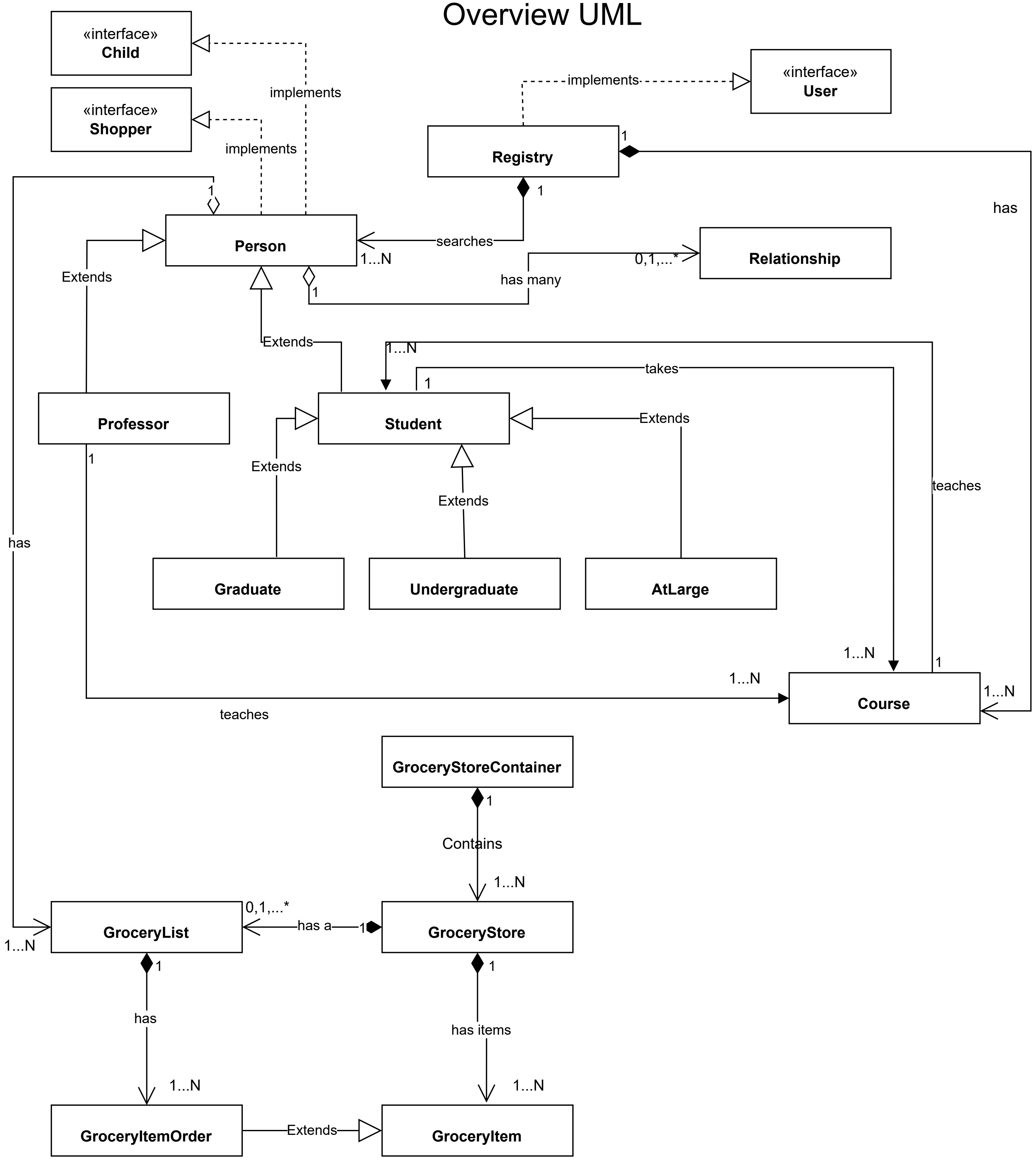
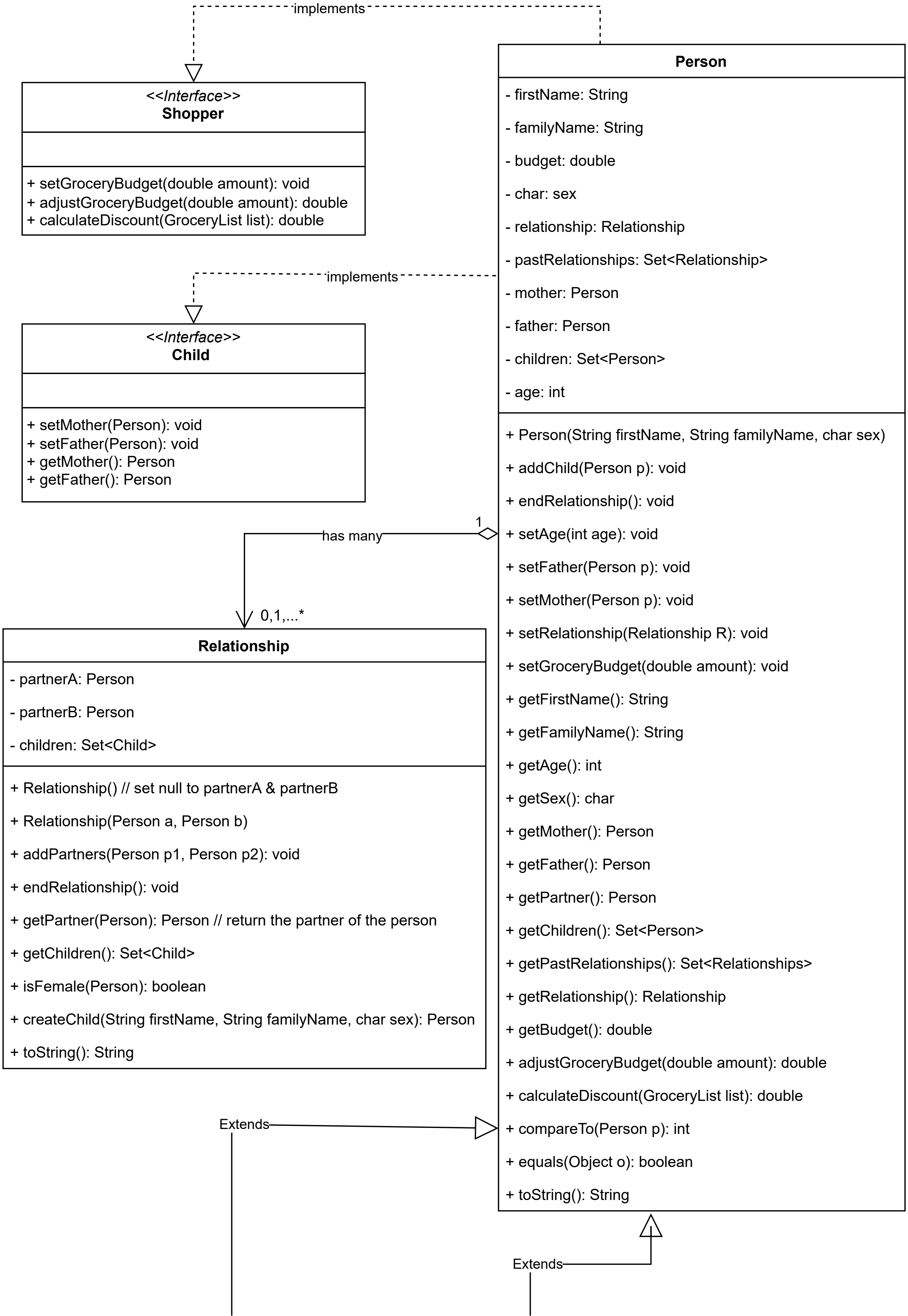
Overview UML



