10. Inheritence
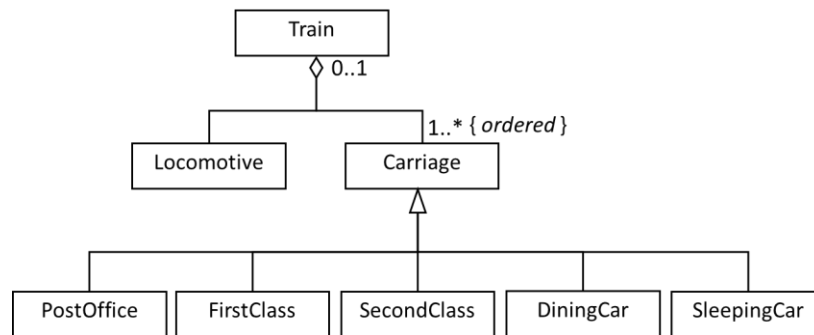
Topics: Inheritence, override of virtual functions, runtime polymorphism.

1. A train consists of a locomotive and at least one passenger carriage. A train has the locomotive at the front and carriages are connected to the locomotive according to a given order. A train might contain various types of carriages. The possible types of carriages are: first-class carriage, second-class carriage, railway post office, dining car, sleeping car.

```
                          ┌──────────┐
                          │  Train   │
                          └──────────┘
                              ◇ 0..1
                    ┌──────────┴──────────┐
                                      1..* { ordered }
              ┌────────────┐    ┌────────────┐
              │ Locomotive │    │  Carriage  │
              └────────────┘    └────────────┘
                                       △
         ┌──────────┬──────────┬──────┴──────┬──────────┐
   ┌───────────┐ ┌──────────┐ ┌───────────┐ ┌─────────┐ ┌────────────┐
   │ PostOffice│ │FirstClass│ │SecondClass│ │DiningCar│ │SleepingCar │
   └───────────┘ └──────────┘ └───────────┘ └─────────┘ └────────────┘
```
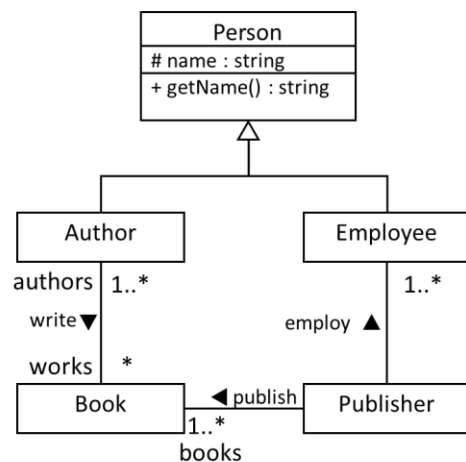
2. Every book has at least one author and it is published exactly by one publisher. A publisher issues at least one book. A publisher employs at least one employee. An employee is employed by exactly one publisher. Employees and authors are persons, and authors are not employees of publishers.
   Tasks: How many books has been written by a given author? Which author has written the most number of books published by a given publisher?

Problem: How many books has been written by a given author?

