Retour sur les matrices et les images

Commencez par relire l'introduction du TP8 sur les images, puis téléchargez l'image `TP12Image.bmp` ainsi que le fichier `TP12.py` (qui contient la trame du TP dans laquelle vous écrirez votre code), vous mettrez ces deux fichiers dans un même dossier appelé `TP12`. Puis tester et comprendre ce qu'il y a dans le fichier `TP12.py`.

Exercice 1 (𝄞⋆)**.** Considérons une image ayant `n` lignes et `p` colonnes. On souhaite réduire cette image, c'est-à-dire créer une nouvelle image ayant moins de lignes et de colonnes que cette image originale. Fixons r_ℓ (coefficient de réduction de lignes) et r_c (coefficient de réduction de colonnes) deux entiers naturels non nuls (éventuellement distincts). On souhaite conserver une ligne sur r_ℓ et une ligne sur r_c en commençant à 0. Ainsi, les lignes conservées seront 0, r_ℓ, $2r_\ell$, $3r_\ell$ etc. et les colonnes conservées seront $0, r_c, 2r_c$ etc. Ainsi, il faut garder les pixels de l'image originale dont la ligne et la colonne sont conservées.

1. Si on prend une ligne sur r_ℓ et une colonne sur r_c, déterminer le nombre de lignes et de colonnes que l'on obtiendra.
2. Créer une fonction `Reduction(Img,rl,rc)` qui effectue cette réduction sur une image `Img`. Dans cette fonction, commencer par créer `M` une matrice de 0 ayant la taille désirée, puis à l'aide d'une double boucle on affectera à `M[i][j]` la valeur désirée.

On rappelle que Python gère un type d'objet qui s'appelle *tuple* (n-uplet) en français, il s'agit de regrouper des objets sous la forme, par exemple `(a,b,c)` où a, b, c sont trois objets. En particulier, Python est capable de faire du *dépaquetage de tuple* c'est-à-dire que la commande `(e,f,g)=(a,b,c)` va, à partir de trois variables `a`, `b` et `c` existantes, créer le tuple `(e,f,g)` en créant les variables `e`, `f` et `g`, `e` vaudra `a`, `f` vaudra `b` et `g` vaudra `c`.

Exercice 2. Créer une fonction `Cycle(a,b,c,d)` qui, à quatre objets, les permute en renvoyant `(b,c,d,a)`.

Exercice 3 (𝄞⋆⋆)**.** Soit une image carrée (ayant le même nombre de lignes et de colonnes). On souhaite faire tourner l'image d'un quart de tour. Pour cela, on va écrire une fonction `rotation(I,x,y,n)` qui va faire tourner l'image `I` d'un quart de tour seulement dans le carré délimité par les sommets (x, y), $(x, y+n-1)$, $(x+n-1, y+n-1)$ et $(x+n-1, y)$. Si $n > 1$, on pose p le quotient de la division euclidienne de n par 2. Divisons le carré en 4 petits carrés, puis permuter ces quatre carrés en permutant les pixels de coordonnées (i, j), $(i, p+j)$, $(p+i, p+j)$ et $(p+i, j)$, pour tout $(i, j) \in [\![x\,;x+p-1]\!] \times [\![y\,;y+p-1]\!]$. Puis, par récursivité, tourner d'un quart de tour les quatre petits carrés. Attention, cette fonction va directement modifier `I` par effet de bord, il n'y aura donc pas de `return`. Tester la fonction avec `rotation(Img,0,0,len(Img))`.
On peut aussi faire d'abord tourner les quatre petits carrés puis les permuter.

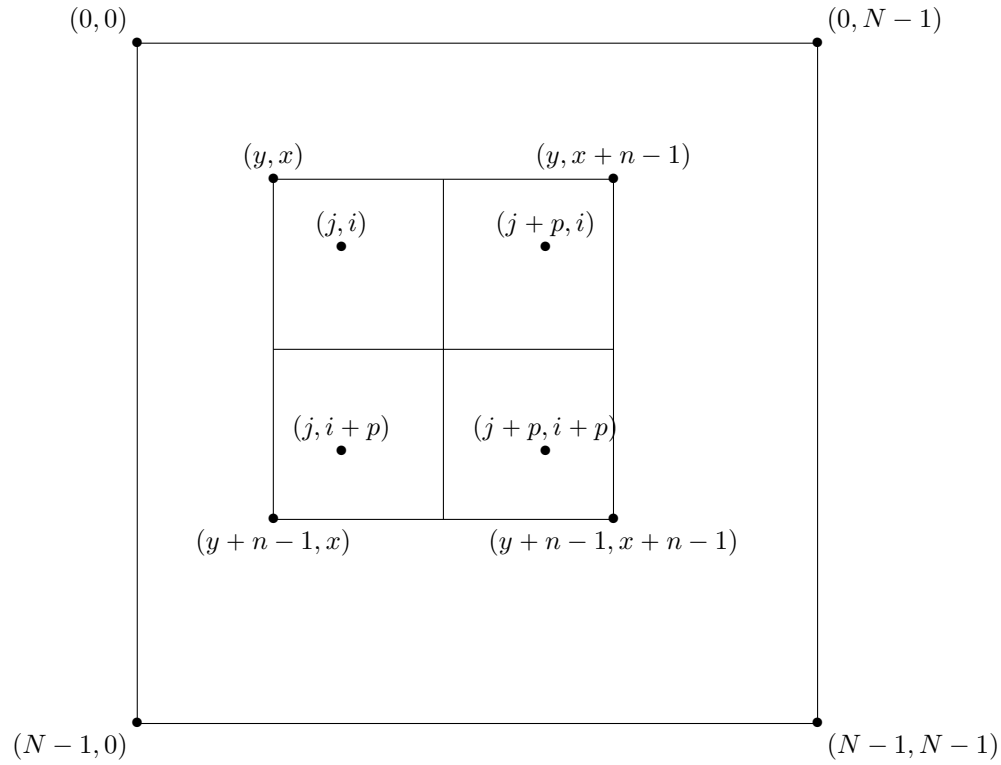

FIGURE 1 – Découpage en quatre du carré de côté n partant du sommet (y, x) de longueur n. Rappel : dans `M[i][j]`, le i désigne un indice de ligne (donc une ordonnée sur le schéma), et le j désigne un indice de colonne (donc une abscisse sur le schéma).

Retour sur les algorithmes glouton

Exercice 4 (𝄞⋆⋆)**.** Un professeur d'informatique, désireux de s'offrir de belles vacances, a planifié un cambriolage dans une bijouterie. Il a parfaitement étudié son coup et sait que la bijouterie contient :

- des sacs de diamants de 9 kg valant 999€
- des sacs de pépites de 12 kg valant 696€
- des sacs de topazes de 2 kg valant 100€
- des sacs d'émeraudes de 7 kg valant 301€
- des sacs de bijoux en or de 5 kg valant 400€

N'étant pas très musclé, il ne peut porter que 15kg de bijoux. Le but est de maximiser le montant de butin. Écrire une fonction `ButinMax(systeme)` qui renvoie le contenu du butin et son montant sachant qu'il espère récolter un montant

maximal selon le principe d'un algorithme glouton. On réfléchira d'abord sur quel `système` il est intéressant de travailler en entrée pour maximiser le butin.