# Exercice 1

► **Q1.**

```
def dimensions(G):
    p = len(G)
    q = len(G[0])
    for i in range(1, p):
        if len(G[i]) != q:
            return (0, 0)
    return (p, q)
```

► **Q2.**

```
def EstRectangle(G, l1, l2, c1, c2):
    for i in range(l1, l2):
        for j in range(c1, c2):
            if G[i][j] != 0:
                return False
    return True
```

► **Q3.** Voici des exemples de tests (avec `G` la grille de l'énoncé) :

```
assert EstRectangle(G, 2, 5, 1, 4)      # un rectangle de 0
assert not EstRectangle(G, 2, 6, 1, 4) # un rectangle avec un 1
assert EstRectangle(G, 0, 1, 2, 4)      # une ligne de 0
assert not EstRectangle(G, 0, 1, 2, 6) # une ligne avec un 1
assert EstRectangle(G, 0, 1, 2, 3)      # une case 0
assert not EstRectangle(G, 0, 1, 0, 1) # une case 1
```

► **Q4.** Il y a $(l2-l1)(c2-c1)$ passages dans les boucles et chaque passage a une complexité constante. La complexité est donc en $O((l2-l1)(c2-c1))$ et *a fortiori* en $O(pq)$.

► **Q5.**

```
def PlusGrandRectangle(G):
    p, q = dimensions(G)
    taille_max = 0
    for l1 in range(p):
        for l2 in range(l1+1, p+1):
            for c1 in range(q):
                for c2 in range(c1+1, q+1):
                    if EstRectangle(G, l1, l2, c1, c2) and (l2-l1)*(c2
                        -c1) > taille_max:
                        taille_max = (l2-l1)*(c2-c1)
    return taille_max
```

► **Q6.** Le nombre d'itérations est :

$$\sum_{i_1=0}^{n-1}\sum_{i_2=i_1}^{n}\sum_{j_1=0}^{n-1}\sum_{j_2=i_1}^{n} 1$$

et à chaque itérations il y a un au plus 3 opérations, un test, éventuellement 3 opérations et une affectation mais il y a aussi l'appel à la fonction `EstRectangle` qui a une complexité en $O((l2-l1)*(c2-c1))$.
On a donc un nombre d'itérations en $O(n^4)$ avec, pour chaque itération, un nombre d'opérations en $O(n^2)$.
On a donc une complexité en $O(n^6)$.

► **Q7.** Tout d'abord pour $i$ fixé : $\text{hmax}_{i,i}$ est la hauteur minimale entre les positions $i$ et $i$ donc c'est $\texttt{H}[i]$.
Par ailleurs, la définition de $\text{hmax}_{i,j}$ suppose que $i \leqslant j$ donc $\text{hmax}_{i,j} = 0$ lorsque $j < i$.
Enfin, si $i < j$ alors $\text{hmax}_{i,j}$ est la hauteur minimale entre les positions $i$ et $j$ : il s'agit de la même hauteur minimale qu'entre les positions $i$ et $j-1$ sauf si la colonne $j$ est plus basse donc c'est la plus petite valeur entre $\text{hmax}_{i,j-1}$ et la hauteur «supplémentaire» $\texttt{H}[j]$.
On a donc :

$$\text{hmax}_{i,i} = \texttt{H}[i] \text{ et } \text{hmax}_{i,j} = \begin{cases} 0 & \text{si } j < i \\ \min\left(\text{hmax}_{i,j-1}, \texttt{H}[j]\right) & \text{si } j > i \end{cases}.$$

► **Q8.**

```
def CalculHauteurs(H):
    n = len(H)
    hmax = [[0 for j in range(n)] for i in range(n)]
    for i in range(n):
        hmax[i][i] = H[i]
        for j in range(i+1, n):
            if H[j] < hmax[i][j-1]:
                hmax[i][j] = H[j]
            else:
                hmax[i][j] = hmax[i][j-1]
    return hmax
```

► **Q9.**

```
def ColonneZero(G):
    p, q = dimensions(G)
    tab = [[0 for j in range(q)] for i in range(p)]
    for j in range(q):
        tab[p-1][j] = 1 - G[p-1][j]
    i = p-2
    while i >= 0:
        for j in range(q):
            if G[i][j] == 1:
                tab[i][j] = 0
            else:
                tab[i][j] = 1 + tab[i+1][j]
        i = i - 1
    return tab
```

► **Q10.**

```
def PgrUltime(G):
    H = ColonneZero(G)
    n = len(H)
    maxi = 0
    for i in range(n):
        histo = H[i]
        t = PgrHistogramme(histo)
        if t > maxi:
            maxi = t
    return maxi
```

► **Q11.**

```
def convertir(image):
    p, q = dimensions(image)
    g = [[0 for j in range(q)] for i in range(p)]
    for i in range(p):
        for j in range(q):
            a, b, c = image[i][j]
            if (a + b + c)/3 > 120:
                g[i][j] = 1
    return g
```

# Exercice 2 – base de données

► **Q12.**

```
SELECT nom, prenom
  FROM Etudiants
  WHERE groupe = "A"
```

► **Q13.**

```
SELECT DISTINCT colleur
  FROM Creneaux
  WHERE jour = "jeudi"
```

► **Q14.**

```
SELECT note, semaine
  FROM Colles
  WHERE idetud = 11
  ORDER BY note DESC
```

► **Q15.**

```
SELECT AVG(c.note)
  FROM Etudiants e JOIN Colles c ON e.id = c.idetud
  WHERE e.nom = "MELLEREAU"
```

► **Q16.**

```
SELECT e.nom, e.prenom, MAX(c.note)
  FROM Etudiants e JOIN Colles c ON e.id = c.idetud
  GROUP BY e.id
```

► **Q17.**

```
SELECT x.colleur, AVG(c.note) as moyenne
  FROM Creneaux x JOIN Colles c ON x.numero = c.numero
  GROUP BY x.colleur
  ORDER BY moyenne DESC
  LIMIT 2
```

► **Q18.**

```
SELECT e.nom, e.prenom, AVG(c.note) as moyenne
  FROM Etudiants e JOIN Colles c ON e.id = c.idetud
  GROUP BY e.id
  HAVING moyenne >= 14
```