**Proiect: Game of Life**

**Introducere**

**Game of Life** este un joc conceput de matematicianul John Conway, care simulează evoluția celulelor pe o grilă bidimensională. Fiecare celulă poate fi vie sau moartă și își schimbă starea în funcție de vecinii săi, conform unui set de reguli.

**Caracteristici principale**

* **Interfață grafică (GUI)**: Utilizarea Tkinter pentru a desena și interacționa cu grila.
* **Control manual și automat**: Posibilitatea de a avansa manual generațiile sau de a porni automat evoluția.
* **Generare aleatorie**: Generare aleatorie a celulelor vii.
* **Personalizare**: Posibilitatea de a modifica dimensiunile grilei.

**Începerea utilizării**

**Cerințe**

* Python 3.x
* Biblioteca Tkinter (inclusă de obicei cu instalările standard de Python)
* Biblioteca NumPy
* Biblioteca **rule-engine**

python game\_of\_life.py

**Primii pași**

După rularea scriptului, se va deschide o fereastră Tkinter cu grila de celule. Puteți să adăugați sau să eliminați celule prin clic pe grilă și să utilizați butoanele pentru a avansa sau reseta jocul.

**Utilizare**

* **Adăugare/Eliminare celule**: Click pe o celulă pentru a o schimba între starea vie (negru) și moartă (alb).
* **Next Step**: Avansează jocul cu o generație.
* **Start/Stop**: Pornește sau oprește evoluția automată a celulelor.
* **Reset**: Resetează grila la starea inițială, toate celulele fiind moarte.
* **Random Generate**: Generează aleatoriu un număr specificat de celule vii.

**Structura proiectului**

* **\_\_init\_\_**: Inițializează fereastra Tkinter, grila de celule și controlează interfața grafică.
* **draw\_board**: Desenează grila de celule pe canvas-ul Tkinter.
* **modify\_cell**: Modifică starea unei celule când se face click pe ea.
* **verify\_condition**: Verifică dacă există cel puțin 3 celule vii.
* **next\_generation**: Calculează și actualizează starea celulelor pentru următoarea generație conform regulilor.
* **start\_stop**: Controlează rularea automată a jocului.
* **reset**: Resetează grila.
* **run\_game**: Rulează jocul automat cu o întârziere între generații.
* **create\_menu**: Creează meniurile de configurare.
* **explanation**: Afișează explicația jocului.
* **get\_random\_input**: Solicită utilizatorului numărul de celule vii pentru generare aleatorie.
* **random\_generate**: Generează aleatoriu celule vii.
* **create\_info\_labels**: Creează etichetele de informare despre numărul de celule vii și moarte.
* **update\_info\_labels**: Actualizează etichetele de informare.
* **modify\_dimensions**: Modifică dimensiunile grilei.

**Exemple de intrare/ieșire**

**Exemplu 1: Configurație manuală**

1. Adăugați câteva celule vii prin clic pe grilă.
2. Apăsați butonul "Next Step" pentru a avansa cu o generație.
3. Observați cum celulele își schimbă starea în funcție de regulile jocului.

**Exemplu 2: Generare aleatorie**

1. Selectați meniul "Random Generate" și introduceți numărul de celule vii.
2. Apăsați "OK" pentru a genera celulele.
3. Apăsați butonul "Start/Stop" pentru a porni evoluția automată a celulelor.

**Decizii tehnice**

1. **Utilizarea Tkinter**: Tkinter a fost ales pentru simplitatea și disponibilitatea sa în instalările standard de Python. Este suficient pentru crearea unei interfețe grafice simple și interactive.
2. **Utilizarea NumPy**: NumPy oferă suport eficient pentru manipularea matricelor și a fost utilizat pentru a gestiona stările celulelor pe grilă.
3. **Reguli personalizabile cu rule-engine**: Biblioteca **rule-engine** a fost utilizată pentru a permite definirea și evaluarea flexibilă a regulilor jocului, facilitând extinderea sau modificarea lor în viitor.

**Raționamentul din spatele deciziilor tehnice**

Proiectul de mai sus implementează "Game of Life" (Jocul Vieții), un automaton celular inventat de matematicianul John Conway, folosind biblioteca **tkinter** pentru interfața grafică și **numpy** pentru manipularea matricei care reprezintă tabla de joc. Iată raționamentul din spatele principalelor decizii tehnice:

1. **Utilizarea bibliotecii tkinter**:
   * **tkinter** este o bibliotecă standard în Python pentru crearea interfețelor grafice. A fost aleasă pentru simplitatea și integrarea sa nativă cu Python, permițând dezvoltarea rapidă a aplicațiilor GUI.
2. **Utilizarea bibliotecii numpy**:
   * **numpy** este o bibliotecă eficientă pentru manipularea array-urilor multidimensionale. A fost aleasă pentru manipularea matricei de celule datorită performanței sale superioare în comparație cu listele native Python, mai ales la operațiuni care implică calcule matriciale.
3. **Funcții pentru numărarea celulelor**:
   * **count\_ceels** și **count\_neighbor\_values** sunt funcții auxiliare care numără celulele vii și moarte din matrice și valorile vecinilor unei celule specifice. Aceste funcții sunt necesare pentru a implementa regulile Jocului Vieții.
4. **Clasa GameOfLife**:
   * Aceasta clasează întreaga logică și interfața jocului. Metodele și variabilele clasei sunt bine organizate pentru a separa logica jocului de interfața grafică și pentru a facilita extensibilitatea și mentenabilitatea codului.
5. **Metoda draw\_board**:
   * Responsabilă de desenarea tablei de joc pe canvas-ul **tkinter**. Această metodă este chemată de fiecare dată când se modifică starea tablei pentru a reflecta vizual starea actuală a celulelor.
6. **Metoda modify\_cell**:
   * Permite utilizatorului să modifice starea celulelor (vii/moarte) prin click-uri pe canvas. Aceasta este o interacțiune de bază necesară pentru a seta condițiile inițiale ale jocului.
7. **Reguli pentru evoluția celulelor**:
   * Metodele **verify\_condition** și **next\_generation** folosesc **rule\_engine** pentru a defini și evalua regulile jocului. Acest lucru asigură că logica jocului este clar separată și modulară.
8. **Metoda run\_game**:
   * Aceasta gestionează rularea continuă a jocului. Este apelată atunci când utilizatorul apasă butonul de start și continuă să genereze următoarele stări ale tablei până când jocul este oprit sau condițiile nu mai sunt îndeplinite.
9. **Meniuri și controale**:
   * **create\_menu**, **create\_info\_labels**, **update\_info\_labels**, **modify\_dimensions**, și alte metode asociate gestionează interfața utilizatorului, incluzând meniuri pentru setarea dimensiunilor tablei, generarea aleatorie a celulelor vii și afișarea informațiilor despre numărul celulelor vii și moarte.
10. **Reguli în rule\_engine**:
    * Aceste reguli definesc condițiile pentru supraviețuirea, moartea sau nașterea celulelor, asigurând o separare clară a logicii jocului de restul codului.

**Lista membrilor echipei**

Echipa care a lucrat la proiectul "Game of Life" include următorii membri:

1. Leonard Muşat
2. Luca Nastasa Baras
3. Mototolea David Cosmin