Le but de ce TP est de faire un programme affichant $n = 8$ reines sur échiquier $n \times n$ sans aucune menace, par exemple pour $n = 4$

	Q		
			Q
Q			
		Q	

1 Compter

La représentation en base b d'un entier $x = \sum_{i=0}^{n-1} x_i b^i$, avec $x_i \in \{0, \ldots, b-1\}$, est le tableau X tel que $X[n-i-1] = x_i$. Par exemple, pour $b = 2$ et $n = 3$, on peut représenter tous les entiers de 0 à 7:

$$\begin{array}{ccc} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 0 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \\ 1 & 1 & 1 \end{array}$$

Après l'initialisation d'un tableau `X` de taille `n=len(X)` et d'un entier `b`, par exemple

```
b=2
n=3
X=[0 for i in range(n)]
```

Programmer une fonction récursive `count(i)` telle que l'appel à `count(0)` affichera l'expression en base b de tous les entiers de 0 à $b^n - 1$.

Pour concevoir le code, on fera en sorte que:

1. Trivialement `count(n-1)` affiche, en dernière colonne `X[n-1]` tous les entiers de 0 à $b^{n-(n-1)} - 1$. Donc, pour $b = 2$ et $n = 3$, `count(2)` doit afficher

   $$\begin{array}{ccc} * & * & 0 \\ * & * & 1 \end{array}$$

   (où *=0 si `X` a été initialisé à 0, sinon * n'est pas prévisible)

2. Si `count(i+1)` affiche, sur `X[i+1]` ... `X[n-1]` tous les entiers de 0 à $b^{n-(i+1)} - 1$, alors `count(i)` affiche, sur `X[i]` ... `X[n-1]` tous les entiers de 0 à $b^{n-i} - 1$. Donc, pour $b = 2$ et $n = 3$, `count(1)` doit afficher

   $$\begin{array}{ccc} * & 0 & 0 \\ * & 0 & 1 \\ * & 1 & 0 \\ * & 1 & 1 \end{array}$$

2 Les permutations

Pour énumérer toutes les $n!$ permutations, par exemple pour $n = 3$

$$\begin{array}{ccc} 0 & 1 & 2 \\ 0 & 2 & 1 \\ 1 & 0 & 2 \\ 1 & 2 & 0 \\ 2 & 0 & 1 \\ 2 & 1 & 0 \end{array}$$

il faut initialiser, par exemple:

```
n=3
X=[0 for i in range(n)]
U=[0 for i in range(n)]
```

où `U[i]==1` si $i \in \{0, 1, \ldots, n-1\}$ est présent dans `X`, et `U[i]==0` sinon. Modifier `compt(i)` en une fonction récursive `permut(i)` telle que l'appel à `permut(0)` affiche ainsi toutes les permutations.

3 Les stables d'un graphe

Pour énumérer tous les stables d'un graphe G à n sommets avec une liste d'arêtes ij, par exemple

```
n=5
X=[0 for i in range(n)]
EG=[[0,1],[0,2],[1,2],[0,4],[2,3],[3,4]]
```

Il faut énumérer tous les vecteur $x \in \{0, 1\}^n$ et vérifier qu'ils caractérisent bien des stables, c'est-à-dire que pour tout ij dans EG, on a $x_i = 0$ ou $x_j = 0$.

4 Les reines d'un échiquier

L'énumération des permutations que l'on a vu correspond, pour $n = 8$, à l'énumération de tous les placements de 8 tours sur un échiquier sans menace.

Un autre exemple est l'énumération des placements de 8 reines dans un échiquier sans aucune menace entre elles. L'échiquier sans pièce est représenté par

```
n=8
Q=[-1 for i in range(n)]
r=[0 for i in range(n)]
c=[0 for i in range(2*n-1)]
d=[0 for i in range(2*n-1)]
```

Les tableaux `Q` ayant des valeurs `Q[i]` dans $\{0, 1, \ldots, 7\}$ sont en bijection avec les échiquiers contenant exactement une reine par colonne. L'instruction `Q[i]=j` place une reine en colonne i ligne j, et

1. `r[j]=1` signifie que la $j$ème ligne
2. `c[i+j]=1` signifie que la $(i + j)$ème diagonale Nord-Ouest
3. `d[i-j+n-1]=1` signifie que la $(i - j + n - 1)$ème diagonale Nord-Est

est alors menacée par cette reine. Programmer une fonction récursive `queen(i)` telle que l'appel à `queen(0)` affiche tous les placements possibles de reines sans menace.