Java Programming Task: University Management System

Objective:

Develop a University Management System in Java, showcasing inheritance, polymorphism, abstraction, and encapsulation. The system should handle courses, students, and instructors using `HashMaps`, `ArrayLists`, and `Sets` to manage relationships and data efficiently.

Task Requirements:

1. **Base Entity Class:**
   - Create an abstract class `Entity` with private attributes:
     - `id` (String)
     - `name` (String)
   - Define a constructor that initializes these attributes.
   - Provide getters and setters for each attribute.
2. **Instructor Class:**
   - Derive a class `Instructor` from `Entity`.
   - Add private attributes:
     - `salary` (double)
     - `coursesTaught` (Set<String> to hold course IDs)
   - Implement methods to add and remove a course from `coursesTaught`.
   - Override a method to display detailed information about the instructor, including the courses taught.
3. **Student Class:**
   - Derive a class `Student` from `Entity`.
   - Add private attributes:
     - `creditsEarned` (int)
     - `enrolledCourses` (ArrayList<String> to hold course IDs)
   - Implement methods to enroll and drop a course from `enrolledCourses`.
   - Override a method to display detailed information about the student, including the courses enrolled in.
4. **Course Class:**
   - Create a class `Course` with attributes:
     - `courseId` (String)
     - `title` (String)
     - `credits` (int)
     - `instructorId` (String to link to an Instructor)
   - Include a constructor and appropriate getters and setters.

5. **University Management System:**
   - Use `HashMaps` to manage entities:
     - `HashMap<String, Instructor>` to store instructors, where keys are `id` values.
     - `HashMap<String, Student>` to store students, where keys are `id` values.
     - `HashMap<String, Course>` to store courses, where keys are `courseId` values.
   - Implement CRUD operations for each entity, including complex operations such as:
     - `assignInstructorToCourse(String instructorId, String courseId)`: Assigns an instructor to a course.
     - `registerStudentForCourse(String studentId, String courseId)`: Registers a student for a course.
     - `displayAllCoursesTaughtByInstructor(String instructorId)`: Lists all courses taught by a specific instructor.
     - `listStudentsEnrolledInCourse(String courseId)`: Lists all students enrolled in a specific course.

Additional Complex Methods:

1. **Calculating Workload:**
   - Implement a method in the `Instructor` class, `calculateWorkload()`, that calculates the instructor's workload based on the number of courses taught and credits per course.
2. **Academic Progress Tracker:**
   - Implement a method in the `Student` class, `calculateAcademicProgress()`, that calculates the student's academic progress based on credits earned vs. total credits available.

Implementation Notes:

- Ensure all classes are properly encapsulated and use collections (`ArrayLists`, `Sets`, and `HashMaps`) effectively.
- The `Entity` class should be abstract and must not be instantiated.
- Demonstrate polymorphism by implementing method overriding where appropriate.
- The system should prevent assigning an instructor to a course they are already teaching and enrolling a student in a course they are already enrolled in.