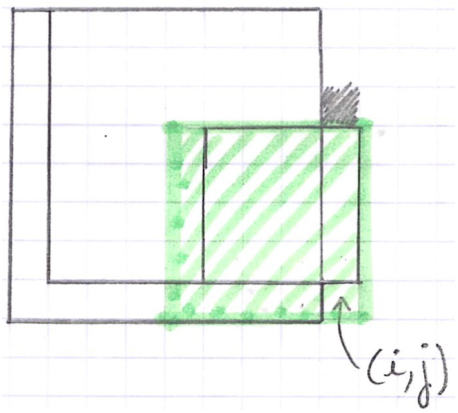
02 - Recherche du plus grand carré blanc dans une grille - Corrigé

1. Si `PGCB(3,7)=3`, les 9 pixels qui forment le carré blanc ont pour coordonnées :
   $(1,5),(1,6),(1,7),(2,5),(2,6),(2,7),(3,5),(3,6),(3,7)$.
2. Si `G[i][j]=0`, alors le pixel de coordonnées (i,j) est noir, donc `PGCB(i,j)`doit renvoyer 0.
3. Si `G[i][j]=1`, et si ($\mathrm{i}=0$ ou $\mathrm{j}=0$), alors le pixel de coordonnées (i,j) est blanc, mais est situé sur le bord supérieur de l'image, ou sur le bord gauche. On ne peut donc pas le prolonger par le haut et à gauche. Donc `PGCB(i,j)` doit renvoyer 1.

   On suppose pour les questions 4. et 5. que le pixel de coordonnées (i,j) est blanc, mais qu'il n'est ni situé sur le bord supérieur de l'image, ni sur le bord gauche.
4. Si `PGCB(i-1,j-1)=7`, `PGCB(i-1,j)=4`, `PGCB(i,j-1)=8`, alors `PGCB(i,j)`$=5$.



5. De façon générale, on observe (il n'est pas demandé de démonstration) que :

   `PGCB(i,j)`$=1+$MIN(`PGCB(i-1,j-1)`,`PGCB(i-1,j)`, `PGCB(i,j-1)`)
6. Voici le code de la fonction `PGCB` :

```
D={}

def PGCB(i,j):
    if (i,j) in D:
        return D[(i,j)]
    if G[i][j]==0: #cas d'un pixel noir
        D[(i,j)]=0
        return 0
    else: #cas d'un pixel blanc
        if i==0 or j==0: #si on est sur le bord supérieur ou sur le bord gauche
            D[(i,j)]=1
            return 1
        else:
            D[(i,j)]=1+min([PGCB(i-1,j-1),PGCB(i,j-1),PGCB(i-1,j)])
            return D[(i,j)]
```

7. Pour la grille donnée en exmple , déterminons la plus grande valeur que renvoie la fonction `PGCB` ainsi qu'un couple (i0,j0) qui atteint cette plus grande valeur :

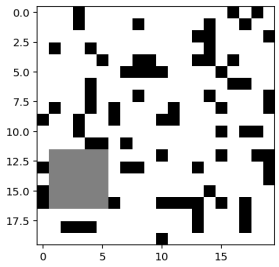
```
maxi=0
i0=0
j0=0
for i in range(20):
    for j in range(20):
        valeur=PGCB(i,j)
        if valeur>maxi:
            maxi=valeur
            i0=i
            j0=j
print(i0,j0,maxi)
```

On trouve que le plus carré blanc a pour sommet en bas à droite le pixel $(16, 5)$, et que ce carré est de côté 5.

8. .
On va colorier le carré en question en gris et faire afficher la grille ainsi modifiée (la variable `maxi` contient la taille du côté de plus grand carré blanc) :

```
for i in range(i0-maxi+1,i0+1):
    for j in range(j0-maxi+1,j0+1):
        G[i][j]=0.5

plt.imshow(G,cmap="gray")
```



9. Si on veut s'amuser un peu : voici comment générer facilement des grilles aléatoires avec le module `random` :

```
import random as rd
# la fonction rd.random() génère un réel aléatoire entre 0 et 1

def génération_grille(n,p,coeff):
    ''' coeff entre 0 et 1 pour la proportion de carrés blancs ou noirs '''
    G=[]
    for i in range(n):
        ligne=[]
        for j in range(p):
            if rd.random()<coeff:
                ligne.append(0) #on génère un carré noir
            else:
                ligne.append(1) #on génère un carré blanc
        G.append(ligne)
    return G
```