Spring 2026

Programmation 2

TP 4 — *Lisez attentivement l'ensemble du document.* — *Nabil Mustafa*

Consignes.

(a) **L'objectif est d'apprendre la programmation en Python.**

Il n'y a pas besoin de se précipiter !

Prenez votre temps et terminez correctement tous les TP précédents avant de commencer avec ce TP.

(b) *Vous devriez lire attentivement les diapositives, pour réviser ce que nous avons fait en CM.*

(c) *Il est important de se concentrer et de lire le texte attentivement et lentement. Si quelque chose dans le texte n'est pas clair après l'avoir lu plusieurs fois, demandez au professeur.*

Tâche 1-a: Exercises

(a) Connectez-vous à `LINUX` avec votre compte utilisateur.

(b) Téléchargez le Jupyter Notebook fichier `Lec3-boucles.ipynb`, disponible sur le site web du module.

(c) Pour le lire, ouvrez un `terminal` dans le *même dossier* que le fichier `Lec3-boucles.ipynb`, et exécuter le command :

```
jupyter-notebook Lec3-boucles.ipynb
```

(d) Pour exécuter un code Python dans le Jupyter Notebook, cliquez sur le code et appuyez sur "SHIFT + ENTER".

(e) **Lisez toutes les informations** dans cette Jupyter Notebook.

(f) **Faites tous les exercices** dans cette Jupyter Notebook.

Tâche 2: Exécuter Python

(a) Connectez-vous à `LINUX` avec votre compte utilisateur.

(b) **Téléchargez** le fichier `gol.py`.

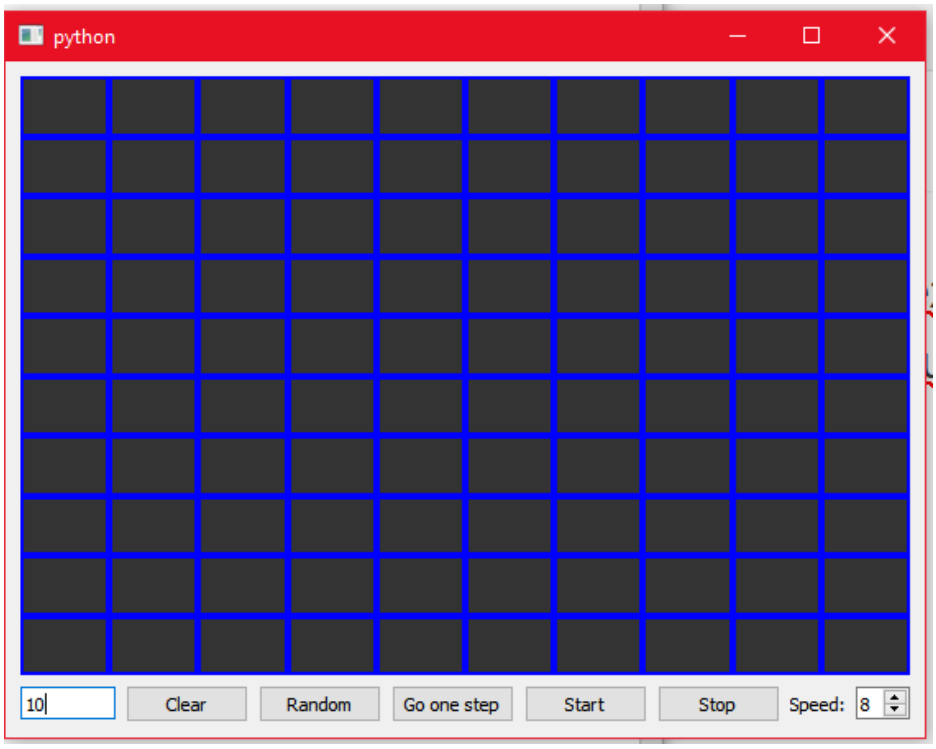
(c) **Éditer** ce fichier avec l'éditeur de votre choix. Par exemple, ouvrez un `terminal` dans le même dossier que le fichier `gol.py`, et exécuter le command:

```
kate gol.py
```

(d) **Exécuter** le code avec `Python`: ouvrez un `terminal` dans le *même dossier* que le fichier `gol.py`, et exécuter le command :

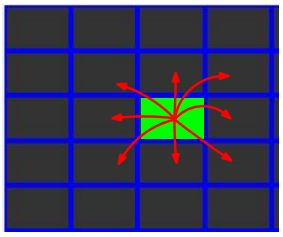
```
python3 gol.py
```

Vous verrez cette interface que j'ai créée pour vous. Vous pouvez essayer tous les boutons, ainsi que cliquer sur la grille pour changer l'état des cellules.

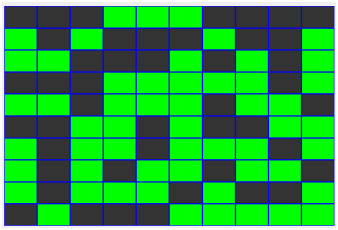


Une description du jeu de la vie

Le jeu se déroule sur une grille des cellules à deux dimensions. Une cellule possède huit voisins, qui sont les cellules adjacentes horizontalement, verticalement et diagonalement:

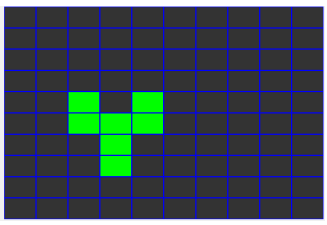


Chaque cellule peut prendre deux états distincts : **vivante** ou **morte**. Voici une exemple d'une grille de taille 10×10:



Le jeu consiste en de nombreuses itérations:

Au début: l'utilisateur décide, pour chaque cellule, si elle est **vivante** ou **morte**. Par exemple, voici une configuration de départ :

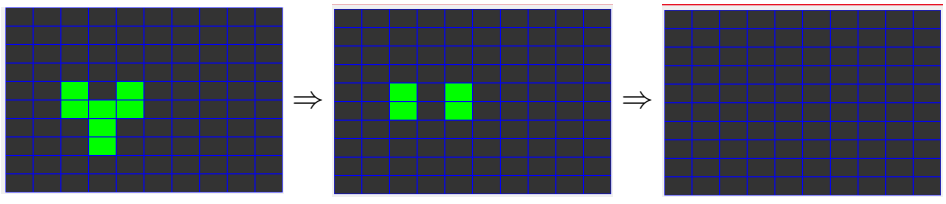


Chaque itération: l'état d'une cellule est entièrement déterminée par l'état de ses huit cellules voisines, selon les règles suivantes :

- **une cellule morte possédant exactement trois cellules voisines vivantes devient vivante; sinon elle reste morte.**
- **une cellule vivante possédant deux ou trois cellules voisines vivantes le reste; sinon elle meurt.**

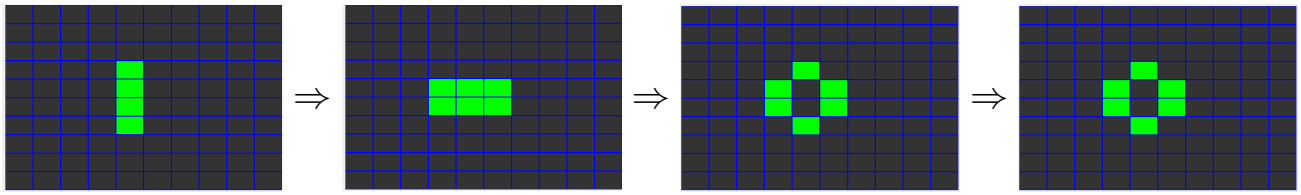
Notez que l'état de toutes les cellules à l'instant $t + 1$ est déterminé exclusivement à partir de l'état de toutes les cellules à l'instant t. Cela signifie que les mises à jour sont simultanées : vous ne pouvez pas modifier l'état d'une cellule et utiliser immédiatement ce nouvel état pour décider du sort de ses voisines.

Voici l'évolution du jeu sur deux itérations :

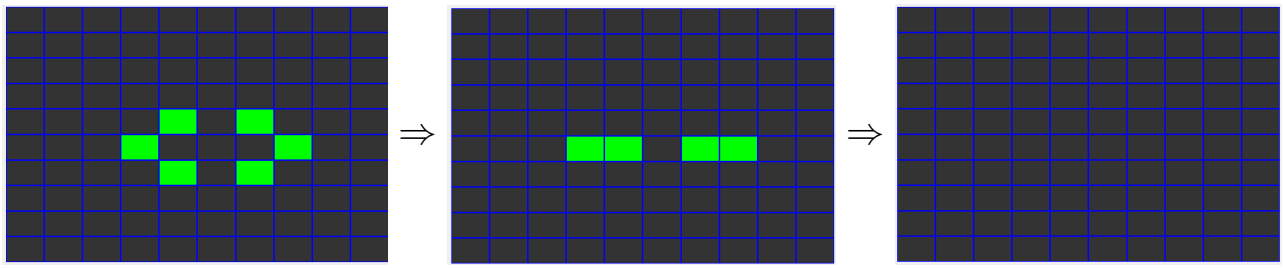


Exemples

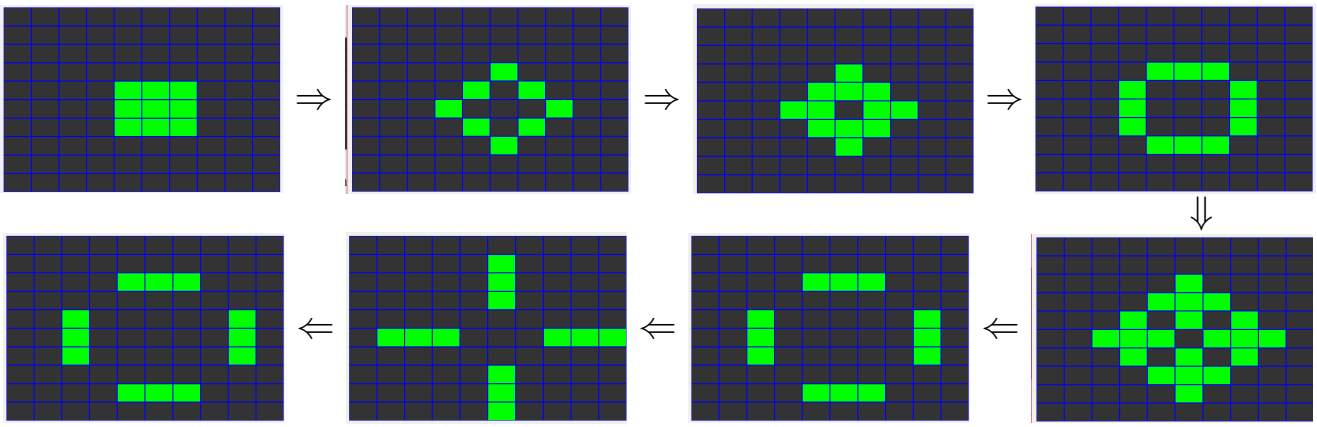
Exemple 1:



Exemple 2:



Exemple 3:



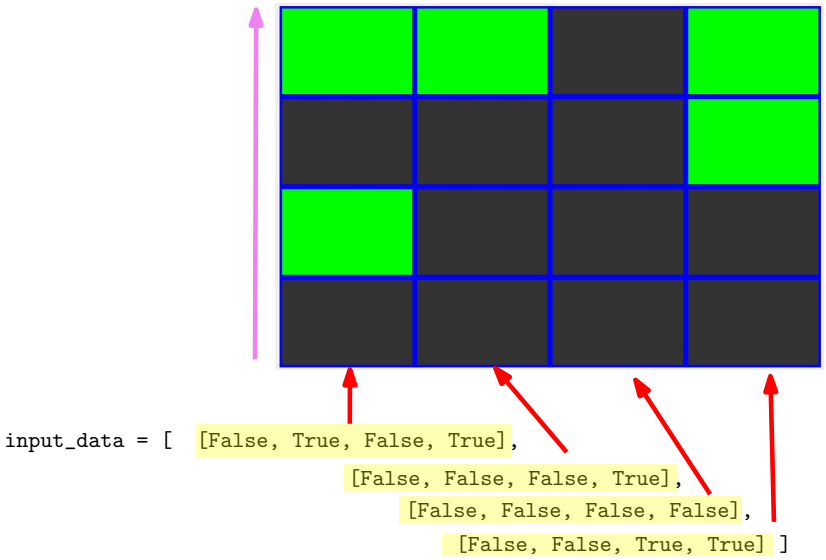
Tâche 3: Ecrivez un algorithme pour Jeu de la vie

Votre objectif est d'ajouter le code dans le fonction `nextState(input_data)` pour calculer et mettre le prochain état de la grille dans le variable `output_data`, qui, à la fin, est retourné par la fonction.

Il renverra la nouvelle grille `output_data` représentant l'état suivant.

La grille est stockée dans le variable `input_data`, qui est une liste de listes. Chaque colonne, de bas en haut, est une liste dans `input_data`.

Par exemple, le grille suivante est stockée comme:



Voici la seule fonction dans `gol.py` que vous devez modifier:

```
def nextState(input_data):
    n = len(input_data)

    """initialization de prochain grille output_data, avec chaque
       valeur de output_data[i][j] = False."""
    output_data = []
    for i in range(0, n):
        output_data.append( [False]*n )

    #ECRIVEZ VOTRE CODE ICI pour remplir output_data correctement

    return output_data
```