Person-Professor-Students-Relationship



<<*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

Person

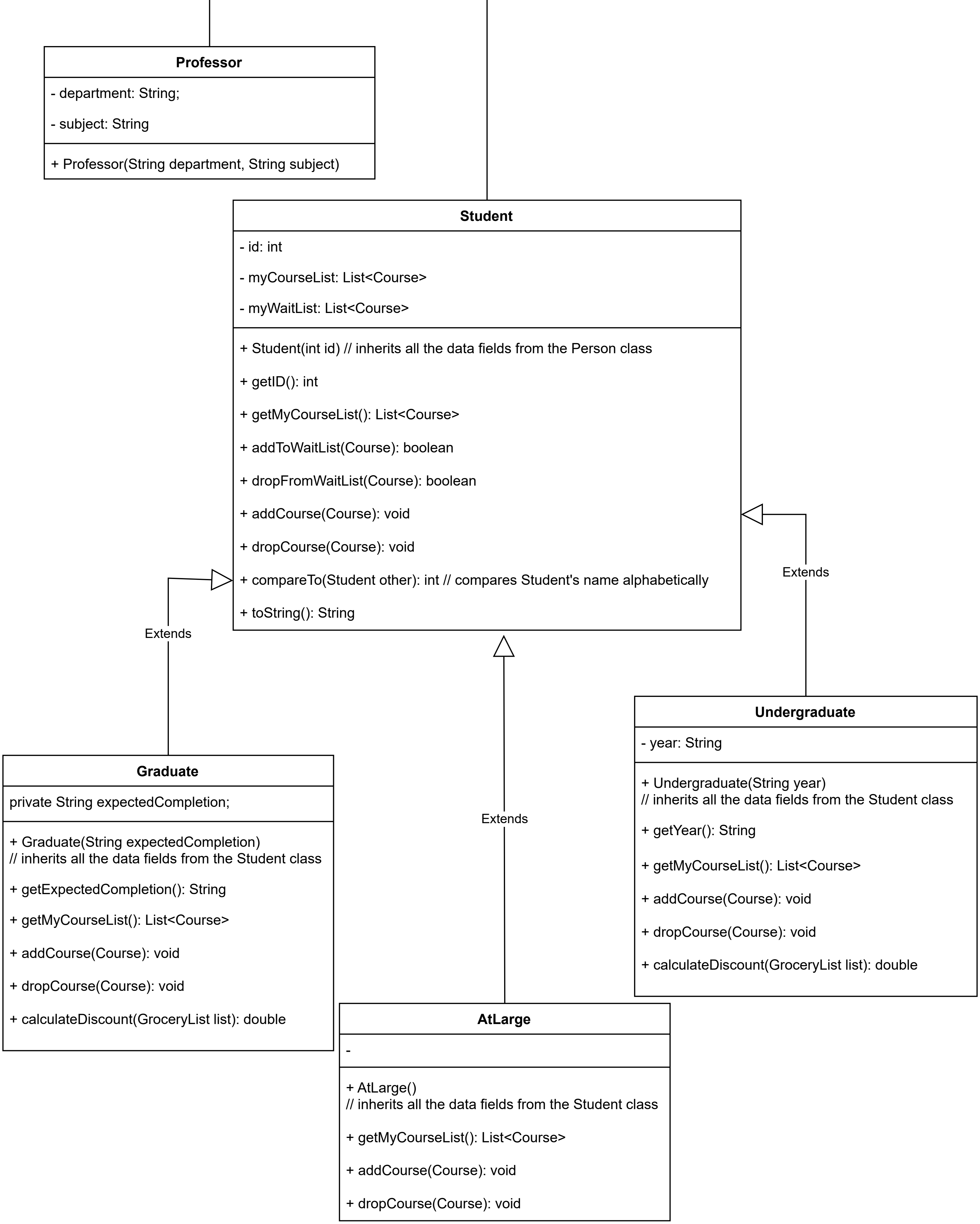
- firstName: String
- familyName: String
- budget: double
- char: sex
- relationship: Relationship
- pastRelationships: Set<Relationship>
- mother: Person
- father: Person
- children: Set<Person>
- age: int

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

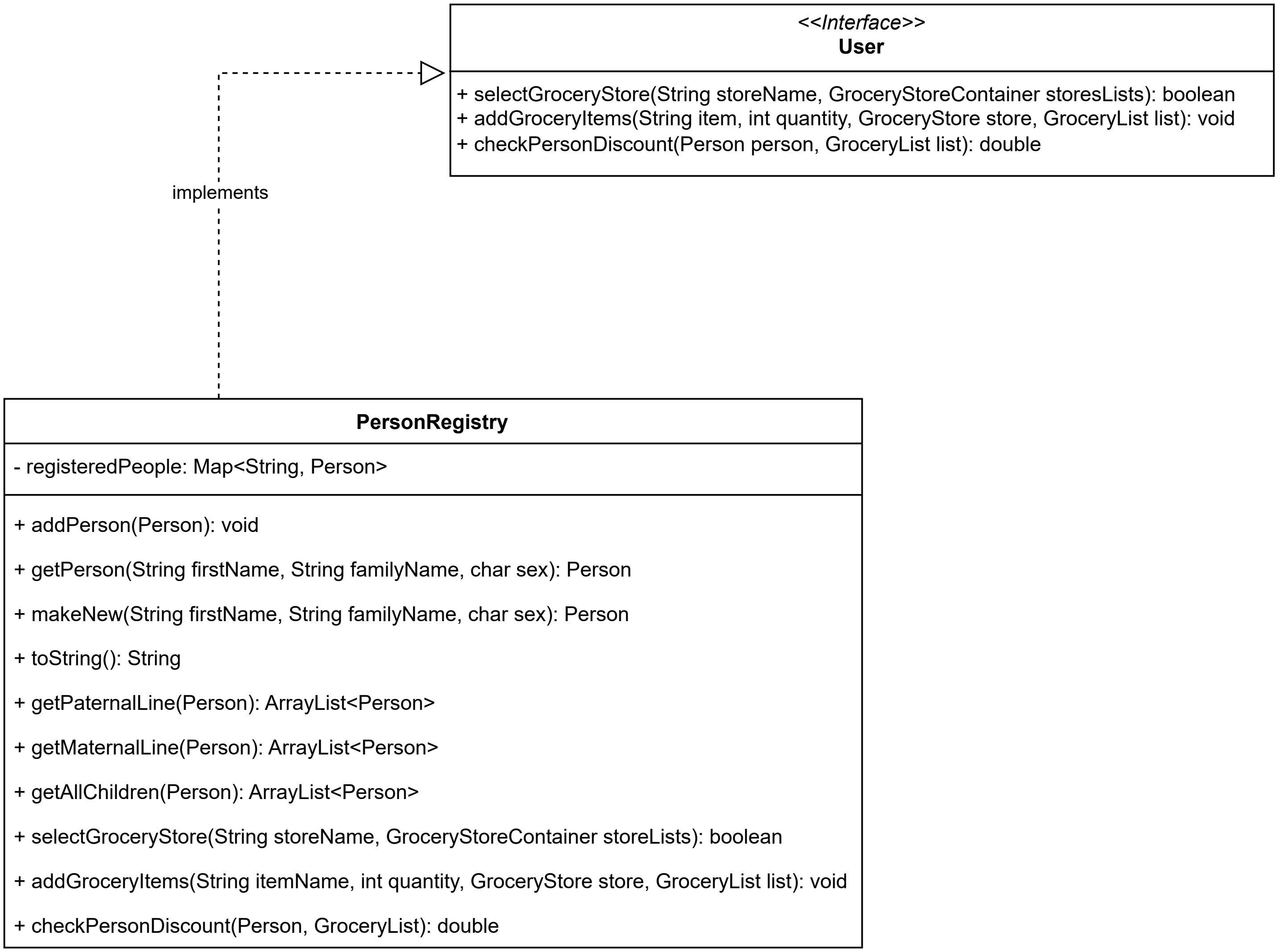
Relationship

- partnerA: Person
- partnerB: Person
- children: Set<Child>

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


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

Person Registry



<<Interface>>
User

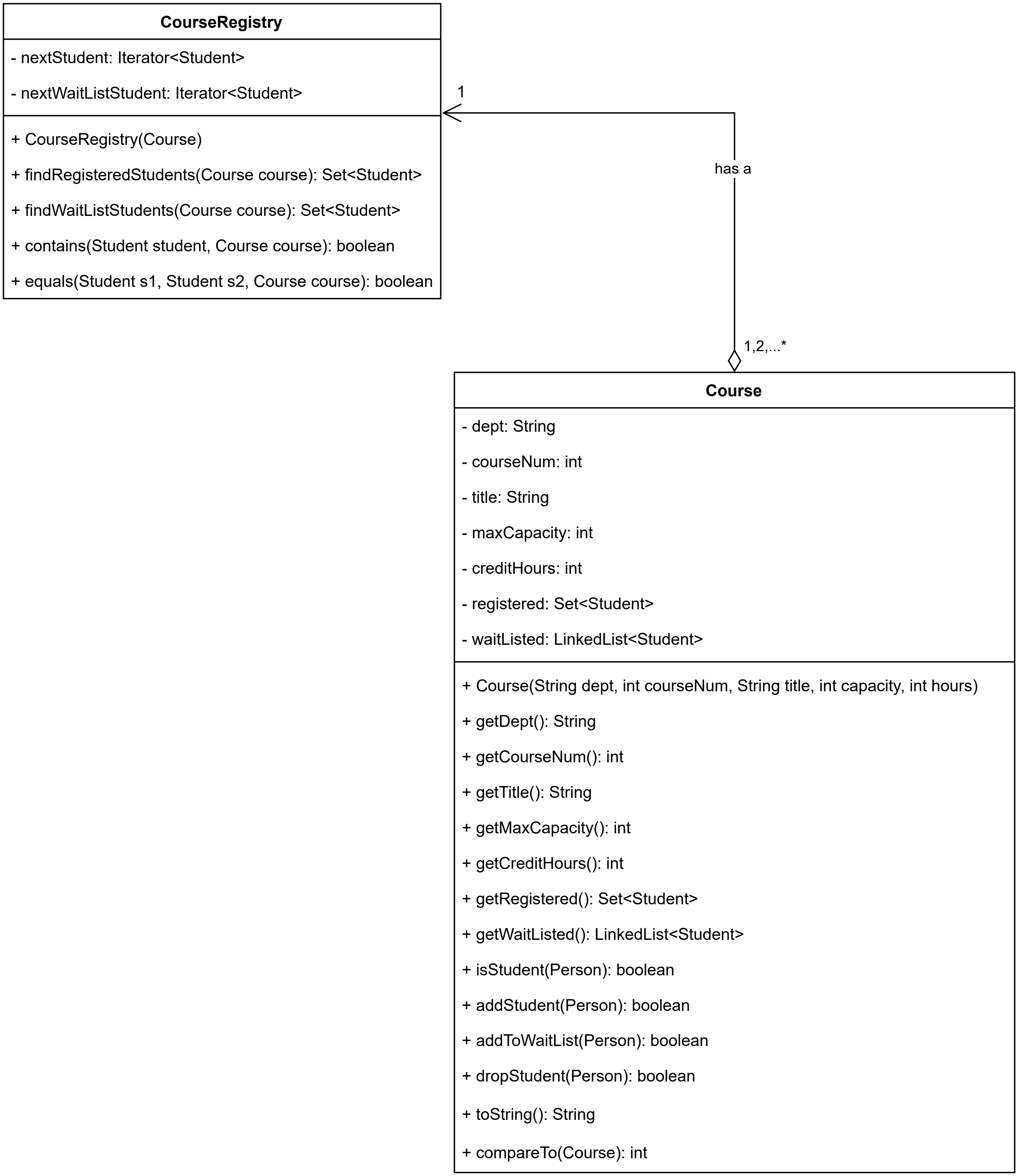
+ selectGroceryStore(String storeName, GroceryStoreContainer storesLists): boolean
+ addGroceryItems(String item, int quantity, GroceryStore store, GroceryList list): void
+ checkPersonDiscount(Person person, GroceryList list): double

PersonRegistry

- registeredPeople: Map<String, Person>

+ addPerson(Person): void
+ getPerson(String firstName, String familyName, char sex): Person
+ makeNew(String firstName, String familyName, char sex): Person
+ toString(): String
+ getPaternalLine(Person): ArrayList<Person>
+ getMaternalLine(Person): ArrayList<Person>
+ getAllChildren(Person): ArrayList<Person>
+ selectGroceryStore(String storeName, GroceryStoreContainer storeLists): boolean
+ addGroceryItems(String itemName, int quantity, GroceryStore store, GroceryList list): void
+ checkPersonDiscount(Person, GroceryList): double

Course



CourseRegistry

- nextStudent: Iterator<Student>

- nextWaitListStudent: Iterator<Student>

+ CourseRegistry(Course)

+ findRegisteredStudents(Course course): Set<Student>

+ findWaitListStudents(Course course): Set<Student>

+ contains(Student student, Course course): boolean

+ equals(Student s1, Student s2, Course course): boolean

Course

- dept: String

- courseNum: int

- title: String

- maxCapacity: int

- creditHours: int

- registered: Set<Student>

- waitListed: LinkedList<Student>

+ Course(String dept, int courseNum, String title, int capacity, int hours)

+ getDept(): String

+ getCourseNum(): int

+ getTitle(): String

+ getMaxCapacity(): int

+ getCreditHours(): int

+ getRegistered(): Set<Student>

+ getWaitListed(): LinkedList<Student>

+ isStudent(Person): boolean

+ addStudent(Person): boolean

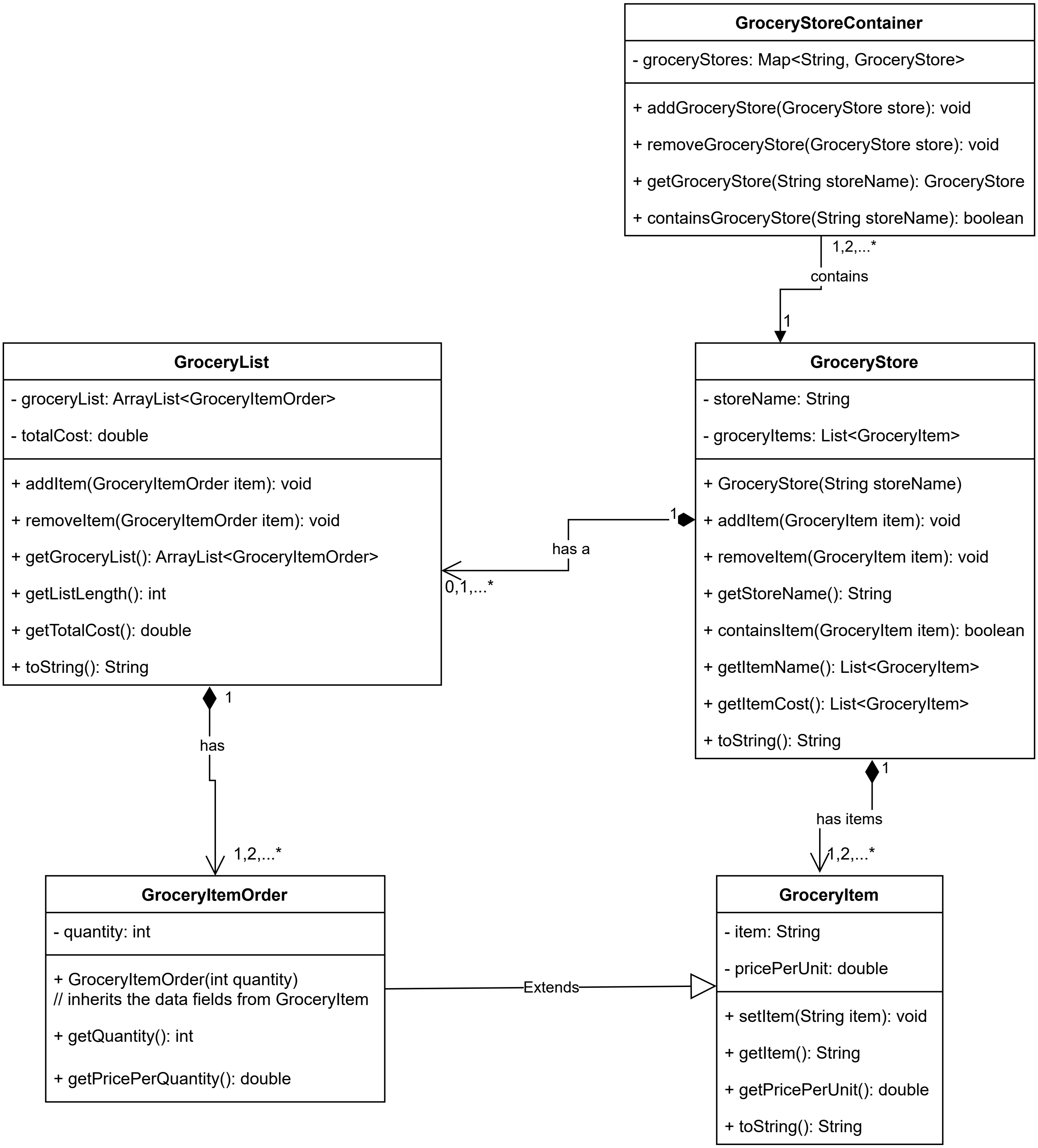
+ addToWaitList(Person): boolean

+ dropStudent(Person): boolean

+ toString(): String

+ compareTo(Course): int

Grocery



GroceryStoreContainer

- groceryStores: Map<String, GroceryStore>

+ addGroceryStore(GroceryStore store): void

+ removeGroceryStore(GroceryStore store): void

+ getGroceryStore(String storeName): GroceryStore

+ containsGroceryStore(String storeName): boolean

GroceryList

- groceryList: ArrayList<GroceryItemOrder>

- totalCost: double

+ addItem(GroceryItemOrder item): void

+ removeItem(GroceryItemOrder item): void

+ getGroceryList(): ArrayList<GroceryItemOrder>

+ getListLength(): int

+ getTotalCost(): double

+ toString(): String

GroceryStore

- storeName: String

- groceryItems: List<GroceryItem>

+ GroceryStore(String storeName)

+ addItem(GroceryItem item): void

+ removeItem(GroceryItem item): void

+ getStoreName(): String

+ containsItem(GroceryItem item): boolean

+ getItemName(): List<GroceryItem>

+ getItemCost(): List<GroceryItem>

+ toString(): String

GroceryItemOrder

- quantity: int

+ GroceryItemOrder(int quantity)
// inherits the data fields from GroceryItem

+ getQuantity(): int

+ getPricePerQuantity(): double

GroceryItem

- item: String

- pricePerUnit: double

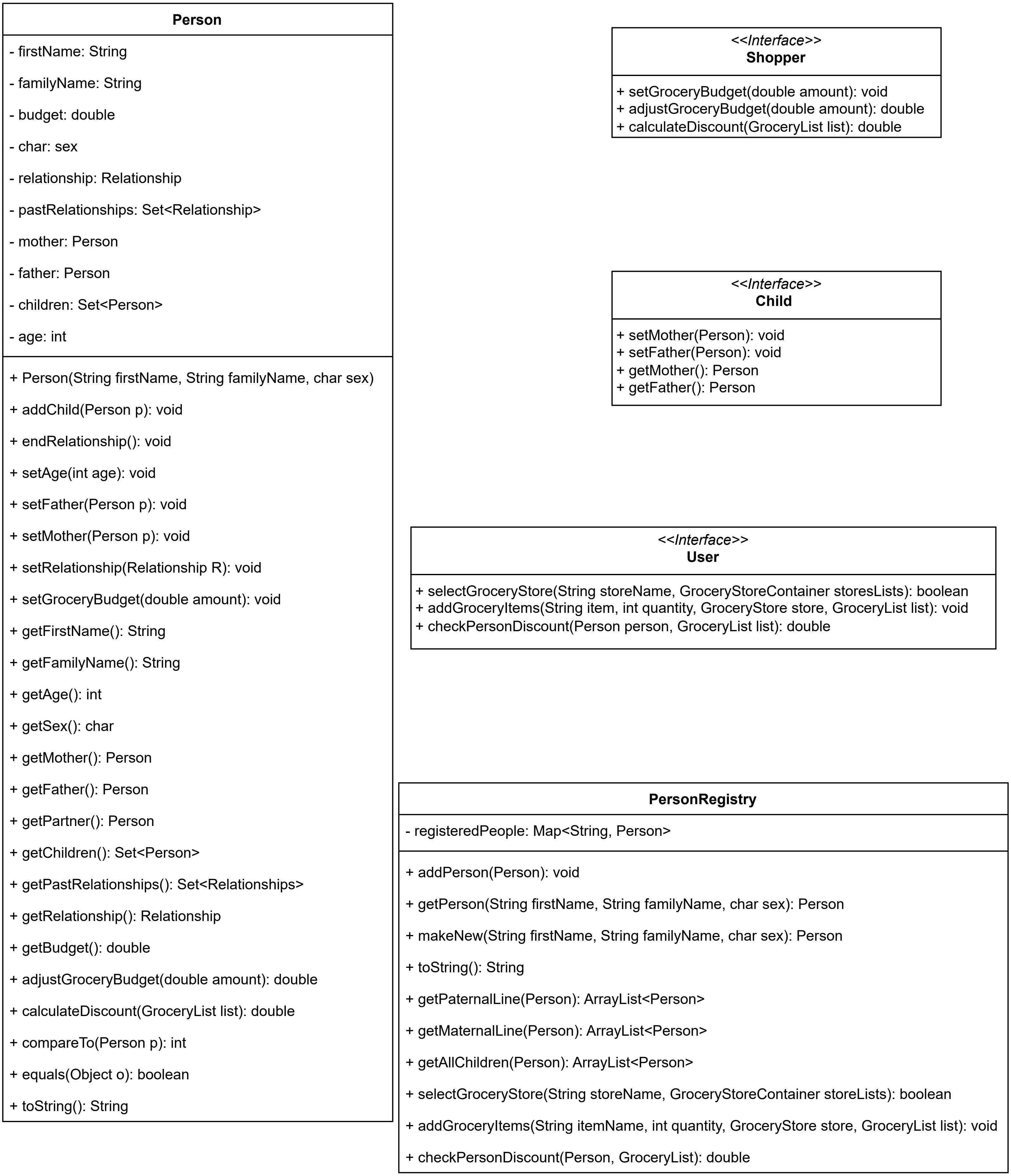
+ setItem(String item): void

+ getItem(): String

+ getPricePerUnit(): double

+ toString(): String

Final UML



Person

- firstName: String

- familyName: String

- budget: double

- char: sex

- relationship: Relationship

- pastRelationships: Set<Relationship>

- mother: Person

- father: Person

- children: Set<Person>

- age: int

+ Person(String firstName, String familyName, char sex)

+ addChild(Person p): void

+ endRelationship(): void

+ setAge(int age): void

+ setFather(Person p): void

+ setMother(Person p): void

+ setRelationship(Relationship R): void

+ setGroceryBudget(double amount): void

+ getFirstName(): String

+ getFamilyName(): String

+ getAge(): int

+ getSex(): char

+ getMother(): Person

+ getFather(): Person

+ getPartner(): Person

+ getChildren(): Set<Person>

+ getPastRelationships(): Set<Relationships>

+ getRelationship(): Relationship

+ getBudget(): double

+ adjustGroceryBudget(double amount): double

+ calculateDiscount(GroceryList list): double

+ compareTo(Person p): int

+ equals(Object o): boolean

+ toString(): String

<<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

PersonRegistry

- registeredPeople: Map<String, Person>

+ addPerson(Person): void

+ getPerson(String firstName, String familyName, char sex): Person

+ makeNew(String firstName, String familyName, char sex): Person

+ toString(): String

+ getPaternalLine(Person): ArrayList<Person>

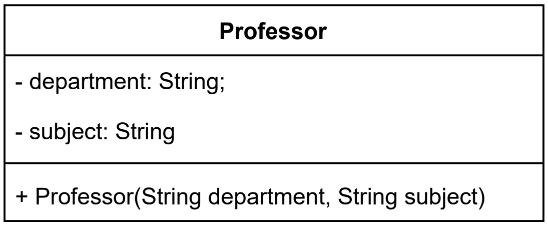
+ getMaternalLine(Person): ArrayList<Person>

+ getAllChildren(Person): ArrayList<Person>

+ selectGroceryStore(String storeName, GroceryStoreContainer storeLists): boolean

+ addGroceryItems(String itemName, int quantity, GroceryStore store, GroceryList list): void

+ checkPersonDiscount(Person, GroceryList): double

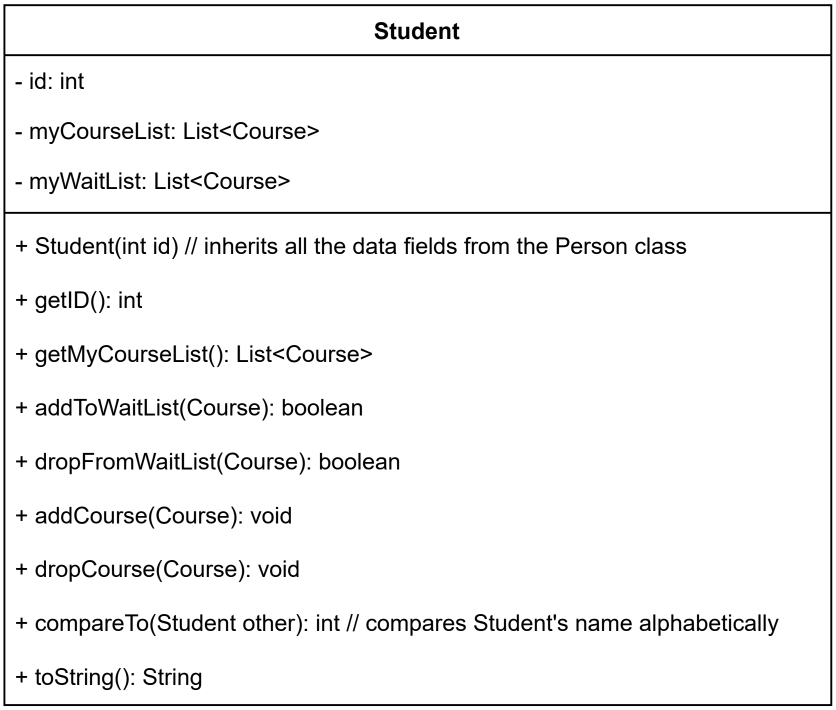


Professor

- department: String;

- subject: String

+ Professor(String department, String subject)



Student

- id: int

- myCourseList: List<Course>

- myWaitList: List<Course>

+ Student(int id) // inherits all the data fields from the Person class

+ getID(): int

+ getMyCourseList(): List<Course>

+ addToWaitList(Course): boolean

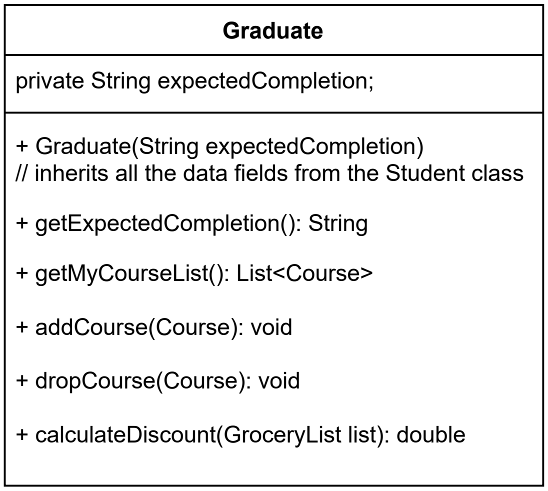
+ dropFromWaitList(Course): boolean

+ addCourse(Course): void

+ dropCourse(Course): void

+ compareTo(Student other): int // compares Student's name alphabetically

+ toString(): String



Graduate

private String expectedCompletion;

+ Graduate(String expectedCompletion)
// inherits all the data fields from the Student class

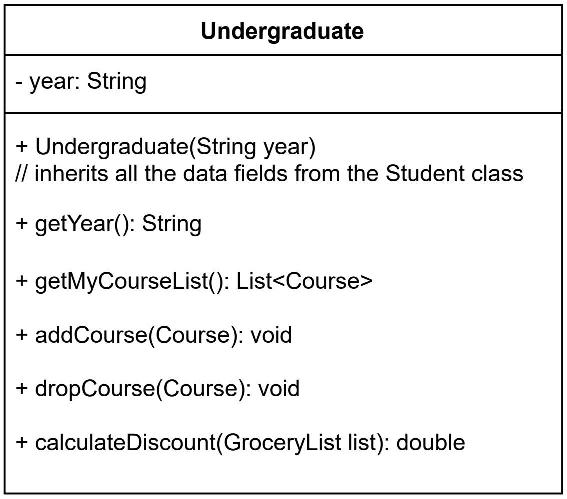
+ getExpectedCompletion(): String

+ getMyCourseList(): List<Course>

+ addCourse(Course): void

+ dropCourse(Course): void

+ calculateDiscount(GroceryList list): double



Undergraduate

- year: String

+ Undergraduate(String year)
// inherits all the data fields from the Student class

