recipoGraph.jar가 JDK 22에서는 그래픽 오류로 인해 열리지 않았으나, JDK 8을 다시 설치하고 실행했을 때 문제가 해결된 이유는 여러 가지 기술적 차이 때문입니다. 그 이유를 설명하겠습니다.

**1. Java 모듈 시스템**

* **JDK 9**부터 Java는 모듈 시스템을 도입했습니다. 이는 Java의 구조를 더 안전하고 효율적으로 관리하기 위한 변화였으나, 이전에 모듈화를 고려하지 않고 개발된 애플리케이션(특히 Java 8 이전의 프로그램)은 모듈 시스템으로 인한 호환성 문제가 발생할 수 있습니다.
* **Java 8**에서는 모듈 시스템이 존재하지 않기 때문에, 이전 방식대로 라이브러리와 클래스에 대한 접근이 자유로워 recipoGraph.jar와 같은 애플리케이션이 정상적으로 작동할 수 있습니다.

**2. 그래픽 API와 Java3D, JOGL의 호환성**

* **Java3D**와 **JOGL** 같은 그래픽 라이브러리들은 Java 8을 기준으로 설계된 경우가 많습니다. JDK 22에서 이러한 라이브러리를 사용할 때, 최신 버전에서 변경된 그래픽 환경 또는 내부 API 변화로 인해 오류가 발생할 수 있습니다. 특히 JDK 22의 최신 그래픽 라이브러리와 호환되지 않는 경우 ArrayIndexOutOfBoundsException과 같은 오류가 발생할 수 있습니다.
* **Java 8**에서는 해당 라이브러리들이 그 당시의 그래픽 환경과 일관되게 동작하기 때문에 호환성 문제가 없습니다.

**3. JVM 옵션의 차이**

* JDK 9 이후로 모듈 시스템이 도입됨에 따라, 특정 클래스를 외부에서 접근하려면 JVM 옵션(예: --add-opens)을 추가해야 합니다. 이러한 추가 설정이 없으면 **Java 9 이상**에서 그래픽 라이브러리와의 호환성 문제가 발생할 수 있습니다. 하지만 Java 8에서는 이러한 설정이 필요하지 않아 애플리케이션이 정상적으로 동작합니다.

**4. 기존 라이브러리와의 호환성**

* 일부 **구형 라이브러리**는 JDK의 최신 버전에서 더 이상 지원되지 않거나, 그 동작 방식이 변경될 수 있습니다. JDK 22는 새로운 기능과 보안 향상을 제공하지만, 이전 버전의 라이브러리나 코드가 이를 제대로 활용하지 못할 수 있습니다. 반면, Java 8은 이러한 구형 라이브러리와 더 잘 호환됩니다.

**결론**

* **JDK 22**는 최신 기능과 모듈화를 제공하지만, 이는 기존 애플리케이션이나 라이브러리와의 호환성 문제를 일으킬 수 있습니다. **Java 8**은 이러한 최신 기능이 없기 때문에 기존 애플리케이션이 문제없이 동작할 가능성이 더 큽니다. recipoGraph.jar도 이러한 이유로 JDK 8에서 정상적으로 실행된 것입니다.