STUDENT MANAGEMENT SYSTEM

Members

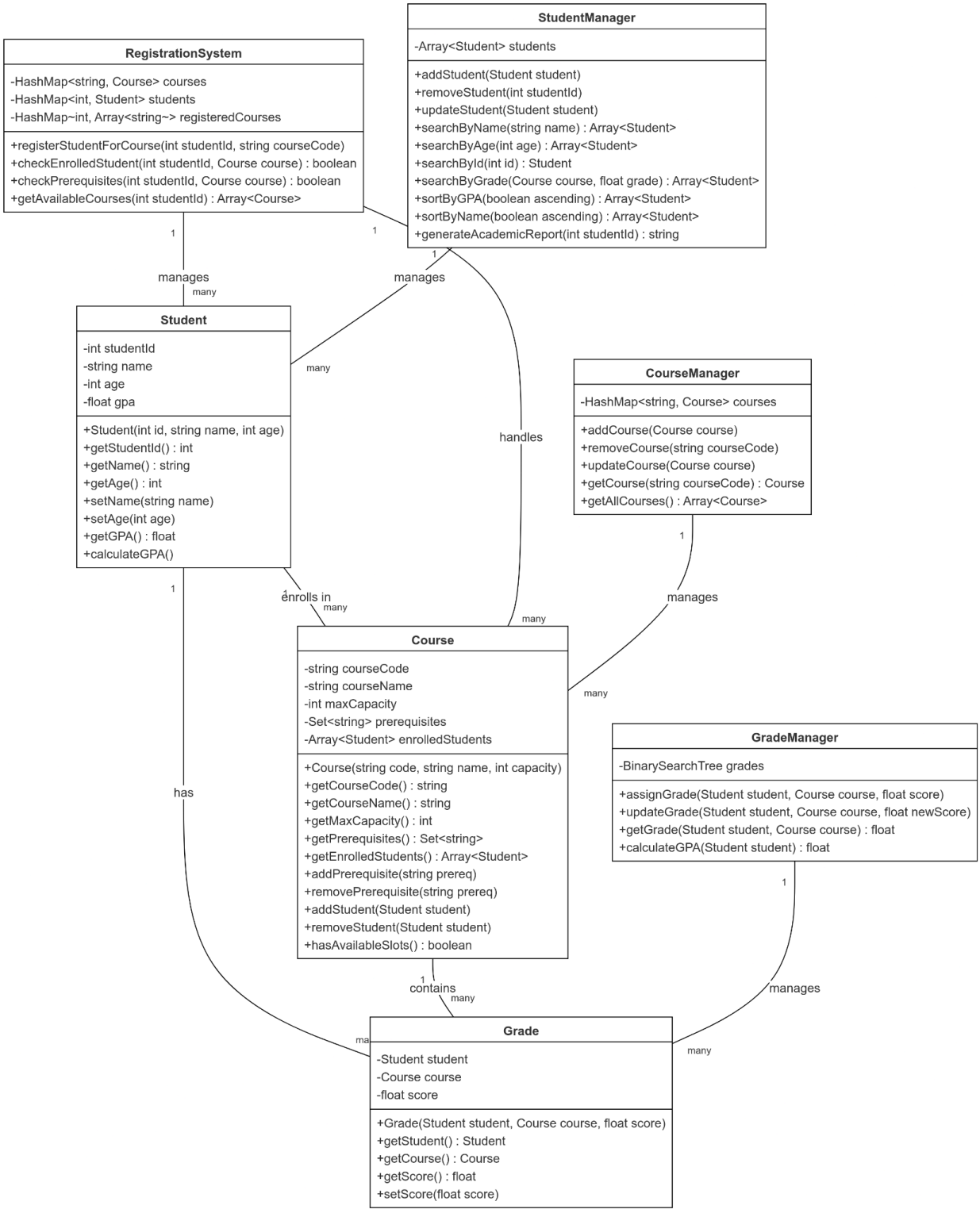
- Đào Đình Trung (22070932)
- Trần Mạnh Đức (22070372)
- Nhữ Quang Minh (22070406)
- Lã Nhật Bảo Duy (22071044)

