**Case Study: School Management System**

**Context:**

You are required to develop a console-based School Management System using Core Java. The application will allow school administrators to manage students, teachers, courses, and academic records. The application will utilize Java Collections for storing student and teacher information, implement Object-Oriented Programming principles, and include exception handling to manage potential errors.

**Requirements:**

**Features:**

1. **Add and Remove Students:**
   * Allow school administrators to add new students to the system.
   * Allow school administrators to remove students from the system.
2. **Update Student Details:**
   * Allow school administrators to update details of existing students, such as name, grade, and contact information.
3. **Add and Remove Teachers:**
   * Allow school administrators to add new teachers to the system.
   * Allow school administrators to remove teachers from the system.
4. **Update Teacher Details:**
   * Allow school administrators to update details of existing teachers, such as name, subject taught, and contact information.
5. **Manage Courses:**
   * Allow school administrators to add new courses to the system.
   * Allow school administrators to remove courses from the system.
6. **Assign Students to Courses:**
   * Allow school administrators to assign students to courses.
   * Ensure that students are assigned to appropriate courses based on their grade level.
7. **Assign Teachers to Courses:**
   * Allow school administrators to assign teachers to courses.
   * Ensure that teachers are assigned to courses they are qualified to teach.

**Classes and Objects:**

1. **Student Class:**
   * Attributes: id, name, grade, contactNumber.
   * Methods: Constructors, getters and setters, toString method.
2. **Teacher Class:**
   * Attributes: id, name, subject, contactNumber.
   * Methods: Constructors, getters and setters, toString method.
3. **Course Class:**
   * Attributes: id, name, teacherId, studentIds.
   * Methods: Constructors, getters and setters, toString method.
4. **SchoolManagementSystem Class:**
   * Attributes: HashMap<Integer, Student> students, HashMap<Integer, Teacher> teachers, HashMap<Integer, Course> courses.
   * Methods:
     + addStudent(Student student): Adds a new student to the system.
     + removeStudent(int studentId): Removes a student from the system.
     + updateStudent(int studentId, Student updatedStudent): Updates details of an existing student.
     + addTeacher(Teacher teacher): Adds a new teacher to the system.
     + removeTeacher(int teacherId): Removes a teacher from the system.
     + updateTeacher(int teacherId, Teacher updatedTeacher): Updates details of an existing teacher.
     + addCourse(Course course): Adds a new course to the system.
     + removeCourse(int courseId): Removes a course from the system.
     + assignStudentToCourse(int studentId, int courseId): Assigns a student to a course.
     + assignTeacherToCourse(int teacherId, int courseId): Assigns a teacher to a course.
     + Helper methods for input validation and exception handling.

**Deliverables:**

1. Complete source code for the School Management System.
2. Documentation including:
   * How to run the application.
   * Instructions for each feature.
   * Explanation of the exception handling implemented.
3. A brief report on the application design and how Object-Oriented principles were applied.