a : Author, c : int

c := | a.works |

```
              ┌──────────────────────┐
              │        Person        │
              ├──────────────────────┤
              │ # name : string      │
              ├──────────────────────┤
              │ + getName() : string │
              └──────────────────────┘
                         △
             ┌───────────┴───────────┐
      ┌────────────┐          ┌────────────┐
      │   Author   │          │  Employee  │
      └────────────┘          └────────────┘
   authors  1..*                       1..*
   write ▼                    employ ▲
   works   *
      ┌────────────┐ ◀publish ┌────────────┐
      │    Book    │          │  Publisher │
      └────────────┘          └────────────┘
          1..*
          books
```
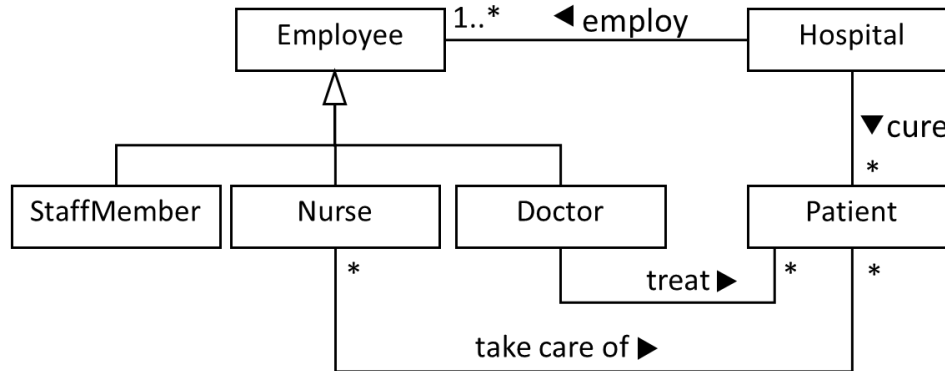
Problem: Which author has written the most number of books published by a given publisher?
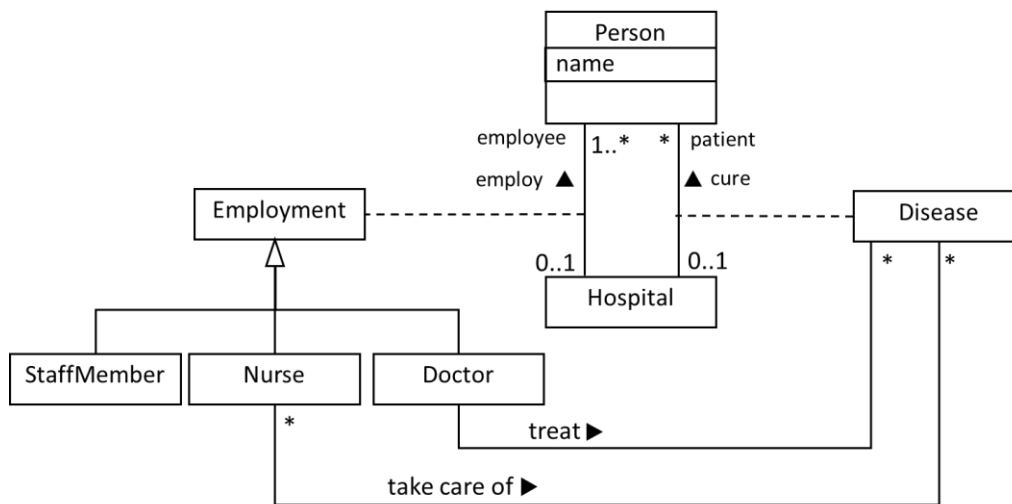
name : string, p : Publisher, b : Bag

```
b.erase()
forall e in p.books loop
   forall a in e.authors loop
      b.putIn(a)
   endloop
endloop
name := b.mostFrequent().getName()
```
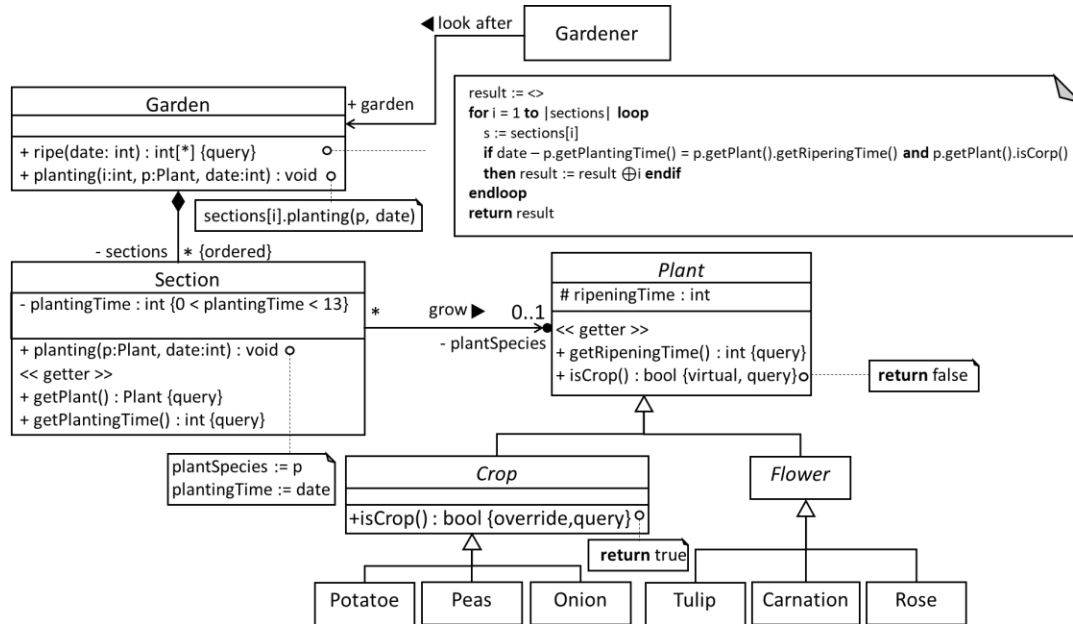
3. In a hospital there is at least one employee – doctor or nurse or hospital staff member. Patients are settled in hospitals, treated by doctors and taken care by nurses. A patient is treated exactly by one doctor, and is taken care by any number of nurses. Every doctor can treat any number of patients, also each nurse can take care of any number of patients.
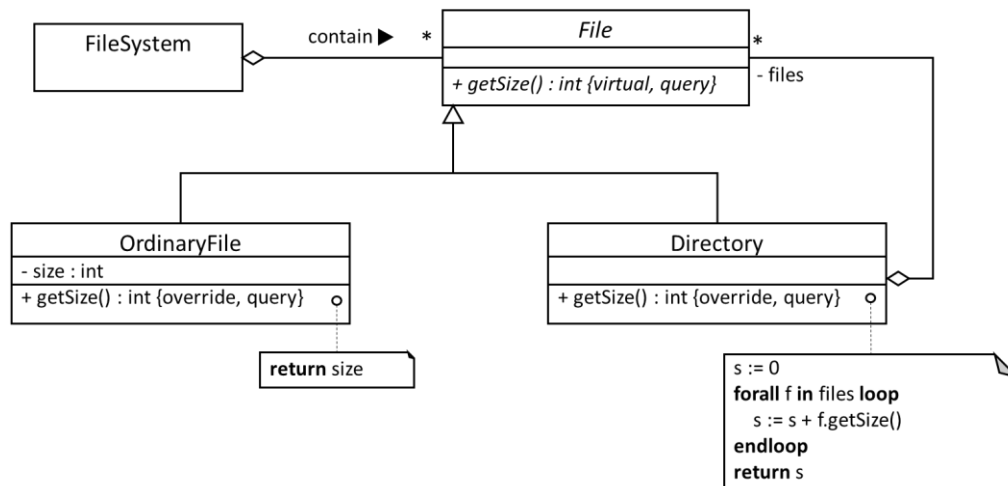