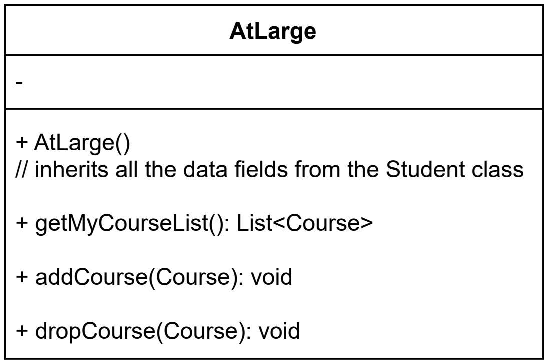
+ getYear(): String

+ getMyCourseList(): List<Course>

+ addCourse(Course): void

+ dropCourse(Course): void

+ calculateDiscount(GroceryList list): double



AtLarge

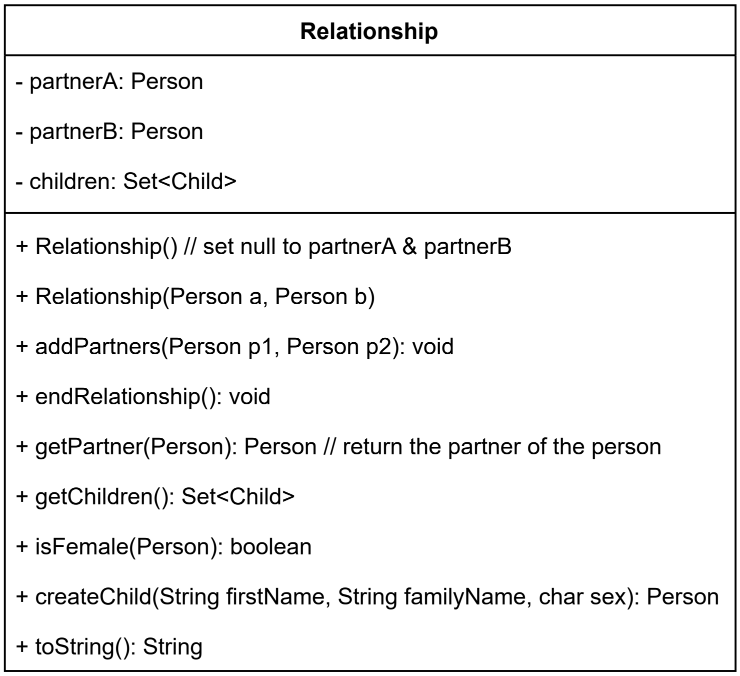
-

+ AtLarge()
// inherits all the data fields from the Student class

+ getMyCourseList(): List<Course>

+ addCourse(Course): void

+ dropCourse(Course): void



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

+ endRelationship(): void

+ getPartner(Person): Person // return the partner of the person

+ getChildren(): Set<Child>

+ isFemale(Person): boolean

+ createChild(String firstName, String familyName, char sex): Person

+ toString(): String

Course

```
- dept: String
- courseNum: int
- title: String
- maxCapacity: int
- creditHours: int
- registered: Set<Student>
- waitListed: LinkedList<Student>
------------------------------------------------------------
+ Course(String dept, int courseNum, String title, int capacity, int hours)
+ getDept(): String
+ getCourseNum(): int
+ getTitle(): String
+ getMaxCapacity(): int
+ getCreditHours(): int
+ getRegistered(): Set<Student>
+ getWaitListed(): LinkedList<Student>
+ isStudent(Person): boolean
+ addStudent(Person): boolean
+ addToWaitList(Person): boolean
+ dropStudent(Person): boolean
+ toString(): String
+ compareTo(Course): int
```

CourseRegistry

```
- nextStudent: Iterator<Student>
- nextWaitListStudent: Iterator<Student>
------------------------------------------------------------
+ CourseRegistry(Course)
+ findRegisteredStudents(Course course): Set<Student>
+ findWaitListStudents(Course course): Set<Student>
+ contains(Student student, Course course): boolean
+ equals(Student s1, Student s2, Course course): boolean
```

GroceryStoreContainer

```
- groceryStores: Map<String, GroceryStore>
------------------------------------------------------------
+ addGroceryStore(GroceryStore store): void
+ removeGroceryStore(GroceryStore store): void
+ getGroceryStore(String storeName): GroceryStore
+ containsGroceryStore(String storeName): boolean
```

GroceryList

```
- groceryList: ArrayList<GroceryItemOrder>
- totalCost: double
------------------------------------------------------------
+ addItem(GroceryItemOrder item): void
+ removeItem(GroceryItemOrder item): void
+ getGroceryList(): ArrayList<GroceryItemOrder>
+ getListLength(): int
+ getTotalCost(): double
+ toString(): String
```

GroceryStore

```
- storeName: String
- groceryItems: List<GroceryItem>
------------------------------------------------------------
+ GroceryStore(String storeName)
+ addItem(GroceryItem item): void
+ removeItem(GroceryItem item): void
+ getStoreName(): String
+ containsItem(GroceryItem item): boolean
+ getItemName(): List<GroceryItem>
+ getItemCost(): List<GroceryItem>
+ toString(): String
```

GroceryItemOrder

```
- quantity: int
------------------------------------------------------------
+ GroceryItemOrder(int quantity)
// inherits the data fields from GroceryItem
+ getQuantity(): int
+ getPricePerQuantity(): double
```

GroceryItem

```
- item: String
- pricePerUnit: double
------------------------------------------------------------
+ setItem(String item): void
+ getItem(): String
+ getPricePerUnit(): double
+ toString(): String
```