Exercise Description:

You are tasked with developing a Java program that models different types of students, specifically undergraduate and graduate students. The program will store these students in a list and display their information and type.

Details:

1. **Student Class (Student.java)**:
    - This is an abstract base class representing a student.
    - **Attributes**:
        - `name`: The name of the student.
        - `department`: The department the student belongs to.
    - **Constructor**:
        - Initializes the student's name and department.
    - **Methods**:
        - `printInfo()`: Prints the student's name and department.
        - `printType()`: An abstract method that must be implemented by subclasses to print the type of student (e.g., undergraduate or graduate).
2. **GraduateStudent Class (GraduateStudent.java)**:
    - A subclass of `Student` representing a graduate student.
    - **Additional Attribute**:
        - `supervisor`: The name of the graduate student's supervisor.
    - **Constructor**:
        - Initializes the name, department, and supervisor.
    - **Methods**:
        - `printInfo()`: Overrides the `printInfo()` method to include the supervisor's name.
        - `printType()`: Implements the `printType()` method to print "I am a graduate student".
3. **UnderGradStudent Class (UnderGradStudent.java)**:
    - A subclass of `Student` representing an undergraduate student.
    - **Additional Attribute**:
        - `year`: The year of study the student is in.
    - **Constructor**:
        - Initializes the name, department, and year of study.
    - **Methods**:
        - `printInfo()`: Overrides the `printInfo()` method to include the year of study.
        - `printType()`: Implements the `printType()` method to print "I am an undergraduate student".
4. **Main Class (Main.java)**:
    - The main class demonstrates the creation of student objects and stores them in an `ArrayList`.

- **Process**:
    - Several students are created, including both undergraduate and graduate students.
    - The program then iterates through the list of students, printing their details using `printInfo()` and indicating their type using `printType()`.

Task:

1. Implement the `Student` abstract class with methods for printing information and an abstract method for printing the student type.
2. Create two subclasses, `GraduateStudent` and `UnderGradStudent`, each with their own attributes and method implementations for printing information and type.
3. In the `Main` class, create instances of both types of students, store them in a list, and print their details and type by iterating through the list.

Bonus:

- Add additional attributes or methods to further differentiate between student types, such as GPA for undergraduate students or research topics for graduate students.
- Implement error handling to ensure valid data is entered for the year of study or supervisor name.

This exercise emphasizes object-oriented programming concepts such as inheritance, polymorphism, and abstraction.