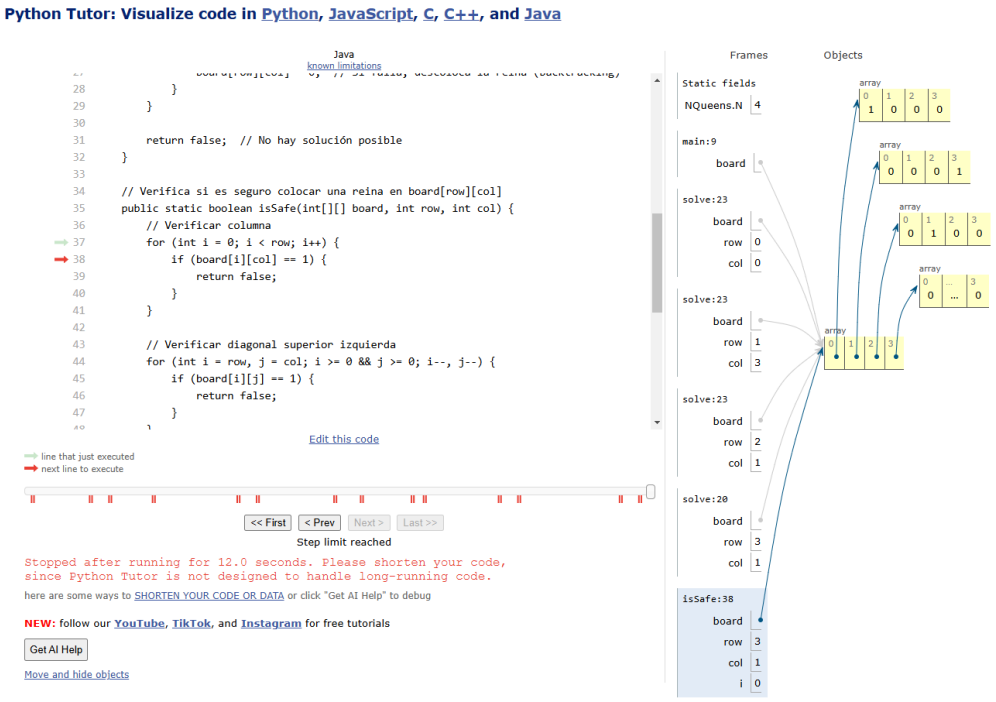
Actividad 1 - Análisis de Código



Explicación del Backtracking en el Problema de N-Reinas

1. ¿Cómo funciona el backtracking en este problema?

   > Objetivo: el algoritmo trata de poner una reina en cada fila, empezando desd ela primera (Fila 0)

   En cada fila, prueba cada columna. Si la posición es segura (sin otra reina en la msima columna o diagonla), entocnes coloca una reina.

   ☐ Camino Feliz 😊→ Si se consigue poner una reina en todas las columnas

   ☐ Camino Triste 😟 → SI no se puede poner una reina en ninguna columna de una fila, "retrocede" , es decir utiliza `backtracking`, desechará esa última reina mal colocada y prueba la siguiente opción.

2. **¿Qué pasa cuando el algoritmo encuentra una solución?**

   Cuando el algoritmo coloca reinas en todas las filas (hasta la fila `N`), sencuentra una solución correcta y la función `solve` devolverá `true`.

   ☐ Cuando se encuentra una solución se imprimirá el tablero con las reinas correctamente colocadas.

3. **¿Qué ocurre cuando no puede colocar más reinas?**

   Si llega a una fila y no puede poner una reina en ninguna columna (todas son incorrectas), `retrocede`, quitando la reina de la fila anterior y prueba otras opciones ahí.

   Si retrocede hasta la primera fila y no encuentra más opciones válidas, concluye que no hay soluciones (para `N=4`, solo hay dos configuraciones posibles).

4. **¿Qué sucede cuando el algoritmo "retrocede"?**

   Cuando retrocede, deshace el último movimiento en el tablero, quitando la última reina colocada y marcando esa posición como libre (la vuelve a `0`).

   Luego intenta poner una reina en la siguiente columna disponible. Este retroceso permite al algoritmo probar diferentes configuraciones hasta encontrar una solución.

5. **Modificaciones para aumentar `N` a 8:**

   - Para cambiar el tamaño del tablero a 8×8, simplemente cambio el valor de `N` a 8:

     ```
     private static final int N = 8;
     ```

   - El tiempo de ejecución aumentará significativamente porque el número de combinaciones posibles crece exponencialmente. Sin embargo, el backtracking reduce el tiempo de ejecución al descartar configuraciones inválidas antes de explorarlas por completo.

6. **Importancia del método `isSafe`:**

   El método `isSafe` es crucial porque verifica si una posición es segura antes de colocar una reina. Sin esta verificación, el algoritmo podría colocar reinas en posiciones conflictivas, lo que resultaría en soluciones inválidas.

   `isSafe` permite que el backtracking sea eficiente al evitar configuraciones de tablero que no cumplen las restricciones, reduciendo así el número de llamadas recursivas necesarias.