**Documentación del Código de Simulación de Enjambre de Partículas**

**Descripción General**

Este programa simula el comportamiento de un enjambre de partículas utilizando el algoritmo de optimización de enjambre de partículas (PSO, por sus siglas en inglés). La interfaz gráfica está construida con tkinter, y las partículas se representan en un gráfico de dispersión. Cada partícula busca minimizar la función objetivo f(x,y)=x2+y2, es decir, se mueve hacia el origen (0, 0).

**Clases y Métodos**

**1. Clase Particula**

Representa una partícula dentro del enjambre. Cada partícula tiene una posición, velocidad, memoria, coeficientes cognitivo y social, y una mejor posición personal.

**Métodos:**

* **\_\_init\_\_(self, id, x\_min, x\_max, y\_min, y\_max)**  
  Constructor de la clase que inicializa la partícula con una posición aleatoria dentro de los límites definidos por x\_min, x\_max, y\_min, y\_max. También asigna valores aleatorios para la velocidad, coeficientes y la memoria de la partícula.
* **evaluar(self, posición)**  
  Función objetivo que evalúa la calidad de una posición dada. En este caso, la función es f(x,y)=x2+y2 .
* **actualizar(self, mejor\_global)**  
  Actualiza la velocidad y la posición de la partícula basándose en su mejor posición personal y la mejor posición global encontrada hasta el momento. La actualización de la velocidad se realiza utilizando los coeficientes cognitivo y social.

**2. Clase Enjambre**

Esta clase representa el enjambre completo, que está compuesto por varias partículas. Permite mover las partículas y gestionar la mejor posición global.

**Métodos:**

* **\_\_init\_\_(self, cantidad, x\_min, x\_max, y\_min, y\_max)**  
  Constructor de la clase que crea un número específico de partículas (cantidad) con posiciones aleatorias dentro de los límites dados.
* **obtener\_dataframe(self)**  
  Devuelve un DataFrame de pandas con los datos de las partículas, incluyendo su ID, posición X, posición Y, velocidad, memoria (cuántas posiciones han guardado), coeficientes cognitivo y social, y el mejor valor obtenido por cada partícula.
* **obtener\_mejor\_global(self)**  
  Calcula y devuelve la mejor posición global encontrada por el enjambre, es decir, la posición de la partícula que tiene el menor valor de la función objetivo.
* **mover\_particulas(self)**  
  Mueve todas las partículas del enjambre basándose en la mejor posición global. Actualiza las posiciones y velocidades de las partículas.

**3. Clase AplicacionEnjambre**

Es la interfaz gráfica que permite interactuar con el enjambre de partículas. Esta clase usa tkinter para la interfaz y matplotlib para mostrar los gráficos de las partículas.

**Métodos:**

* **\_\_init\_\_(self, root)**  
  Constructor que inicializa la ventana de la aplicación y configura los elementos de la interfaz gráfica, como las entradas de parámetros y los botones.
* **generar\_enjambre(self)**  
  Lee los parámetros introducidos por el usuario (cantidad de partículas, límites de X e Y) y crea un enjambre de partículas. Luego muestra los datos de las partículas en una tabla y genera la gráfica de dispersión.
* **mover\_enjambre(self)**  
  Llama al método mover\_particulas del enjambre, actualizando las posiciones de las partículas. Muestra los nuevos datos en la tabla y la gráfica.
* **mostrar\_tabla(self, df)**  
  Muestra los datos del enjambre en una tabla usando ttk.Treeview. Los datos mostrados incluyen ID, posiciones, velocidad, memoria, coeficientes y el mejor valor de cada partícula.
* **mostrar\_grafica(self, enjambre)**  
  Dibuja un gráfico de dispersión con matplotlib, donde cada partícula es representada como un punto en el plano XY. También muestra la ID de cada partícula sobre su punto en el gráfico.

**Interfaz Gráfica**

La aplicación usa tkinter para crear una ventana con los siguientes componentes:

* **Campos de Entrada**: Permiten al usuario ingresar los parámetros del enjambre, como el número de partículas y los límites de las coordenadas (X y Y).
* **Botones**:
  + "Crear Enjambre": Crea un enjambre con las partículas de acuerdo a los parámetros ingresados y muestra los datos en la tabla y el gráfico.
  + "Mover Enjambre": Actualiza la posición de las partículas en base a la mejor posición global encontrada y muestra los resultados.
* **Tabla**: Muestra los datos de las partículas en un formato de tabla utilizando ttk.Treeview.
* **Gráfico de dispersión**: Muestra la posición actual de las partículas en un gráfico utilizando matplotlib. Las partículas se representan como puntos azules, y su ID se muestra sobre cada punto.

**Uso**

1. **Crear Enjambre**:
   * El usuario ingresa los valores para el número de partículas y los límites de las coordenadas X e Y.
   * Se genera un enjambre de partículas con posiciones aleatorias dentro de esos límites.
   * Se muestra la tabla con los datos de las partículas y un gráfico de dispersión con sus posiciones iniciales.
2. **Mover Enjambre**:
   * Las partículas se mueven hacia la mejor posición global encontrada por el enjambre.
   * Se actualizan las posiciones y velocidades de las partículas, y se muestra la tabla y el gráfico actualizados.

**Dependencias**

Este código utiliza las siguientes bibliotecas externas:

* tkinter: Para la interfaz gráfica de usuario.
* matplotlib: Para crear los gráficos de dispersión.
* pandas: Para manejar los datos de las partículas y mostrarlos en una tabla.

Asegúrate de tener estas bibliotecas instaladas en tu entorno de Python para ejecutar el programa correctamente.