Modify the class diagram according to this case: employees can also get sick, their employment status will not be terminated.

4. A garden is looked after by one gardener. The garden consists of sections, only one species of plant can be planted into each section. The time of planting (month) is recorded. Plants can be crops as potatoe, peas, onion; or flowers as tulip, carnation, rose. The ripening time of plants are given in months. List the sections of crops ripening in a given month.


```
result := <>
for i = 1 to |sections|  loop
    s := sections[i]
    if date − p.getPlantingTime() = p.getPlant().getRiperingTime() and p.getPlant().isCorp()
    then result := result ⊕i endif
endloop
return result
```

Garden
+ ripe(date: int) : int[*] {query}
+ planting(i:int, p:Plant, date:int) : void

+ garden

sections[i].planting(p, date)

- sections * {ordered}

Section
- plantingTime : int {0 < plantingTime < 13}
+ planting(p:Plant, date:int) : void
<< getter >>
+ getPlant() : Plant {query}
+ getPlantingTime() : int {query}

grow ▶   0..1
- plantSpecies

plantSpecies := p
plantingTime := date

Plant
ripeningTime : int
<< getter >>
+ getRipeningTime() : int {query}
+ isCrop() : bool {virtual, query}

return false

Crop
+isCrop() : bool {override,query}

return true

Flower

Potatoe   Peas   Onion   Tulip   Carnation   Rose

5. In the hierarchical file system ordinary files are stored in directories. Every directory can contain any number of ordinary files or directories. The topmost directory in such a filesystem, which is not contained inside another directory, is called the root directory. In the file system, files can be stored directly in the root directory, or in any directory. Determine the storage space consumed by a given directory.

FileSystem   contain ▶   *

File
+ getSize() : int {virtual, query}

*
- files

OrdinaryFile
- size : int
+ getSize() : int {override, query}

return size

Directory
+ getSize() : int {override, query}

```
s := 0
forall f in files loop
    s := s + f.getSize()
endloop
return s
```

Exercise:

A train consists of a locomotive and at least one passenger carriage. A train has the locomotive at the front and carriages are connected to the locomotive according to a given order. A train might contain various types of carriages (with different lengths). The possible types of carriages are: first-class carriage, second-class carriage, railway post office, dining car, sleeping car, freight wagon, flat wagon.

A train is either a passenger train (not containing freight or flat wagon) or a goods train (consisting of freight and flat wagons only). At the same time one locomotive can only be connected to one train. It is also the same for carriages. Determine the length of a given train.