Students enroll in Courses; Professors teach Courses (association & communication)

Student

- studentId: Int
- name: String

+ enrollCourse(c:Course)
+ viewGrades()

Course

- courseId: Int
- title: String

+ assignProfessor(p:Professor)
+ addStudent(s:Student)

Professor

- professorId: Int
- name: String

+ teachCourse()
+ assignGrade()
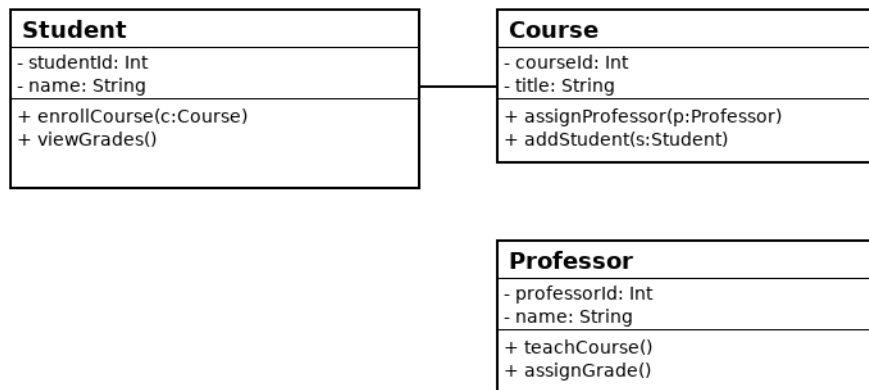
Customer places Order; Order aggregates Product(s)

Customer
- customerId: Int
- name: String
+ placeOrder()
+ viewOrders()

Order
- orderId: Int
- date: Date
- total: Double
+ addProduct(p:Product)
+ checkout()

Product
- productId: Int
- title: String
- price: Double
+ getDetails()
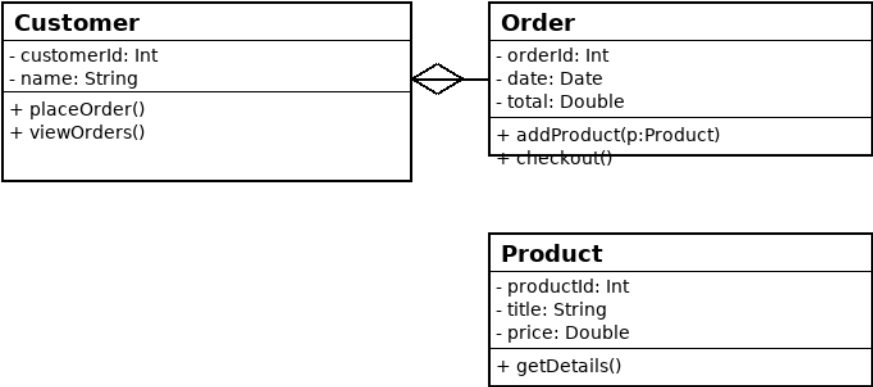
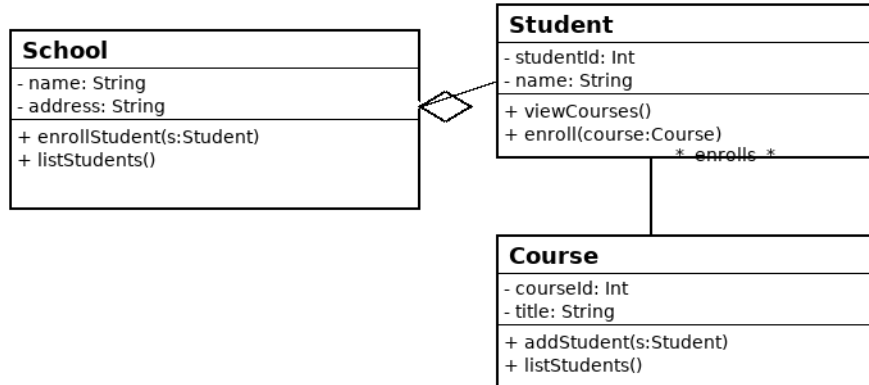Doctors and Patients: Association with communication (consult())

Hospital

- name: String
- address: String

+ registerPatient()
+ schedule()

Doctor

- doctorId: Int
- name: String
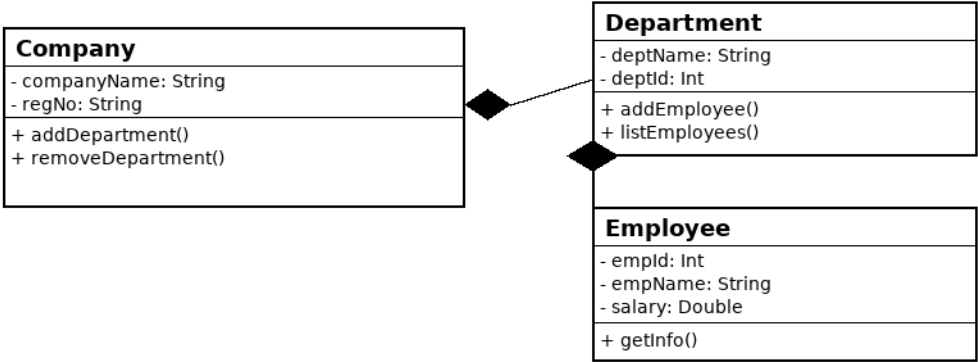- specialization: String

+ consult(p:Patient)
+ prescribe()

consults *

Patient

- patientId: Int
- name: String

+ viewHistory()
+ requestConsultation()

School-Student aggregation, Student-Course association (many-to-many)

Student
- studentId: Int
- name: String

+ viewCourses()
+ enroll(course:Course)

School
- name: String
- address: String

+ enrollStudent(s:Student)
+ listStudents()

* enrolls *

Course
- courseId: Int
- title: String

+ addStudent(s:Student)
+ listStudents()

Composition (Company ◆— Department ◆— Employee)

Company

- companyName: String
- regNo: String

+ addDepartment()
+ removeDepartment()

Department

- deptName: String
- deptId: Int

+ addEmployee()
+ listEmployees()

Employee

- empId: Int
- empName: String
- salary: Double

+ getInfo()

Association (Bank — Customer)

Bank
- bankName: String
- branchCode: String
+ openAccount(c:Customer)
+ getDetails()

Customer
- name: String
- customerId: Int
+ viewBalance()
+ deposit()

1..* associates 0..*
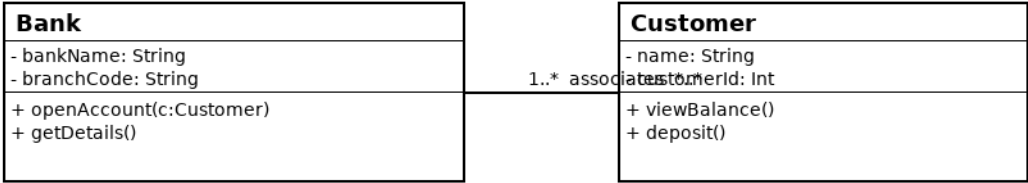
Aggregation (Library ◇— Book)

Library

- name: String
- address: String

+ addBook(b:Book)
+ showBooks()

Book

- title: String
- author: String
- isbn: String

+ getDetails()