# Conway's Game of Life

Le jeu de la vie

# Objectif:

**à regarder :** https://www.youtube.com/watch?v=S-W0NX97DB0

Le but du projet est de coder l’algorithme permettant de simuler le jeu de la vie.

Les règles du jeu sont les suivantes (source : https://fr.wikipedia.org/wiki/Jeu\_de\_la\_vie) :

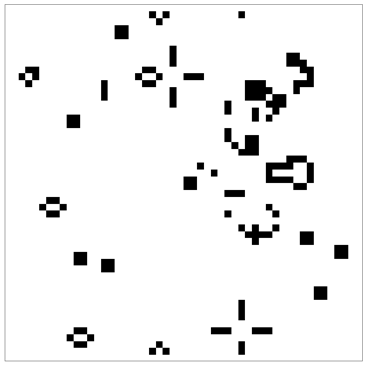
Le jeu se déroule sur une grille à deux dimensions (matrice) dont les cases sont appelées « cellules », par analogie avec les cellules vivantes.

Chaque cellule peut prendre deux états distincts : « vivante » ou « morte » (1 ou 0).

Une cellule possède huit voisins, qui sont les cellules adjacentes horizontalement, verticalement et diagonalement.

À chaque itération, l'état d’une cellule est entièrement déterminé par l’état de ses huit cellules voisines, selon les règles suivantes :

* une cellule morte possédant exactement trois cellules voisines vivantes devient vivante (elle naît) ;
* une cellule vivante possédant deux ou trois cellules voisines vivantes le reste, sinon elle meurt ;



# Instructions :

Pré-requis :

* Une installation python
* Un environnement virtuel avec numpy installé dessus

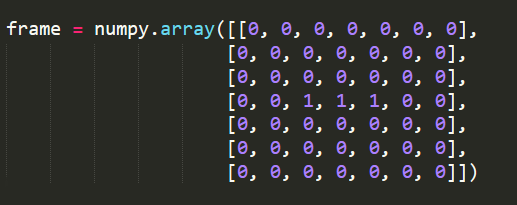
1. Créez un répertoire de projet dans lequel vous créez un script : main.py

Nous voulons modéliser une grille du jeu de la vie de taille 7x7 pour commencer.

On considérera dans la suite du projet que la grille est une matrice de dimension (7, 7) dont les éléments peuvent valoir 0 ou 1 (cellule = élément) :

* 0 si l'élément est mort
* 1 si l'élément est vivant

1. Importez numpy à partir de votre script main.py et créez une variable frame tel que:



*Pour savoir si une cellule va prendre vie ou mourir il faut :*

*connaître son état (1 ou 0) et connaître le nombre de voisins qu’elle possède.*

Nous allons voir maintenant comment calculer le nombre de voisins d’une cellule.

Pour calculer le nombre de voisin vivant d’une cellule il suffit de sommer les valeurs de ses 8 cases voisines. Le résultat de cette somme est égale au nombre de voisins.

**Problème : Comment faisons-nous pour les cases se situant sur les bords de la matrice ? Elles n’ont pas 8 cellules voisines !**

**Solution : Le zero padding !**

Nous allons entourer notre matrice avec des 0 pour créer une sorte de bordure artificielle.

Nous rajoutons donc :

* 2 lignes avec que des 0 (en haut et en bas).
* 2 colonnes avec que des 0 (droite et gauche)
* Le contenu de la grille du jeu de la vie dans la matrice avec bordures commence donc à la ligne / colonne 1.
* Le contenu de la grille du jeu de la vie dans la matrice avec bordure se termine donc à la ligne m - 2 et à la n -2 pour une grille de jeu de taille (m,n)

L'intérêt de rajouter des bordures (zero padding) est de nous permettre de calculer plus facilement le nombre de voisins sans avoir à se soucier de l’effet de bord lors des calculs.

En effet on peut parcourir la nouvelle matrice générée avec le padding de la ligne 1 à la ligne m-2 et de la colonne 1 à la colonne n-2.

Avec ce parcours sur cette matrice avec bordure, tous les éléments de la grille auront 8 voisins y compris ceux qui causaient problème.

1. Voici le squelette de l’algorithme avec du pseudo code:

