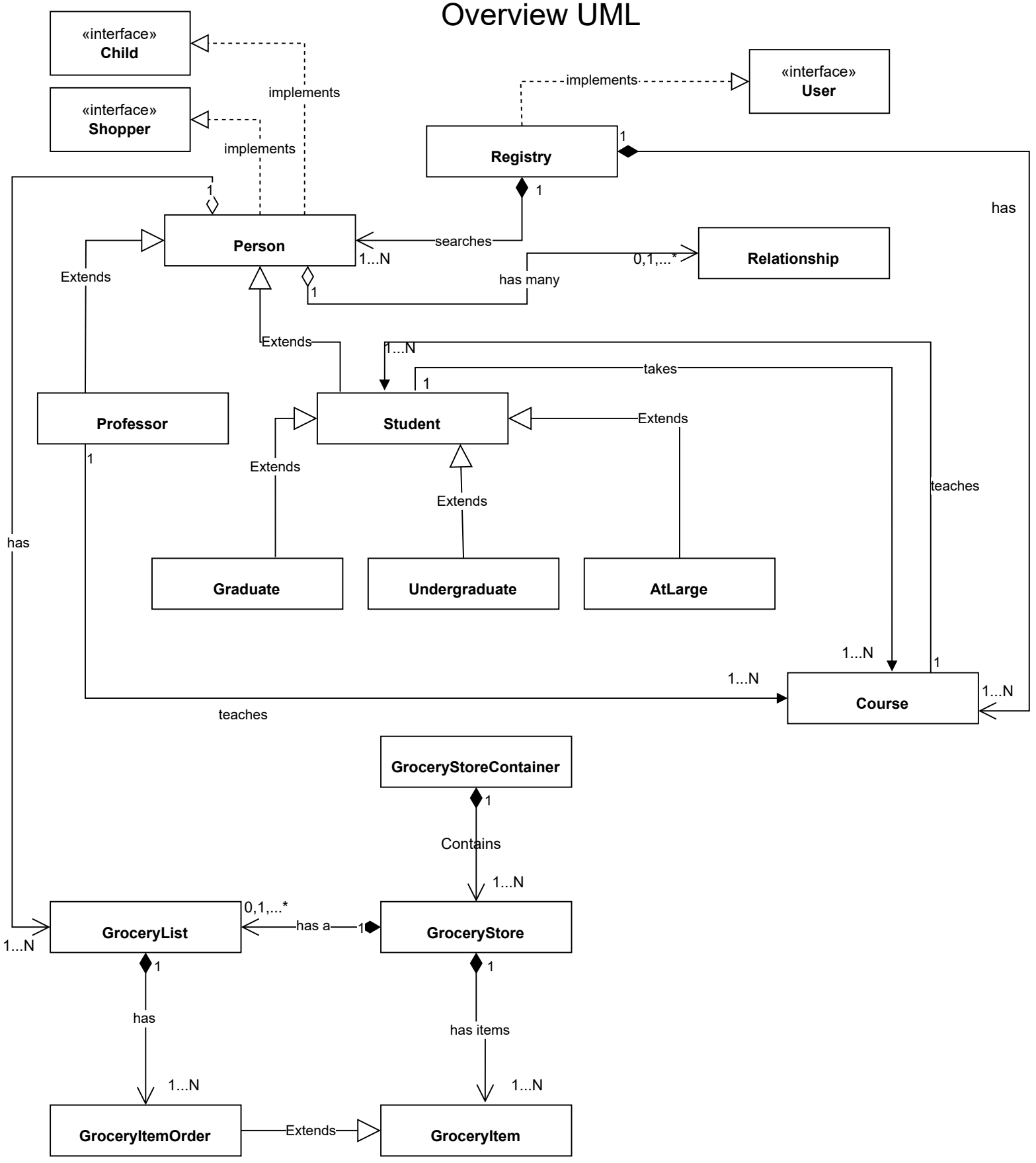
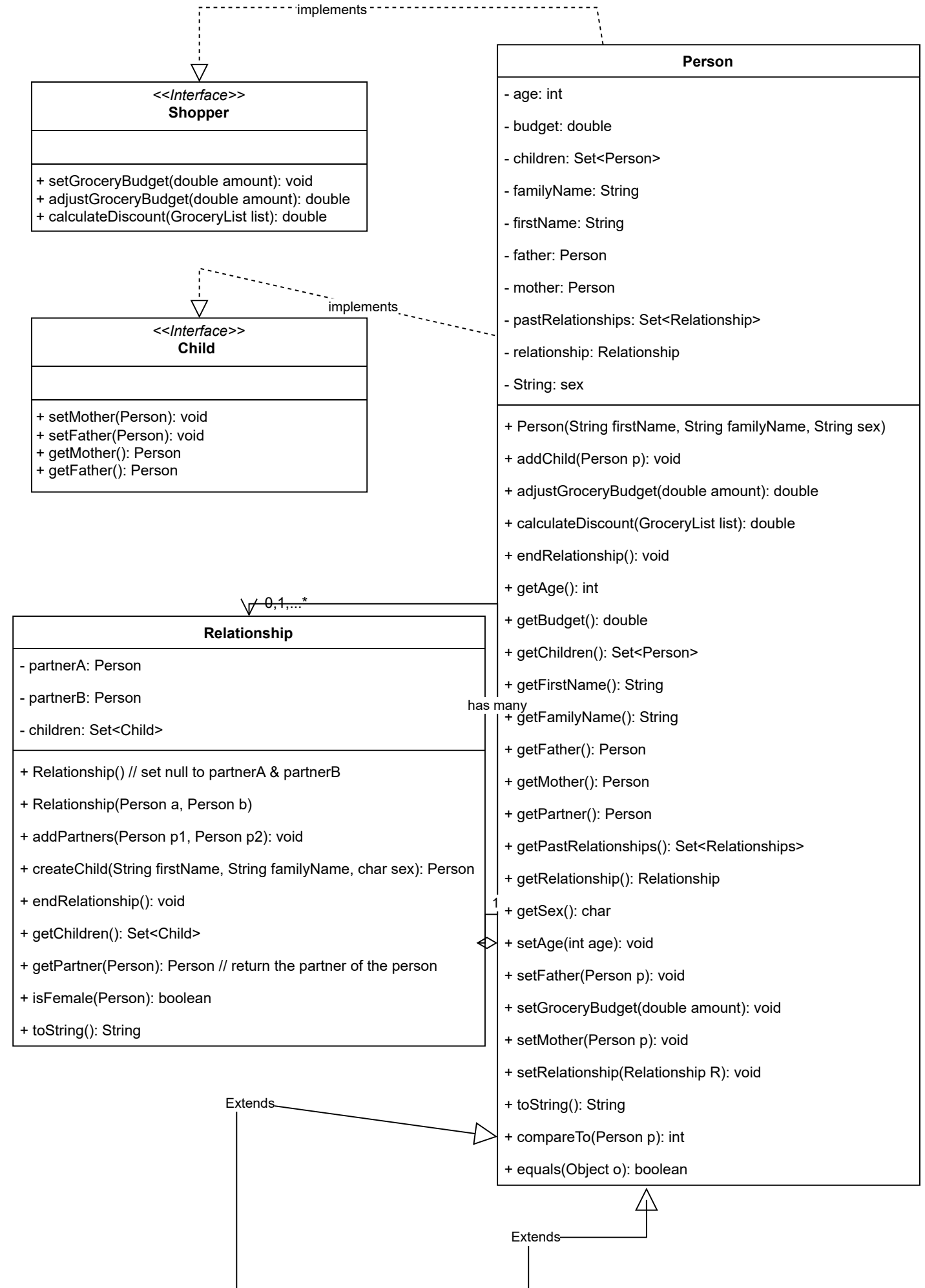
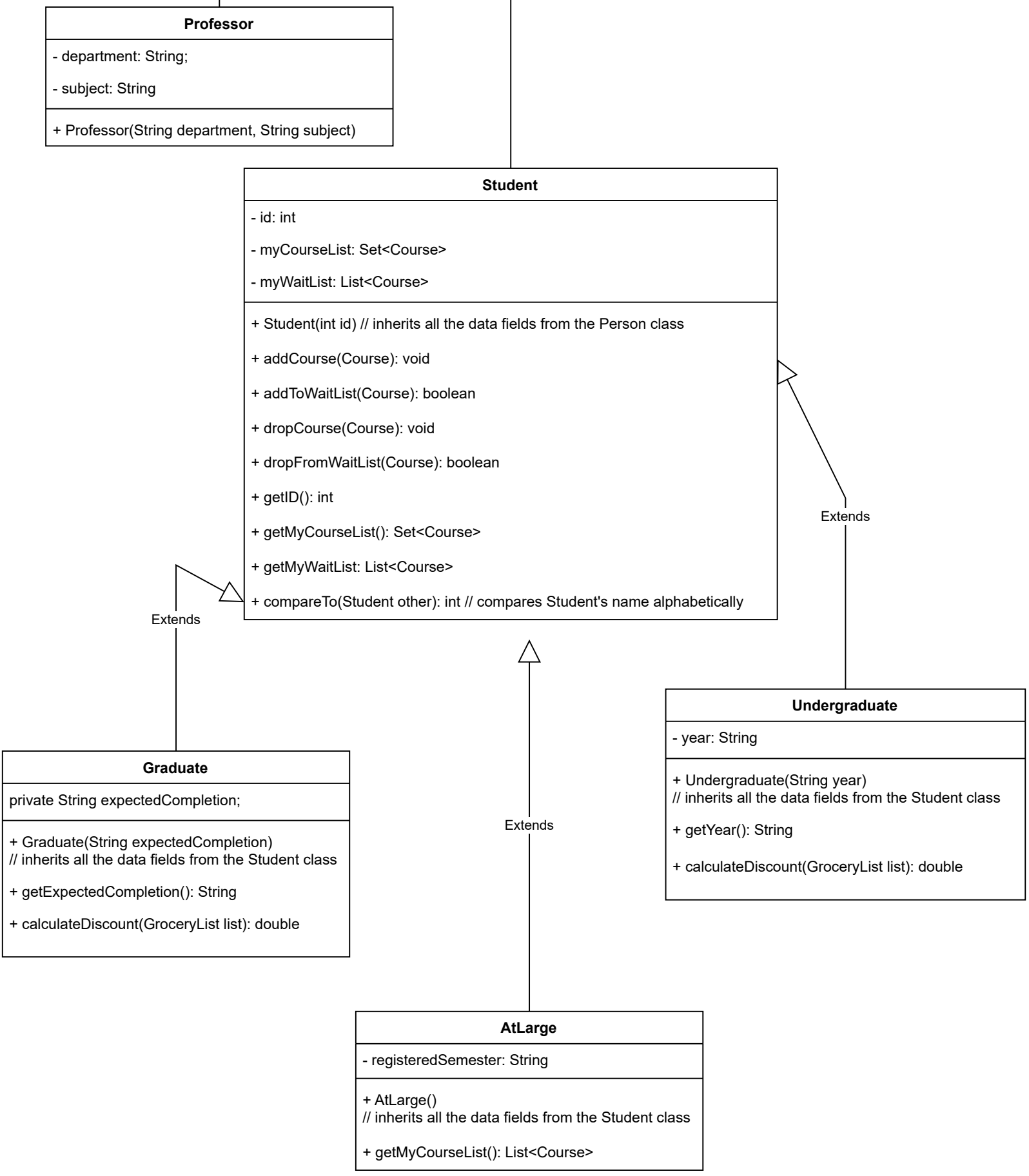
Overview UML

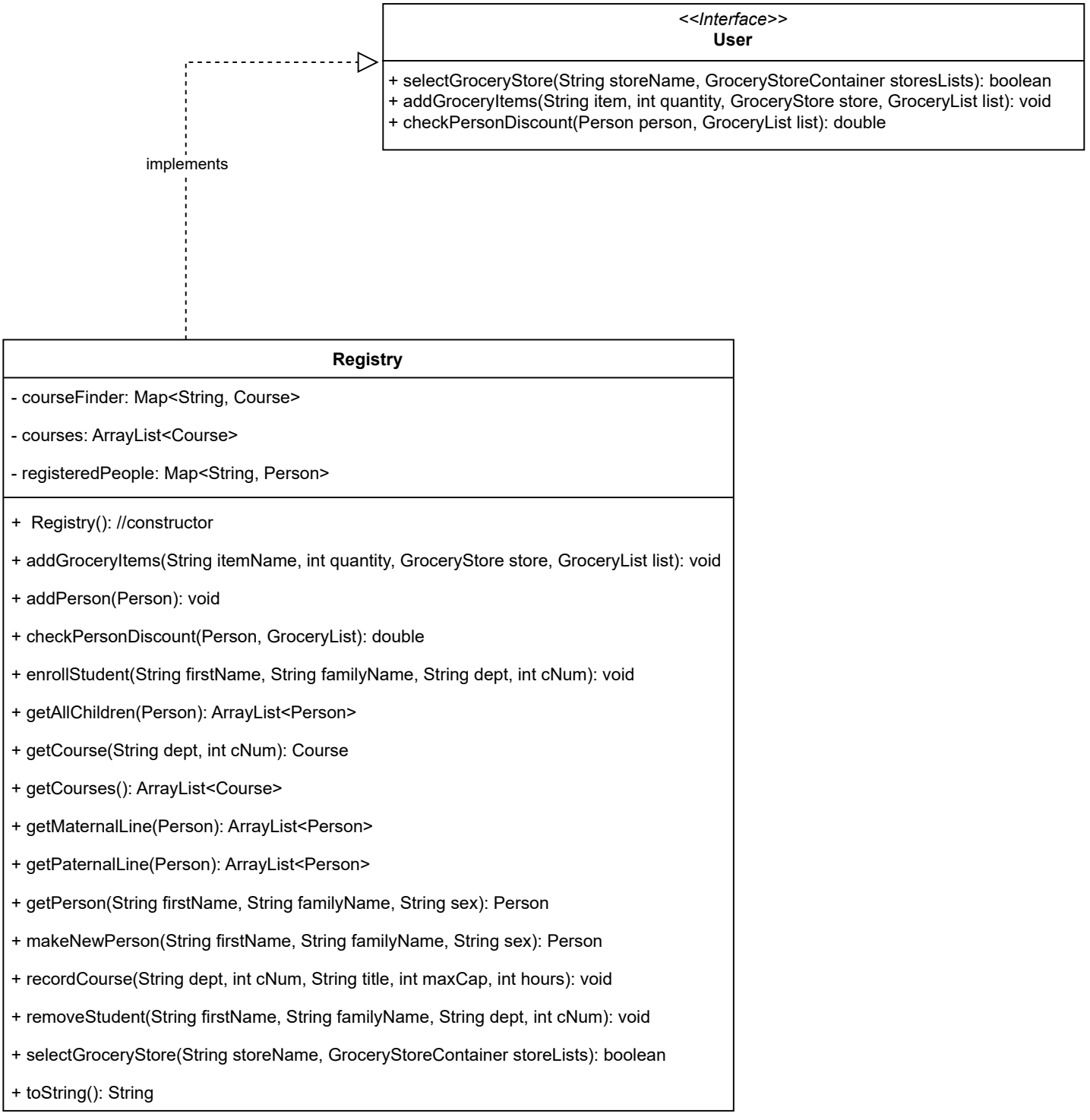


Person-Professor-Students-Relationship

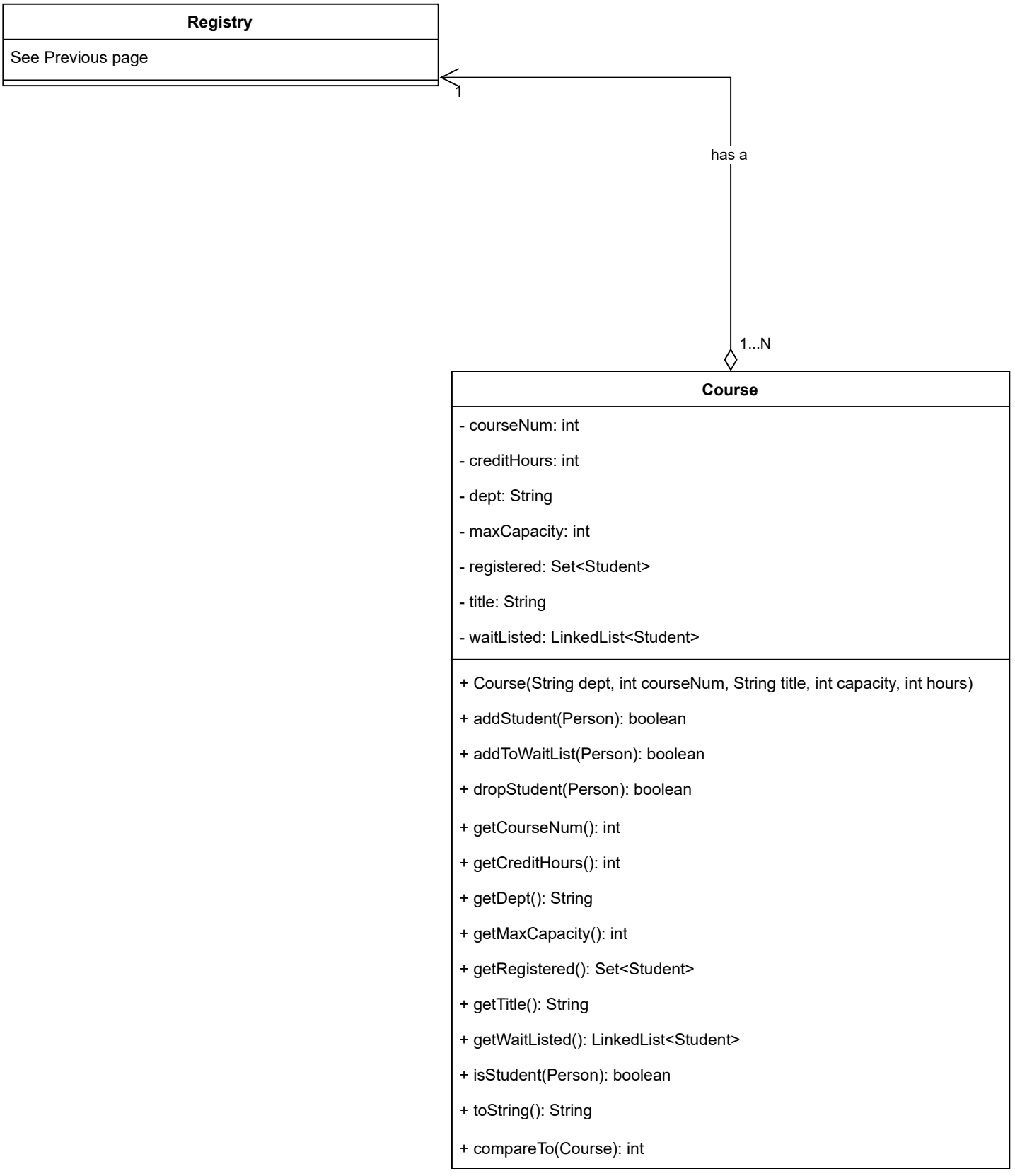



Professor
- department: String;
- subject: String
+ Professor(String department, String subject)
Student
- id: int
- myCourseList: Set<Course>
- myWaitList: List<Course>
+ Student(int id) // inherits all the data fields from the Person class
+ addCourse(Course): void
+ addToWaitList(Course): boolean
+ dropCourse(Course): void
+ dropFromWaitList(Course): boolean
+ getID(): int
+ getMyCourseList(): Set<Course>
+ getMyWaitList: List<Course>
+ compareTo(Student other): int // compares Student's name alphabetically
Extends
Extends
Extends
Graduate
private String expectedCompletion;
+ Graduate(String expectedCompletion)
// inherits all the data fields from the Student class
+ getExpectedCompletion(): String
+ calculateDiscount(GroceryList list): double
Undergraduate
- year: String
+ Undergraduate(String year)
// inherits all the data fields from the Student class
+ getYear(): String
+ calculateDiscount(GroceryList list): double
AtLarge
- registeredSemester: String
+ AtLarge()
// inherits all the data fields from the Student class
+ getMyCourseList(): List<Course>

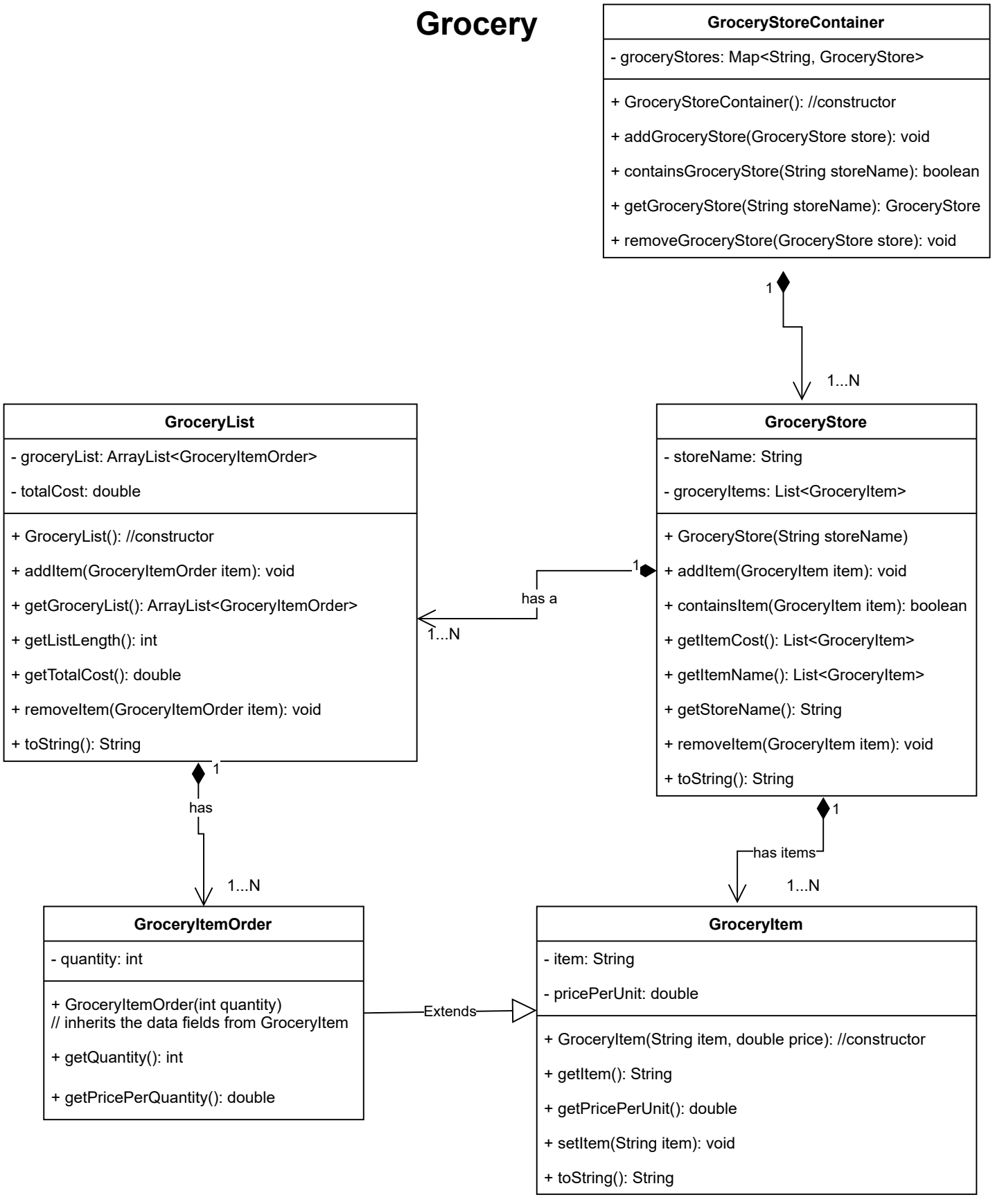
Registry


<<Interface>>
User
+ selectGroceryStore(String storeName, GroceryStoreContainer storesLists): boolean
+ addGroceryItems(String item, int quantity, GroceryStore store, GroceryList list): void
+ checkPersonDiscount(Person person, GroceryList list): double
implements
Registry
- courseFinder: Map<String, Course>
- courses: ArrayList<Course>
- registeredPeople: Map<String, Person>
+ Registry(): //constructor
+ addGroceryItems(String itemName, int quantity, GroceryStore store, GroceryList list): void
+ addPerson(Person): void
+ checkPersonDiscount(Person, GroceryList): double
+ enrollStudent(String firstName, String familyName, String dept, int cNum): void
+ getAllChildren(Person): ArrayList<Person>
+ getCourse(String dept, int cNum): Course
+ getCourses(): ArrayList<Course>
+ getMaternalLine(Person): ArrayList<Person>
+ getPaternalLine(Person): ArrayList<Person>
+ getPerson(String firstName, String familyName, String sex): Person
+ makeNewPerson(String firstName, String familyName, String sex): Person
+ recordCourse(String dept, int cNum, String title, int maxCap, int hours): void
+ removeStudent(String firstName, String familyName, String dept, int cNum): void
+ selectGroceryStore(String storeName, GroceryStoreContainer storeLists): boolean
+ toString(): String

Course


Registry
See Previous page
1
has a
1...N
Course
- courseNum: int
- creditHours: int
- dept: String
- maxCapacity: int
- registered: Set<Student>
- title: String
- waitListed: LinkedList<Student>
+ Course(String dept, int courseNum, String title, int capacity, int hours)
+ addStudent(Person): boolean
+ addToWaitList(Person): boolean
+ dropStudent(Person): boolean
+ getCourseNum(): int
+ getCreditHours(): int
+ getDept(): String
+ getMaxCapacity(): int
+ getRegistered(): Set<Student>
+ getTitle(): String
+ getWaitListed(): LinkedList<Student>
+ isStudent(Person): boolean
+ toString(): String
+ compareTo(Course): int

Grocery


GroceryStoreContainer
- groceryStores: Map<String, GroceryStore>
+ GroceryStoreContainer(): //constructor
+ addGroceryStore(GroceryStore store): void
+ containsGroceryStore(String storeName): boolean
+ getGroceryStore(String storeName): GroceryStore
+ removeGroceryStore(GroceryStore store): void
1
1...N
GroceryList
- groceryList: ArrayList<GroceryItemOrder>
- totalCost: double
+ GroceryList(): //constructor
+ addItem(GroceryItemOrder item): void
+ getGroceryList(): ArrayList<GroceryItemOrder>
+ getListLength(): int
+ getTotalCost(): double
+ removeItem(GroceryItemOrder item): void
+ toString(): String
GroceryStore
- storeName: String
- groceryItems: List<GroceryItem>
+ GroceryStore(String storeName)
+ addItem(GroceryItem item): void
+ containsItem(GroceryItem item): boolean
+ getItemCost(): List<GroceryItem>
+ getItemName(): List<GroceryItem>
+ getStoreName(): String
+ removeItem(GroceryItem item): void
+ toString(): String
1
has a
1...N
1
has
1...N
1
has items
1...N
GroceryItemOrder
- quantity: int
+ GroceryItemOrder(int quantity)
// inherits the data fields from GroceryItem
+ getQuantity(): int
+ getPricePerQuantity(): double
Extends
GroceryItem
- item: String
- pricePerUnit: double
+ GroceryItem(String item, double price): //constructor
+ getItem(): String
+ getPricePerUnit(): double
+ setItem(String item): void
+ toString(): String

Final UML

Person

- age: int
- budget: double
- children: Set<Person>
- familyName: String
- firstName: String
- father: Person
- mother: Person
- pastRelationships: Set<Relationship>
- relationship: Relationship
- String: sex

+ Person(String firstName, String familyName, String sex)
+ addChild(Person p): void
+ adjustGroceryBudget(double amount): double
+ calculateDiscount(GroceryList list): double
+ endRelationship(): void
+ getAge(): int
+ getBudget(): double
+ getChildren(): Set<Person>
+ getFirstName(): String
+ getFamilyName(): String
+ getFather(): Person
+ getMother(): Person
+ getPartner(): Person
+ getPastRelationships(): Set<Relationships>
+ getRelationship(): Relationship
+ getSex(): char
+ setAge(int age): void
+ setFather(Person p): void
+ setGroceryBudget(double amount): void
+ setMother(Person p): void
+ setRelationship(Relationship R): void
+ toString(): String
+ compareTo(Person p): int
+ equals(Object o): boolean

<<Interface>>
Shopper

+ setGroceryBudget(double amount): void
+ adjustGroceryBudget(double amount): double
+ calculateDiscount(GroceryList list): double

<<Interface>>
Child

+ setMother(Person): void
+ setFather(Person): void
+ getMother(): Person
+ getFather(): Person

<<Interface>>
User

+ selectGroceryStore(String storeName, GroceryStoreContainer storesLists): boolean
+ addGroceryItems(String item, int quantity, GroceryStore store, GroceryList list): void
+ checkPersonDiscount(Person person, GroceryList list): double

Registry

- courseFinder: Map<String, Course>
- courses: ArrayList<Course>
- registeredPeople: Map<String, Person>

+ Registry(): //constructor
+ addGroceryItems(String itemName, int quantity, GroceryStore store, GroceryList list): void
+ addPerson(Person): void
+ checkPersonDiscount(Person, GroceryList): double
+ enrollStudent(String firstName, String familyName, String dept, int cNum): void
+ getAllChildren(Person): ArrayList<Person>
+ getCourse(String dept, int cNum): Course
+ getCourses(): ArrayList<Course>
+ getMaternalLine(Person): ArrayList<Person>
+ getPaternalLine(Person): ArrayList<Person>
+ getPerson(String firstName, String familyName, String sex): Person
+ makeNewPerson(String firstName, String familyName, String sex): Person
+ recordCourse(String dept, int cNum, String title, int maxCap, int hours): void
+ removeStudent(String firstName, String familyName, String dept, int cNum): void
+ selectGroceryStore(String storeName, GroceryStoreContainer storeLists): boolean

+ selectGroceryStore(String storeName, GroceryStoreContainer storeLists): boolean

+ toString(): String

Professor
- department: String; - subject: String
+ Professor(String department, String subject)

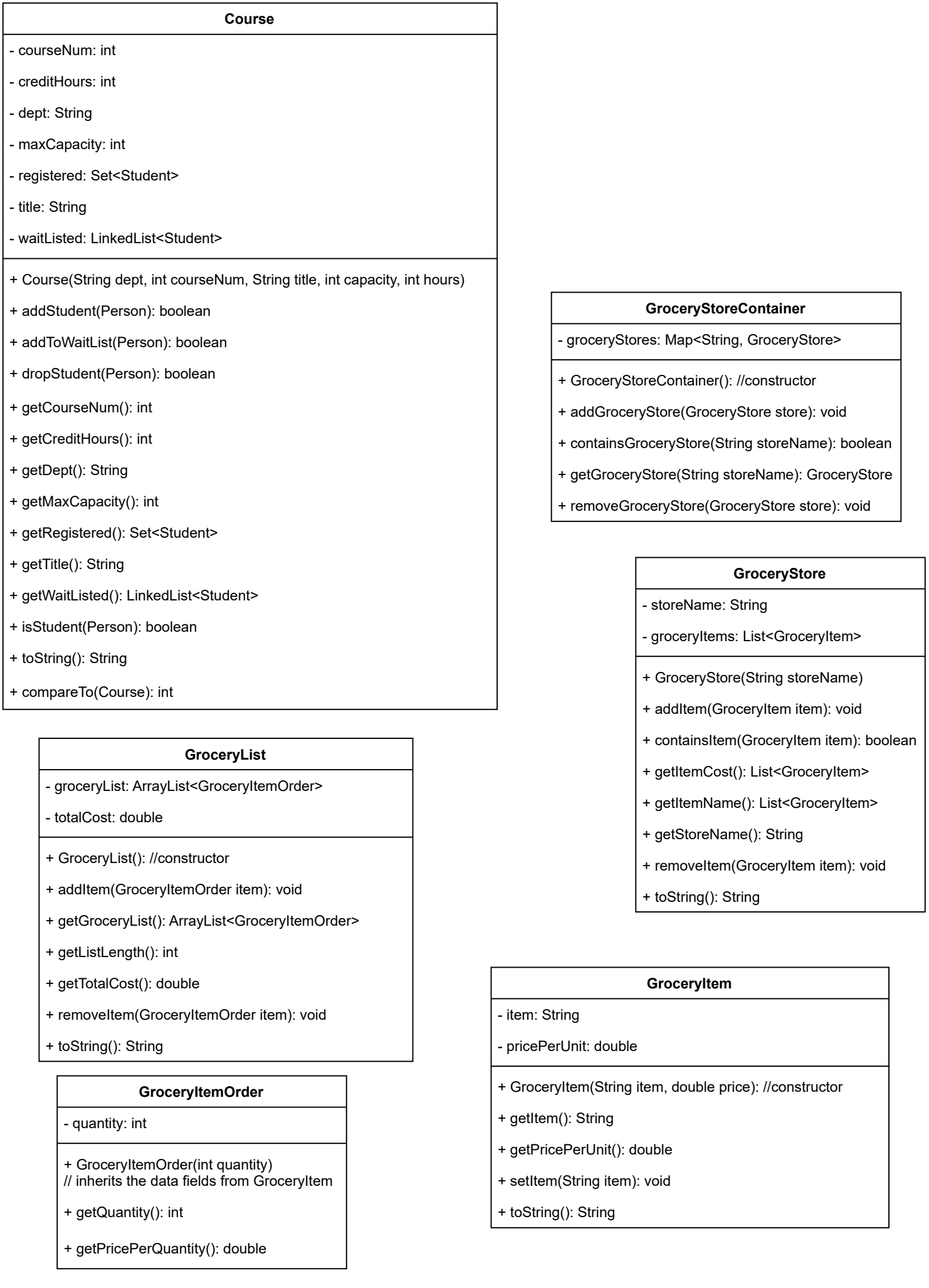
Student
- id: int - myCourseList: Set<Course> - myWaitList: List<Course>
+ Student(int id) // inherits all the data fields from the Person class + addCourse(Course): void + addToWaitList(Course): boolean + dropCourse(Course): void + dropFromWaitList(Course): boolean + getID(): int + getMyCourseList(): Set<Course> + getMyWaitList: List<Course> + compareTo(Student other): int // compares Student's name alphabetically

Graduate
private String expectedCompletion;
+ Graduate(String expectedCompletion) // inherits all the data fields from the Student class + getExpectedCompletion(): String + calculateDiscount(GroceryList list): double

AtLarge
- registeredSemester: String
+ AtLarge() // inherits all the data fields from the Student class + getMyCourseList(): List<Course>

Undergraduate
- year: String
+ Undergraduate(String year) // inherits all the data fields from the Student class + getYear(): String + calculateDiscount(GroceryList list): double

Relationship
- partnerA: Person - partnerB: Person - children: Set<Child>
+ Relationship() // set null to partnerA & partnerB + Relationship(Person a, Person b) + addPartners(Person p1, Person p2): void + createChild(String firstName, String familyName, char sex): Person + endRelationship(): void + getChildren(): Set<Child> + getPartner(Person): Person // return the partner of the person + isFemale(Person): boolean + toString(): String


Course
- courseNum: int
- creditHours: int
- dept: String
- maxCapacity: int
- registered: Set<Student>
- title: String
- waitListed: LinkedList<Student>
+ Course(String dept, int courseNum, String title, int capacity, int hours)
+ addStudent(Person): boolean
+ addToWaitList(Person): boolean
+ dropStudent(Person): boolean
+ getCourseNum(): int
+ getCreditHours(): int
+ getDept(): String
+ getMaxCapacity(): int
+ getRegistered(): Set<Student>
+ getTitle(): String
+ getWaitListed(): LinkedList<Student>
+ isStudent(Person): boolean
+ toString(): String
+ compareTo(Course): int
GroceryStoreContainer
- groceryStores: Map<String, GroceryStore>
+ GroceryStoreContainer(): //constructor
+ addGroceryStore(GroceryStore store): void
+ containsGroceryStore(String storeName): boolean
+ getGroceryStore(String storeName): GroceryStore
+ removeGroceryStore(GroceryStore store): void
GroceryStore
- storeName: String
- groceryItems: List<GroceryItem>
+ GroceryStore(String storeName)
+ addItem(GroceryItem item): void
+ containsItem(GroceryItem item): boolean
+ getItemCost(): List<GroceryItem>
+ getItemName(): List<GroceryItem>
+ getStoreName(): String
+ removeItem(GroceryItem item): void
+ toString(): String
GroceryList
- groceryList: ArrayList<GroceryItemOrder>
- totalCost: double
+ GroceryList(): //constructor
+ addItem(GroceryItemOrder item): void
+ getGroceryList(): ArrayList<GroceryItemOrder>
+ getListLength(): int
+ getTotalCost(): double
+ removeItem(GroceryItemOrder item): void
+ toString(): String
GroceryItem
- item: String
- pricePerUnit: double
+ GroceryItem(String item, double price): //constructor
+ getItem(): String
+ getPricePerUnit(): double
+ setItem(String item): void
+ toString(): String
GroceryItemOrder
- quantity: int
+ GroceryItemOrder(int quantity)
// inherits the data fields from GroceryItem
+ getQuantity(): int
+ getPricePerQuantity(): double