**Описание домашнего задания**

**Цель**

Расширить реализацию лабораторной работы 8, добавив графический пользовательский интерфейс для визуализации мира, управления симуляцией и отображения статистики.

**Основные требования**

1. **Окно приложения** с тремя ключевыми зонами:
   * **Верх**: ползунок времени суток (слайдер), позволяющий прокручивать симуляцию вперёд/назад.
   * **Центр**: карта мира в виде canvas (Graph-элемент PySimpleGUI), где по координатам рисуются:
     + lumiere — жёлтые квадраты.
     + obscurite — синие квадраты.
     + demi — серые квадраты.
     + Животные:
       - malheureux — фиолетовые кружки, размер пропорционален их «масштабу».
       - pauvre — жёлтые кружки, аналогично по размеру.
   * **Низ**: панель статистики (текстовые поля/таблица), показывающая численность каждого вида, средний радиус обзора животных и другие ключевые метрики.
2. **Взаимодействие**
   * Перемещение ползунка сразу обновляет кадр симуляции.
   * При движении мыши по карте можно отображать радиус обзора выбранного животного (полупрозрачное кольцо вокруг кружка).
3. **Дополнительные детали**
   * Каждый агент хранит параметр vision\_radius, влияющий на группирование/разобщение.
   * При наведении на агент отображается его обзор и информация об окружении (например, ближайшие соседи в пределах vision\_radius).
   * Интерфейс реагирует на события: запуск/паузу симуляции, смену скорости, сброс мира.

**3. Рекомендации по реализации**

1. **Пример разметки окна (PySimpleGUI)**

import PySimpleGUI as sg

layout = [

[sg.Text('Time:'), sg.Slider(range=(0, 24), orientation='h', key='-TIME-', enable\_events=True)],

[sg.Graph(canvas\_size=(600, 400), graph\_bottom\_left=(0,0), graph\_top\_right=(100,100), key='-MAP-')],

[sg.Multiline(size=(80,5), key='-STATS-', disabled=True)]

]

window = sg.Window('Ecosystem Simulator', layout, finalize=True)

1. **Отрисовка карты**
   * Используйте метод Graph.draw\_rectangle/draw\_circle для растений и животных соответственно.
   * Цвета задаются фиксировано: жёлтый — '#FFFF00', синий — '#0000FF', серый — '#808080', фиолетовый — '#800080'.
   * Для отображения обзора — Graph.draw\_circle с прозрачностью (через параметр fill\_color и line\_color).
2. **Обработка событий**
   * В основном loop’е — event, values = window.read(timeout=100);
   * По событию -TIME- обновлять состояние мира и перерисовывать canvas;
   * Для hover-событий можно отслеживать координаты мыши через window['-MAP-'].get\_last\_mouse\_position().
3. **Статистика**
   * В функции stats.py собрать данные о числе особей каждого вида, среднем радиусе обзора и т.п.
   * Обновлять содержимое Multiline через window['-STATS-'].update(new\_text).