About the topic

-

System Overview

Entity Design



Core components

1. Student Data Management

Class `Student:`

- **Purpose**: Represents individual student records.
- **Key Attributes**:
  - studentId (int): Unique identifier
  - name (string): Student's full name
  - age (int): Student's age
  - gpa (float): Calculated Grade Point Average
- **Key Methods**:
  - Calculate GPA
  - Get/Set basic information
- **Data Structures**:
  - Array/Linked List for reserving student records

Class `StudentManager:`

- **Purpose**: Handles operations on student collections
- **Key Features**:
  - Advanced search functionality (by name, age, ID, grade)
  - Sorting capabilities (by GPA, name)
  - Academic report generation
- **Algorithms**:
  - Uses sorting algorithms for organizing student data
  - Implements search algorithms for efficient lookup

2. Course Management

Class **Course:**

- **Purpose:** Maintains course information and enrollment.
- **Key Attributes:**
  - courseCode (string): Unique identifier generated randomly
  - courseName (string): Name of the course
  - maxCapacity (int): Maximum allowed students
  - prerequisites (Set<string>): Required courses
  - enrolledStudents (Array<Student>): Currently enrolled students
- **Key Methods**:
  - Manage prerequisites for the courses
  - Handle student enrollment

- Check availability

Class **CourseManager**:

- **Purpose**: Manages course operations
- **Data Structures:** HashMap for O(1) course lookup
- **Key Operations:**
  - Add/Remove courses
  - Update course information
  - Retrieve course details

3. Grade Management

Class **Grade**:

- **Purpose**: Links students, courses, and scores
- **Key Attributes:**
  - student (Student): Reference to student
  - course (Course): Reference to course
  - score (float): Numerical grade

Class **GradeManager**:

- **Purpose**: Handles grade operations and calculations
- **Data Structure**: Binary Search Tree for efficient grade organization
- **Key Features:**
  - Grade assignment and updates
  - GPA calculation
  - Grade retrieval

4. Course Registration System

Class **RegistrationSystem**:

- **Purpose**: Manages course registration process
- **Key Components**:
  a. courses (HashMap<string, Course>)
  b. students (HashMap<int, Student>)
  c. registeredCourses (HashMap<int, Array<string>>)
- **Key Operations**:
  a. Register Student For Course:
     - Validates student and course existence

        - Checks course capacity
        - Verifies prerequisites
        - Handles enrollment
    b. Check Enrollment Status
    c. Verify Prerequisites
    d. List Available Courses

Data Structures Used For This Project

HashMap / Dictionary

- **Used for**: Course lookup, student registration, course registration
- **Benefits:** O(1) access time for quick lookups

Array / Linked Lists

- **Used for:** Storing multiple student records, enrolled students
- **Benefits:** Sequential access, easy iteration

Sets

- **Used for:** Course prerequisites
- **Benefits:** No duplicate entries, efficient lookup

Binary Search Tree

- **Used for:** Grade organization
- **Benefits:** Efficient searching and sorting, O(log n) operations

Why do we opt for these data structures?

- For the sake of boosting the performance of the system, we are considering these criterias:
    - HashMap: usage ensures O(1) lookup time for frequent operations
    - Binary Search Tree: provides efficient grade organization
    - Optimized search algorithms for student lookup
    - Efficient data structures for managing relationships between entities

Error Handling

Implementing some comprehensive error methods that checking for:

1. Course Registration:
    - Invalid student/course IDs
    - Duplicate enrollments
    - Capacity limits
    - Prerequisite requirements
2. Data Validation:
    - Student information
    - Course details
    - Grade entries

Advanced Features (Optional)

1. **Search Capabilities**:
    - Multiple search criteria support
    - Flexible student lookup
    - Course availability checking
2. **Sorting Functions**:
    - GPA-based sorting
    - Alphabetical name sorting
    - Both ascending and descending options
3. **Reporting**:
    - Comprehensive academic reports
    - Enrollment statistics
    - Grade distributions
4. **Potential areas for expanding the system** (**optional**):
    - Attendance tracking
    - Schedule management
    - Academic performance analytics
    - Integration with external systems
    - Advanced reporting capabilities