Use the attached UML diagram to create the five classes and their relationships. Remember to correspond UML symbols with the appropriate code modifiers. Create three constructors, one default, one that sets only the data items provided by that class and one that sets all inherited and class data items. Use the attached output from the test program to help implement the `toString` methods for each class. Do not hardcode a year into the `calcAge` method. Use the `GregorianCalendar` class to determine the current year. Read up on modular arithmetic (Java operator: `%`) to use in the `calcHeight` method.

Students are eligible for graduation based on the following conditions:

**IT:** 123+ credit hours and have completed a senior project  
**CS:** 123+ credit hours and have finished programming courses  
**ENG:** 30+ credit hours, 2.7+ GPA, and finished programming courses (to transfer to GaTech)

The test program that is included will be used to test your programs. Use this to test your classes. The output from your program should match the included output.
public class TestStudent {

    public static void main(String[] args) {

        int compare = 0;

        // Create three students
        Student csStudent = new ComputerScienceStudent();
        Student itStudent = new InformationTechnologyStudent();
        Student engStudent = new EngineeringStudent();

        // Set values for csStudent
        csStudent.setGender("male");
        csStudent.setYearOfBirth(1982);
        csStudent.setCreditHours(45);
        csStudent.setGpa(3.2);
        csStudent.setHeight(71);
        csStudent.setZipcode(31419);
        System.out.println(csStudent);
        System.out.println();

        // Set values for itStudent
        itStudent.setGender("female");
        itStudent.setYearOfBirth(1985);
        itStudent.setCreditHours(12);
        itStudent.setGpa(3.6);
        itStudent.setHeight(62);
        itStudent.setZipcode(31406);
        System.out.println(itStudent);
        System.out.println();

        // Set values for engStudent
        engStudent.setGender("male");
        engStudent.setYearOfBirth(1987);
        engStudent.setCreditHours(6);
        engStudent.setGpa(3.0);
        engStudent.setHeight(68);
        engStudent.setZipcode(31412);
        System.out.println(engStudent);
        System.out.println();

        // compare students
        System.out.println("This student has the higher GPA: \n" +
                max(itStudent, engStudent));
        System.out.println("This student has the higher GPA: \n" +
                max(csStudent, engStudent));
        System.out.println("This student has the higher GPA: \n" +
                max(itStudent, csStudent));
    }

    public static Object max(Object o1, Object o2) {
        if (((Comparable)o1).compareTo(o2) > 0) {
            return o1;
        } else {
            return o2;
        }
    }
}
Age: 23  
Height: 5 feet, 11 inches  
Gender: male  
Zipcode: 31419  
GPA: 3.2  
Major: Computer Science  
Credit Hours: 45.0  
Finished Programming: false  
Eligible for Graduation: false

Age: 20  
Height: 5 feet, 2 inches  
Gender: female  
Zipcode: 31406  
GPA: 3.6  
Major: Information Technology  
Credit Hours: 12.0  
Finished Senior Project: false  
Eligible for Graduation: false

Age: 18  
Height: 5 feet, 8 inches  
Gender: male  
Zipcode: 31412  
GPA: 3.0  
Major: Engineering  
Credit Hours: 6.0  
Finished Programming: false  
Eligible for Graduation: false

This student has the higher GPA:
Age: 20  
Height: 5 feet, 2 inches  
Gender: female  
Zipcode: 31406  
GPA: 3.6  
Major: Information Technology  
Credit Hours: 12.0  
Finished Senior Project: false  
Eligible for Graduation: false

This student has the higher GPA:
Age: 23  
Height: 5 feet, 11 inches  
Gender: male  
Zipcode: 31419  
GPA: 3.2  
Major: Computer Science  
Credit Hours: 45.0  
Finished Programming: false  
Eligible for Graduation: false

This student has the higher GPA:
Age: 20  
Height: 5 feet, 2 inches  
Gender: female  
Zipcode: 31406  
GPA: 3.6  
Major: Information Technology  
Credit Hours: 12.0  
Finished Senior Project: false  
Eligible for Graduation: false