# 1 TP bonus : le jeu de la vie

**Notions pratiquées** : listes à 2 dimensions.

Dans ce TP on se propose d'implémenter le jeu de la vie de Conway[1]. Le jeu de la vie de Conway est un automate cellulaire dont les règles sont extrêmement simples. On considère une grille carrée de cellules qui peuvent être soit vivantes soit mortes. L'évolution de l'état d'une cellule dépend de son nombre de voisines vivantes parmi les 8 cellules qui l'entourent (cf figure 1):

- Une cellule morte qui a exactement 3 voisines vivantes devient vivante (elle naît);
- Une cellule vivante qui a exactement 2 ou 3 voisines vivantes reste vivante, sinon elle meurt.

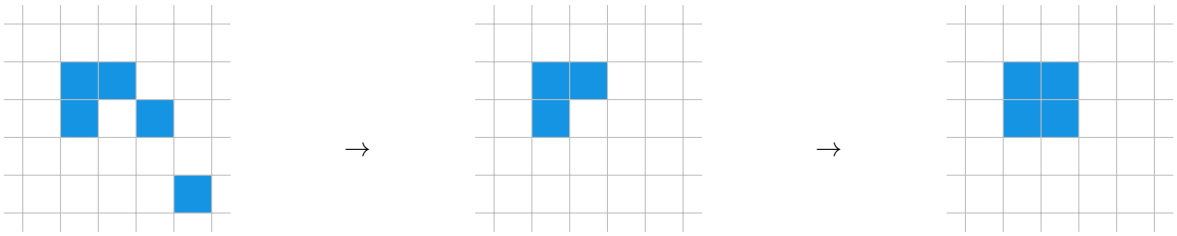

Figure 1: Un exemple d'évolution sur trois étapes d'un extrait de grille dans le jeu de la vie. Les cellules mortes sont en blanc, les cellules vivantes en bleu.

## 1.1 Version textuelle

On choisit ici de représenter cette grille de la manière suivante :

- L'état de chaque cellule est un booléen : True pour vivante, False pour morte.
- La grille est une liste de listes : chaque élément de la liste est une liste contenant les états de toutes les cellules de cette ligne.
- On appelle taille de la grille le nombre de cellules par côté. Une grille de taille N est donc une liste de N listes de N éléments.

Par exemple la grille de gauche sur la figure 1 ci-dessus est de taille 5 et est représentée par la liste suivante :

```
[[False, False, False, False, False],
 [False, True,  True,  False, False],
 [False, True,  False, True,  False],
 [False, False, False, False, False],
 [False, False, False, False, True ]]
```

1. Écrire une fonction `initGrilleM(N)` qui reçoit en argument la taille N de la grille carrée, et initialise et renvoie une grille carrée de cette taille (N cellules par N cellules) contenant uniquement des cellules mortes. **Attention** à créer des listes différentes pour chaque ligne, et non pas des références à la même liste.
2. Écrire une fonction `initGrilleV(N,V)` qui appelle la fonction précédente pour créer une grille de taille N de cellules mortes, puis qui place **exactement** V cellules vivantes (à des positions au hasard), et renvoie cette grille. On suppose que V a une valeur correcte c'est-à-dire positif et inférieur ou égal à N*N. *Indice: il s'agit, pour chaque cellule vivante à placer, de tirer au hasard des coordonnées dans la grille, jusqu'à trouver une cellule morte, puis de rendre cette cellule vivante.*
3. Écrire une fonction `afficheSolution` qui reçoit en paramètre une grille et l'affiche de la manière suivante : les cellules vivantes sont représentées par le caractère '*' (étoile), et les cellules mortes par le caractère '.' (point), sans espace entre les valeurs. Par exemple la grille de booléens ci-dessus (qui correspond à la grille de gauche sur la figure) sera affichée sous la forme suivante :

[1] Voir plus de détails sur Wikipedia `https://fr.wikipedia.org/wiki/Jeu_de_la_vie`, et une réalisation (dans un autre langage) ici: `https://nausikaa.net/wp-content/uploads/conway-fr.html`

```
.....
.**..
.*.*.
.....
....*
```

4. Écrire une fonction `grilleEtendue` qui reçoit une grille de taille N, et qui calcule et renvoie une grille de taille N+2 en rajoutant des cellules mortes (valeur False) tout autour de la grille reçue. Cela permettra d'éviter les effets de bord dans le comptage des voisins vivants des cellules. *On rappelle les fonctions* ***`insert`*** *et* ***`append`*** *pour insérer un élément dans une liste ou à la fin d'une liste respectivement.* Par exemple pour la grille de taille 5 ci-dessus, cette fonction renvoie la grille de taille 7 suivante :

```
.......
.......
..**...
..*.*..
.......
.....*.
.......
```

5. Écrire une fonction `compteVoisins(grille,i,j)` qui reçoit la grille de taille N, utilise la fonction ci-dessus pour calculer la grille étendue de taille N+2, puis compte et renvoie le nombre de voisins vivants de la cellule de la ligne d'indice i et colonne d'indice j de la grille. **Attention** : les indices reçus sont supposés corrects, et sont exprimés dans la grille de taille N, avant extension : ils sont donc compris entre 0 et N-1. Grâce à l'extension de la grille, toutes les cellules ont exactement 8 voisines et il n'est donc pas nécessaire de gérer des cas particuliers dans les coins ou sur les bords.
*Par exemple pour la grille ci-dessus et les indices (0,1), la fonction compte les voisins vivants de la cellule de la 1e ligne et 2e colonne de la grille initiale (numérotées à partir de 0), c-à-d de la 2e ligne et 3e colonne dans la grille étendue, et renvoie 2 (une cellule vivante dessous, et une cellule vivante dessous à droite).*

6. Écrire une fonction `evoluerGrille` qui reçoit la grille en argument, et crée et renvoie une nouvelle grille contenant le nouvel état de toutes les cellules après une évolution. La grille initiale n'est pas modifiée. Toutes les cellules évoluent **simultanément**, c'est-à-dire qu'on compte le nombre de voisins vivants dans l'ancienne grille (celle reçue en paramètre), afin de calculer le nouvel état qui sera stocké dans la nouvelle grille (celle renvoyée par la fonction). Pour créer la nouvelle grille, on pourra utiliser la fonction `initGrilleM` définie ci-dessus.

7. Écrire une fonction `nombreVivantes` qui reçoit une grille en argument, et parcourt cette grille pour calculer et renvoyer le nombre de cellules vivantes qu'elle contient.

8. Écrire un programme principal qui appelle les fonctions définies ci-dessus pour :

   - Initialiser une grille carrée de taille 5 par 5, avec 7 cellules vivantes, et afficher cette grille initiale.
   - Faire évoluer cette grille pendant 10 tours, en affichant à chaque tour le numéro du tour, la nouvelle grille, et le nombre de cellules vivantes.
   - Ce programme s'arrête après les 10 tours, ou avant dès qu'il n'y a plus aucune cellule vivante dans la grille.

## 1.2 Version graphique

Dans cette partie, on propose de réaliser une visualisation graphique avec turtle.

- écrire une fonction qui affiche une grille dans la fenêtre graphique. On veut afficher le quadrillage, les cellules mortes en blanc, et les cellules vivantes en couleur (choisissez votre couleur).
- ajouter une interface graphique à votre jeu de la vie
- Bonus: permettre l'initialisation manuelle de la grille en cliquant pour générer des cellules vivantes